ITHEA

International Journal INFORMATION THEORIES & APPLICATIONS

2008 Volume 15 Number 4

International Journal
INFORMATION THEORIES & APPLICATIONS
Volume 15 / 2008, Number 4

IJ ITA is official publisher of the scientific papers of the members of
the ITHEA International Scientific Society,
the Association of Developers and Users of Intellectualized Systems (ADUIS)
and the Association for Development of the Information Society (ADIS)

IJ ITA welcomes scientific papers connected with any information theory or its application.
IJ ITA rules for preparing the manuscripts are compulsory.
The **rules for the papers** for IJ ITA as well as the **subscription fees** are given on *www.foibg.com/ijita*.
The camera-ready copy of the paper should be received by e-mail: *info@foibg.com*.
Responsibility for papers published in IJ ITA belongs to authors.

General Sponsor of IJ ITA is the **Consortium FOI Bulgaria** (www.foibg.com).

International Journal "INFORMATION THEORIES & APPLICATIONS" Vol.15, Number 4, 2008

Printed in Bulgaria

Edited by the **Institute of Information Theories and Applications FOI ITHEA**, Bulgaria,
in collaboration with the V.M.Glushkov Institute of Cybernetics of NAS, Ukraine,
and the Institute of Mathematics and Informatics, BAS, Bulgaria.

Publisher: **Institute of Information Theories and Applications FOI ITHEA**
Sofia, 1000, P.O.B. 775, Bulgaria. www.ithea.org, www.foibg.com, e-mail: info@foibg.com

ISSN 1310-0513 (printed) **ISSN 1313-0463 (online)** **ISSN 1313-0498 (CD/DVD)**

AN ONTOLOGY- CONTENT-BASED FILTERING METHOD

Peretz Shoval, Veronica Maidel, Bracha Shapira

Abstract: *Traditional content-based filtering methods usually utilize text extraction and classification techniques for building user profiles as well as for representations of contents, i.e. item profiles. These methods have some disadvantages e.g. mismatch between user profile terms and item profile terms, leading to low performance. Some of the disadvantages can be overcome by incorporating a common ontology which enables representing both the users' and the items' profiles with concepts taken from the same vocabulary.*

We propose a new content-based method for filtering and ranking the relevancy of items for users, which utilizes a hierarchical ontology. The method measures the similarity of the user's profile to the items' profiles, considering the existing of mutual concepts in the two profiles, as well as the existence of "related" concepts, according to their position in the ontology. The proposed filtering algorithm computes the similarity between the users' profiles and the items' profiles, and rank-orders the relevant items according to their relevancy to each user. The method is being implemented in ePaper, a personalized electronic newspaper project, utilizing a hierarchical ontology designed specifically for classification of News items. It can, however, be utilized in other domains and extended to other ontologies.

Keywords*: Ontology, Retrieval models, Information filtering, Content-based filtering, User profiles.*

ACM Classification Keywords*: H.3 Information Storage and Retrieval, H.3.1 Content Analysis and Indexing, H.3.3 Information Search and Retrieval, I.7 Document and Text Processing.*

1. Introduction

In content-based filtering, the representations of the content of items (e.g. documents, News), i.e. the items' profiles, are compared with the representation of the users, i.e. the users' profiles. In order to enable matching and measuring the similarity between the profiles, it is assumed that a user's profile and an item's profile share a common method of representation (e.g., by keywords). The output of the matching process can be expressed as a ranking score, indicating the similarity between the user's profile and each item.

A user profile can be generated in various ways, including explicit definition by the user, or implicit analysis of the user's behavior (e.g. by logging and analyzing what the user read). An item's profile too can be generated in various ways, e.g. explicitly, by asking the originator (author) to specify proper index terms, or automatically, using a text classification algorithm which extracts terms representing the item's content in the best way. At any rate, no matter which method is used for creating either type of profile, content-based filtering has drawbacks due to well known problems of term ambiguity. For example, different terms may be used to represent the same content or the same user (synonymy); or, the same term may be used to represent different contents or different users (homonymy).

A possible way to overcome such problems of ambiguity might be through the use of ontology, i.e., a controlled vocabulary of terms or concepts, and semantic relationships among them. Ontology can bridge the gap between the terms in the users' profile and the terms representing the items. Ontology can be organized in various ways. For example, a taxonomy is a hierarchical structure with is-a relationships; in a thesaurus there are a few more types of relationships, e.g. BT/NT (broader-term; narrower terms) and general relatedness. Note that a thesaurus is a graph, not a hierarchy, because a term may have many NTs and more than one BT. In the newspapers domain, which is exemplified in this study, there is an ontology specifically generated for classification of News named NewsCodes, created by IPTC [Le Meur and Steidl, 2004].

Assuming that there exists ontology of a specific domain, which is used for representing users (user profiles), and contents of items (item profiles), the research question we deal with is how exactly to match and measure the similarity between a user's profile and an items' profile. Obviously, if a user's profile includes exactly the same concept (terms) as an item's profile, there is some similarity between them; but the two profiles may include different concepts and still be similar to a certain degree – depending on if and how "close" the concepts are in the two profiles with respect to the common ontology.

This research is conducted within the framework of *ePaper*, an electronic personalized newspaper research project, which is aimed to provide a personalized electronic newspaper to each reader. In the News domain, instant filtering of News items is important. Since a new item has no reading history, the filtering and personalization cannot rely on collaborative filtering (as opposed to other domains such as recommendation of books, movies, etc.), but rather need to rely on content-based filtering, so that once a new item arrives to the News repository, the content-based filtering algorithm can perform the necessary matching with the users' profiles and determine the degree of relevancy of each item to the potential users. If many News items accumulate in a certain period of time, the content-based filtering algorithm can rank-order the items according to their relevancy to each of the potential users.

The remaining of this paper is structured as follows: The next section provides a background on content-based filtering and on ontological modeling, and reviews related research on conceptual and ontological modeling employed in content-based filtering. Section 3, the main section of the paper, presents the proposed method for the ontology- content-based filtering, along with an example. Section 4 describes the evaluations that we plan to conduct with the proposed method, and the last section summarizes and proposes further research and extensions to the proposed method.

2. Background on Content-Based Filtering and Ontological Modeling

2.1. Content-Based Filtering

The information filtering approach is based on the information retrieval (IR) domain and employs many of the same techniques [Hanani et al., 2001]. One aspect by which information filtering differs from IR is with respect to the uses' interests. While in IR the users poses ad-hoc queries, in information filtering the users have profiles which represent their long-term interests, and the filtering system tries to provide to each user relevant items on a long-term basis. As said, the user profiles, as well as the item profiles, may consist of sets of terms. Based on some measure of similarity between the respective profiles, the filtering system selects and rank-orders the relevant items and provides them to the user.

The actual relevancy of an item provided by the system to a user can be determined by explicit or implicit user feedback. Explicit feedback requires the user to express the degree of relevancy of the provided item, while in implicit feedback the relevancy of the item is inferred by observing the user's behavior, e.g. the reading time. Implicit feedback may be more convenient for the user but more difficult to implement and less accurate. User feedback enables to update the user's profile according to what he/she actually read, liked or disliked.

There exist two main approaches in information filtering: collaborative and content-based. In collaborative filtering, the system selects and rank-orders items for a user based on the similarity of the user to other users who read/liked similar items in the past. In content-based filtering, the system selects and rank-orders items based on the similarity of the user's profile and the items' profiles.

A major advantage of content-based filtering is that users can get insight into the motivation why items are considered relevant to them, because the content of each item is known from its representation. Content-based filters are less affected by problems of collaborative filtering systems [Claypool et al., 1999] such as "cold start" and sparsity: if a new item is added to the repository, it cannot be recommended to a user by a collaborative filter before enough users read/rated it. Moreover, if the number of users is small relatively to the volume of items in the repository, there is a risk of the ratings coverage becoming very sparse, thinning the collection of recommendable items [Balabanovic and Shoham, 1997]. For a user whose tastes are unusual compared to the rest of the population, the system will not be able to locate users who are particularly similar, leading to poor recommendations.

However, content-based filtering has disadvantages too, for example the fact that it focuses on keyword similarity. This approach is incapable of capturing more complex relationships at a deeper semantic level, based on different types of attributes associated with structured objects of the text [Dai and Mobasher, 2001]. Consequently, many items are missed and many irrelevant items are retrieved [Blair and Maron, 1985].

Unlike humans, content-based techniques have difficulty in distinguishing between high quality and low quality information, since both good and bad information might be represented by the same terms. As the number of items increases, the number of items in the same content-based category increases too, further decreasing the effectiveness of content-based approaches [Claypool et al., 1999]. Another disadvantage of content-based methods is that they require analyzing the content of the document, which is computationally expensive and even impossible to perform on multimedia items which do not contain text.

To expand the first point of the disadvantages, it can be added that there is a tremendous diversity in the words that people use to describe the same concept (synonymy) and this places strict and low limits on the expected performance of keyword-based systems. If the user uses different words from the organizer (indexer) of the information, relevant materials might be missed. On the other hand, the same word can have more than one meaning (homonyms), leading to irrelevant materials being retrieved [Dumais et al., 1988]. This disadvantage is added to the fact that the basic models of content-based filtering assume a representation of documents as sets or vectors of index-terms and typically employ only primitive search strategies based solely on the occurrence of term or combinations of terms [Knappe, 2005].

Thus, extensions to the traditional content-based filtering methods should be considered. Extensions may include additional knowledge in the form of a coherent taxonomy of concepts in the domain spanned by the items. This type of conceptual knowledge would provide means for item and user profile representation.

There is a need for devising a content-based approach which extends the classical models, where the use of simple natural language analysis in combination with the knowledge contained in an ontology forms the basis for representations of both user profiles and items. Consequently, items can be described using a concept language and be directly mapped into the ontology. The similarity between such representation of the user and the representations of the items will be based on the proximity principle stating that the distance of two descriptive items in the ontology is directly related to their similarity [Knappe, 2005].

2.2. Ontological Modeling

Ontology is a specification of a conceptualization. It can be described by defining a set of representational concepts. These definitions are used to associate the names of entities in the universe (e.g., classes, relations, functions or other objects) with human-readable text, describing what the names mean, and formal axioms that constrain the interpretation and focus the well-formed use of these concepts [Khan, 2000]. When constructing ontology, not only concepts and relationships are defined, but also the context in which the concept (relationship) applies. Therefore, ontology defines a set of representational terms which are called concepts, and the interrelationships among the concepts.

Linguistic ontologies (e.g., WordNet) and thesauri express various relationships between concepts (e.g. synonyms, antonyms, is-a, contains-a), and have a hierarchical structure based on the relations between concepts. But they do not explicitly and formally describe what a concept means [Khan, 2000]. WordNet, for example, is an electronic lexical database that contains nouns, verbs, adjectives and adverbs which are organized into synonym sets (*synsets*), each representing one underlying lexical concept [Magnini and Strapparava, 2001]. It is offering two distinct services: a vocabulary which describes the various word senses, and an ontology which describes the semantic relationships among senses [Guarino et al., 1999].

An example of domain ontology is the IPTC NewsCodes [Le Meur and Steidl, 2004]. This is a 3-level hierarchical ontology of concepts targeted to News description; it currently contains approximately 1,400 concepts. A first level concept of NewsCodes is called Subject; a second level – Subject Matter, and a third – Subject Detail. Figure 1 demonstrates an example.

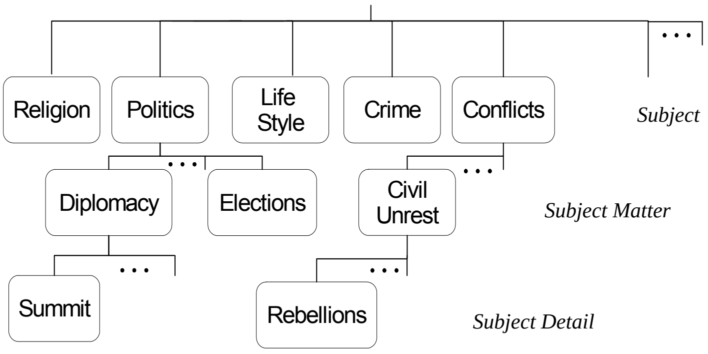


Figure 1: Example of IPTC NewsCodes ontology

2.3. Related Work

Savia et al. [1998] was one of the first to present a hierarchical representation for describing documents and user profiles by attaching metadata to each document and using the same method to generate a compatible representation of users' interests. A hierarchical representation was chosen in order to develop a document classification system understandable to humans and yet not restricted to text documents. Savia et al. chose to take advantage of the hierarchical metadata concepts along with an asymmetric distance measure, which considers not only the concepts appearing both in the user's profile and in the document's profile, but also concepts appearing in the document's profile which do not appear in the user's profile. The underlying assumption for the asymmetric measure was that the best matching documents are not necessarily those that cover all the interests at the same time. Distance computations were performed on different levels of the hierarchy and the metadata was represented in a fuzzy distribution among the leaf nodes of the concept tree.

Ontological and conceptual modeling was used in order to extract user profiles, such as the four-level ontology used in the Quickstep system [Middleton et al., 2001] which recommends papers to researchers by combining both content-based and collaborative filtering techniques. Papers were represented as term vectors with term frequency normalized by the total number of terms used for a term's weight. Whenever a research paper was browsed and had a classified topic, it accumulated an interest score of that topic for the particular user. In the ontology-based user profile, whenever a topic received some interest all its super classes gained a share: the immediate super-class gained 50% of the main topics value; the next super-class gained 25%, and so on. This way, general topics rather than just the most specific ones were also included in the profile and thus produced a broader profile. Recommendations were computed based on the correlation between the user's topics of interest and papers classified to those topics.

Another work which used ontology for content-based retrieval was the electronic publishing system CoMet [Puustjärvi and Yli-Koivisto, 2001]. CoMet extracted metadata information both about users and about contents of documents (document profiles) and stored the metadata in hierarchical ontology structures. Comparison between a user's profile and the documents' profiles was performed by finding the largest combined hierarchy (LCH), which is the largest hierarchy that the user profile and the document profile share in the ontology. By using the weights on the nodes in each level, a similarity measure was calculated between the documents that had an LCH with a user profile. In a weighted LCH-matching, the deepness of the LCH was emphasized in the matching calculations, since the depth of the hierarchy has a significant effect on the expression power of the incorporated ontology. The depth of the profile was also suggested to be used as a generalization tool. For example, if a user is interested in news items on F1 (a sport car), one can assume that she would like to view other motor sport related items when F1 news items are not available. The result of the matching generated a set of news items most suitable for the user according to the calculation result of LCH-matching.

Pereira and Tettamanzi [2006] illustrated a novel approach to learning users' interests on the basis of a fuzzy conceptual representation of documents, by using information contained in ontology. Instead of a keyword representation, documents were represented as a vector of components expressing the “importance” of the concepts. In order to choose the concepts that would represent a document, they considered both the leaf concepts and the internal nodes of the ontology. The internal nodes were implicitly represented in the importance vector by “distributing” their importance to all their descendants down to the leaf concepts. All documents with a certain level of similarity were grouped together into fuzzy clusters, in order to express user interests with respect to clusters instead of individual documents. Since the clusters were fuzzy, each document received its membership degree for that cluster, meaning it could belong to more than one cluster. A user model was represented as a vector of membership degrees which described the model’s guess of the extent to which the user was interested in each document cluster. A user profile was set up by adding to the list of its interest groups the instances of clusters with features similar to those requested by the user.

In the above survey we have emphasized methods involving the incorporation of ontologies both for user profile generation and for representation of items. Some of the methods employed ontology in order to acquire user profiles more accurately, while others used ontology in order to perform disambiguation of a user profile. In most cases, the ontology was used in all of the steps taken towards the retrieval of items according to the user profile. All studies which incorporated ontology in their content-based filtering method provided better and more accurate results compared to traditional content-based methods. This encouraged us to adopt the ontology approach and inspired us to introduce a novel filtering method which incorporates ontology.

3. The New Method for Ontological-Content-based Filtering

3.1. Research Goal

The aim of this research is to develop, implement and evaluate a new ontology-based filtering method, which filters and ranks relevant items by measuring the similarity of user profiles and item profiles, both consisting of ontology concepts, by considering the "closeness" (or distance) of concepts in the profiles, based on their location in the ontology. We utilize the method in the News domain, as part of *ePaper*, a research project which includes the development of a personalized electronic newspaper system. In this research we incorporate ontology for the News domain and exploit its three-level hierarchy in the representation of user profiles and News items profiles, and in the process of matching between them.

3.2. The Ontological-Content-based Filtering Method

The filtering method, initially proposed by Shoval [2006], is based on the assumption that each item (e.g. a News item) and each user profile (e.g. a reader of the *ePaper*) is represented with a set of concepts taken from the ontology. In the *ePaper* system we use the IPTC NewsCodes ontology, which is exemplified in Figure 1. It may be assumed that the generation of an item's representation (profile) is done automatically, utilizing some classification technique which analyses both the metadata describing the item and the actual text of the item. (We do not elaborate here on how this is done because as it is not an essential part of the proposed method.) Similarly, it may be assumed that an initial user profile is generated explicitly by the user who selects concepts from the ontology and assigns them weights of importance. Subsequently, the concepts in the initial user profile and their weights are updated implicitly, based on monitoring the items actually read by the user and considering the ontology concepts by which those items are represented. (This part is not elaborated here as well; suffice is to know that at any point in time a user's profile contains an up-to-date weighted set of ontology concepts.)

Following are the details of the filtering method.

Representation of contents – an item's profile:

An item's profile consists of a set of ontology concepts which represent its content. The concepts representing an item are the most specific ones in a certain branch of the hierarchy. For example, if an item deals with 'sport' and specifically with 'football', it is represented with 'football' concept only; the ontology can tell that the latter is a child (subtype) of the former.

Obviously, an item may be represented with many ontology concepts; each concept may appear in any branch of the ontology hierarchy and at any level – all depending on the actual content of that item. For example, an item's profile may include the concepts 'politics' (a top-level concept), 'football' (child of 'sport') and 'rebellions' (grandchild of 'conflicts'). Note that the profile may include sibling concepts, i.e. children of the same super concept. For example, an item's profile may include both 'football' and 'basketball' (children of 'sport').

Note that we do not assume that the concepts representing an item are weighted, although the proposed filtering algorithm can be adjusted for such possibility.

Representation of users – a user's profile:

A user's content-based profile consists of a weighted list of ontology concepts representing his/her interests. Obviously, a user's profile may consist of many ontology concepts, each appearing in different branches and at different levels of the hierarchy. For example, a user's profile may include the concept 'sport' only, or 'sport' and 'football', or 'football' and 'basketball', or all the three – besides many other concepts. This means that a certain concept in an item's profile may be "matched" (i.e. compared) with more than one equivalent concept in the user's profile. For example, if an item's profile includes 'football' and a user's profile includes both 'sport' and 'football', then there is a "perfect match" between the two profiles due to the common concept 'football', and also a "partial match" due to the parent concept 'sport'.

As stated before, the user's content-based profile may be generated initially by the user who selects concepts from the ontology and assign them weights of importance. (The total of the weights is normalized 100%). Then, the user's profile is constantly updated according to implicit feedback from the user: when a user reads an item and finds it interesting, the concepts in that item's profile which are not yet in the user's profile are added to it, and the weights of all concepts in this profile are recalculated as follows: a new concept is added with 1 'click' (a 'click' indicates how many times that concept was found relevant to the user) and the weight of an existing concepts is increased by 1 'click'. The weight of each concept in the user's profile is the number of its 'clicks' divided by the total number of 'clicks' in the user's profile. (Hence, the weights sum up to 100 %.)

Measuring similarity between an item and a user:

An item and a user are similar to a certain degree if their profiles include common (the same) concepts or related concepts, i.e. concepts having some kind of parent-child relationship. An item's profile and a user's profile may have many common or related concepts; obviously, the more common or related concepts, the stronger is the similarity between them. For example, if a user's profile includes 'football' and 'sport', this profile is similar (to a certain degree) to an item including these two concepts, but it is less similar to an item including just 'sport', and is more similar to an item including 'sport', 'football' and 'basketball'.

In the *ePaper* project, we adopted the 3-level NewsCodes ontology, so related concepts may be only one or two levels apart (parent-child or grandparent-grandchild), but generally concepts may be more levels apart. It is obvious that the closer two concepts are in the ontology, the closer are the two objects which they represent (i.e. the user and the item).

When dealing with related concepts appearing a user's profile and in an item's profile, two different cases can be distinguished: in one case, the concept in the user's profile is more general than the related concept in the item's profile (one or two levels apart), meaning that the user has a more general interest in the topic which the item deals with. In the other case, the concept in the user's profile is more specific than the related concept in the item's profile (one or two levels apart), meaning that the user has more specific interests in the topic dealt in the item. In any of the above cases of "partial match" between the user and item concepts, we should also consider the distance between the concepts: two related concepts which are only one level apart are closer (i.e. more similar) than two concepts which are two levels apart.

Scores of similarity:

Based on the above, in a 3-level hierarchical ontology we can distinguish between 9 different possible cases of similarity between concepts in a user's profile and an item's profile, as portrayed in Figure 2.

- **"Perfect match"**: the concept appears both in the user's profile and in the item's profile. I1, I2, I3 (see Figure 2) denote the level of a concept in an item's profile, and U1, U2, U3 - the level of a concept in the user's profile. A 'perfect match' can occur in 3 cases:
  - I1=U1 (e.g. both item and user profiles include 'sport')
  - I2=U2 (e.g. both item and user profiles include 'football')
  - I3=U3 (e.g. both item and user profiles include 'Mondeal games')

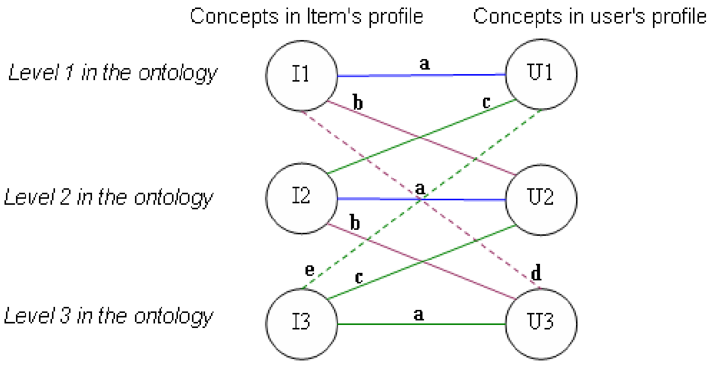


Figure 2: Hierarchical similarity measure

- **"Close match"**: a concept appears only in one of profiles, while a parent or child of that concept appears in the other profile. A 'close match' can occur in 2 **pairs** of cases:
  - I1=U2 (e.g. item concept is 'sport', while user concept is 'football')
  - I2=U3 (e.g. item concept is 'football' while user concept is 'Mondeal games')

  In the above 2 cases, the item's concept is more general than the user's concept (1 level apart), i.e. the user interest is more precise/specific than the item.

  - I2=U1 (e.g. item concept is 'basketball' while user concept is 'sport')
  - I3=U2 (e.g. item concept is 'Euro league' while user concept is 'basketball')

  In the above 2 cases, the item concept is more specific than the user concept, i.e. the user's interest is more general.

Note that in all the above 4 cases there may be more than one occurrence of 'close match' between the concepts. For example, in the case I1=U2, assume the item's concept is 'sport' while the user's profile includes both 'football' and 'basketball' concepts. When measuring similarity, we have to consider all possible 'close matches' between parent and children concepts.

- **"Weak match":** a concept appears in one profile, while a grandparent concept or a grandchild concept appears in the other profile (concepts are 2 levels apart). A 'weak match' can occur in 2 cases:
  - I1=U3 (e.g. item concept is 'sport' while user concept is 'Mondeal games') – in this case the item is much more general than the user's interest.
  - I3=U1 (e.g. item concept is 'Euro league' while user concept is 'sport') – in this case the item is much more specific than the user's interest.

  Recall that there may be more than one occurrence of 'weak match' between the concepts. For example, in the case I3-U1 the user concept is 'sport' while the item concepts include 'Euro league' and 'Mondeal games'.

For each of the 9 possible cases we determine a score of similarity. In the 3 cases of "perfect" match' labeled 'a' (see Figure 2) the score is 1 (maximal); in all other cases the score should be less than 1, depending if it is a 'close' or a 'weak' match and on the "direction" of the relationship, i.e., whether the user's concept is more general or more specific than the item's concept. For example, the score for the case I1=U2 (the item's concept is more general than the user's concept) may be 2/5, while the case I2=U1 (the item's concept is more specific than the user's concept) may score 2/3 – higher. The rationale for this may be that in the first case the item deals with a more general concept than the user's interest, yielding lower Precision than in the other case, where the item deals with a more specific concept than the user's interest, thus yielding higher Precision. But this assumption, as well as the exact scores of similarity for all possible cases is subject to experimentation.

The following is a possible scoring scheme for the 9 possible cases:

- I1=U1 → 1; I2=U2 → 1; I3=U3 →1 (3 cases of "perfect match"; marked **a** in Figure 2)
- I1=U2 → 2/5; I2=U3 → 2/5 (2 cases of "close match" - item concept is more general; marked **b**)
- I2=U1 → 2/3; I3=U2 → 2/3 (2 cases of "close match" - item concept is more specific; marked **c**)
- I1=U3 →1/3 (case of "weak match" - item concept is much more general; marked **d**)
- I3=U1 → 1/2 (case of "weak match" - item concept is much more specific; marked **e**)

Measure of similarity between item and user:

The similarity of an item's profile to a user's profile is based on the number of "perfect match", "close match" and "weak match" of concepts between the two profiles, and on the weights of the concepts in the user's profile. The overall Item Similarity score (IS) is computed as follows:

$$IS = \frac{\sum_{i \in Z} N_i \cdot S_i}{\sum_{j \in U} N_j}$$

where:

Z - number of concepts in item's profile
U - number of concepts in user's profile
i - index of the concepts in item's profile
j - index of the concepts in user's profile
S_i - score of similarity, depending if it is a "perfect", "close" or a "weak" match of concept *i* to a respective concepts in the user's profile. (Note that in case of no match at all, $S_i = 0$.)
N_i - number of clicks on the concept (used to determine the concepts' weights)

The matching algorithm:

The algorithm can be applied for measuring the similarity of a single item to a single user, or for rank ordering by relevancy a batch of items for a single user, or for rank ordering by relevancy a batch of users for a single item, or for rank ordering by relevancy a batch of items for a batch of users – all depending on the specific need/application.

The algorithm described below is applied for measuring the similarity of a single item's profile to a single user's profile. The algorithm is expressed in pseudo-code; it does not refer to any specific programming language, database system and other implementation aspects. However, it may be assumed that due to size on one hand and efficiency on the other hand, during execution the ontology resides in memory.

Since a user's profile may include many concepts (depending, among else, on how many items he already read), some with very low weights ('clicks'), it might be worthwhile to include in the computation of similarity only the most important concepts, e.g., the top 10 concepts or the concepts having weight above a certain threshold. The exact number of concepts has to be determined in experiments.

The algorithm consists of two loops: one over the concepts in the Item-list (i.e., list of concepts in the item's profile), searching for matches in the User-list (i.e., list of concepts in the user's profile); the other loop is over the User-list, searching for matches in the Item-list. Within each loop, if there is no "perfect match" the search is for a match with the parent or grand-parent of the item. (There is no need to search for children and grandchildren, a time-consuming task, because the first loop finds matches from the other list of concepts.)

Legend:

- Score: total score of similarity b/w item and concept
- I-concept: a concept in Item-list
- U-concept: a concept in User-list
- w: weight of concept in User-list that is being matched.

```
Begin
Score=0
Repeat for each I-concept in Item-list:
  Do case:
    - If I-concept is in User-list then Score= ++1*w     /*"perfect match"/
    - If parent of I-concept is in User-list then Score= ++ 2/3*w     /*"partial match" type c: I2=U1 or I3=U2/
    - If grandparent of I-concept is in User-list then Score= ++ 1/2*w     /*"weak match" type e: I3=U1/
  End case.
Until end of Item-list.
Repeat for each U-concept in User-list:
  Do case:     /*no need to check again for "perfect match" between concepts of same item and user profiles/
    - If parent of U-concept) is in Item-list then Score= ++ 2/5*w     /*"partial match" type b: I1=U2 or I2=U3/
    - If grandparent of U-concept is in Item-list then Score= ++ 1/3*w     /*"weak match" type d: I1=U3/
  End case.
Until end of User-list.
End.
```

Notes:

1) The scores for each type of match used in the algorithm are given as examples, as described above.

2) Not all user concepts must participate in the computation; as said, only the n-top concepts might be considered.

3.3. Example

The following example demonstrates the application of the filtering method using a few simulated items' profiles and a user's profile. The calculations are based on the matching scores demonstrated above.

Items' Profiles:

Item #	Ontology concepts representing the item*
Item 1	Crime → Laws
	Unrest → Civil unrest → Social conflict
Item 2	Sport → American Football
	Health → Injury
Item 3	Science → Natural science → Astronomy
Item 4	Life style and leisure
	Disaster and accident → Emergency incident

A User's Profile:

Ontology concepts in the user's profile	Number of clicks (weight)
Sport	20
Health	12
Crime → Laws → Criminal	3
Unrest	10
Lifestyle and leisure → Fishing	8

** An arrow represents parent-child relationship. The item's profile includes only the lower-level concepts.*

The application of the algorithm yields the following rank ordered list of items:

Item #	Ranking score
Item 2	0.40
Item 1	0.11
Item 4	0.06
Item 3	0.00

It can be observed that Item 2 gets the highest score because its profile includes 'American football', a child of 'Sport' in the user's profile; and 'Injury', a child of 'Health' in the user's profile – and both concepts in the user's profile have relatively high weights. Here is the exact computation of the ranking score, assuming we consider the scoring scheme in which I2=U1 → 2/3:

$$IS \ = \ \frac{\frac{2}{3}\cdot 20+\frac{2}{3}\cdot 12}{20+12+3+10+8}=0.4$$

Item 1 gets the second highest score because of the 'close match' between its 'Laws' concept and 'Crime' in the user's profile, and also because of the 'weak match' between its 'Social conflict' concept and 'Unrest' in the user's profile. Item 1 gets a lower ranking than Item 2 because of two reasons: 1) lower scores of similarity; 2) lower weight of the matched concepts. Item 4 gets even a lower ranking because it has only one concept having any match with the user's profile: its concept 'Lifestyle and leisure' is a 'close match' with 'Fishing' in the user's profile. Item 3 gets a ranking score 0 because it has no match at all with the user's profile.

4. Evaluations of the Filtering Method

We plan to evaluate the filtering method in a controlled setting utilizing a prototype of the *ePaper* system. The main objective of the evaluations is to examine the effectiveness of the method, including the contribution of the various matching types (i.e. "perfect", "close" and "weak" matches) to performance, and to determine the optimal values for the various matching scores.

4.1. Measures of Effectiveness

Traditional measures of effectiveness of information retrieval systems usually include Precision and Recall. But these measures may not be appropriate for evaluating the quality of rank ordered items because the user might read only some of the top ranked items, while Precision and Recall are based on the total number of relevant or retrieved items, respectively. We are considering several rank accuracy measures which are more appropriate to evaluate rank-ordered results, and where the users' preferences in recommendations are non-binary. Following Herlocker et al. [2004], we are considering the following measures:

- *Rank Correlations*, such as Spearman's ρ and Kendall's Tau, which measure the extent to which two different rankings agree independent of the actual values of the variables.
- *Half-life Utility* metric, which attempts to evaluate the utility of a ranked list to the user. The utility is defined as the difference between the user's rating for an item and the "default rating" for an item.
- *NDPM Measure*, which is used to compare two different weakly-ordered ratings.

4.2. What will be Evaluated

The evaluations will include the following objectives:

1. **Determination of the matching scores**: The filtering method assumes different matching scores to the various possible types of matching between concepts in the user's profile and the item's profile: the highest score (1) is given to a "perfect" match, while a "close" match and a "weak" match get lower scores, considering also the direction of the hierarchical relation between the concepts (i.e., whether the user's concept is more general or more specific than the item's concept). This part of the experimental evaluations is aimed to determine the optimal scores for the different types of match.
2. **Evaluation of the contribution of the various types of match between user concepts and item concepts:** It is obvious that the more common concepts appear in both the user's profile and the item's profile, and the closer the user's concepts is to the item's concepts - the more relevant is the item to the

user. The question is: what is the residual contribution of the different types of match (i.e. "closeness") to the quality of the results. For example, what is the quality of results if only "perfect" matches are considered? What is the additional contribution of "close" matches? What is the additional contribution of "weak" matches? Results of these evaluations may enable us to determine if it is worthwhile to consider all types of relatedness, or perhaps only some of them are sufficient to obtain quality results.

3. **Considering more than one match between related concepts in the user's and item's profiles:** A user's profile may contain concepts from various levels of one branch of the hierarchy (e.g., the profile may include the concepts 'sport' and 'football'). The question is whether all concepts along the branch should be considered when compared to the item's profile, or perhaps only the concept having the highest score*weight. (Note that the score itself is determined according to the "closeness" factor, while the weight is determined according to the number of read items which included the concept).
4. **Determining the number of concepts in a user's profile to consider:** A user's profile may include many concepts, each having a certain weight (as explained above). Considering all concepts in the profile might be time consuming (in terms of processing time). It is likely that concepts having low weights will not contribute much to the quality of the filtering results. We will examine the contribution of low-weight concepts in order to determine a threshold for an optimal number of concepts or for concept weights. Initially, the algorithm will consider all concepts; then we will omit certain concepts (beyond a certain number or below a certain weight) and see to what degree it affects performance.

4.3. The Evaluation Plan

We plan to conduct user studies with real users (subjects), each having a content-based profile representing his interests. Some of the subjects will have similar (overlapping) profiles and some will have dissimilar profiles, in order to find out how the filtering method affects similar and dissimilar subjects.

The subjects will read News items delivered to them by the *ePaper* system and rate each item as "interesting" or "not interesting". Alternatively we are considering to use a scale bar (say from 1 to 5) to expresses the level of interest.

Some of the items read by a subject will be used for updating his/her profile (training set), while the remaining items will be used for the various tests described above (test sets). A test set of items, rank ordered by the algorithm (in any of its variations) will be compared to the subject's ratings of the items, and the measures of effectiveness (as described above) will be applied to determine the quality of the result. As described, the algorithm will vary from test to test:

1. As a result of the first set of tests (determination of the matching scores) we will adopt the best set of matching scores of similarity; these will be used in all the subsequent evaluations.
2. As a result of the second set of tests (evaluation of the contribution of the various types of match between user concepts and item concepts) we will determine the contribution of each level of proximity relatively to the performance obtained at the prior level, and determine if it is worthwhile to consider all or only parts of match types (e.g. only 'perfect' and 'close' matches).
3. As a result of the third set of tests (considering more than one match between related concepts in the user's and item's profiles) we will determine whether all concepts along a branch should be considered or only the concept having the highest score*weight. Based on that, the filtering algorithm will be adjusted.
4. As a result of the fourth set of tests (determining the number of concepts in a user's profile to consider) we will calibrate the algorithm to consider only a certain number of concepts in the user's profile, limited by a threshold number or weight.

5. Summary and Further Research

We presented a new content-based filtering method that uses ontology for representing user and item profiles, and for ranking items according to their relevancy in the electronic newspapers domain. The method is being implemented in the *ePaper* system for personalized electronic newspaper. The filtering method considers the hierarchical distance, or closeness, between concepts in the user's profile and concepts in the items' profile.

The method can be enhanced in various aspects. One possible enhancement is to assign more importance to concepts co-occurring in items read in the past by the user. An item which includes co-occurring concepts might get a higher score than an item including the same concepts that did not co-occur in past read items. The added value of the incorporation this enhancement will be examined before being implemented in the method.

Another possible enhancement of the method is to consider penalty scores for concepts appearing in an item but not in the user's profile. This idea, which was adopted from Savia et al. [1998], means that a concept in an item's profile which does not appear in the user's profile might be given a negative (penalizing) score. The contribution of such penalty to the quality of the filter can be determined in empirical experiments.

The proposed filtering method utilizes a 3-level hierarchical ontology of News. It can, however, be generalized to other domains with their specific ontologies; and it must not be restricted to three levels. Moreover, the method can be enhanced to deal not just with a hierarchical but also with a network-based (DAG) ontology, where a concept may have many parent concepts, not only child concepts. Another possible extension to the method is to consider more types of relations between concepts, besides parent-child and grandparent-grandchild, e.g. twins of concepts. For example, a use's profile may include 'football' while an item may include 'basketball'. These extensions will be dealt with in further research.

Acknowledgment

This work was performed as part of a Deutsche-Telekom / Ben-Gurion University joint research project.

Bibliography

[Balabanovic et al., 1997] Balabanovic, M. & Shoham, Y. Fab: Content-based, collaborative recommendation. Communications of the ACM, 40(3), 66-72.

[Blair and Maron, 1985] Blair, D.C. & Maron, M. E. An evaluation of retrieval effectiveness for a full-text document retrieval system. Communications of the ACM, 28, 289-299.

[Claypool et al., 1999] Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D. & Sartin M. Combining content-based and collaborative filters in an online newspaper. Proc. of ACM SIGIR Workshop on Recommender Systems.

[Dai and Mobasher, 2002] Dai, H., & Mobasher, B. Using ontologies to discover domain-level web usage profiles. Proc. of the Second Semantic Web Mining Workshop at PKDD 2001, Helsinki, Finland.

[Dumais et al., 1988] Dumais, S.T., Furnas, G.W., Landauer, T.K. & Deerwester, S. Using latent semantic analysis to improve information retrieval. Proc. of CHI'88 Conf. on Human Factors in Computing, New York: ACM, 281-285.

[Guarino et al., 1999] Guarino, N., Masolo, C. & Vetere, G. OntoSeek: Content-based access to the Web. IEEE Intelligent Systems 14(3), 70-80.

[Hanani et al., 2001] Hanani, U., Shapira, B. & Shoval, P. Information filtering: overview of issues, research and systems. User Modeling and User-Adapted Interaction (UMUAI), 11(3), 203-259.

[Herlocker et al., 2004] Herlocker, J.L., Konstan, J.A., Terveen, L.G. & Riedl, J.T. Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems, 22(1), 5-53.

[Khan, 2000] Khan, L. Ontology-based Information Selection. Ph.D. Thesis, University of South California.

[Knappe, 2005] Knappe, R. Measures of semantic similarity and relatedness for use in ontology-based information retrieval. Ph.D. Thesis, Roskilde University, Department of Communication, Journalism and Computer Science.

[Le Meur and Steidl, 2004] Le Meur, L. & Steidl, M. NewsML 1.2 – Guidelines V1.00. Int'l Press Telecommunications Council. Retrieved: Dec. 07, 06: http://www.newsml.org/IPTC/NewsML/1.2/documentation/NewsML_1.2-doc-Guidelines_1.00.pdf

[Magnini and Strapparava, 2001] Magnini, B. & Strapparava, C. Improving user modelling with content-based techniques. Proc. of the 8th Int'l Conference on User Modeling 2001. M. In: Bauer, P., Gmytrasiewicz, J. & Vassileva, J. (Eds.): Lecture Notes in Computer Science, 2109. Springer-Verlag, London, 74-83.

[Middleton et al., 2001] Middleton, S.E., De Roure, D.C. & Shadbolt, N.R. Capturing knowledge of user preferences: ontologies in recommender systems. Proc. of 1st Int'l Conf. on Knowledge Capture, 100-107, Victoria, BC, Canada.

[Pereira and Tettamanzi, 2001] Pereira, C. C. & Tettamanzi, A. G. An ontology-based method for user model acquisition. Soft Computing in Ontologies and Semantic Web, Berlin, Springer.

[Puustjärvi and Yli-Koivisto, 2001] Puustjärvi, J. & Yli-Koivisto, J. Using metadata in electronic publishing. Project internal publication, available at http://www.soberit.hut.fi/comet/.

[Savia et al., 1998] Savia, E., Koskinen, T. & Jokela, S. Metadata based matching of documents and user profiles. Proc. of Finnish Artificial Intelligence Conference, STeP'98.

[Shoval, 2006] Shoval, P. Ontology and content-based filtering for the ePaper project. Working Paper, BGU.

Authors' Information

Peretz Shoval *(Department of Information Systems Engineering, Ben-Gurion University, Beer-Sheva 84105, Israel; e-mail: shoval@bgu.ac.il) is a Professor at the Dept. of Information Systems Engineering of Ben-Gurion University. He earned his Ph.D. in Information Systems from the University of Pittsburgh, where he specialized in expert systems for information retrieval. In 1984 he joined Ben-Gurion University, where he founded the Information Systems Program and later on founded and headed the Dept. of Information Systems Engineering. Prior to moving to academia, Shoval held professional and managerial positions in computer and software companies. Shoval's research interests include information systems analysis and design methods, data modeling, and information retrieval and filtering.*

Veronica Maidel *(Department of Information Systems Engineering, Ben-Gurion University, Beer-Sheva 84105, Israel; e-mail: maidel@bgu.ac.il) received her B.Sc. from Tel-Aviv University in 2001 and is currently a graduate student at the Dept. of Information Systems Engineering of Ben-Gurion University. Her research is on content-based filtering. This paper is part of her research.*

Bracha Shapira *(Department of Information Systems Engineering, Ben-Gurion University, Beer-Sheva 84105, Israel; e-mail: bshapira@bgu.ac.il) is Senior Lecturer at the Department of Information Systems Engineering of Ben-Gurion University. She holds a M.Sc. in Computer Science from the Hebrew University in Jerusalem and a Ph.D. in Information Systems from Ben-Gurion University. Her research interests include Information Retrieval and Filtering, specializing in various aspects of user profiling and personalization. In addition, she has worked on privacy preservation while browsing and on formal models of Information Retrieval systems. She is leading research projects in these domains at the Deutche-Telekom research lab at Ben-Gurion University.*

LOGIC BASED PATTERN RECOGNITION - ONTOLOGY CONTENT (2)[1]

Levon Aslanyan, Vladimir Ryazanov

Abstract: *Logic based Pattern Recognition extends the well known similarity models, where the distance measure is the base instrument for recognition. Initial part (1) of current publication in iTECH-06 reduces the logic based recognition models to the reduced disjunctive normal forms of partially defined Boolean functions. This step appears as a way to alternative pattern recognition instruments through combining metric and logic hypotheses and features, leading to studies of logic forms, hypotheses, hierarchies of hypotheses and effective algorithmic solutions. Current part (2) provides probabilistic conclusions on effective recognition by logic means in a model environment of binary attributes.*

1. Introduction

Pattern Recognition consists in reasonable formalization (ontology) of informal relations between object's visible/measurable properties and of object classification by an automatic or a learnable procedure [1]. Similarity measure [1] is the basic instrument for many recognition formalisms but additional means are available such as logical terms discussed in part (1) of current research [2]. Huge number of recognition models follows the direct goal of increasing recognition speed and accuracy. Several models use control sets above ordinary learning sets, others use optimization and other direct forces. Besides, more alternative notions are available to describe algorithmic properties. In existing studies the role of these notions is underestimated and less attention is paid to these components. In part (1) the attention is paid to implementing the learning set through its pairs of elements rather than the elements separately. The following framework is considered: given a set of logical variables

[1] The research is supported partly by INTAS: 04-77-7173 project, http://www.intas.be

(properties) $x_1, x_2, ..., x_n$ to code the studied objects, and let we have two types/classes for classification of objects: K_1 and K_2. Let $\beta \in K_1$, and $\gamma \in K_2$, and α is an unknown object in sense of classification. We say, that γ is separated by the information of β for α if $\beta \oplus \gamma \leq \beta \oplus \alpha$, where $\oplus$ is $mod\,2$ summation. Formally, after this assumption, the reduced disjunctive normal forms of two complementary partially defined Boolean functions appear to describe the structure of information enlargement of the learning sets. The idea used is in knowledge comparison. α is an object of interest. Relation $\beta \oplus \gamma \leq \beta \oplus \alpha$ informs that the descriptive knowledge difference of β and α is larger than the same difference of β and γ. This approach we call logic separation. While notion of similarity gives the measure of descriptive knowledge differences, the logic separation describes areas which are preferable for classes and learning set elements. In general the question is in better use of learning set. The learning set based knowledge, which is used by recognition procedure, at least is supposed to reconstruct the learning set itself. It is indeed negative when this information is not able to reconstruct the learning set. It is easy to check that the similarity knowledge can't reconstruct an arbitrary learning set, and only special sets allow reconstructing of objects by their distances [3]. Restructuring power is high when comparison is used for the set of all attribute subsets. Theoretically such structures are studied in discrete tomography problems [4], but practically even the use of pairs draws to known hard computational area of disjunctive normal forms.

Consider pairs of elements of the learning set, where each pair contains elements of different classes (the case of 2 learning classes is supposed). It was shown [2] that the logical separators divide the object space into three areas, where only one of these areas needs to be treated afterward by AEA (algorithms of estimation analogies – voting algorithms) [1]. This set is large enough for almost all weakly defined Boolean functions, but for the functions with compactness property it is small. Let, for $0 \leq k_0 < k_1 \leq n$, F_{n,k_0,k_1} be the set of all Boolean functions defined as follows: each of them has zero (false) value on the vertices of k_0-sphere centered at $\tilde{0}$, and has one (true) value on ($n - k_1$)-sphere centered at $\tilde{1}$. On the remainder vertices of n-cube the assignment/evaluation is arbitrary. These functions (for appropriate choice of k_0 and k_1) satisfy the compactness assumptions [8], and their quantity is not less than $2^{\varepsilon(n)2^n}$ for an appropriate $\varepsilon(n) \to 0$ with $n \to 0$. For these functions we have also, that for recovering the full classification by means of logical separators procedure, it is enough to consider a learning set which consists of any $n2^{n-\varepsilon(n)\sqrt{n}}$ or more arbitrary points. This is an example of postulations which will be considered below. It is relating the metric and logic structures and suppositions, although separately studies of these structures are also important. The follow up articles will describe the mixed hierarchy of recognition metric-logic interpretable hypotheses, which helps to allocate classification algorithms to the application problems.

2. Structuring by Logic Separation

Let f be a Boolean function (it might be partially or completely defined). Let N_f denotes the reduced disjunctive normal form of f and sets $N_0^f, ..., N_3^f$ [2] define areas, in which N_f and $N_{\bar{f}}$ take values {0,1}, {1,0}, {0,0} and {1,1} correspondingly. Identical to $N_0^f, ..., N_3^f$, similar areas are defined by logic separation - $M_0^f, ..., M_3^f$.

Let $f_0 \in P_2(n)$ (a completely defined Boolean function of n variables) and $f_0(\tilde{\alpha}) = 1$. Denote by $t(f_0, \tilde{\alpha})$ the number of k-subcubes included in N_{f_0} and covering the vertex $\tilde{\alpha}$. Let m_k is the average number of $t(f_0, \tilde{\alpha})$ calculated for all $f_0 \in P_2(n)$ and $\tilde{\alpha} \in N_{f_0}$. It is easy to check that $m_k = \frac{2^n C_n^k 2^{2^n - 2^k}}{2^n 2^{2^n - 1}} = \frac{C_n^k}{2^{2^k - 1}}$.

Dispersion d_k of the same value $t(f_0, \tilde{\alpha})$ is expressed as $d_k = C_n^k \sum_{j=0}^{k} C_k^j C_{n-k}^{k-j} 2^{-2^{k+1} + 2^j + 1} - \left(\frac{C_n^k}{2^{2^k - 1}} \right)^2$.

Applying the Chebishev inequality to above measures $t(f_0,\tilde{\alpha})$, m_k, d_k leads to the conclusion:

Proposition 1(8). $t(f_0,\tilde{\alpha}) \sim \frac{C_n^k}{2^{2^k-1}}$ for almost all pairs $f_0 \in P_2(n)$ and $\tilde{\alpha} \in N_{f_0}$, when $n \to \infty$ and $\frac{C_n^k}{2^{2^k}} \to \infty$.

Taking into account that for almost all Boolean functions the number of 1-vertices is equivalent to $2^{n-1}, n \to \infty$, we obtain that for almost all functions $f_0 \in P_2(n)$, almost all 1-vertices are covered by the number of k-intervals from N_{f_0}, which is equivalent to $\frac{C_n^k}{2^{2^k-1}}$, when $n \to \infty$ and $\frac{C_n^k}{2^{2^k}} \to \infty$. Particularly, this fact might be used to adjust the postulation in Proposition 7, [2]. Indeed, the $\frac{C_n^k}{2^{2^k-1}}$ intervals, coming from a common fixed vertex, cover not less than $\frac{C_n^k}{2^{2^k-1}}$ vertices of an n-cube.

Now consider arbitrary placement of any l points into the vertices of an n-cube M. Estimate for almost all functions $f_0 \in P_2(n)$ (see Proposition 1(8)) the main value of the number of vertices $\tilde{\alpha} \in N_{f_0}$, which are not covered by any of the k-intervals included in N_{f_0} which is pricked by our l vertices:

$$\mu(n,k,l) \prec (1+\varepsilon_1(n))2^{n-1}\frac{C^l_{2^n-(1+\varepsilon_2(n))C_n^k/2^{2^k-1}}}{C^l_{2_n}}, n \to \infty,\ \varepsilon_1(n) \to 0,\ \varepsilon_2(n) \to 0, \text{ and } \frac{C_n^k}{2^{2^k}} \to \infty.$$

Proposition 2(9). If $\frac{C_n^k}{2^{2^k}} \to \infty$ and $l \ge \varphi(n)\frac{2^n 2^{2^k}}{C_n^k}$, where $\varphi(n) \to \infty$ as $n \to \infty$, then random l vertices for almost all functions $f_0 \in P_2(n)$ prick such sets of k-subcubes included in N_{f_0}, which cover almost all N_{f_0}.

In case of $k = [\log\log n]$ we conclude that the minimal number l satisfying the above proposition, is not greater than $2^n n^2 / C_n^{[\log\log n]}$.

Notice, that in conditions of Proposition 7 [2] and Proposition 2(9) only the usability of condition F_0 (logic separation) is mentioned, so that these are the conditions, when usage of F_0, as a rule, doesn't imply to significant errors. Also, it is important, that we applied the condition F_0 to the whole class $P_2(n)$, although it was supposed for problems, satisfying compactness suppositions. So, it is interesting to know how completely the class $P_2(n)$ satisfies to these suppositions.

Let us bring now a particular justification of compactness conception [8]. Let $f_0 \in P_2(n)$. We call the vertex $\tilde{\alpha} \in M$ boundary vertex for function f_0, if the sphere $S(\tilde{\alpha},1)$ of radius 1 centered at $\tilde{\alpha}$, contains a vertex for which f_0 has the opposite value to $f_0(\tilde{\alpha})$. Denote by $\Gamma(f_0)$ the set of all boundary vertices of f_0. We will say that the function hipping (completion) procedure obeys the compactness conditions, if $|\Gamma(f_0)| = o(2^n), n \to \infty.$ It is easy to calculate that the average number of boundary vertices of functions $f_0 \in P_2(n)$ is almost 2^n. This shows that $P_2(n)$ contradicts the compactness conditions. The same time we proved that the use of the F_0 rule in a very wide area $P_2(n)$ doesn't move to a sensitive error. Below we consider an example problem, which obeys the compactness assumptions, and will follow the action of the rule F_0 on that class. Before that we justify some estimates for the set M_3^f.

Consider the class $\Phi_2(n,k(n),l(n))$ of all of partial Boolean functions, for which $|M_0| = l(n)$ and $|M_1| = k(n)$. We'll deal with the case $l(n) = o(2^n)$ and $k(n) = o(2^n)$. Estimate now the quantitative characteristics of sets M_0^f, M_1^f and M_3^f.

First estimate the average number of vertices of the cube, which are achievable from set M_0:

$$C_{03} \geq 2^n \frac{C^{k(n)}_{2^n - l(n) - 2^j}}{C^{k(n)}_{2^n - l(n)}} \left(1 - \frac{C^{l(n)}_{2^n - \sum_0^j C_n^j}}{C^{l(n)}_{2^n}} \right), \; j = 0,1,\cdots$$

Proposition 3(10). If $k(n)$ and $l(n)$ are $o(2^n)$, $n \to \infty$ and there exists a j_0, that $k(n)2^{j_0 - n} \to 0$ and $2^{-n} \sum_0^{j_0} C_n^i l(n) \to \infty$, then for almost all functions of class $\Phi_2(n,k(n),l(n))$, $|M_1^f| \approx o(2^n), n \to \infty$.

To except the trivial cases in the pattern recognition problems we have to suppose, that $k(n) \cong l(n), n \to \infty$. Then it is clear that choosing appropriate values for j_0 we get $|M_1^f| = o(2^n)$ and $|M_0^f| = o(2^n)$ for almost all functions of class $\Phi_2(n,k(n),l(n)), n \to \infty$.

Let us give an other estimation of c_{03}: $c_{03} \geq \sum_{j=0}^{n} C_n^j \frac{C^{l(n)}_{2^n - 2^j}}{C^{l(n)}_{2^n - 1}}$.

If $\lambda(n)$ is the minimal value for which $\sum_{i = n/2 - \lambda(n)}^{n/2 + \lambda(n)} C_n^i \sim 2^n, n \to \infty$, then $\lambda(n) \approx \sqrt{n}$.

From here we conclude:

Proposition 4(11). If $l(n) \geq 0$ and $2^{-n} k^2(n) 2^{C(n)\sqrt{n}} \to 0$ as $n \to \infty$ for $\forall c(n)$ - restricted, then almost ever $M_1^f \sim o(2^n)$.

So, for the small values of $k(n)$ and $l(n)$ from the each vertex of set $M_0 \cup M_1$, almost all vertices of the n-unite-cube almost ever are achievable. Comparing this, for example with [5] we find that for these classes F_0 works ineffectively.

3. Logic Separation on Compact Classes

Consider problems, satisfying the compactness assumptions. First of all it is evident, that for $M_0 \cup M_1 \supseteq \Gamma(f_0)$ the continuation of function f made on base of F_0, exactly correspond to the final result f_0. Taking into account that by the given description of the compactness assumptions $\Gamma(f_0) = o(2^n), n \to \infty$, we receive that in problems, satisfying the compactness assumptions we can point out learning sets of size $o(2^n), n \to \infty$, which allow to complete and exact continuation of function f_0 on base of condition F_0 only.

Let $\tilde{\alpha} \in M$ and $0 \leq k_1 \leq k_2 \leq n$. Consider functions $f_0 \in P_2(n)$, for which $M_0(f_0) \supseteq S(\tilde{\tilde{\alpha}}, n - k_2)$, $M_1(f_0) \supseteq S(\tilde{\alpha}, n - k_1)$ and which receive arbitrary values on vertices of sets $S(\tilde{\alpha}, k_2 - 1) \supseteq S(\tilde{\alpha}, k_1)$.

Denote the class of these functions by $K(n)$. It is evident, that for $|S(\tilde{\alpha}, k_2 - 1) \setminus S(\tilde{\alpha}, k_1)| = o(2^n)$ all the constructed functions satisfy the given formalisms for the compactness assumptions, and that the quantity of these functions is not less than $2^{\varepsilon_1(n)2^n}$, where $\varepsilon_1(n)$ is an arbitrary function of n, $\varepsilon_1(n) \to 0$ with the $n \to \infty$.

Take a point $\tilde{\beta} \in M, \rho(\tilde{\alpha}, \tilde{\beta}) = k, k < k_1$. It is evident that no more than $C_n^{[n/2]} \cong 2^n \sqrt{\frac{2}{\pi n}}, n \to \infty$ subsets of any fixed size are coming out from any point of n-cube. From the other hand it is evident, that it is enough to take $k_1 - k = o(\sqrt{n})$ as $n \to \infty$ to get the $|S(\tilde{\alpha}, k_1) \setminus S(\tilde{\alpha}, k)| = o(2^n)$. Suppose, that $|M_0(f_0) \cup M_1(f_0)| = l$, and that l points appear as the result of their appropriate placement on the vertices of the n-cube M, when all of these placements are equally probable. Estimate the probability of reaching of this point $\tilde{\beta}$ from zeros of function f_0.

$$\tau_l \le 2^n \sqrt{\frac{2}{\pi n}} \frac{C_{2^n - 2^{\varepsilon 2(n)\sqrt{n}}}}{C_{2^n}^l}, \varepsilon_{2(n)} \to 0 \text{ with } n \to \infty.$$

From here we conclude the

Proposition 5(12). Let $f_0 \in K(n, k_1, k_2)$ and f_1 -- is the continuation of function f on base of condition F_0. If $l \ge n 2^{n - \varepsilon_2(n)\sqrt{n}} = o(2^n)$, $n \to \infty$ and the set $M_0(f) \cup M_1(f)$ is formed as a random collection of points of size l from the set M, then almost ever the function f_1 is the continuation for f, which converges to the f_0 by the accuracy, tending to 1 with the $n \to \infty$.

Conclusion

Logic Separation is an alternative approach to pattern recognition hypotheses and formalisms, while the base concept uses the similarity approach. Structures appearing in this relation are based on terms of Reduced Disjunctive Normal Forms of Boolean Functions. Propositions 1-5(8-12) provide additional knowledge on quantitative properties of areas appearing in extending classification by means of compactness and logic separation principles.

Bibliography

1. Zhuravlev Yu. I. On an algorithmic approach to the problems of recognition and classification. Problemi Kibernetiki, 33, (1978) 5--68.
2. Aslanyan L. and Castellanos J., Logic based pattern recognition – ontology content (1), iTECH-06, 20-25 June 2006, Varna, Bulgaria, Proceedings, pp. 61-66.
3. Gavrilov G. and Sapojenko A., Collection of exercises of discrete mathematics, NAUKA, Moscow, 1973, 368 p.
4. Herman G.T. and Kuba A., editors. Discrete Tomography: Foundations, Algorithms and Applications. Birkhauser, 1999.
5. Zhuravlev Yu. I. Selected Scientific Works, Publishing House Magister, Moscow, (1998) 417p.
6. Aslanyan L. H. On a pattern recognition method based on the separation by the disjunctive normal forms. Kibernetika, 5, (1975), 103--110.
7. Vapnik V. and Chervonenkis A. Theory of Pattern Recognition. “Nauka”, 1974.
8. Aslanyan L. H. The Discrete Isoperimetric Problem and Related Extremal Problems for Discrete Spaces. Problemy Kibernetiki, 36, (1979), 85--128.
9. Nechiporuk E. I. On topological principles of self-correcting. Problemy Kibernetiki, 21, (1969), 5--102.
10. Graham N., Harary F., Livingston M. and Stout Q. Subcube Fault-Tolerance in Hypercubes. Information and Computation 102 (1993), pp. 280{314.
11. Glagolev V. V. Some Estimations of D.N.F. for functions of Algebra of Logic. Problemy Kibernetiki, 19, (1967), 75--94.

Authors' Information

***Levon Aslanyan** – Institute for Informatics and Automation Problems, NAS Armenia, P.Sevak St. 1, Yerevan-14, Armenia; e-mail: lasl@sci.am*

***Vladimir Ryazanov** - Computing Centre of the Russian Academy of Sciences, 40 Vavilova St., Moscow, GSP-1, 119991, Russian Federation, rvvccas@mail.ru*

DYNAMIC ONTOLOGIES IN INFORMATION SECURITY SYSTEMS[1]

Vladimir Jotsov

Abstract: *Different types of ontologies and knowledge or metaknowledge connected to them are considered and analyzed aiming at realization in contemporary information security systems (ISS) and especially the case of intrusion detection systems (IDS) or intrusion prevention systems (IPS). Human-centered methods INCONSISTENCY, FUNNEL, CALEIDOSCOPE and CROSSWORD are algorithmic or data-driven methods based on ontologies. All of them interact on a competitive principle 'survival of the fittest'. They are controlled by a Synthetic MetaMethod SMM. It is shown that the data analysis frequently needs an act of creation especially if it is applied to knowledge-poor environments. It is shown that human-centered methods are very suitable for resolutions in case, and often they are based on the usage of dynamic ontologies .*

Introduction

Contemporary ISS and especially the web-based systems are primarily using intelligent methods. IDS or IPS are machine learning oriented, and some of them are using knowledge discovery and data mining [1,2]. Such sophisticated technologies are time- and labor-consuming, and it is very hard to make them satisfy the standard demands for convergence of the results and/or comparatively low computational complexity. However designers and customers accept such difficulties trying to gain from higher reliability of such applications. The base concept of the presented paper is to make a powerful human-centered system combined with firewalls, IPS or other security tools. It could make a complex defense against different groups of intruders. Aiming at that, we should introduce different ontologies to support the IDS work or the system will be not enough reliable. In the next two sections we'll show the usage of ontologies in different decision support methods and applications in data mining, web mining and/or other computation discovery or evolutionary systems.

Usually ontologies are issued to support methods or applications to probabilistic, fuzzy inference or uncertainty processing [3-6]. Our research shows [7] other, nonstandard ways that are not excluding the other contemporary research but are making something in addition to well known methods, and so are useful to be combined with. The next Section is dedicated to a new self-learning method that constantly searches for knowledge conflicts or its ultimate case-contradictions-and tries to resolve them [8]. It is found to be the best way to self-improvement via the constant correction of knowledge incompleteness or inconsistency. On contrary to other machine learning methods, our proposal is ontology-driven and it is much less heuristic by nature than the other well known methods from the field. In this case the keyword self-learning is introduced to emphasize the above quoted differences.

In section 3 different human-centered methods are used to check the truth value of one or group of statements. Those statements are named below in the text: definition of the problem, target question or a *goal* for short, e.g. goal to detect a possible intrusion. We are not trying to elaborate completely automatic systems. Since the first knowledge discovery systems, it is seen that making more or less automatic inference system makes it over fulfilled of heuristics which restricts its future development. Instead we offer making the machine the best human's advisor. The machine founds some interesting patterns and represents it in an user-friendly manner. Some of similar ideas are used in cognitive graphics but our methods are absolutely different and we prefer to name the field: *human-machine creation*. It is shown that in many cases the act of creation isn't something very difficult, it may resemble a human-machine brainstorm method where the machine 'mechanical' part of work aims at catching some repeating/resembling patterns or show other relations or regularities to the user who may take his fuzzy part of the investigation.

[1] The paper is supported by NSF grant 'Dynamic Ontologies in Multiagent Information and Control Systems'– Bulgarian Ministry of Education and Science

Application results are considered in section 4. Never the above quoted research has been realized in 'all in one' system because of its high complexity. However we used a big variety of method combinations under the SMM, synthetic metamethod control. Those allow us make rather effective inference machines.

2. Ontology-Based Machine Learning

Let the strong (classical) negation is denoted by '$\neg$' and the weak (conditional, paraconsistent [9]) negation by '~'. In case of an evident conflict (inconsistency) between the knowledge and its ultimate form–the contradiction–the conflict situation is determined by the direct comparison of the two statements (the *conflicting sides*) that differ one form another just by a definite number of symbols '$\neg$' or '~'. For example, A and $\neg$A; B and not B ($\neg$ is equivalent to 'not'), etc. η is a negation type, in case strong classical negation, and square brackets embrace all possible words used to represent explicit strong negations in texts.

$$\{\eta\}\ [\text{no, not, не, нет}]. \tag{1}$$

The case of implicit (or hidden) negation between two statements A and B can be recognized only by an analysis of a present ontologies of type (2).

$$\{U\}\ [\eta\text{: A, B}]. \tag{2}$$

where U is a statement with a validity including the validities of the concepts A and B and it is possible that more than two conflicting sides may be present. Below it is accepted that the contents in the figure brackets U is called *an unifying feature*. In this way it is possible to formalize not only the features that separate the conflicting sides but also the unifying (or common) concepts. For example the intelligent detection may be either automated or of a man-machine type but the conflict cannot be recognized without the investigation of the following conflict ontology (3).

$$\{\text{detection procedures}\}\ [\neg\text{: automatic, interactive}]. \tag{3}$$

Ontologies (1) or (2) describe situations where conflict the sides mutually negate one another. In the majority of situations the sides participate in the conflict only under definite conditions: $\chi_1, \chi_2, \ldots\chi_z$.

$$\{U\}\ \ [\eta\text{: } A_1,A_2,\ldots A_p]\ \ <\tilde{\chi}_1 * \tilde{\chi}_2 * \ldots * \tilde{\chi}_z>. \tag{4}$$

where $\tilde{\chi}$ is a literal of χ, i.e. $\tilde{\chi}\equiv\chi$ or $\tilde{\chi}\equiv\eta\chi$, * is the logical operation of conjunction, disjunction or implication.

Let the ultimate form of conflict, contradiction is investigated. The syntactic contradiction ontology is depicted in fig. 1, and the semantic variant is considered in fig. 2. It is obvious that the contradictions are very different but their base ontologies seem quite similar. The reason is that the essential part of both conflicts or contradictions from (2) and (3) isn't an ordinary ontology knowledge itself but is a form of *metaknowledge* that controls the usage of ontologies or parts of them. The bottom level objects from fig. 1 unconditionally refute each other. We may find some cases where the same system have been automatic one, and after some time it became an interactive system, but this case is so labor consuming that actually we speak about a new, different system.

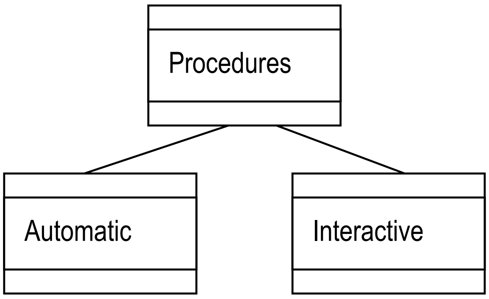


Figure 1. Ontology for a syntactic contradiction

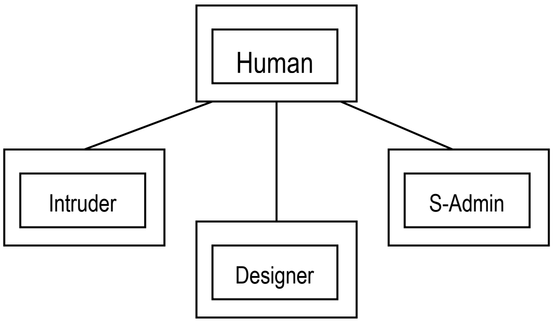


Figure 2. Ontology for a semantic contradiction

What is depicted in fig. 2 shows a different situation, concerned with 'IDS-humans' or three major groups of people dealing with IDS: intruders; security experts or designers (designer); security administrators (S-admin). Weak negation is used in case, in the bottom level objects, because the security administrator may be former expert or he may be a designer of another system, and also former hackers may be engaged as experts. The semantic contradiction will appear iff all the following conditions are satisfied: T (the same time) and I (the same system) and U (the same person) and P (the same place). Next figures 3 and 4 give more details for the case.

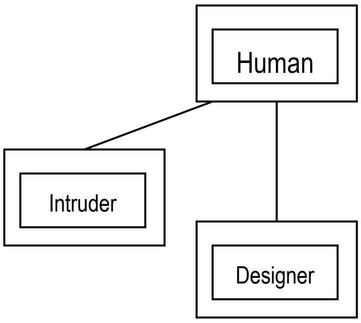


Figure 3. Ontology for conflict situations

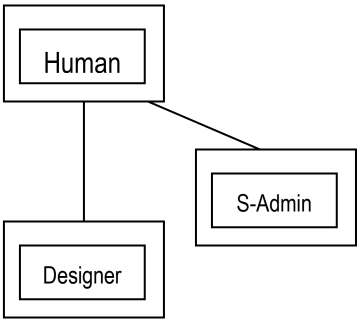


Figure 4. Ontology for contradiction situations

Fig. 3 concerns the part of ontology from fig. 2 when the security administrator is eliminated. Let all the quoted above conditions are satisfied: T (the same time) and I (the same system) and U (the same person) and P (the same place). Still we couldn’t define the situation in fig. 3 as a contradiction, say because the designer may test the IDS system. To resolve this situation we may use knowledge type exclusion and defeasible inference or other well known inference schemes. This is an example of knowledge conflict, not a contradiction, and only additional investigations may result in semantic contradiction.

Fig. 4 shows the semantic contradiction, and if the conditions $T \wedge I \wedge U \wedge P$ are satisfied, then the contradiction appears: ‘nobody can occupy both positions’. Thus fig. 2 contains different types of ontology knowledge inside it. The above given examples aim to show that processes based on knowledge conflicts or contradictions couldn’t be thoroughly described by an ontology knowledge, and using only static situations. We need the dynamic picture to decide if we have no conflict or contradiction. On the other side, when the situation dynamics is investigated, pretty often we turn on to ontology corrections due to its incompleteness or incorrectness. In this situation the main conclusion for us is the following. We need to use metaknowledge and dynamic ontologies to cope with conflict or contradiction identification. The conflict identification is almost always much more complicated than the contradiction case.

The ontology-based contradiction identification is followed by its resolution [8]. The proposed resolution methods are applications of ideas from nonclassic logics and they are one of base parts of the presented research in analogy inference machines, case-based methods, data mining, etc. Contradiction resolution depends on the situation and types of contradiction sides. Our research [8] revealed five main groups of resolution scenarios. Currently we make investigations to elaborate new contradiction resolution scenarios. The research shows that automatic contradiction resolution processes may stay active constantly using free computer resources. Also they may be directly activated by user. In the first case the knowledge and data bases will be constantly improved by continuous elimination of incorrect information or by improving the existing knowledge as a result of revealing and resolving contradictions.

Only two-sided contradictions are considered because most of multi-sided contradictions are represented as a set of two-sided contradictions. The automatic contradiction resolution process starts when one of the sides is very weak, say machine hypothesis while the other side is rather strong, say expert knowledge. The resolution finishes with an instant elimination of the weak side. This situation is used to filter machine hypotheses. If we use the notion ‘conflict’ instead of ‘contradiction’, then in the same situation analogically another filter is to be built up.

Another automatic resolution process is where no resolution is needed at all. Say, if the task is to know the speed of light, then we don’t need to resolve the contradiction ‘is the light wave or particle?’ This way is widespread in multi-agent systems. It is supposed the agents may return without collisions to previous positions before the conflict. Whenever possible, we use each possibility to escape from contradiction resolution process but each detection of conflicts/contradictions should be alerted.

A third group of methods for automatic resolution is the following. If {P} is a set of parameters in the considered model, and every $p \in P$ is strictly defined: $a \le p_i \le b$, and the considered value is outside the model range, say $p_i = b+10$, then there exists a contradiction with the model, and this contradiction is simply resolved by issuing a warning: 'p_i exceeds the range limit'. After that the model and/or some factors connected to the considered parameter should be corrected. Of course the above considered example could include ontologies and nonnumeric information, say 'we thought you are using statistical methods and they aren't'.

All the other ways use human-centered methods where an expert or even non-advanced user contribute to the resolution process. The machine prepares all the necessary information including all the ontologies involved with some additional features, say dynamics of their changes. If an inference to the sides of the contradiction exists then the inference tree is represented. All knowledge and data is to be represented using below considered CALEIDOSCOPE method. The purpose of this group of methods is to reveal hidden regularities to the user and to group all the information so that to ease his act of creation.

As a result the considered contradiction resolution methods have been upgraded to a machine learning method i.e. learning without teacher which is rather effective in case of ISS.

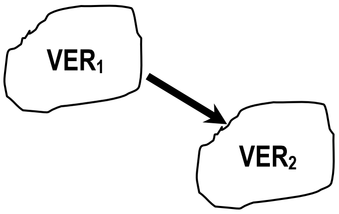


Figure 5. Standard way to changing ontologies.

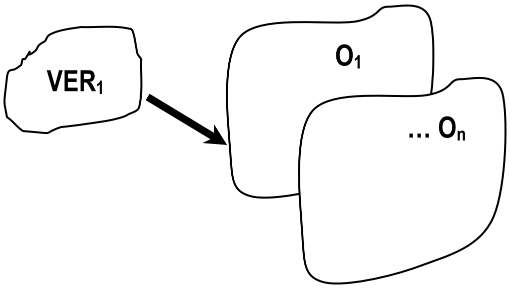


Figure 6. Changing and defeating ontologies.

Ontologies involved in the contradiction resolution process are divided in the following two groups. Denote $o^*_i \in O^c$ are ontologies from the model of syntactic/semantic contradiction, $objs(o^*_i)$ are objects from all the ontology levels from O^c and $o_i \in O$ are all ontologies concerning the sides of the contradiction. Let neg(a,b) denote the arguments of neg unconditionally refute each other or they refute each other while all the conditions are executed. If $neg(o^*_U , o^*_V)$, then the set O^c should be altered by addition and/or elimination of its elements. On the other hand, O^c contains ontology descriptions linked to the ontology from square brackets in the left hand side of (1) up to (4). Hence the contradiction resolution process leads to dynamic ontologies in the knowledge base (**KB**). If a negation is revealed between objects from different levels of one or different ontologies: $neg(objs(o^*_I) , objs(o^*_J))$, then it doesn't mean the contradiction exists but the situation should be saved and described to the domain experts. Even if no contradiction, it may shift the ontologies involved. If $neg(o_S , o_T)$, then the contradiction exists but the models of contradictions O^c are not the reason for its appearance. In this case other domain knowledge including ontologies $o_i \in O$ should be altered and/or appended which may lead to changes in existing ontologies. The existing software, say RDF, OWL or Protégé, contains standard set of features to shift/append ontologies while keeping its history [10,11]. The process of changing ontologies is depicted in fig. 5 where ontology version 1 (ver_1) is substituted by ver_2 but the transition from version 1 to version 2 could be checked at any time. On contrary to the traditional scheme, we offered in [12] a defeasible inference scheme with the following outcome. The original ontology (ver_1) will be substituted by one or a set of ontologies using exclusions while its (ver_1) is still valid in an a priory given possible world. Thus the revealed contradiction is resolved by the transfer of the inconsistent sides or derivatives to different possible worlds.

Example.

Let an ISS is applied into a sport environment and following groups of people may be involved in the training process of the security administrator.

{training of security administrators} [~: trainer, instructor, manager/policymaker, developer]
$< T \wedge I \wedge U \wedge P >$. (5)

If the environment is shifted from sports to government, then instead of trainer an expert will be used and the model will be as follows.

{training of security administrators} [~: expert, instructor, manager/policymaker, developer] < $T \wedge I \wedge U \wedge P$>. (6)

Thus an invariant (*strong*) part of the set is observed which isn't changed by shifting environments, and the difference is the following. In sport environment the meaning sports expert is defeated by trainer. Models (5) and (6) are correctly functioning in *different possible worlds* and there is no inconsistency because of shown differences between (5) and (6). Metaknowledge is used to address (5) to sports and (6) to government.

Designers are seldom involved in the administration training. When the sport environment is shifted by another conditions, say (6), it may be seen that the above written is the same everywhere. Thus the designer branch from the set of the involved ontologies from (5) or (6) is to be pruned, making the work with ontologies more effective. Hence the higher quality of ontologies is obtained. The same manner we may decrease some ontology levels using dynamic transfers like (5) to (6).

Generally, the ontologies dynamics from O^c is is based on contradictions resolution which leads to changes in the set or defeasible inference attachments to the set [12]. After that the quality of ontologies from O^c grows higher because they are becoming less incomplete or incorrect, and because ineffective branches from the set have been pruned if necessary.

To conclude this section, best ontologies used in contradiction resolution processes are dynamic ontologies. The resolution is an evolutionary process [13] and it brings dynamics to knowledge involved, leading to new forms for ontology processing. Two contemporary concepts may be shown how to make machine self-improvement leading to self-learning. The first one is based on the usage of artificial neural networks (ANN), or similar heuristic methods. The ANN methods show low learning rate and high design costs. On contrary, we offer machine self-improvement via contradiction or knowledge conflict resolution. KB is improving after every resolution process, and this gives dynamics to ontology descriptions. After the resolution, the *invariant* part of knowledge or method remains that makes it stronger and more flexible. This self-improvement needs only one time-consuming resource: juxtapositions between different groups of knowledge. It needs the human help only in some complex situations. The considered machine learning is an evolutionary process [13] and it gives better results if the intermediate solutions (*hypotheses*) are tested in different models [14]. The system has many resources to constantly resolve the contradictions when no goal is given or in parallel to main jobs. We can't escape from heuristics but they are passed to the decision maker via productive human-machine interactions mechanism thus making the system alone more effective and less complex. Some part of heuristics is hidden in ontologies driving the process of learning. Most of the presented computational discovery/data mining methods are data-driven. The considered research is more ontology-driven than data-driven but it belongs to the same group of methods. The below presented methods allow us to use not only statistical methods but also other knowledge acquisition methods for knowledge discovery.

This type of machine learning is novel and original in both theory and applied aspects.

3. Method Interactions under SMM Synthetic Metamethod Control

The described below methods interact under the common control of a new type of a synthetic metamethod (***SMM***). The considered metamethod avoids or *defeats* crossovers, phenotypes, mutations, or other elements from traditional evolutionary computation [13, 15]. The formal description below is appended with few explanations in an analogous manner as the way to reduce extra descriptions, because the general scheme of the chosen strategy is rather voluminous. *SSM* swallows and controls the following methods:

I. **INCONSISTENCY**: contradictions detection and resolution method;

II. **CROSSWORD** method;

III. **FUNNEL** method;

IV. **CALEIDOSCOPE** method.

A. CROSSWORD Method

Let somebody tries to solve a problem with a complex sentence of 200+ letters with vague for the reader explanations. Let the unknown sentence be horizontally located. The reader can't solve the problem in an arbitrary manner, because the number of combinations is increased exponentially. Now it is convenient to **facilitate** the solution by linking the well known to the reader information with the complex one from the same model. The reader tries to find vertical words that he is conscious about, say 'non-stream ciphers' (=block ciphers). The more crossings lead to the easier solution of the horizontal sentence. The approach for the CROSSWORD is *even easier*. Here both the easy meanings and difficult ones are from one domain, therefore an additional help exists to find the final solution.

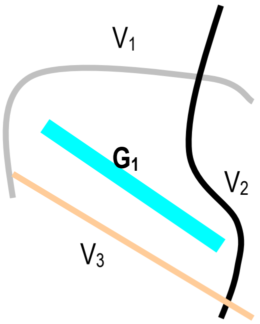


Figure 7. Example of nonlinear constraints

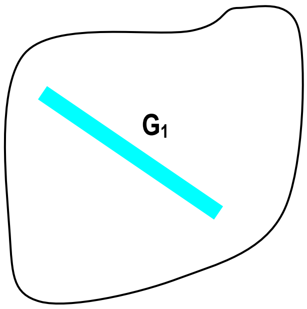


Figure 8. The goal is inside an ontology

The difference of the CROSSWORD method from the usual crosswords is in its highly dimensional spaces and of course the analogy is rather far and is used only for the sake of brevity. Let G be a goal that must be solved, and it is decomposed into two types of subgoals: G_2 is deduced in the classical deductive manner using Modus Ponens (MP), and G_1 is explored in the area defined by the constraints V_1-V_2-V_3 in fig. 7.

The constraints V_j are not necessarily linear. Nonlinear V_j are depicted in fig. 7. Let all the constraints are of different types. Denote V_1 is a curve dividing two groups: knowledge inconsistent with G_1 is located above V_1 and consistent knowledge is below the curve. Let V_1 divides the knowledge having accordance to G_1 from knowledge conflicting with the subgoal. In the end, let V_3 is a linear constraint e.g. x>1997. The solution to the subgoal lays inside the area depicted in fig. 7 and the goal resolution complexity falls significantly.

Another situation reducing the resolution process is depicted in fig. 8 where the same subgoal G_1 is located inside and ontology which gives the search constraints. Sometimes the proof leading to the situation in fig. 8 is the proof *on contrary* when it is impossible the goal to be outside the considered ontology. Comparisons between two examples from fig. 7 and fig. 8 show that using ontologies to reduce the research area is more natural way and is much more effective than standard constraint satisfaction methodology.

Let subgoal G_1 is indeterminate or it is defined in a fuzzy way. Then the introduced algorithm is defined in the following way.

$$\begin{gathered} K_i \in K,\ i=1,2,\ldots n:\ G_1 \cap K_i \neq \varnothing; \\ L_j \in L,\ j=1,2,\ldots m:\ G_1 \cap L_j = \varnothing; \\ S=(G_1 \cap K_1), T=(G_1 \cap K_n); S \neq T; x_1, y_1, z_1 \in S;\ x_2, y_2, z_2 \in T; \\ \frac{x-x_1}{x_2-x_1} = \frac{y-y_1}{y_2-y_1} = \frac{z-z_1}{z_2-z_1} \end{gathered} \tag{7}$$

where x_1, y_1, z_1 and x_2, y_2, z_2 are the coordinates of the respective boundary points from S and T from the set K whilst x, y and z are the coordinates of the points from the slice that tethers the explored area. In this way (by two sticking points) the goal search is restricted from an infinite space to a slice in the space. The introduced method is realized in an iterative manner: the goal place from (7) is replaced by K_i from the previous iteration and so on.

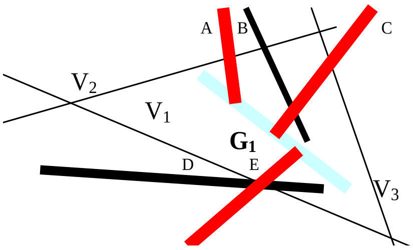


Figure 9. CROSSWORD method: constraints and binding to G

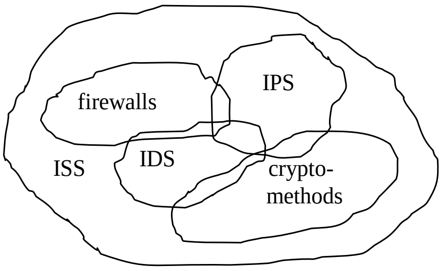


Figure 10. 'Fuzzy' intersections

Fig. 9 illustrates an example with three elements of K={A,B,C} where L={D,E} contains two elements. The example illustrates the benefit from the elements of L and from the spatial constraints V_u even in the case n>1. It is conspicuous that the direction of G_1 most often does not predetermine the integral decision and that the elements D,E and V_u decrease the number of the concurrent alternatives.

Different types of connections are depicted in fig. 9. The search restriction is done by V_1, V_2, and V_3 as considered above in fig. 7. The constraints B and D also restrict the search for G_1 but this restriction is dot-shape because B and D lay not in the search area bounded by V_j. On the other hand, those dots make tight fixation to $G_{1\,,}$ so the are denoted fixation constraints. In the end, A, C and E are resolution constraints because they intersect G_1 and give us parts of the solution to the problem.

The CROSSWORD purpose is to bind and unknown with the known knowledge. Ontologies have been used aiming at realization of different constraints (fig. 8 and fig. 9) and binding elements (fig. 9). Each ontology substitutes a complex set of (non)linear constraints, say in an ontology intrusion which varies in different situations. Thus complex logic and qualitative calculations have been substituted by a keyword/meta tag search. What is depicted in fig. 10 is an application result of methodology for defining inexact borders between few ontology examples. Indeed, if only a little part of IDS is given, then it may be mistaken with a part from IPS or other ISS. The fuzzy ontology border is limited by a knowledge area in the form 'this is exactly not that ontology'. If the system is passive then it is a firewall or similar but not IDS.

One of the intersections from fig. 10 is shown in details in fig. 11. Dynamic ontologies are well suited for operating with such imprecise information. If a contradiction appears then it should be resolved as shown in previous section.

The intersections in fig. 9 are parts of the considered goal from the ontology. The binding element, say B in fig. 9, is knowledge concerning the goal, in other words this knowledge enlarges the belief that the goal is true. In dynamic ontologies this type of knowledge is a target to be included in further ontology versions.

Example.

A security administrator of IPS or IDS receives an email offering perspective positions in the field. He replies and receives large files explaining job offers, and is invited to describe all his project activities. Soon after that the system alerts an increased number of false alarms.

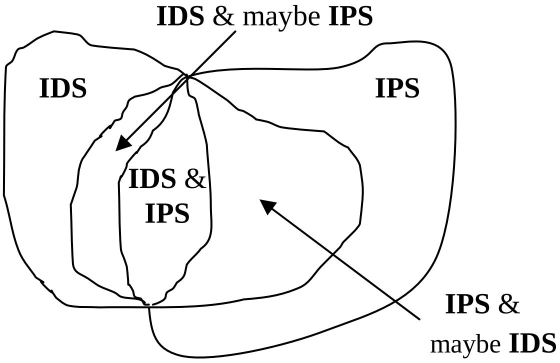


Figure 11. Different grades of intersection.

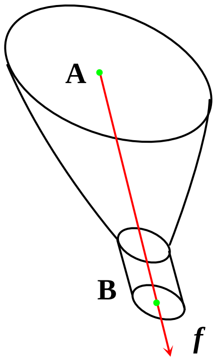


Figure 12. FUNNEL method.

The ontology used here is a hoax. Coincidences in receiving job offer and large files are analogical to binding elements B or D in fig. 9. They don't contain pieces of the goal but are its close neighbors. Large files in this case may purpose preparations to larger traffic, drone software, influencing the artificial neural network from ISS or say ISS shutdown because of false alarms appeared it time of critical work. Intruders are trying to force the administrator to lower the system sensitivity or to turn off the alarms. It is only one intersection with the goal in case: an increasing number of false alarms but because of the other information it is enough to issue an alert and investigate for possible intrusion.

Example

After a visit to some of non-trusted sites, security administrator activates few spyware programs but all of them lead to system shutdown. The administrator switches off the network cable, and observes during the next reset a trial to connect to an unknown .org site. Next few days this PC is working offline and waiting for new Windows installation. After that and before any changes, the spyware shows no threads, the computer functions normally and only the browser is turning on very slowly.

If no dynamic ontologies are used then no threat will be detected. The well investigated situation from the past is connected to present fuzzy situation in case Past knowledge will force further investigations, and new system installs.

B. FUNNEL Method

We denote with f(t$_0$) a fitness function in the point t$_0$. In the common case f(t$_0$) may vary according to its environment – the position in the space and other impacts over the point. In this paper the function is linear and it does not change in the whole domain, *f*=f(t$_0$). In this way f(t$_0$) is reduced to a free vector *f*. Let f(t$_0$) is one of the intermediate solutions to the goal when the process has reached up to t$_0$. f(t$_0$) points only to the *recommendable* direction for the evolution of the solution [13], so the movement in this direction shall be realized only if there are no other alternatives. Here we may use a 'gravity' analogy: it is too weak in case of e.g. jets but still it is enough strong not to be underestimated. f(t$_0$) is combined with a system of spatial constraints in the following way:

f(t$_0$) is the goal function;

f$_i$(t$_0$) is a set of functions which affect t$_0$.

$$A\frac{d^n x}{d^n t}+B\frac{d^n y}{d^n t}+C\frac{d^n z}{d^n t}\le D \qquad (8)$$

$$Ex+Fy+Gz\le H \qquad (9)$$

where (8) is a system of non-linear constraints and (9) is a system of linear constraints. Then the direction of the solution in f*(t$_0$) is defined as a sum of the vectors multiplied by the respective coefficients k$_i$; the existing system of constraints is presented by (8) and (9).

$$f^*(t_0)=f(t_0)+\sum_i k_i f_i(t_0) \qquad (10)$$

Let's assume you have a *plastic funnel*. If you fix it vertically above the ground, you can direct a stream of water or of vaporous drops etc. If you change the funnel direction, then the stream targeting will be hampered, if the stream hasn't enough *inertia power*. Fixing the funnel horizontally makes it practically useless. Analogically in the evolutionary method the general direction in numerical models is determined likewise. In other words this is a movement along the predefined gradient of the information. Just like in the case of the physical example, there are lots of undirected hazardous steps towards conclusions and hypotheses in the beginning.

This paper offers the following modification of FUNNEL. Let k$_i$ be not constants:

$$k_i(t_0)=\frac{k_i^0}{1+D_0-D} \qquad (11)$$

where k_i^0 are the initial meanings coinciding with k$_i$ from (10) and (t$_0$) are the respective coefficients in point t$_0$, D is the initial point in the investigated domain–a beginning of the solution and D$_0$ is an orthogonal projection of t$_0$ upon the straight line L parallel to *f* where D∈L. In this case moving away from the beginning D the solution depends more and more on the fitness function but the other external factors influence it less and less.

The FUNNEL method can be indirectly based on inconsistency tests with known information. The method may be used also in the other parts of *SMM*, e.g. in the CROSSWORD method it assists the determination of the direction of the explored goal. The graphical representation of the FUNNEL main idea is represented in fig. 12.

It is a data driven method, so intruders haven't possibility to predict the results. The direction *f* from the figure is the goal, e.g. the fitness function from genetic algorithms. Unlike the other contemporary methods, the FUNNEL method gives the ISS freedom to choose and update the hierarchy of goals. In 'the loose part' A in fig. 12, if a new goal appears and promises large gains, and if there is still a long way to resolve *f*, then ISS will try to reach the nearest goal, after that it will return to its way for *f*. The 'edge' constraints in FUNNEL are function of the following parameters: the 'stream inertia' of the intermediate solutions, 'gravity', etc. The next Sections show that INCONSISTENCY method also can be applied to define constraints in the FUNNEL method.

C. CALEIDOSCOPE Method

The CALEIDOSCOPE is the visualization method: it presents the current results or the solution to the security expert. Apart from other interfaces, here some cognitive elements have been applied that help the user make conclusions using notions still unknown to the machine: 'beauty', 'useful', etc. Here the system role is mainly to inspire the decision making imagination and to give him the interesting results: repetitive patterns, etc.

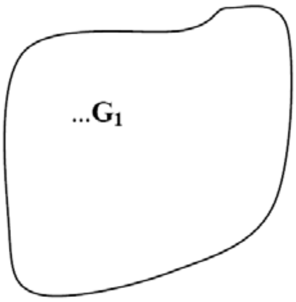


Figure 13.
CROSSWORD -1.
Location of the goal

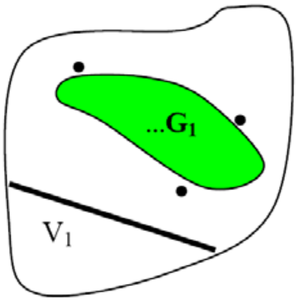


Figure 14.
CROSSWORD -2.
Two different constraint types

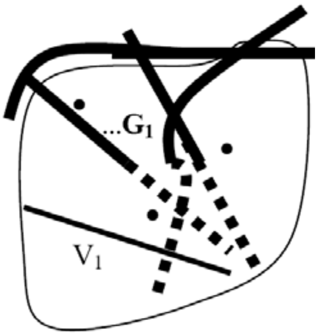


Figure 15.
CROSSWORD -3

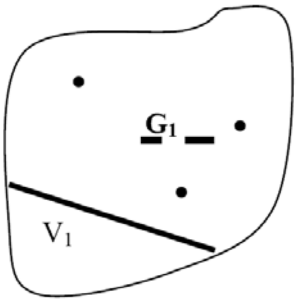


Figure 16.
CROSSWORD -4

Many of the above described methods contain enough visualization elements; in these cases the CALEIDOSCOPE method makes only graphic interpretations of results. In other cases it should make an optimal rotation of the pattern or show intersections of pattern or make other processing helping the user make the decision in best comfort conditions.

Fig 13 shows an example when the decision to the goal is located in the depicted ontology area, and all the other domain knowledge may be considered only if has some relation to the ontology. Let restriction constraint V_1 from fig. 14 is found e.g. 'show only new results', and three fixation constraints are found: the intersection of the curves with the ontology field is represented as three dots. Both two types of constraints make rough solutions thus helping to restrict the search area and make the method complexity better.

Fig. 15 depicts same constraints and three resolution constraints making intersections with the desired goal G_1. A part of other constraints helping define the three resolution constraints is depicted. It is shown in Fig. 16 that two right intersection parts are joined, and the left part is enlarged using knowledge modeling, binding and logic methods. Thus a big part of the goal is known and the security administrator will make correct conclusions.

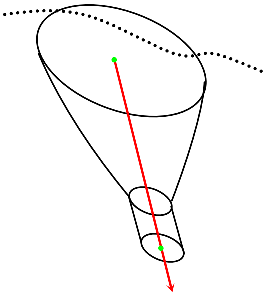


Figure 17. The fixed fitness function

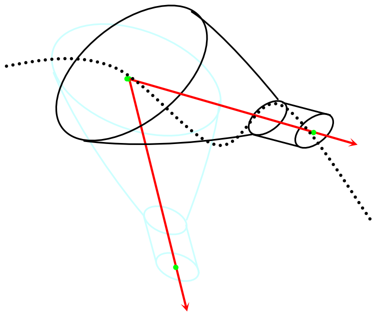


Figure 18. An adjustable FUNNEL method

The visualization of the FUNNEL method results is considered in Fig. 17. Let the solution to the problem, dotted line in fig. 17, has a ‘strong inertia force’ thus leaving the desired area. The interpretation shows that the fitness function in this example should be shifted as shown in fig. 18, and then the solutions will go the desired direction. We use only a set of static pictures but it is obvious that multimedia and visualization of dynamic processes will make greater effect. The hope is to realize it in ongoing projects.

D. Interactions

Briefly, the synthetic SMM control means that the overall result is defined pretty much ‘by the design’, by interactions between the methods than by the outcomes from each method itself.

Method interactions between INCONSISTENCY, CROSSWORD, FUNNEL, and CALEIDOSCOPE are discussed in this section. There exist much more methods under SMM control, say induction, juxtapositions etc. Not everything is described to the sake of clarity and brevity. In general all the methods are collaborative as shown above: FUNNEL-INCONSISTENCY-CALEIDOSCOPE in fig. 18 or CROSSWORD-CALEIDOSCOPE in fig. 15. Briefly, all of those interactions have been visually depicted in pictures above. On the other hand, all methods are competitive, and ‘the fittest survives’ principle means the following. As described, many processes should be executed in parallel but the computer resources are reserved for the high priority methods. The lowest priority belongs to the constantly active test on inconsistency which runs if any free resources. The highest priority belongs to user modeling, intruder modeling and expert- or user-ordered goals. Methods that brought a lot of successful results in the past gather higher priority. The security administrator may shift the set of priorities at any time.

Well known query processing, statistical inference and other knowledge discovery technologies will easily collaborate with the presented methods but they are included under the same SMM control. As stated above, our goal isn’t a method substituting the best contemporary methods but making a good addition to them. The wide part of the funnel in fig. 18 shows that the resolution process may start using statistics in the lack of knowledge and then go to the desired direction when statistical methods are shifted by other knowledge acquisition methods. This important part of SMM is described in [13].

4. Realizations

The presented system source codes are written in different languages: C++, VB, and Prolog. It is convenient to use the applications in freeware like RDF, OWL, Ontoclean or Protégé. Many of the described procedures rely on the usage of different models/ ontologies in addition to the domain knowledge thus the latter are metaknowledge forms. In knowledge-poor environment the human-machine interactions have a great role, and the metaknowledge helps make the dialog more effective and less boring to the human. The dialog forms are divided in 5 categories from 1=’informative’ to 5=’silent’ system. Knowledge and metaknowledge fusion is always documented: where the knowledge comes from, etc. This is the main presented principle: every part of knowledge is useful and if the system is well organized, it will help us resolve some difficult situations.

We rely on nonsymmetrical reply ‘surprise and win’, on the usage of unknown codes in combination with well known methods, and on the high speed of automatic reply in some simple cases e.g. to halt the network connection when the attack is detected. If any part of ISS is infected or changed aiming at reverse engineering or other goals, then the system will automatically erase itself and in some evident cracking cases a harmful reply will follow. The above represented models of users and environment are used in the case. Therefore different SMM realizations are not named IDS but ISS because they include some limited automatic reply to illegal activities.

The success of the presented applications is hidden in a rather simple realization of the presented methods. We tried to make complex applications using reasoning by analogy, machine learning or statistical data mining methods but in this case the complexity of SMM is greater than NP-hard.

5. Conclusions and Future Work

The main conclusion is that all ontologies make ISS more flexible, especially dynamic ones. In the beginning almost all ontologies are poor modeled, incomplete or even partially incorrect. To improve they should evolve, and we found that conflicts or contradictions are best driving factors of such an evolution. Hence the resolution of contradictions is one of driving factors to the usage of dynamic ontologies. The detection or resolution process uses knowledge or metaknowledge concerning evolving ontologies. Different forms of resolutions have been considered. Say, defeasible inference using exclusions allows the system to transfer the sides of conflict or contradictions to different possible worlds.

Cases have been considered where no other solution exists but only using dynamic ontologies. It is derived that many processes concerning human-machine creation are ontology-based. Our additional purpose is to show that when the machine helps to resolve the problem using its strongest features then it uses the formal, mechanical part of the research, while the heuristic part, emotions or notions like simple, beautiful, interesting should be left to the decision maker. Such human-centered methods are much more effective and less complex than automatic heuristic methods. One of the considered methods, CALEIDOSCOPE may be considered a far analogy to human-computer brainstorming methods.

To make an advanced system, we should define and use many labor-consuming models and/or ontologies. In perspective we hope that the usage of machine learning or other knowledge acquisition methods will help to construct ontologies automatically. In parallel we use the considered methods in information security projects [16].

Bibliography

[1] M. Miller. Absolute PC Security and Privacy. SYBEX Inc., CA, 2002.

[2] D. Song, M. Heywood, A. Zincir-Heywood. Training Genetic Programming on Half a Million Patterns: An Example From Anomaly Detection, IEEE Trans./Evolutionary Computation, no. 3, pp. 225-239, 2005.

[3] H. Kyburg*, Probability and Inductive Logic*, Progress, Moscow, 1978.

[4] The Handbook of Data Mining, N. Ye (Ed.), Lawrence Erlbaum Associates, NJ, 2003.

[5] S. Denchev and D. Hristozov. Uncertainty, Complexity and Information: Analysis and Development in Fuzzy Information Environment. Zahari Stoyanov, Sofia, 2004.

[6] G. Klir and B. Yuan. *Fuzzy Sets and Fuzzy Logic: Theory and Applications*. Upper Saddle River, Prentice Hall, NJ, 1997.

[7] V. Jotsov. "Knowledge discovery and data mining in number theory: some models and proofs," Proc. Methods and Algorithms for Distributed Information Systems Design. Institute for Information Transmission Problems of RAS, Moscow, pp.197-218, 1997.

[8] V. Zgurev and V. Jotsov, "An approach for resolving contradictions," *J. Controlling Systems and Machines* Vol. 7-8, pp. 48-59, 1992.

[9] A. Arruda, "A survey on paraconsistent logic," *in Math. Logic in Latin America*, A. Arruda, C. Chiaqui, N. Da Costa, (Eds.), North-Holland, Berlin NY, pp. 1-41, 1982.

[10] Essential RDF features: www.w3schools.com/rdf

[11] Introduction to OWL: www.w3.org/TR/owl-ref/

[12] Jotsov V., Semantic Conflict Resolution Using Ontologies, Proc. 2nd Intl. Conference on System Analysis and Information Technologies, SAIT 2007, RAS, Obninsk, September 11-14, 2007, vol. 1, pp. 83-88.

[13] V. Jotsov. "Evolutionary parallels," *Proc. First Int. IEEE Symp. 'Intelligent Systems'*, T. Samad and V. Sgurev (Eds.), Varna, Bulgaria, vol. 1, pp. 194-201, 2002.

[14] V. Jotsov. "Knowledge acquisition during the integer models investigation," *Proc. XXXV Int.Conf. "Communication, Electronic and Computer Systems"*, Tecnical University of Sofia, pp. 125-130, 2000.

[15] A. Goel, "Design, analogy and creativity," *IEEE Expert/Intelligent Systems and Their Applications*, vol. 12, no. 3, May 1997.

[16] V. Jotsov, V. Sgurev. "An investigation on software defence methods against an illegal copying," *Proc. IV Int. Sci. Conf. 'Internet - an environment for new technologies'*, vol. 7, V. Tarnovo University 'St. St. Kiril and Metodius', pp. 11-16, 2001.

Author's Information

Vladimir S. Jotsov (В.С. Йоцов): e-mail i@AAAaaa.biz

Intsitute of Information Technologies of the Bulgarian Academy of Sciences;

State Institute of Library Studies and Information Technologies; P.O.Box 161, Sofia 1113, BULGARIA;

DOMAINS WITH COMPLICATED STRUCTURES AND THEIR ONTOLOGIES[1]

Irene Artemieva

Abstract: *The article defines the class of domains with complicated structures, gives the definition of multilevel ontologies and determines the method for developing such ontologies.*

Keywords: *Domains with complicated structures, multilevel ontologies*

ACM Classification Keywords: *I.2.4 Knowledge Representation Formalisms and Methods, F4.1. Mathematical Logic*

Introduction

At present there are the following ontology application categories: Knowledge management systems, Controlled vocabulary, Web site or document organization and navigation support, Browsing support, Search support (semantic search), Generalization or specialization of search, Sense "disambiguation" support, Consistency checking (use of restrictions), Auto-completion, Interoperability support (information/process integration), Support validation and verification testing, Configuration support, Support for structured, comparative, and customized search [Denny, 2002], [Gavrilova, 2006], [Zagorulko et al, 2006], [Ontos Miner], [http://www.alphaworks.ibm.com/contentnr/semanticsfaqs]. The goal of research into ontologies is to create explicit, formal catalogues of knowledge that can be used by intelligent systems (see http://www.aaai.org/AITopics/html/ontol.html). The following definitions of an ontology are used:

- An ontology defines the terms used to describe and represent an area of knowledge. Ontologies are used by people, databases, and applications that need to share domain information (a domain is merely a specific subject area or area of knowledge, such as petroleum, medicine, tool manufacturing, real estate, automobile repair, financial management, etc.). Ontologies include computer-usable definitions of basic concepts in the domain and the relationships among them. They encode knowledge in a domain and also knowledge that spans domains. In this way, they make that knowledge reusable. (see http://www.w3.org/TR/webont-req/).

- An ontology comprises a formal explicit description of concepts (often called classes) in a domain of discourse, properties (sometimes called slots) of each concept describing various features and attributes of the concept, and restrictions on properties (sometimes called facets). An ontology together with a set of individual instances of classes constitutes a knowledge base. In reality, there is a fine line between where the ontology ends and the knowledge base begins, and a fine line between a class and an instance. Classes are the focus of most ontologies. Classes describe concepts in the domain. (see http://www.alphaworks.ibm.com/contentnr/semanticsfaqs).

At present there are various ontology-design methodologies [Cristani & Cuel, 2005], [Denny, 2002], [Gavrilova, 2006], [Jones, et al], [Corcho et al, 2003] for the above applications. Many methodologies include defining classes in the ontology, arranging the classes in a taxonomic (subclass–superclass) hierarchy, defining slots and describing set of values for these slots, filling in the values for slots for instances [Noy, 2001]. Different ontologies (see, for example, http://musing.deri.at/ontologies/v0.3/ and http://www.daml.org/ontologies/) designed to be used in program systems of the above classes were developed using these methodologies.

However the specified tasks are only a subset of applied tasks for the automated solutions of which intelligent systems are required, and a number of domains have characteristics that are not taken into consideration by the existing ontology-design methodologies. The aim of this article is describing the class of such domains, structure of their ontologies, and method of developing their ontologies.

[1] This paper was made according to the program № 14 of fundamental scientific research of the Presidium of the Russian Academy of Sciences, the project “Intellectual systems based on multilevel models of domains”.

Domain class definition

Domains with complicated structures have the following characteristics:

- they have sections that are described in different but resembling notion systems [Kleshchev & Artemjeva, 2005a];
- the sections have subsections that are described in different but resembling notion systems;
- any subsection can have subsections with the above characteristic.

Sections (and subsections) of domains with complicated structures are also domains with their own kinds of activity and sets of applied tasks; some applied tasks from different sections may be similar. Notions of ontologies and knowledge of different sections can be used to solve applied tasks in domains with complicated structures.

Medicine is an example of a domain with complicated structure [Kleshchev et al, 2005a]. The examples of sections of this domain are therapeutics, surgery and others. The ontology terms of each section are names of diseases studied in this section and names of signs the values of which are used to diagnose diseases. Each section has its own sets of names of diseases and signs.

Chemistry is another example of a domain with complicated structure. The examples of its sections are physical chemistry, organic chemistry and analytical chemistry. Physical chemistry deals with the physicochemical processes [Artemieva, Tsvetnikov, 2002]. These processes are described in terms of characteristics of matters and reactions that take part in the processes. Organic chemistry adds terms relating to structural properties of matters [Artemieva et al, 2005, 2006]. Analytical chemistry studies processes of influence on matters with various kinds of radiation [Artemieva, Miroshnichenko, 2005]. Chemical thermodynamics and chemical kinetics are examples of sections of physical chemistry; sections of analytical chemistry depend on analysis techniques (for example, X-ray fluorescence analysis).Another example of a domain with complicated structure is the domain of program transformations. The processes of changing programs as a result of applying different transformations are studied in this domain [Artemieva et al., 2002]. The examples of the sections are transformations of structural programs and transformations of parallel programs. The transformations are described in terms of properties of languages of these programs.

Examples of tasks solved in domains with complicated structures are diagnosing, designing, planning, etc. One needs terms required for defining characteristics of a patient (object of diagnosis) which are input data of the task, intermediate data or the result of the task and not for defining characteristics of a disease (that are stored in knowledge bases) to specify a medical diagnosis task. Terms that describe characteristics of substances used in experiments, reactions, chemical process conditions [Artemieva, Reshtanenko, 2006] and not known characteristics of substances and reactions that are traditionally stored in chemistry databases are necessary for specifying a task of planning a chemical experiment.

Some problems arise when intelligent systems for domains with complicated structures are designed. The first problem is integration of knowledge from various sections and subsections within the framework of one knowledge base. A means of such integration is ontology that must take into account that notion systems (ontologies of sections and subsections) used in different sections and subsections differ. The second problem is a way of integration of notion systems (ontologies). A means of such integration can be ontology of higher level of generality.

Defining level of generality of ontologies

Ontologies are used to verbally represent information. Verbal representation of information is a mapping of a finite set of terms into a set of possible values of terms. Verbal representation of information has level 0.

Ontology with the system of knowledge that specifies a particular set of verbal representations of information has level 1. At level 1 ontology terms has no values. Setting values to ontology terms make the verbal representation of the particular information have level 0.

Ontology without knowledge system has level 2. When different knowledge is added, different specifications of verbal representations of information are received.

Ontology in terms of which ontology of level i can be specified has level i + 1. All ontologies of levels more than 2 are metaontologies.

In every hierarchy the language of specification of ontologies has level that is 1 more than the highest level in the hierarchy (and in this hierarchy there are no ontologies of higher levels).

Let us explain the difference between level 1 and level 2. Ontology defines a set of terms used to verbally represent information, a set of possible values of each term, and relations between terms (ontological constraints). Ontology is the result of agreement among people that use the same information in their professional activity; therefore ontology is obviously the result of agreement among these people on what verbal representations in the domain have meaning. We will call a set of all verbal representations of information that have meaning conceptualization. Knowledge imposes additional constraints on a set of verbal representations and picks a subset out of conceptualization. Thus, level 2 specifies conceptualization; level 1 specifies its subset, where knowledge defines characteristics of this subset.

Ontology of the next level specifies a larger set of verbal representations of information as compared with ontology of the previous level. Transition from level i to level i-1 restricts this set and defines its subset. Transition from ontology of level i-1 to ontology of level i is considering ontology of level i-1 as verbalized information.

Let us consider domains as examples of verbalized information. Knowledge is represented verbally if it is specified as an array of pairs consisting of a term and its value. Terms included into the ontology of knowledge is used to verbally represent knowledge [Kleshchev & Artemjeva, 2006]. If verbalized information is a base of knowledge of a domain, then its representation has level 0. If this is a case, level 1 is ontology of knowledge consisting of definitions of terms with their sets of values and knowledge consistency constraints as well. Level 2 specifies sets of ontologies of knowledge.

Let us consider the example when information is the description of a state of affairs of the domain [Kleshchev & Artemjeva, 2006]. In respect of physical chemistry level 0 represents the information of a certain physicochemical process that took place at a certain period of time and under certain external conditions. To describe the process one may use the terms with the following values: “process steps”, “chemical substances at each process step”, “chemical reactions at each process step”, etc. The terms used for describing level 0 for the domain form the ontology of reality. Level 1 specifies the reality model of the given domain and describes all possible chemical processes the information about which can be represented in terms of the ontology; the representation of the information about each chemical process is not inconsistent with the ontological constraints and domain knowledge. The domain knowledge describes laws for going of chemical reactions, formation laws of substance from chemical elements, etc. Level 2 specifies the conceptualization of reality that is an idea about reality that the domain specialist has. This level defines concepts for this reality description.

Regarding X-ray fluorescence analysis, a section of analytical chemistry, level 0 represents the information about a certain physical process that took place at a certain period of time and under certain external conditions. To describe the process one may use the terms with the following values: “analytes”, “sample qualitative composition”, “percentage of an analyte in a sample” [Artemieva, Miroshnichenko, 2005]. The domain knowledge describes laws of physical processes during high-frequency electromagnetic radiation directed at a sample, values of characteristic radiation of analytes, etc.

The domain knowledge defines characteristics if its reality as sets of states of affairs that can take place in it. If the domain knowledge can be verbally represented, then the ontology of level 2 contains sets of terms for their representation. This ontology is a pair of two ontologies (reality and knowledge) and relations between them (additional ontological constraints). If the domain knowledge cannot be verbally represented (e.g. physics), then the ontology of level 2 coincides with the ontology of reality. In this case, the ontology of level 3 specifies a set of ontologies of reality.

There are domains where only part of knowledge is verbally represented. In this case, the ontology of knowledge contains terms that can represent this knowledge. This is characteristic of physical chemistry: knowledge about various properties of chemical elements (atomic weight, atomic number, etc.), physicochemical characteristics of substances (density, formula, etc.), reaction properties (e.g. catalyst) and so on is verbally represented in it. Laws of physical processes cannot be represented verbally.

Properties of multilevel ontologies for domains with complicated structures and method of their development

For the domain with complicated structure the level with the maximum number n is ontology of the domain. The level contains the terms with the help of which the ontology of the next level is determined. The transition to the next level means specifying ontology terms and ontological constraints of the next level [Artemieva, 2006]. If all the domain knowledge and ontological constraints of all the levels can be represented verbally, then the ontology of level n defines properties of all sets of terms of all ontologies of lower levels. The simplified ontology of medical diagnosis has this property [Kleshchev & Artemieva, 2005b].

The ontology of level n-1 consists of modules. Each module defines the ontology of a certain section of the domain. The ontology of level n-2 also consists of modules. Each module defines the ontology of a subsection. All the ontologies of level lower than n are modular. The domain knowledge base is also modular.

Let us describe the method for developing the multilevel ontology of the domain.

The development starts with defining verbalized information about the domain reality and terms for its verbal representation. The domain expert participate in this work. The knowledge engineer and the expert make a list of terms used for representing the reality, record meanings of terms and values, principles of representing states of affairs with their help. A set of all possible meanings is defined for each term (denotation of the term). Ontological constraints specifying constraints for a set of meanings of terms are formed (domain state of affairs consistency constraints).

A set of applied tasks of the professional activity is analyzed in order to define what information about the reality is to be verbally represented. Terms used for specifying input data and their results and terms used for representing values of intermediate data are defined. Ontological constraints specifying relations between all these terms are also defined. A set of tasks of the professional activity unambiguously defines the domain. Thus, the ontology of the reality contains terms that are used in applied tasks to specify input data and results of solutions, intermediate data, and relations between terms of the three groups. Then, the knowledge system, probably defining the reality of the domain more accurately, is to be designed.

Developing the ontology of level 2 starts with answering the question whether the domain knowledge can be represented verbally. If the answer is in the negative (i.e. the knowledge cannot be represented verbally), the knowledge system is developed in the same form as the system of ontological constraints (in terms of the ontology of the reality). If the answer is in the affirmative (i.e. the knowledge can be represented verbally), a list of terms for representing the domain knowledge is made with the help of an expert, definitions of these terms are developed, knowledge consistency constraints and interrelations between the reality and the knowledge are formulated. This list of terms for representing the domain knowledge and the set of knowledge consistency constraints form the ontology of the knowledge of this domain. If only a part of the knowledge can be represented verbally, a list of terms for representing only this part is made. The system of knowledge consists of two components: a set of assertions in terms of the ontology of the reality and mapping of a set of terms for representing knowledge into a set of values. The ontology of the reality, the domain knowledge ontology, and interrelations between the reality and the knowledge form the ontology of level 2 for the domain.

The ontology of level 3 (and all following levels) can be developed if the ontologies of level 2 (and all the previous levels) of several sections of the domain are developed since this ontology specifies a set of ontologies of level 2. This step starts with answering whether the ontologies of level 2 can be represented verbally. If the answer is in the negative, developing stops. If the answer is in the affirmative, a list of terms for its representing is made, definitions of these terms are developed, consistency constraints for the ontology of level 2 are formulated. The task of this level and the following ones is searching for “regularities” in the ontology of the previous level, grouping terms with some similar properties into one set, formulating the term properties of these sets and relations between them.

Let us consider a fragment of the ontology of level 4 for chemistry represented by an applied logic language to exemplify the usage of a top-level ontology in the domain [Kleshchev & Artemjeva, 2005b].

1. sort Types of objects: {}N \ ∅

   Term "Types of objects" means non-empty set of names of object types of the domain.

2. (Type: Types of objects) sort Type: {}(R ∪ I ∪ N ∪ L)

Each type of objects is a set of objects; each object can be named, represented with a number, can be logical value.

3. sort Types of object components: Types of objects → {} Types of objects

   Term "Types of object components" means the function that maps object type on non-empty set of names of object types. If Types of object components(t)=t' and t'={t'_1,t'_2} then objects from the set t'_1 or the set t'_2 that can be components of objects with type t.

4. Set of objects ≡ {(Type: Types of objects) j(Type)}

   This auxiliary term means a set of objects of all types.

5. Own properties of objects ≡ (λ(Type: Types of objects) (λ(Area of possible values: {}(Value sets ∪ {}Value corteges)) (j(Type) → Area of possible values))

   Term "Own properties of objects" means the function the argument of which is object type and the result of which is a set of functions. The argument of each function is a set of values or a set of corteges; the result is a set of functions. If Own properties of objects(t) = f1 and f1(m)=f2 then an argument of the function f2 is an object with type t and the result is an element of the set m.

6. Properties of components of given types ≡ (λ(Type1: Types of objects) (Type2: Types of object components (Type1)) (λ(Area of possible values: {}(Value sets ∪ {}Value corteges)) (Object that has type 1 → j(Type1), Object that has type 2→ Object components(Type1, Type2)(Object that has type 1)) → Area of possible values))

   Term "Properties of components of given types" means the function the arguments of which are two object types - t1 and t2, and the result is a set of functions the argument of each one is m set of values or corteges of values, and the result is a set of functions the arguments of each one is object of t1 type and object of t2 type which is a component of object of type t1, and the result is m set element.

7. sort Number of process steps: I[0,∞)

   Term "Number of process steps" means a number of steps included in a physicochemical process.

8. sort Types of process objects: {} Types of objects \ ∅

   Term "Types of process objects" means a set of object types that are considered as components of a physicochemical process.

9. Process components ≡ (λ(Type: Type of process objects) (I[1, Number of process steps] → {} {(v: Set of objects) Object type(v) = Type} \ ∅)

   Term "Process components" means a function the argument of which is t object type, and the result is a set of functions the argument of each one is the number of a process step, and the result is a set of components of a process – non-empty subset of objects of type t.

10. Properties of process components ≡ (λ(Type: Types of process objects) (λ(Area of possible values: {}(Value sets ∪ {}Value corteges)) (Step number→ I[1, Number of process steps], Process component → Process components(Type)(Step number)) → Area of possible values))

    Term "Properties of process components" means the function the argument of which is t object type, and the result is the function the argument of which is m set of values or corteges of values, and the result is the function the arguments of which are the number of a process step and a component of this step (object of type t), and the result is m set element.

Let us now consider the example of using the ontology of level 4 when defining the ontology of level 3 for X-ray fluorescence analysis [Artemieva, Miroshnichenko, 2005]. First, let us define the values of the parameters of ontology of level 4 (a set of terms of ontology of level 3).

1. Types of objects ≡ {Shells of chemical element atoms, Radiation transition of orbital electrons, Chemical elements}

   The ontology defines the objects of the given types. This set specifies types of objects that are studied in the section of the domain.

2. Types of object components ≡ (λ(Type: {Shells of chemical element atoms, Radiation transition of orbital electrons, Chemical elements}) (Type = Chemical elements ⇒ Radiation transition of orbital electrons), (Type ≠ Chemical elements ⇒ ∅}

   Energy levels and radiation transition of orbital electrons are defined for chemical elements; energy levels are defined for shells. Objects of other types do not have components.

3. Types of process objects ≡ {Chemical elements, Radiant energies}

   Types of chemical process objects are chemical elements and radiant energies.

Now let us define examples of ontological constraints that are part of ontology of level 3.

1. Shells of chemical element atoms ⊂ {}N \ ∅

   Shells of chemical element atoms is name of set. This set consists of designation for shells.

2. Chemical elements ⊂ {}N \ ∅

   Chemical elements is name of set. This set consists of designation of elements.

3. Radiation transition of orbital electrons ⊂ {}N \ ∅

   Radiation transition of orbital electrons is name of set. This set consists of designation of transitions.

Then let us define examples of terms that are part of ontology of level 3 that mean names of functions.

1. Own properties of shells ≡ Own properties of objects(Shells of chemical element atoms)

   Term "Own properties of shells" means the function the argument of which is a set of values or set of corteges of values m, and the result is the function the argument of which is shell, and the result is an element of m set.

2. Own properties of radiation transitions ≡ Own properties of objects (Radiation transition of orbital electrons)

   Term "Own properties of radiation transitions" means the function the argument of which is a set of values or set of corteges of values m, and the result is the function the argument of which is radiation transition, and the result is an element of m set.

3. Properties of radiation transition of orbital electrons of elements ≡ Properties of components of the given types (Chemical elements, Radiation transition of orbital electrons).

   Term "Properties of radiation transition of orbital electrons of elements" means the function the argument of which is a set of values or set of corteges of values m, and the result is the function the arguments of which are chemical element and its radiation transition, and the result is an element of m set.

4. Properties of elements of a sample ≡ Properties of process components (Chemical elements)

   Term "Properties of elements of a sample" means the function the argument of which is a set of values or set of corteges of values m, and the result is the function the arguments of which are the number of a process step and chemical element of this step, and the result is an element of m set.

Finally let us define examples of terms that are part of ontology of level 2.

1. sort Binding energy of electrons on an energy level for an element: Properties of energy levels for an element (R(0, ∞))

   Binding energy of electrons on an energy level for an element is a function the first argument of which belongs to the set with name Chemical elements, and the second argument of which belongs to the set with name Energy level. The result of the function belongs to the set of real number.

2. sort Characteristic radiation frequency: Properties of radiation transition of orbital electrons of elements (R(0, ∞))

   Characteristic radiation frequency is a function the first argument of which belongs to the set with name Chemical elements, and the second argument of which belongs to the set with name Radiation transition of orbital electrons. The result of the function belongs to the set of real number.

3. sort Wave-length of characteristic radiation: Properties of radiation transition of orbital electrons of elements (R(0, ∞))

   Wave-length of characteristic radiation is a function the first argument of which belongs to the set with name Chemical elements, and the second argument of which belongs to the set with name Radiation transition of orbital electrons. The result of the function belongs to the set of real number.

4. sort Energy of characteristic radiation: Properties of radiation transition of orbital electrons of elements (R(0, ∞))

   Energy of characteristic radiation is a function the first argument of which belongs to the set with name Chemical elements, and the second argument of which belongs to the set with name Radiation transition of orbital electrons. The result of the function belongs to the set of real number.

Conclusion

Top-level (or upper-level) ontologies are described in many papers. Such ontologies define terms used for highly abstract notions that studied by philosophy. They are aimed at defining all meanings of these terms.

However in domains of professional activity considered in this paper it is assumed that meanings of domain ontology terms are fixed. Terms of ontology of higher level of generality defined in the article specify properties of sets of these terms and have fixed meanings themselves.

The method of development of multilevel ontologies was used to create multilevel ontology of chemistry [Artemieva et al, 2005, 2006], [Artemieva, Miroshnichenko, 2005], [Artemieva, Tsvetnikov, 2002], and multilevel ontology of domain “Optimization of sequential programs” [Artemieva et al, 2002].

Properties of an intelligent system based on multilevel ontology were described in the article [Artemieva, Reshtanenko, 2006].

Bibliography

[Artemieva, 2006] Artemieva I.L. Multilevel mathematical models of domains. In Artificial Intelligence, Ukraina, 2006, vol.4: 85-94. – ISSN 1561-5359.

[Artemieva et al, 2002] Artemieva I.L., Knyazeva M.A., Kupnevich O.A. A model of a domain ontology for “Optimization of sequential computer programs”. Terms for optimization process description. In 3 parts. In Scientific & Technical Information. Part 1: 2002. №12: 23-28. Part 2: 2003. №1: 22-29. Part 3: 2003, № 2: 27-34.

[Artemieva et al, 2005] Artemieva I.L., Vysotsky V.I., Reshtanenko N.V. A model for the ontology of organic chemistry. In Scientific & Technical Information, 2005, № 8: 19-27.

[Artemieva et al, 2006] Artemieva I.L., Vysotsky V.I., Reshtanenko N.V. Description of structural formula of organic compounds in the model for the ontology of organic chemistry. In Scientific & Technical Information, 2006, №2: 11-19.

[Artemieva, Miroshnichenko, 2005] Artemieva I.L., Miroshnichenko N.L. A model for the ontology of X-ray fluorescence analysis. In Informatic & Management Systems. 2005. № 2: 78-88. – ISSN 1814-2400.

[Artemieva, Reshtanenko, 2006] Artemieva I.L., Reshtanenko N.V. Specialized computer knowledge bank for organic chemistry and its development based on the ontology. In Artificial Intelligence, Ukraina, 2006, vol.4: 95-106. – ISSN 1561-5359.

[Artemieva, Tsvetnikov, 2002] Artemieva I.L., Tsvetnikov V.A. A fragment of the ontology of physical chemistry and its model // Investigated in Russia [Electronic resource]: multysubject scientific journal / Moscow Institute of Physics and Technology - Dolgoprudny: MIPT. 2002. № 5. P.454-474. - http://zhurnal.ape.relarn.ru/articles/2002/042.pdf

[Corcho et al, 2003] Corcho O., Fernandez-Lopez M., Gomez-Perez A. Methodologies, tools and languages for building ontologies. Where is their meeting point? In Data & Knowledge Engineering, 2003. № 46: 41–64.

[Cristani & Cuel, 2005] A Survey on Ontology Creation Methodologies. In Int. J. on Semantic Web & Information Systems, 2005, 1(2): 48-68.

[Denny, 2002] Denny M. Ontology Building: a Survey of Editing Tools. URL: http://www.xml.com/pub/a/2002/11/06/ontologies.html

[Jones, et al] Jones D., Bench-Capon T. and Visser P. Methodologies for Ontology Development. URL: http://www.iet.com/Projects/RKF/SME/methodologies-for-ontology-development.pdf

[Gavrilova, 2006] Gavrilova T.A. Development of applied ontologies. URL: http://raai.org/resurs/papers/kii-2006

[Kleshchev et al, 2005] Kleshchev A.S., Moskalenko F.M., Chernyakhovskaya M.Yu. A model for the ontology of medical diagnosis. In 2 parts. In Scientific & Technical Information. Part 1: 2005. №.12: 1-7. Part 2: 2006. № 2: 19-30.

[Kleshchev, Artemieva, 2005a] Kleshchev A.S., Artemieva I.L. An analysis of some relations among domain ontologies. In Int. Journal on Inf. Theories and Appl., 2005, vol 12, № 1: 85-93. – ISSN 1310-0513.

[Kleshchev, Artemieva, 2005b] Kleshchev A.S., Artemieva I.L. A mathematical apparatus for ontology simulation. In Int. Journal on Inf. Theories and Appl., 2005, vol 12, №№ 3-4 – ISSN 1310-0513.

[Kleshchev, Artemieva, 2006] Kleshchev A.S., Artemieva I.L. Domain ontologies and their mathematical models. In the Proceedings of the XII-th International Conference "Knowledge-Dialog-Solution" - KDS 2006, June 20-25, Varna, Bulgaria, Sofia: FOI-COMMERGE-2006: 107-115. – ISBN 954-16-0038-7.

[Noy, 2001] Noy N., McGuinness D. Ontology Development 101 : A Guide to Creating Your First Ontology.- Knowledge Systems Lab, Stanfo. URL: http://protege.stanford.edu/publications/ontology_development/ontology101-noy-mcguinness.html

[Ontos Miner] Ontos Miner: Data Mining System from Text Documents in Russian. URL: http://www.avicomp.ru/rus/ontos/academic_solutions_rus.php.

[Zagorulko et al., 2006] Zagorulko Yu. A., Borovikova O.I., Kononenko I.S., Sidorova E.A. Approach to developing domain ontology for knowledge portal of computational linguistics. In Computational linguistics and intellectual technologies: Papers of International Conference “Dialogue 2006” (Bekasovo, 31 May – 4 June, 2006). – Moscow: RSUH Publishing Centre, 2006: 148-151.

Authors’ Information

Irene L. Artemieva – *artemeva@iacp.dvo.ru*

Institute for Automation & Control Processes, Far Eastern Branch of the Russian Academy of Sciences; 5 Radio Street, Vladivostok, Russia

ONTOLOGY VIEW ON AUTOMATA THEORY

Sergey Krivoi, Lyudmila Matveyeva, Yelena Lukianova, Olga Sedleckaya

***Abstract**: The summary of automata theory ontology is presented in the paper. It is based on the following dependences: a type of an automaton – the language accepted by the automaton – applications. The given ontology does not claim to be exhaustive as automata theory is very extensive and it is a complicated problem to survey all its aspects within one article. Only the main properties of automata and their applications are considered.*

1. Introduction

Correct design and verification of hardware and software systems are very important current problems. Design problem is difficult to formalize and consequently difficult to automatize; verification problem is hard as it accumulates a variety of different tasks engaged from adjacent scientific domains. One of such adjacent scientific domains is automata theory which is applied for partial solution of design and verification problems. Automata theory plays a crucial role particularly for reactive systems properties verification since such important problems as properties recognition, a definite state or a set of states reachability, and accepted language emptiness are decidable for most types of automata.

The paper presents brief automata theory ontology based on the following dependences: ***an automaton – the language accepted by this automaton – applications***. Only the main properties of automata and their applications are considered here due to limited size of the given paper.

2. Finite Automata on Finite Words

All the main problems in theory of finite automata working on finite words are already decided, so from this point of view the theory is complete. The main results received at present in this domain have an applied nature, i.e. methods of automata theory are used now for specified applied domains tasks solving. However, finite automata theory has exerted influence on further development of general automata theory. It is showed up via numerous variations of the finite automaton notion: finite automata over infinite words [3, 12], timed automata [2], hybrid automata [9], automata over trees [10], etc.

The notion of a finite automaton on finite words (finite sets of symbols) in finite alphabet is a basic notion on which ontology is built up. Three types of such automaton are in common use: *Mealy automata, Moore automata,* and *X–automata (or automata without outputs).* Let us define these types.

Mealy Automata. Let X = {x, x,…, x } and Y = {y, y,…, y } be finite alphabets, i.e. finite sets of pair-wise different elements, which are named as symbols or signals.

Definition 1. Mealy automaton is the 5-tuple *(A, X, Y, f, g)* such that *A* is a finite set of automaton states, *X* is a finite set called the input alphabet, *Y* is a finite set called the output alphabet, $f : A \times X \to A$ is a transition function, and $g: A \times X \to Y$ is an output function.

Usually, an automaton is denoted by a symbol of a set of its states, i.e. *A = (A, X, Y, f, g).* If *f(a,x) = a'*, then it is said, that automaton *A* passes on to the state *a'* under the action of input signal $x \in X$ or the signal *x* transfers the automaton A from the state *a* to the state *a'*. If *g(a,x) = y*, then it is said, that the automata A transforms input signal $x \in X$ into output signal $y \in Y$ being in the state a.

An automaton operation is described in the following way. A signal (symbol) $x \in X$ is given at the input of the automaton, then the automaton state is changed as a result in accordance with the current automaton state and its transition and output functions; and an output signal (symbol) $y \in Y$ appeared at its output. A word (a sequence of output symbols) appears at an automaton output if a word (a sequence of input symbols) is given at its input too. One may consider in that case the automaton A as an alphabetical information transformer which maps semigroup *F(X)* words into the semigroup *F(Y)* words. One may consider the sequence of input signals as a function of the natural argument – discrete automaton time. This circumstance allows us to consider an automaton as a discrete dynamic system which changes its states in time under the action of internal and external factors.

Automaton *A* is named *initial*, if a certain state a_0 is chosen (which is named *initial (start) state*) in the automaton set of states. It is considered that initial automaton is at initital state a_0 at initial point of time (before giving some word $p \in F(X)$ at its input).

Automaton *A = (A, X, Y, f, g)* is named *finite* if all three sets *A, X,* and *Y* are finite and — *infinite* if even one of them is infinite.

An automaton is called *complete* or *completely specified* if its transition function and output function are completely specified and — *partial* if even one of these functions is partial.

An automaton, which component *f* is not a function but is a certain relation (i.e. the condition of a transition univocacy is not carried out in that sort of automata), is called *nondeterministic*. If the relation *f* is the function then the automaton is called *deterministic*. Therefore, the state *b* is unambiguously found for deterministic automaton *A* with start state a_0, when the automaton passes on in state *b* under the action of the word $p \in F(X)$. But there may be several states of this kind in a nondeterministic automaton. It is clear that the class of Mealy nondeterministic automata is the subclass of the nondeterministic automata class.

Moore Automata. Moore automaton presents the special case of Mealy automaton.

Definition 2. The automaton *(A, X, Y, f, g)* is named *Moore automaton* if its output function *g(a,x)* can be expressed by means of a transition function *f(a,x)* via the eguation *g(a,x) = h(f(a,x)),* where $h: A \to Y.$ The function *h* is named as an *automaton marking function* and its value *h(a)* on the state *a* is named as the *mark* of this state.

Despite the fact that Moore automaton is the special case of Mealy automaton, Moore automata are studied separately in automata theory as far as in a number of cases their specific properties provide the opportunity to build more substantial theory then Moor automata one.

X-automata. *X-automaton* is 4-tuple *(A, X, f, F)*, but if *X*-automaton is initial, then it is 5-tuple $(A, X, f, a_0, F),$ where $f: A \times X \to A$ is the transition finction, $F \subseteq A$ is a certain subset of the automaton states which elements are called final states and $a_0 \in A$ is a start state of the automaton.

3. Brief Finite Automata Ontology

Brief finite automata ontology is introduces in figure 1 which is presented in the form of a graph. Arcs of the graph link together different scientific domains which are sided with one or another automata theory or its applications.

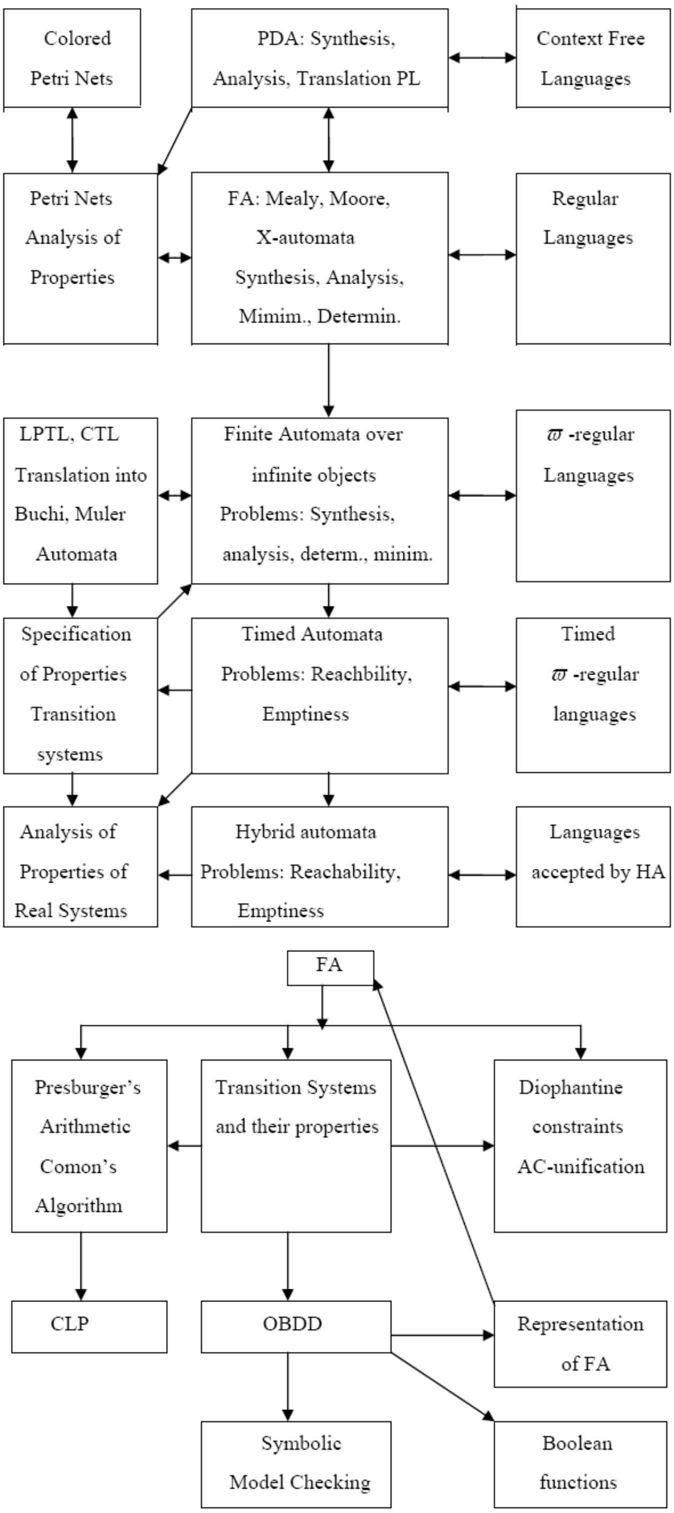

Colored Petri Nets
PDA: Synthesis, Analysis, Translation PL
Context Free Languages
Petri Nets Analysis of Properties
FA: Mealy, Moore, X-automata Synthesis, Analysis, Mimim., Determin.
Regular Languages
LPTL, CTL Translation into Buchi, Muler Automata
Finite Automata over infinite objects Problems: Synthesis, analysis, determ., minim.
ϖ -regular Languages
Specification of Properties Transition systems
Timed Automata Problems: Reachbility, Emptiness
Timed ϖ -regular languages
Analysis of Properties of Real Systems
Hybrid automata Problems: Reachability, Emptiness
Languages accepted by HA
FA
Presburger's Arithmetic Comon's Algorithm
Transition Systems and their properties
Diophantine constraints AC-unification
CLP
OBDD
Representation of FA
Symbolic Model Checking
Boolean functions

Figure 1

4. Finite Automata on Infinite Words

Let X be a finite alphabet. ϖ -language is an arbitrary subset X^{ϖ} of all infinite words in the alphabet X. A ϖ -automaton is finite object for the acceptance of ϖ -languages over a certain alphabet X. Several types of a ϖ -automata exist. Büchi automata and Muller automata are the main of them.

4.1. Büchi Automata and Muller Automata

Definition 3. A Büchi automata is 5-tuple *(A, X, f, A_0, F)*, where X is a finite input alphabet, A is a set of the automaton states, $A_0 \subseteq A$ is a set of start states, $f \subseteq A \times X \times A$ is a set of arcs (a transition relation), and $F \subseteq A$ a set of final states. The tuple *(A, X, f, A_0)* is called a *transition table*. So, a Büchi automaton is the extension of a finite state automaton to infinite inputs. It accepts an infinite input sequence, iff there exists a run of the automaton (in case of a deterministic automaton, there is exactly one possible run) which has infinitely many states in the set of final states.
The automaton starts in one of the start states and if $(s, x, s') \in f$, then the automaton is able to change its state from s to s' by means of reading input symbol $x \in X$. One can say that $r = s_0 x_1 s_1 x_2 s_2 x_3 \ldots$ is the trace of the automathon A over the word $\sigma = x_1 x_2 x_3 \ldots$ in the alphabet X, which joins $s_0 \in A_0$ and $(s_{i-1}, x_i, s_i) \in f$ for all $i \geq 1$. The set *inf(r)* for the trace r consists of the states $s \in A$ such that $s = s_i$ infinitely often, $i \geq 0$. The trace r of the automaton A on the word $\sigma \in X^{\varpi}$ is called the *accepting trace* iff inf$(r) \cap F \neq \varnothing$. In other words, the trace r is accepting trace iff a certain state from the set F is repeated infinitely ofter in the trace r. The language *L(A)* is called the language accepted by automaton A if it consists of the words σ such that the automaton A has the trace r which accepts σ.
ϖ -language is named *regular ϖ -language* if it is accepted by a certain Büchi automata.
Let $L(A) = \{p \in X^{\varpi} \mid A \text{ accepts } p\}$ be the ϖ -language which is accepted by a certain automaton A. If $L = L(A)$ for a certan Büchi automaton, then one can say that the language L is Büchi accepted. Muller atomata are the generalization of Büchi automata relatively the set of final states.
Definition 4. A Muller atomata over an alphabet X is 5-tuple $A = (A, X, f, a_0, F)$, where X is the finite alphabet, A is a finite set of states, $a_0 \in A$ is a start state, $f \subseteq A \times X \times A$ is transition relation, $F \subseteq B(A)$ is a set of subsets of the set of final states. It is said, that an automaton trace $\sigma = s_0 x_1 s_1 x_2 s_2 x_3 \ldots$is *effective*, if certain of the trace states appear infinite of times and serve as states from the set F. Automaton A accepts the word $p \in X^{\varpi}$, if the trace which corresponds to the word p is effective. ϖ -language $L \subseteq X^{\varpi}$ is named as accepted by Muller automaton, if it consists of all ϖ -words which this automaton accepts.

4.2. Properties of Regular ϖ –languages

All the results for ϖ –languages and ϖ -automata are presented in table 1. They mean that the operations are closed with respect to given classes of automata.

Table 1

Class of ϖ language	Operations
MA = BA = DMA $\cup$ DBA	union, intersection, complement union, intersection

5. Timed Automata

Let B be a set of clock variables which are nonnegative rational quantities D+. The set of timing constraints *C(B)* is defined as the following:

a) all the elements, which belong to *C(B),* are inequalities and look like $y \prec c$ and $c \prec y$, where $\prec$ means < or $\leq$, and c is nonnegative rational quantity (number);

b) if ϕ and ψ belong to *C(B)*, then $\phi \wedge \psi \in C(B)$.

Let us notice, that if *B* includes *k* clocks, then every timing constraint defines some convex set of *k*-dimensional Euclidean space. Therefore, if two points satisfy timing constraint, then all the points which belong to the segment connecting these points also satisfy this timing constraint.

Definition 5. Timed automaton (TA) *A* is 6-tuple (X, A, A_0, B, I, T), where *X* is a finite alphabet TA, *A* is a finite set of states, $A_0 \subseteq A$ is a set of start states, *B* is a set of clocks, $I : A \to C(B)$ is the mapping of a set of states into timing constraints, which is named *states invariants*; $T \subseteq S \times X \times C(B) \times 2^B \times A$ gives the set of transitions.

Every 5-tuple (a, x, ϕ, λ, a') corresponds to the transition from the state *a* into the state *a'* marked with the symbol x $\in$X. The constraint ϕ defines the moment of time when this transition becomes possible and clock values from the set $\lambda \in B$ are nulled when this transition is occurred. That is, a transition may be taken only if the current values of the clocks satisfy the associated constraints.

5.1. The Timed Automaton Model

The transition graph $T(B) = (X, V, V_0, R)$ with infinite nodes number serves as the model of TA *A.*

Every node from *V* corresponds to the pair *(a, v)* which consists of the state $a \in A$ ans clock value $v: B \to D+$ (*D*+ — nonnegative rational quantities). The set of start nodes is the following: $V_0 = \{(a,v) : a \in A_0 \wedge \forall y \in B[v(y) = 0]\}$.

It is necessary to introduce some notation for the definition of transitions in *T(B).* Let us define $v[\lambda = 0]$ as the value of clocks for $\lambda \subseteq B$, at that the values coincide with *v* for clocks from $B \setminus \lambda$ and all the clocks from λ have the value 0.

Let us define *v*+*d* for $d \in D+$ as the clock values *v(y)*+*d* which are taken for all clocks from $y \in B$, and the clock values *v*-*d* are defined similarly.

It follows from the definitions above that there are two main types of TA transitions:

- *transition by delay d* corresponds to defined time in case when the automaton is in some state *a*. At that it is written: *(a, v){d} (a, v + d)*, where $d \in D+$, if for all $0 \leq e \leq d$ the invariant *I(a)* is true for *v* + *e*;
- *transition by action* corresponds to running transition from the set *T*. In that case it is written: *(a, v){x}(a', v')*, where $x \in X$, in case if such transition $(a, x, \phi, \lambda, a')$ exists that *v* satisfies ϕ and $v' = v[\lambda = 0]$.

One can receive transition relation *R* for *T(B)* via joining a set of transitions by delay and a set of transitions by action. The notation *(a,v)R(a', v')* or *(a, v){x}(a', v')* means: such elements *a"* and *v"* exist, that *(a, v){d} (a", v"){x} (a', v')*, for some $d \in D+$ and $x \in X$.

5.2. Timed Languages

Let $X = \{x_1,...,x_n\}$ be an alphabet and *R* is the set of rational quantities. A *time sequence* $\tau = \tau_1 \tau_2 ...$ is a infinite sequence of time values $\tau \in R$ with $\tau_i > 0$, satisfying the following constraints:

(1) Monotonicity: τ increases strictly monotonically; that is, $\tau_i > \tau_{i+1}$ for all $i \geq 1$;

(2) Progress: For every $t \in R$, there is some $i \geq 1$ such that $\tau_i > t$.

A *timed word* over an alphabet *X* is a pair (σ, τ), where $\sigma = \sigma_1 \sigma_2 ...$ is a infinite word over *X* and τ is a time sequence. A *timed language* over *X* is a set of timed words over *X*.

A timed word (σ, τ) is named *input word* of an automaton; if this timed word is viewed as an input to an automaton, it presents the symbol σ_i at time τ_i.

Timed languages language-theoretic operations such as union, intersection, and complementation are defined the same way as for regular languages. In addition we define the *Untime* operation (for timed language *L* over *X*) which discards the time values associated with the symbols, that is, it considers the projection of a time trace (σ, τ) on the first component. That is, *Untime(L)* is ϖ-language which include all the words σ such that $(\sigma, \tau) \in L$ for some time sequence τ. Main results for languages and language operations are presented in table 2.

Table 2

Timed languages class	Operations
TMA = TBA	union, intersection
$\cup$	
DTMA	union, intersection, complement
$\cup$	
DTBA	union, intersection

6. Hybrid Automata

Hybrid automata generalize timed automata. A hybrid automaton (HA) is a dynamical system wih both discrete and continuous components. For example, an automobile engine whose fuel injection (continuous) is regulated by a microprocessor (discrete) is a hybrid system. HA provide a mathematical model for different real systems like digital computer systems that interact with analog environment in real time. Particularly, distributed processes with drifting clocks, real-time schedulers, and protocols for the control of manufacturing plants, vehicles, and robots would be modeled by means of HA.

Two problems, that are central to the analysis of HA theory are:

- the reachability problem and
- ϖ -language emptiness problem (more general problem).

Let us present here basic notions and main results of the HA theory. Let $D_{\geq 0}$ means the set of nonnegative real numbers: $D_{\geq 0}$ = {x $\in$ D | x $\geq$ 0}.

Rectangular regions. Given a positive integer *n > 0, n* $\in$ *N*, a subset of D^n is called a *region*. A bounded and closed region is *compact*. The region $R \subseteq D^n$ is called *rectangular* if it is Cartesian product of *n* intervals (possibly unbounded), all of whose finite endpoints are rational. R_i means the projection of *R* on *i*-th coordinate, so that $R = R_1 \times R_2 \times \ldots \times R_n$. The set of all rectangular regions in D^n is denoted R^n **.**

Definition 6. *n-dimensional rectangular automaton* (RA) *A* is 9-tuple *A = (G, X, init, inv, flow, pre, post, jump, obs)*, where *G = (V, E)* – finite directed graph, *X* is finite *observation alphabet*, three vertex labeling functions *init:* $V \rightarrow R^n$ *, inv:* $V \rightarrow R^n$, and *flow:* $V \rightarrow R^n$, and four edge labeling functions *pre:* $E \rightarrow R^n$, *post:* $V \rightarrow R^n$, *jump:* $E \rightarrow 2^{\{1,2,\ldots,n\}}$, and *obs:* $E \rightarrow X$.

RA may have so-called ε *-moves* (empty transitions). *n-dimensional RA with* ε *moves* differs from RA presented above in that the function *obs :* $E \rightarrow X^{\varepsilon}$, where $X^{\varepsilon} = X \cup \{\varepsilon\}$ augments the observation alphabet with the null observation $\varepsilon \notin X$.

The function *init* defines the set of initial states of RA. When the discrete state begins at vertex *v* $\in$ *V*, the continuous state must begin in the initial region *init(v)*.

The functions *pre, post,* and *jump* constraint the behavior of the automaton state during edge steps. The edge *e = (v, w)* may be traversed only if the discrete state resides at vertex *v* and the continuous state lies in the region *pre(v)*. For every *i* in the jump set *jump(e)*, the i-th coordinate of the continuous state is nondeterministically assigned a new value in the interval $post(e)_i$. For each *i* $\notin$ *jump (e),* the *i*-th coordinate of the continuous state is not changed and must lie in $post\ (e)_i$.

The *observation* function *obs* identifies every edge traversal with an observation from *X* or X^{ε} .

The *invariant* function *inv* and *flow* function *flow* constrain the behavior of the automaton state during time steps. While the discrete state resides at vertex *v*, the continuous state nondeterministically follows a smooth (C^{∞}) trajectory within the invariant region *inv(v)*, whose first time derivative remains within the flow region *flow(v)*. RA with ε -moves may traverse ε -edges during time steps.

If we replace rectangular regions with arbitrary linear regions in the definition ofRA, we obtain the *linear hybrid automata* (LHA). Thus RA are the subclass of LHA in which all defining regions are rectangular.

Initialization and bounded nondeterminism. RA *A* is *initialized* if for every edge *e=(v, w)* of *A*, and every coordinate $i \in [1,2,\ldots,n]$ with $flow(v)_i \neq flow(w)_i$, we have $i \in jump(e)$.

It follows that whenever the i-th continuous coordinate of an initialized automaton changes its dymanics, as given by the *flow* function, then its value is nondeterministically reinitialized according to the *post* function.

RA *A* has *bounded nondeterminism* if

(1) for every vertex $v \in V$, the regions *init(v)* and *flow(v)* are bounded, and

(2) for every edge $e \in E$, and every coordinate $i \in [1,\ldots,n]$ with $i \in jump(e)$ the interval $post(e)_i$ is bounded.

Note that bounded nondeterminism does not imply finite branching. It ensures that the edge and time successors of a bounded region are bounded.

7. Automata Applications for Systems Verification

Finite-state automata are well used for modeling of concurrent and interacting reactive systems. This modeling imposes that either the set of an automaton states, or the symbols of its input alphabet represent the states of modeled system. Main advantage at that automata use for systems verification is the following: both the system model and its specification are presented equally. In that case, Kripke model is directly correlated with ϖ - automaton (Büchi or Muller automaton), which all states are final, and the set of possible system behaviors is set by ϖ -language *L(A)*, which is accepted by corresponding automaton *A*. At that, the algorithm exists which translates a temporal logic formula into ϖ -automaton.

Let AP be a set of atomic propositions, then Kripke model (S, R, S_0, f), where $f : S \rightarrow 2^{AP}$, one may translate into the automaton $A = (A \cup a_0, X, Q, a_0, A \cup a_0)$, which input alphabet is the powerset of the set of atomic propositions $X = 2^{AP}$. At that for every pare of states $a, a' \in A$ relation *Q* includes the triple *(a,x,a')* iff $(a,a') \in R$ and *x = f(a')*, in which connection $(a_0,x,a) \in Q$ iff $a_0 \in S_0$ and *x = f(a)*, where *R* – consequence relation in Kripke model.

8. Examples of Some Properties and Their Specification

Let us consider some hypothetical reactive system. The properties of such systems are divided into two classes:

- ***safety properties***, usually state that something bad never happens**;**
- ***liveness properties*,** which state that something good eventually happens.

Safety properties are usually expressed by means of the following temporal logic formula $\Box \neg p$, where formula *p* expresses an unwanted event (a state) in a system.

The example of safety property is *mutual exclusion* property: $\Box (\neg p \vee \neg q)$, where formulas *p* and *q* defines the states at which a system can never be at the same time. The example of liveness property is *fairness* property; it states that a system must some time answer received inquiry, so fairness is concerned with guaranteeing that processes get a chance to proceed.

Let us extend this list of properties by some additional properties which belong to the class of liveness properties and are the following:

- guarantee states, that some event occurs at least once, but its repetition is not guaranteed. This property is expressed by LPTL-formula as ▱ *p*;

- obligation states, that formula *p* must always be executed or formula *q* is executed in the same state as formula *p*. This property is expressed by LPTL-formula as $\Box p \vee$ ▱ *q*. **obligation** property can be obtained by disjunciton of **safety** and **guarantee** properties;

- response states, that the event defined by formula *p* occurs infinitely often. This property is expressed by LPTL-formula as $\Box$ ▱ *p*;

- **persistence** states, that the event defined by formula *p* occurs continuously from a certain point on. This property is expressed by LPTL-formula as ▱ □ *p*;

- **reactivity** can be obtained by disjunciton of **persistence** and **response** properties**.** This property is expressed by LPTL-formula as ▱ □ *p* ∨ □ ▱ *p*;

- **unconditional fairness** states, that eligible process eventually run or the event, which is defined by formula *q*, occurs infinitely often irrespective of property *p*. This property is expressed by LPTL-formula as □ ▱ *p*;

- **weak fairness** states, if an activity is continually enabled (no temporary disabling) than it has to be executed infinitely often; or in other words, when formula *p* is true all the time, then formula *q* must be true infinitely often. This property is expressed by LPTL-formula as □ *p* → □ ▱ *q*;

- **strong fairness** states, if an activity is infinitely often enabled (not necessarily always) then it has to be executed infinitely often; or in other words, when formula *p* is true infinitely often, then formula *q* must be true infinitely often also. This property is expressed by LPTL-formula as □ ▱ *p*→ □ ▱ *q*.

Bibliography

1. Ахо А., Хопкрофт Дж., Ульман Дж. Построение и анализ вычислительных алгоритмов. -М. :Мир.-1979. - 535 с.
2. Alur R., Dill D.L. A theory of timed automata. - Theoretical Computer Science. -1994. -126. - PP. 183-235.
3. Thomas W. Automata on infinite objects. Handbook on theoretical computer science. - 1990. - PP. 135-191.
4. Годлевский А.Б., Кривой С. Л. Трансформационный синтез эффективных алгоритмов с учетом дополнительных спецификаций. Кибернетика, - 1986. - N 6. - С.34 - 43.
5. Глушков В.М. Абстрактная теория автоматов. - Успехи математических наук. -1961. - 16. - вып. 5. - С. 3-62.
6. Глушков В.М. Синтез цифровых автоматов. - М: Физматгиз. - 1962. - 476 с.
7. Глушков В.М., Цейтлин Г.Е., Ющенко Е.Л. Алгебра, языки, программирование. -Киев: Наукова думка. -1985. 327с.
8. Perrin D. Finite automata. In Handbook of Theoretical Computer Science. vol. 2, -Elsevier. -1990. -PP. 1-58.
9. Henzinger T.A., Kopke P. W, Puri A., Varaiya P. What's Decidable About Hybrid Automata? In the Proceed. of the 27-th Annual ACM Symposium on Theory of Computing (STOC 1995). - 1995. - PP. 373-382.
10. Comon H. Constraint solving on terms: Automata techniques (Preliminary lecture notes). - Intern. Summer School on Constraints in Computational Logics: Gif-sur-Yvette, France, September 5-8. -1999. - 22 p.
11. Капитонова Ю.В., Кривой С. Л., Летичевский А. А., Луцкий Г.М. Лекции по дискретной математике. БХВ: Санкт-Петербург, 2004, 624 с.
12. Трахтенброт Б. А., Барздинь Я. М. Конечные автоматы (Поведение и синтез). -М.: Наука. -1970. - 400 с.
13. Чень Ч., Ли Р. Математическая логика и автоматическое доказательство теорем. -М.: Мир. -1973. - 256 с.
14. Arnold A. Finite Transition Systems: Semantics of Communicating Systems. -Paris: Prentiuce Hall. -1994. - 177 p.
15. Ben-Ari M. Mathematical Logic for Computer Science. Springer Verlag London Limited. - 2001.-305 p.
16. Emerson E.A. Temporal and modal logics. Handbook of Theoretical Computer Science: Elsevier. - vol. B. -1990. - PP.995-1072.
17. Peterson G.L. Myths about the mutual exclusion problem. - Information Processing Letters, - 1981. - v.12.-N 3. - P.115-116
18. Wolper P. Temporal logic can be more expressive. - Information and Control. -v. 99. -1983. -P. 56 - 72.
19. Clarke E.M., Grumberg Jr. O., Peled D. Model Checking. - The MIT Press: Cambridge, Massachusetts, London, England. -2001. -356 p

Authors Information

S.L. Krivoy, L.Ye. Matveyeva *– Institute of Cybernetics, NAS of Ukraine, Kiev, Ukraine, e-mail: krivoi@i.com.ua; luda@iss.org.ua*

E. A. Lukianova *– Crimean Vernadski's University, Crimea, Simferopol, Ukraine*

O. Sedleckaja *– Institute of Theoretical and Applied Informatics, Technical University of Czestochova, Czestochova, Poland*

INFORMATION-ANALYTICAL SYSTEM FOR DESIGN OF NEW INORGANIC COMPOUNDS

Nadezhda Kiselyova, Andrey Stolyarenko, Vladimir Ryazanov, Vadim Podbel’skii

***Abstract**: The principles of design of information-analytical system (IAS) intended for design of new inorganic compounds are considered. IAS includes the integrated system of databases on properties of inorganic substances and materials, the system of the programs of pattern recognition, the knowledge base and managing program. IAS allows a prediction of inorganic compounds not yet synthesized and estimation of their some properties.*

***Keywords**: information-analytical system, knowledge discovery in databases, design of new inorganic compounds, pattern recognition, computer learning, knowledge base, databases on properties of inorganic substances and materials.*

***ACM Classification Keywords**: J.6 Computer-aided Design, J.2 Computer Applications & Chemistry, H.2.8 Scientific Databases, I.2.6 Analogies, I.2.6 Learning, I.2.4 Knowledge Representation Formalisms and Methods, C.2.5 Internet.*

Introduction

The problem of prediction of formation of new compounds and calculation of their properties is one of the most important tasks of inorganic chemistry. Any successful attempt of design of compounds not yet synthesized is of the large theoretical and practical importance. The problem of design of new inorganic compounds can be formulated as follows: it is necessary to find a combination of chemical elements and their ratio (that is, qualitative and quantitative composition) for making (under the given conditions) the predefined space molecular or crystal structure of compound allowing a realization of necessary functional properties. Only properties of chemical elements and data about other already investigated compounds should be used as initial information for calculations. Thus, the problem is concerned with a search for regularities between properties of chemical systems (for example, properties of compounds) and properties of elements, which form these systems.

The decision of a task of design of new inorganic compounds presents severe difficulties. The main difficulty is an extreme complexity of dependences relating property of inorganic compounds with properties of chemical elements. The traditional way of the decision of this task is associated with quantum-mechanical methods, which are based on the Schrodinger’s equation. However in most cases the accurate solution (in analytical functions) of the latter for certain inorganic substances is fraught with great mathematical difficulties, which were been overcome only for the simplest systems. Therefore various approximated methods, as a rule, are used. These methods very much frequently do not give desirable results.

On the other hand, the chemistry had accumulated large information on properties of inorganic substances. There are periodic regularities between properties of compounds and properties of elements, which are included into their composition. This supposition is a consequence of the Periodic Law. Moreover, it is obviously, that already known compounds should be in accordance with these periodic regularities. The aims of our researches are development of methods and creation of computer system for search for these periodic regularities on the basis of analysis of information about already known substances accumulated in databases on properties of inorganic substances and materials. The found regularities are used for design of new inorganic compounds – analogues of already synthesized substances.

Selection of Methods of Search for Regularities in Information of Databases on Properties of Inorganic Substances and Materials

The methods of computer learning in pattern recognition are one of the most effective means of search for regularities in the large arrays of the chemical data [Kiselyova, 2005; Savitski and Gribulya, 1985]. In this case it is possible to connect some discrete parameters of inorganic compounds (for example, possibility of formation of compound or type of its crystal structure under normal conditions) with properties of elements, which are included into their composition, and also to get a threshold estimation of some numerical properties (for example, estimation of the melting point of compound at atmospheric pressure - above or below than certain threshold). It is important, that the fulfillment (though also not so strict) of basic hypothesis of methods of pattern recognition - hypothesis of compactness - is a consequence of the Periodic Law. Let an each compound corresponds to a point in multi-dimensional space of properties of elements. Owing to periodicity of properties of chemical elements points, which correspond to combinations of close on properties elements, combining into compounds, form compact clusters. Thus, the task of search for regularities connecting property of inorganic compounds with properties of chemical elements can be reduced to a problem of computer learning in pattern recognition. In this case the analysis of the information about already known compounds, which are represented as a set of values of properties of chemical elements, allows discovery of classifying regularities. The latter allow separation of known compounds into predetermined classes. It is possible to predict new compounds and estimate their unknown parameters by substitution of the property values of the appropriate chemical elements into the found regularities.

The principal problems at application of methods of pattern recognition to the decision of tasks of inorganic chemistry are following:

1. Small informativeness of attributes - properties of chemical elements.
2. The strong correlation of these attributes owing to their dependence on common parameter - atomic number of chemical elements (it follows from the Periodic Law).
3. Omissions in values of attributes.
4. In many cases - the large asymmetry of a size of classes of training set.
5. Sometimes feature description includes non-numerical attributes.
6. Possibility of experimental mistakes of classification in training sets.

In connection with the above-stated peculiarities of subject domain the search for methods and algorithms of pattern recognition allowing correct solution of these problems was one of the basic tasks of development of information-analytical system (IAS) for computer-aided design of inorganic compounds. It was established during testing various algorithms of computer learning for concrete tasks that it is impossible to specify beforehand, what algorithm is most effective at the decision of the certain chemical task of design of inorganic compounds. Quite often programs, which well have classified training set, obtained bad results at the prediction of unknown compounds. In this connection the most effective way of decision of tasks of predicting properties of new inorganic compounds is concerned with methods of recognition by collectives of algorithms [Zhuravlev et al., 2006]. At synthesis of the collective decision it is possible to compensate mistakes of separate algorithms by the correct predictions of other algorithms. Hence, the developed information-analytical system includes a set of the programs realized algorithms of various types, and also different strategies of collective decisions making.

Other way of increase of accuracy of predicting is a use of dependence of properties of chemical elements on atomic number. On the one hand, this fact complicates a task of search for separate properties that are the most important for classification of because of strong correlation of all used parameters of elements forming feature description. On the other hand, the classifying regularities including values of any subset of properties of chemical elements, which are used for the description of inorganic compounds, should in principle give identical results of classification. I.e. the results of the prediction with use of various subsets of properties of elements should, basically, coincide. This fact allows an additional possibility of collective decision making but already on the basis of collective of feature descriptions which was obtained as a result of division of initial set of properties of chemical elements on partially crossed subsets.

The problem of filling omissions also is partially solved with use of periodic dependences of parameters of elements. Replacement of the omission by average value of given parameter for two chemical elements that are nearest (within the range of group of Periodic System) to the element with omission is used.

After testing the programs the following software of pattern recognition were included into information-analytical system:

- a wide class of algorithms of system RECOGNITION developed by A.A.Dorodnicyn Computer Center of Russian Academy of Sciences (CCAS) [Zhuravlev et al., 2006]. This multifunctional system of pattern recognition includes the well-known methods of *k*–nearest neighbors, Fisher's linear discriminant, linear machine, multi-level perceptron (neural networks), support vector machine, genetic algorithm, and the special algorithms which were developed by CCAS: estimates calculation algorithms, LoReg (Logical Regularities), deadlock test algorithm, statistical weighted syndromes, etc.

- system of concept formation ConFor developed by V.M.Glushkov Institute of Cybernetics of National Academy of Sciences of Ukraine [Gladun, 1995, 2000, 2005]. The system is based on special data structure in a computer memory named as growing pyramidal networks.

It is important, that system RECOGNITION [Zhuravlev et al., 2006] is equipped with a set of algorithms of the decision of tasks of recognition by collectives of various algorithms. In this case task of recognition is decided in two stages. At first various algorithms, which are included into system, are applied independently. Further an optimum collective decision is made automatically with the help of special methods - "correctors". Some of methods of synthesis of the collective decisions - Bayesian corrector, convex stabilizer, some heuristic methods, etc. are used as correctors.

Databases and Knowledge Base of Information-Analytical Systems

The information basis of IAS (fig.1) is the integrated system of databases on properties of inorganic substances and materials [Dudarev et al., 2006; Kiselyova et al., 2005], which now includes:

- DBs containing the brief information on the most widespread properties of inorganic compounds and chemical elements:

1). DB on properties of inorganic compounds "Phases" [Kiselyova, 2005; Kiselyova et al., 2006] which now contains the information on properties more than 43, 000 ternary compounds (i.e. compounds formed by three chemical elements) and more than 15, 000 quaternary compounds, that was extracted from about 20, 000 publications.

2). DB on properties of chemical elements "Elements" which includes the data on more than 90 parameters.

- Specialized DBs which contain the detailed information on industrially vital substances and materials:

1). DB of phase diagrams of systems with intermediate semiconducting phases “Diagram” [Kiselyova, 2005; Khristoforov et al., 2001], that contains information on the most important pressure-temperature-concentration phase diagrams of semiconducting systems evaluated by qualified experts and also on the physical-chemical properties of the intermediate phases. Now DB contains the detailed information on several tens binary and ternary systems extracted from 2000 publications.

2). DB on substances with significant acousto-optical, electro-optical and nonlinear-optical properties “Crystal” [Kiselyova, 2005; Kiselyova et al., 2004] which now includes the information on parameters more than 100 materials.

3). DB on width of the forbidden zone of inorganic substances “Bandgap” [Dudarev et al., 2006] which now contains the data on more than 2, 000 substances.

Cumulative volume of DBs is ~7 GB. All these databases are accessible from Internet (www.imet-db.ru).

The knowledge base (KB) of information-analytical system includes the relational tables containing regularities found during computer learning with the indication about common chemical composition of compounds, set of attributes included into regularity, parameter to be predicted, used algorithm, and also service information (date of updating, surname of the expert who carried out computer learning, etc.). Both the integrated system of DBs on properties of inorganic substances and materials and knowledge base are realized with use of DBMS MS SQL Server. DBs, which are incorporated into the integrated information system, use various DBMS [Dudarev et al., 2006].

The information-analytical system for design of inorganic compounds is intended for users of two levels. Firstly it is a reference system for ordinary specialist. Secondly IAS is a tool for expert estimating the chemical information for computer learning and carrying out a search for regularities in data (fig.1). In last case owing to use of knowledge and experience of the highly skilled experts the mentioned above problems of selection of the most important attributes for the description of compounds and filtration of mistakes of classification of objects of training set are partially decided.

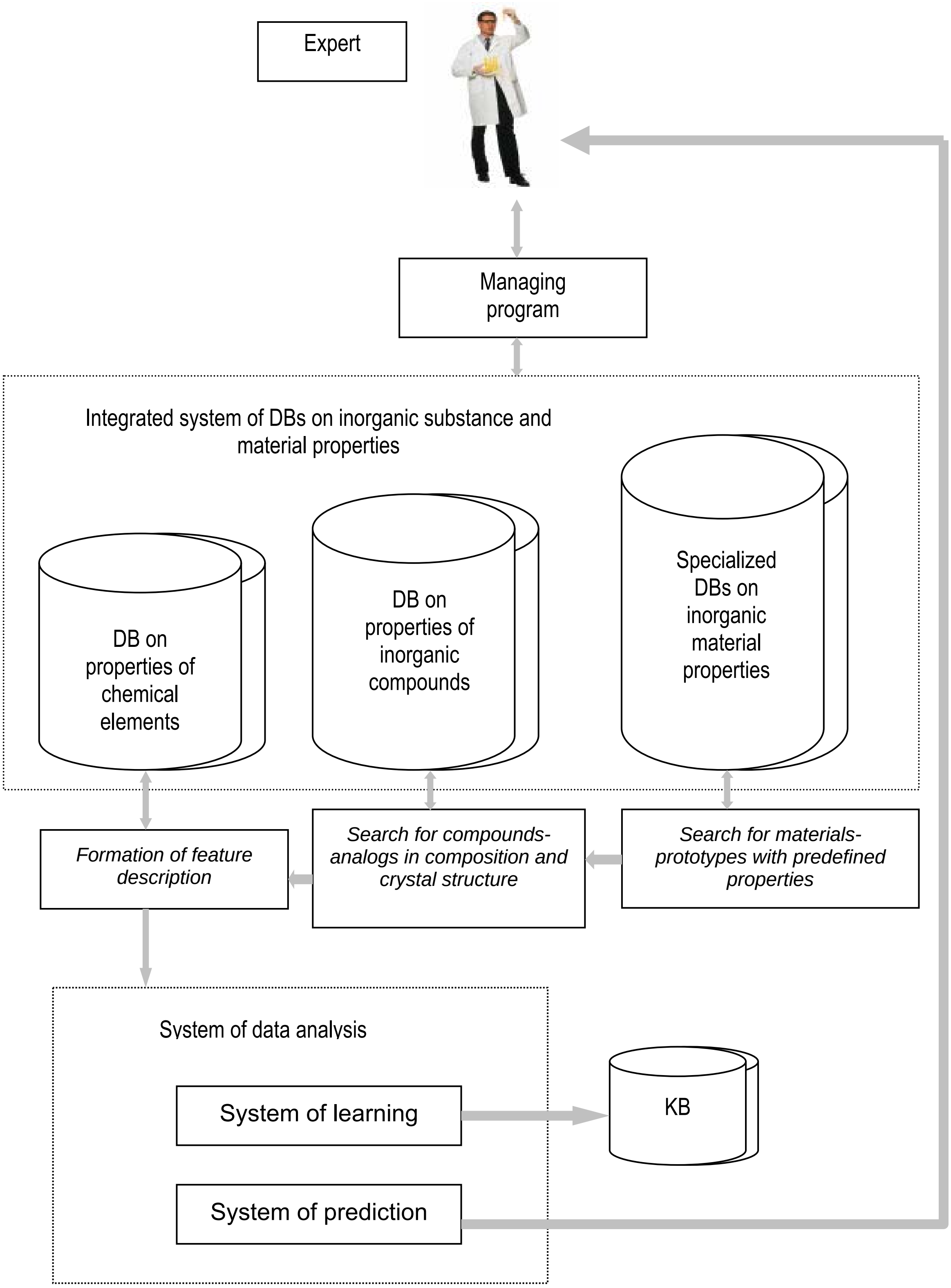

Expert
Managing program
Integrated system of DBs on inorganic substance and material properties
DB on properties of chemical elements
DB on properties of inorganic compounds
Specialized DBs on inorganic material properties
Formation of feature description
Search for compounds-analogs in composition and crystal structure
Search for materials-prototypes with predefined properties
System of data analysis
System of learning
System of prediction
KB

Fig.1. Principal schema of IAS

Program Realization of Information-Analytical System

Feature of program realization of IAS is the design of client module on the basis of the Web-interface completely. The users work with IAS using only Web-browser. Thus, the users do not need to install of any additional programs. It also facilitates an expansion of system by new methods and functionalities: the changes are done only in server where the system is located. Interaction between predicting subsystems, which realize various methods of learning and predicting, and subsystem of the graphic Web-interface is realized on the basis of the interface module-broker or the "shell" giving all necessary functions of system, that also corresponds to the ideas of SOA-approach. The operation of processes of training and recognition is realized by means of asynchronous Web-services. At design of IAS it is necessary to provide storage of the information about the programs of data analysis and methods of recognition realized in them. These data are necessary for a correct invocation of functions of programs and setting of learning methods. The relational DB called as metabase was used for data storage. The knowledge base is realized using SQL-server and Web-server. Web-server is intended for storing all necessary files using special format for subsequent their application to recognizing. SQL-server stores complete information on the obtained regularities.

Conclusion

The information-analytical system, created by us, allows solution of two important tasks of inorganic chemistry. First, it allows the partially automation of analysis of the huge experimental information, accumulated by chemistry, for search for regularities in the data and subsequent design of new compounds with predefined properties. Secondly, it expands opportunities of traditional DBs on properties of substances and materials, giving the user not only information on the already investigated substances, but also predictions of some substances not yet synthesized and estimation of their properties. Essential advantage of developed IAS is Internet-access. In this case user receives operative access to "alive" data and regularities. With the help of IAS it was possible to predict some new inorganic compounds and to estimate their some properties.

Acknowledgements

Financial support from RFBR (grants №06-07-89120 and 05-03-39009) is gratefully acknowledged. We should like to thank our colleagues Prof. V.P.Gladun, Dr. V.Yu.Velichko of the Institute of Cybernetics of the National Academy of Sciences of Ukraine, Prof. V.S.Zemskov, Dr. V.A.Dudarev, Dr. I.V.Prokoshev, V.V.Khorbenko of the Institute of Metallurgy and Materials Science of Russian Academy of Sciences, and Dr. O.V.Sen’ko of the Computer Center of the Russian Academy of Sciences for their help and support.

Bibliography

[Dudarev et al., 2006] V.A.Dudarev, N.N.Kiselyova, V.S.Zemskov. Integrated system of databases on properties of materials for electronics. Perspektivnye Materialy, 2006, N.5 (Russ.).

[Gladun, 1995] V.P.Gladun. Processes of Formation of New Knowledge. SD "Pedagog 6”, Sofia, 1995 (Russ.).

[Gladun, 2000] V.P.Gladun. Partnership with Computer. Port-Royal. Kiev, 2000 (Russ.).

[Gladun, 2004] V.P.Gladun. Growing pyramidal networks. Novosti Iskusstvennogo Intellekta, 2004, №1 (Russ.).

[Khristoforov et al., 2001] Yu.I.Khristoforov, V.V.Khorbenko, N.N.Kiselyova, et al. Internet-accessible database on phase diagrams of semiconductor systems. Izvestiya VUZov. Materialy Elektron.Tekhniki, 2001, №4 (Russ.).

[Kiselyova, 2005] N.N.Kiselyova. Computer Design of Inorganic Compounds. Application of Databases and Artificial Intelligence. Nauka, Moscow, 2005 (Russ.).

[Kiselyova et al., 2005] N.N.Kiselyova, V.A.Dudarev, I.V.Prokoshev, et al. The distributed system of databases on properties of inorganic substances and materials. Int.J."Information Theories & Applications”, 2005, v.12.

[Kiselyova et al., 2006] N.Kiselyova, D.Murat, A.Stolyarenko, et al. Database on ternary inorganic compound properties “Phases” in Internet. Informazionnye Resursy Rossii, 2006, N.4 (Russ.).

[Kiselyova et al., 2004] N.N.Kiselyova, I.V.Prokoshev, V.A.Dudarev, et al. Internet-accessible electronic materials database system. Inorganic Materials, 2004, v.42, №3.

[Savitski and Gribulya, 1985] E.M.Savitski and V.B.Gribulya. Application of Computer Techniques in the Prediction of Inorganic Compounds. Oxonian Press Pvt.Ltd., New Delhi-Calcutta, 1985.

[Zhuravlev et al., 2006] Yu.I.Zhuravlev, V.V.Ryazanov, O.V.Senko. RECOGNITION. Mathematical methods. Software System. Practical Solutions. Phasis, Moscow, 2006 (Russ.).

Authors' Information

Nadezhda Kiselyova – *A.A.Baikov Institute of Metallurgy and Materials Science of Russian Academy of Sciences, P.O.Box: 119991 GSP-1, 49, Leninskii Prospect, Moscow, Russia, e-mail: kis@ultra.imet.ac.ru*

Andrey Stolyarenko –- *A.A.Baikov Institute of Metallurgy and Materials Science of Russian Academy of Sciences, P.O.Box: 119991 GSP-1, 49, Leninskii Prospect, Moscow, Russia, e-mail: stol-drew@yandex.ru*

Vladimir Ryazanov – *A.A.Dorodnicyn Computer Center of Russian Academy of Sciences, e-mail: riazanov@ccas.ru*

Vadim Podbel'skii – *Moscow Institute of Electronics and Mathematics (Technical University), P.O.Box: 109028, B.Trehsvjatitelsky per. 3/12, Moscow, Russia, e-mail: vvp@mitme.ru, vpodbelskiy@hse.ru*

AN IDEA OF A COMPUTER KNOWLEDGE BANK ON MEDICAL DIAGNOSTICS

Mery Chernyakhovskaya, Alexander Kleschev, Filip Moskalenko

Abstract: *The paper is a description of information and software content of a computer knowledge bank on medical diagnostics. The classes of its users and the tasks which they can solve are described. The information content of the bank contains three ontologies: an ontology of observations in the field of medical diagnostics, an ontology of knowledge base (diseases) in medical diagnostics and an ontology of case records, and also it contains three classes of information resources for every division of medicine – observation bases, knowledge bases, and data bases (with data about patients), that correspond to these ontologies. Software content consists of editors for information of different kinds (ontologies, bases of observations, knowledge and data), and also of a program which performs medical diagnostics.*

Keywords: *Medical Diagnostics, ontology model, parallel computing, knowledge bank.*

ACM Classification Keywords: *I.2.1 Applications and Expert Systems, J.3 Life and Medical Sciences.*

Introduction

Computer systems for medical diagnostics are one of applications of AI systems. They can help doctors to improve the quality of their work. The task of such systems is to recognize diseases (one or several), with which a patient is ill, basing on the results of patient's observations. The important components of such systems are a confidence subsystem which can show to the doctors the knowledge base of the system and an explanation subsystem which can show to the doctors the information and reasoning way which were used to produce the result.

Two classes of the systems for solving the task of medical diagnostics have been developed by now, which differ by methods that lie in their base. The systems of the first class are based on statistical and other mathematical models – their bases are mathematical algorithms that perform the search of usually a partial correspondence between the symptoms of the current patient and the symptoms of previous patients for whom the diagnoses are known [1 – 4]. However such systems lack the confidence and explanation subsystems.

The systems of the second class are based on expert knowledge. Their algorithms operate with the information about the patient and with the knowledge about diseases which are represented in a form that is more or less close to the concepts of doctors (and described by expert-doctors). That is achieved by an explicit or implicit usage of ontologies of medical diagnostics. In these systems it is possible to create the subsystems of confidence and explanation which is capable of giving a doctor the results of analysis of patient's state that led to the derived result.

The models of ontologies used in such systems take into account temporal changes of symptom values [1], connections between the symptoms and diseases (for example, using logical rules) [2, 5], division of observations into several groups (for example, clinical, laboratory, morphological data – [4]), representing the state of a patient as a multi-level model [6].

The algorithms in such systems try to imitate the way of reasoning of doctors [7, 8], to look for a correlation between the information about the patient and clinical findings of diseases that were described by experts [9], or to process the rules that describe the links between observations and diseases and are set by experts [5, 8, 10, 11].

An analysis of recent systems for medical diagnostics of the second type shows that the ontologies that they use are rather simple and do not combine such information (widely used by doctors) as: knowledge about the reasons of diseases; knowledge about different types of cause-effect relation between symptoms and diseases; knowledge about the influence of events on symptom values; knowledge about different variants of temporal changes of symptom values, that depend on anatomical-and-physiological features of patients.

In addition, one of the negative properties of modern systems is that they cannot be widely used. This is because they are either prototypes made for research purposes or they are developed for a particular medical institution and are not available outside its LAN. On the other hand, systems that allow a wide access to their resources though modern technologies (like Internet) do not let experts to broaden their knowledge bases, for example – DXplain [2] and Diagnostics of preeclampsia [12].

Thus, to develop a system for medical diagnostics a) which would be based on expert knowledge and on ontology model that takes into account all features of medical knowledge stated above, b) in which the model of knowledge is close to concepts of doctors and c) which allows not only to recognize the diagnosis but also to explain how it was obtained – is an urgent task. Such a system should realize the diagnostics in acceptable (for doctors) time in spite of the fact that a non-trivial ontology of medical knowledge lies in its base. What is more, such a system must provide an access to its resources for as many users as possible for both purposes – performing medical diagnostics and participating in gathering and improving medical knowledge about different diseases.

The goal of this paper is to describe a concept of a network resource on medical diagnostics that possesses all features stated above.

This paper was made according to the project of FEBRAS № 06-III-A-01-457 «Designing, implementing and developing the bank of medical knowledge in the Internet network» and according to the project of RFBR № 06-07 89071 «Research of possibilities of collective control in semantic web for information resources of different types of generality».

1. A theoretical background and general principles for developing the Bank of knowledge on medical diagnostics

Specialists in the field of AI and experts in medical diagnostics have developed several models of ontology for medical diagnostics in recent years. Some of these models were used for developing diagnostic systems that were based on expert knowledge. As it was stated in the introduction, each of them was somehow better and somehow worse than others [1, 2, 4, 5, 6]. In order to unite the advantages of those models and to create a model of ontology that would be close to concepts of experts in the field of medical diagnostics – an ontology and its model were developed in [13,14,15]. This ontology of medical diagnostics describes acute diseases; interaction of cause-and-effect relations of different types is taken into account. The ontology is close to real concepts of medical diagnostics in the Russian Federation and it defines combined and complicated pathology, development of pathological processes in time, influence of treatment and other events on diseases' manifestations. The model of this ontology includes definitions of terms of the knowledge model (parameters), the definitions of terms of the reality model (unknowns) and is an unenriched logical relationship system with parameters, which also describes of integrity constraints for unknowns, and parameters and relationships between them.

The relationships between unknowns and parameters can be divided into the following semantic groups:

1) the relationships between knowledge about cause-and-effect relations and cause-and-effect relations which take place in situations;

2) the relationships which determine cause-and-effect relations that are the reasons of values of each sign during its development intervals;

3) the relationships which determine the properties of borders of intervals of the time axis for each sign;

4) the relationships which determine the reason of each disease from the diagnosis.

In [16] a general problem of medical diagnostics is formulated based on the described model of the ontology: to recognize the possible diagnoses of a patient basing on the domain knowledge and observation data, which are values of symptoms (during the times of observations), values of anatomical-and-physiological features (constant in time) and values of events that took place with the patient (in the moments when they occur), and also for each diagnosis – to state its reason (some event or another disease) and its explanation (by indicating the reasons of observed values of symptoms).

Because the outlined model of the ontology takes into account a large amount of interrelations between the processes that take place in patient's body, it can be expected that any algorithm for solving the formulated above problem of medical diagnostics (which has to analyze all these relations) will be of an intense computing complexity. One of the ways to improve the effectiveness of this algorithm is to parallel it and then to execute it on a multiprocessor computing system. The maximum effect from paralleling can be achieved while solving some particular but nevertheless a practically important problem – when the observed symptoms can be analyzed independently and thus – simultaneously on the nodes of a multiprocessor computing system. Such problem comes out when several restrictions are applied to the used model of the ontology: patient can be either healthy or sick with only one disease; each disease has particularly one period of development.

Such a problem is specified in the terms of the restricted ontology. An algorithm for its solving is presented in [16]. In [17] both a parallel algorithm for solving that problem and an algorithm for disease database optimization (for reduction of the amount of hypothesizes to check) are presented. The results of an experimental research of time-complexity for the optimized algorithm for solving the particular medical diagnostics problem [17] are described in [18]. This research shows that:

a) usage of the optimized disease database remarkably reduces the time of diagnostics, especially when a great number of diseases is stored in database;

b) the maximum speed of diagnostics is achieved when all the nodes of a multiprocessor computing system are used and on each of them only a single copy of a client-part of the algorithm is executed;

c) the time of diagnostics on al test-patterns (while using all available nodes and using the optimized database) did not exceed several minutes.

The described ontology and the algorithm for medical diagnostics can be used for developing a system that will support the process of a coordinated solving the tasks of gathering, formalizing, translating to a computer-readable representation, engineering, storing, managing and processing data and knowledge in the field of medical diagnostics and will be a combination tool for all this information into a single resource with remote access for various users.

The system will be called a Bank of knowledge on medical diagnostics. In order to implement all the mentioned requirements the system should be built in accordance with three-tier software architecture which means the Bank should consist of the following parts:

- information content with some standard interface for access to the stored information and with unified format for storing that data;
- software content which is oriented on intellectual support for users of the Bank and which includes the following: tools for editing the data of the information content, tools for their processing (optimization of knowledge base about diseases and medical diagnostics of a patient) and an administrative subsystem;
- interfaces for access to software components.

2. The information content of the Bank of knowledge on medical diagnostics

The information content can be divided into several partitions – one for each section of medical diagnostics (ophthalmology, cardiology, etc.). Each partition includes three types of bases, all formed in accordance with the model of the ontology for medical diagnostics [13, 14, 15].

The model of ontology consists of three parts:

- an ontology model for observations,
- an ontology model for knowledge about the influence of diseases on symptom values (model of knowledge about diseases),
- an ontology model for patient (an ontology model for patient’s case record).

The ontology model for observations describes the structure of observations and their values. Observations are the observed symptoms, events and anatomical-and-physiological features. The base of observations is built on the basis of this model and it contains the names of events, symptoms and features and also lists of their possible values.

The ontology model for patients’ case records is built on the basis of the ontology model for observations and it sets the structure for describing the state of a patient as a function of time. Events can happen to a patient at different moments of time and can have different values. Symptoms also can have different values at different moments of time and the values of anatomical-and-physiological features of a patient are constant in time. Basing on this model of a patient those who perform the medical diagnostics form the base of patients’ case records, that is create the records of patients in the information content which store the values of symptoms (observed at different moment of time), the values of events (that happen to a patient at different moments of time) and the values of anatomical-and-physiological features.

The ontology model for knowledge about diseases contains the descriptions of basic terms of knowledge (including the relations between them) in terms of the ontology model for observations. They include the descriptions of diseases (and their reasons) and of the normal state of the patient. This scheme is used by experts as a template for describing the particular diseases (symptom values during diseases are defined by clinical manifestations and clinical manifestations modified by event’s influence), their reasons (a set of etiologies is described) and normal state of the patient (normal reactions and reactions to event’s influence are described).

Figure 1 shows the general scheme of the information content for the Bank of knowledge on medical diagnostics. Arrows on the figure show how one component of the information content is used for forming another one.

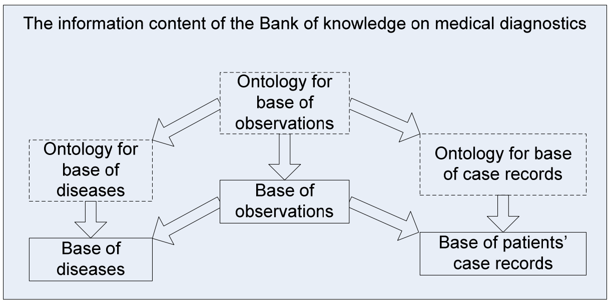

The information content of the Bank of knowledge on medical diagnostics
Ontology for base of observations
Ontology for base of diseases
Ontology for base of case records
Base of observations
Base of diseases
Base of patients’ case records

Figure 1. Information content of the Bank of knowledge on medical diagnostics.

3. Software for problems which can be solved by means of the Bank of knowledge on medical diagnostics by users of different types

As it was stated in section 1, the software content of the Bank consists of software components of three types:

- editors for information content,
- problem solvers,
- an administrative subsystem.

In accordance with the mentioned in section 2 parts of the information content of the Bank of knowledge on medical diagnostics the following editors for them have to be developed:

- an editor for observations bases;
- an editor for bases of knowledge about diseases;
- an editor for bases of patients' case records.

For solving the problem of medical diagnostics the above mentioned algorithm is used [17]. It takes into account all relations between knowledge and reality that are stored in the model of ontology [14,15].

The software tool that performs the optimizing transformation of the knowledge base with diseases' descriptions is an implementation of the algorithm described in [17]. It analyzes the information from the disease base and recognizes during which diseases each value of each symptom can be observed. Basing on that information the diagnostics algorithm forms a list of hypothesizes about diagnosis for checking which probably contains fewer diseases than the set of all the diseases.

The groups of users of Bank of knowledge on medical diagnostics are:

- servicing users (administrators),
- information mediums (experts),
- users (doctors and guests).

An administrator traces the functionality of the entire Bank of knowledge with the use of administrative subsystem, which lets him to perform two functions: working with user accounts and controlling the users' activities.

Experts edit the base of observations and the base of knowledge about diseases in accordance with the correspondent models of ontologies. There is a single base of observations in each partition of the information content of the Bank. As for the base of knowledge about diseases – for each user of this type a personal base is formed in a particular partition and he fills it with knowledge. When he achieves some final and essential results during his research, he sends to the administrator a special request for adding that information into the global knowledge base in the information content. If the administrator approves that – he expands the knowledge base with those new descriptions of diseases with the help of administrative subsystem.

Users are the doctors who work with the base of case records and perform the diagnostics and guests who are able only to view the data in the observations base and in the base with descriptions of diseases that are stored in information content.

4. An implementation of the Bank of knowledge on medical diagnostics

An implementation of the described system can be accomplished with using the Multipurpose Bank of Knowledge (MBK) which has been developed in the IACP FEB RAS [19,20]. It is meant to be used for supporting life-cycle of compatible systems for information processing.

In this case, the software content of the Bank of knowledge on medical diagnostics includes:

- an editor for information of different level of generality (IDLG), which is a part of MBK (this editor is used for editing the models of ontologies and information structures organized according to these models in IDLG language [21,22]);
- an administrative subsystem which is also a part of MBK and which is used for managing user accounts and authorities;
- a solver for the medical diagnostics problem;
- a transformer of the base with diseases' descriptions.

The diagnostics algorithm is implemented as a parallel application in C++ which runs at on a multi-processor computing machine (MPCM) under OS Linux. The knowledge bank has to have an access to that resource. The knowledge bank server contains a low-functional server-part of the solver, implemented in Java, which interacts with the user, executes the diagnostics on the MPCM, receives the results and sends them to user.

As the diagnostics algorithm has to have a rapid access to knowledge and data in order to perform the diagnostics as fast as possible – the data from the diseases knowledge base has to be sent to and stored at

MPCM before any diagnostics. This procedure is performed by a knowledge-base transformation subsystem. When the diagnostics is executed – only the patient's case record is transmitted to MPCM.

When the diagnostics ends all results (rejected and approved hypothesizes about diagnosis with their reasons and explanations for values of observed symptoms) are transmitted from MPCM by server-part of the solver to the user.

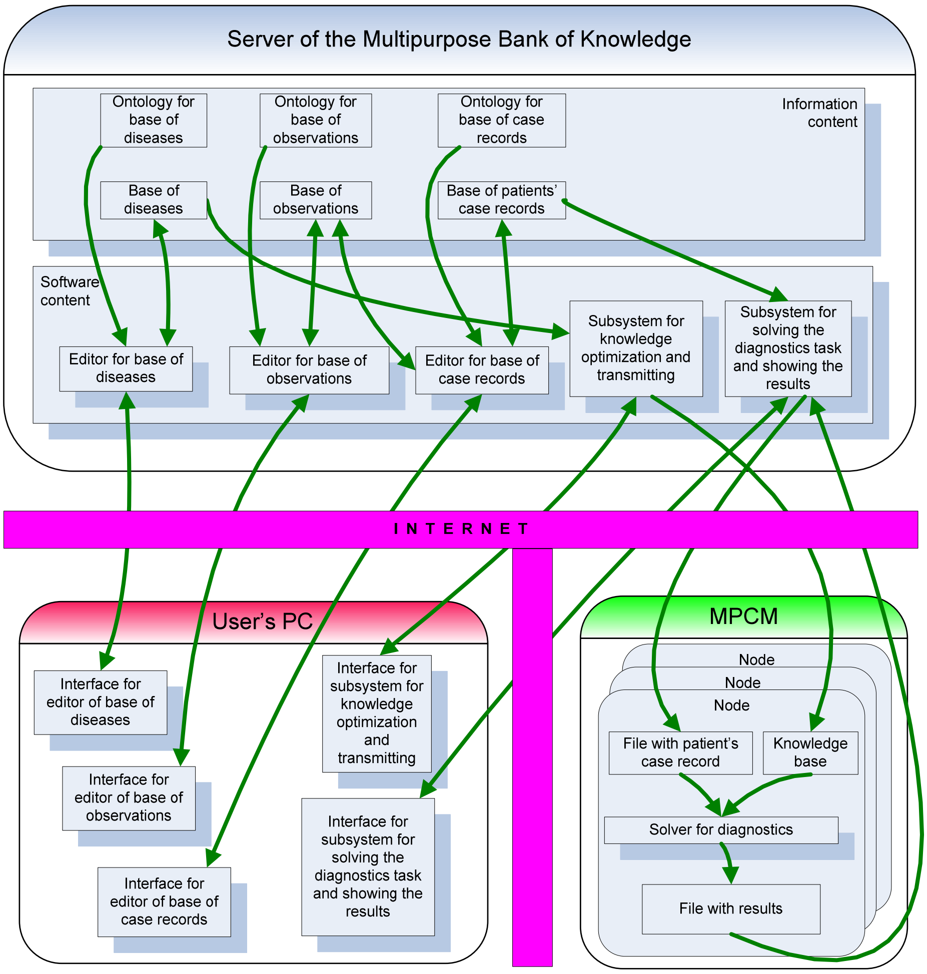

Server of the Multipurpose Bank of Knowledge
Ontology for base of diseases
Ontology for base of observations
Ontology for base of case records
Information content
Base of diseases
Base of observations
Base of patients' case records
Software content
Subsystem for knowledge optimization and transmitting
Subsystem for solving the diagnostics task and showing the results
Editor for base of diseases
Editor for base of observations
Editor for base of case records
I N T E R N E T
User's PC
Interface for editor of base of diseases
Interface for editor of base of observations
Interface for editor of base of case records
Interface for subsystem for knowledge optimization and transmitting
Interface for subsystem for solving the diagnostics task and showing the results
MPCM
Node
Node
Node
File with patient's case record
Knowledge base
Solver for diagnostics
File with results

Figure 2. The architecture of the Bank of knowledge on medical diagnostics.

Figure 2 shows the general architecture of the Bank of knowledge on medical diagnostics in MBK/MPCM environment. As the scheme is too complex the administrative subsystem, its interface and user account database are not shown. Also the editors for ontology models (here – the editor of IDLG) are not shown as they are just used for initializing the correspondent bases and are not used afterwards. One should also understand that the standard (for MBK) editor for IDLG is used in the developed Bank for any base but it can be replaced with any other (more convenient) editor. In this case it has to be implemented and included into the software content.

Conclusion

A conception for the Bank of knowledge on medical diagnostics is presented in the paper. The information content of this Bank consists of three models of ontologies for the domain (model of observations, model of knowledge about diseases, model of patient's case record), knowledge of the domain (base of observations and base of knowledge about diseases) and data of reality (base of patients' case records). Which is more, the Bank contains the software content which consists of editors for models of ontologies, editors for data and knowledge, solver for the task of medical diagnostics, a software tool for transforming the knowledge base and administrative subsystem. The details of implementation of the Bank by means of the Multipurpose Bank of Knowledge (developed in IACP FEB RAS) are described.

Bibliography

1. Genkin A.A. About a sequential Bayes strategy and a mechanism for decision support in intellectual system OMIS. // Clinical laboratory diagnostics. - 1998. - №4.- pp. 42-49. (in Russian)
2. Detmer W.M., Shortliffe E.H. Using the Internet to Improve Knowledge Diffusion in Medicine. 1997. http://www-smi.stanford.edu/pubs/SMI_Reports/SMI-97-0658.pdf
3. Alekseev A.V. Using methods of informatics and computers for differential diagnostics of appendicitis and appendicular colic. http://arkadyal.chat.ru/rar/ddoaak_.rar (in Russian)
4. Burdaev V.P. Applied expert systems based on TECHEXP shell. 2002. http://uacm.kharkov.ua/eng/index.shtml?eexpert.htm (in Russian)
5. Expert system “Diagnostic of comatose states”. Admin LC. 1992. http://www.adminru.com/start_e.htm
6. Chabat F., Hansell D.M., Guang-Zhong Yang. Computerized Decision Support in Medical Imaging. 1997. http://www.doc.ic.ac.uk/~gzy/pub/chabat-decision-support.pdf
7. Gaskov A.P., Valivach M.N. The “Consultation” expert system for medical diagnostics. 2000. http://www.eksi.kz/consilium/librar/esmd_short.htm (in Russian)
8. Lhotska L., Vlcek T. Efficiency enhancement of rule-based expert systems. Proceedings of the 15th IEEE symposium on computer-based medical systems (CBMS 2002). pp.53-58.
9. Matsumoto T., Ueda Y., Kawaji Sh. A software system for giving clues of medical diagnosis to clinician. Proceedings of the 15th IEEE symposium on computer-based medical systems (CBMS 2002). pp. 65-58.
10. Filho M.M., Palombo C.R., Sabbatini R.M. TMJ Plus: A Knowledge Base and Expert System for Diagnosis and Therapeutics of the Temporomandibular Joint Disorders. 1994. http://www.epub.org.br/ojdom/vol03n03.htm
11. Sawar S.J. Diagnostic Decision Support System of POEMS. 1999. http://www.cbl.leeds.ac.uk/sawar/projects/poems/poems-in-detail.html
12. Zilber A.P., Shifman E.M., Pavlov A.G., Belousov S.E. The "Computer diagnostics of preeclampsia" Internet project. 1998. http://critical.onego.ru/critical/medlogic/ (in Russian)
13. Chernyakhovskaya M.Yu., Kleschev A.S., Moskalenko F.M. A metaontology for medical diagnostics of acute diseases. Part 1. An informal description and definitions of basic terms. INFOS 2008, Varna, Bulgaria. (to be published)
14. Chernyakhovskaya M.Yu., Kleschev A.S., Moskalenko F.M. A metaontology for medical diagnostics of acute diseases. Part 2. A formal description of cause-and-effect relations. INFOS 2008, Varna, Bulgaria. (to be published)
15. Chernyakhovskaya M.Yu., Kleschev A.S., Moskalenko F.M. A metaontology for medical diagnostics of acute diseases. Part 3. A formal description of the causes of signs’ values and of diseases. INFOS 2008. (to be published)
16. Moskalenko F.M. A task of medical diagnostics and an algorithm for its solving which can be paralleled // Informatics and control systems. – 2005. – № 2(10). – pp.52-63. (in Russian)
17. Moskalenko F.M. A parallel optimized algorithm for medical diagnostics // Informatics and control systems. – 2006. – № 1(11). – pp.87-98. (in Russian)
18. Moskalenko F.M. An experimental research of time-complexity for a parallel algorithm of medical diagnostics which is based on a real ontology // Informatics and control systems. – 2006. – № 2(12). – pp.42-53. (in Russian)
19. Orlov V.A., Kleschev A.S. Computer knowledge banks. Multipurpose bank of knowledge. // Information technologies – 2006. – №2. – pp.2-8. (in Russian)
20. Orlov V.A., Kleschev A.S. Computer knowledge banks. Requirements for the Multipurpose bank of knowledge. // Information technologies – 2006. – №4. – pp.21-28. (in Russian)
21. Orlov V.A., Kleschev A.S. Computer knowledge banks. A universal approach for solving the problem of information editing. // Information technologies – 2006. – №5. – pp.25-31. (in Russian)
22. Orlov V.A., Kleschev A.S. Computer knowledge banks. Модель процесса редактирования информационного наполнения. // Information technologies – 2006. – №7. – pp.11-16. (in Russian)

Information about authors

Chernyahovskaya M.Yu. **-** *chernyah@iacp.dvo.ru*

Kleschev A.S. **-** *kleschev@iacp.dvo.ru*

Moskalenko Ph.M. **-** *philipmm@yahoo.com*

Institute for automation and control processes, Far-eastern branch of Russian Academy of Sciences, Rusian Federation, Vladivostok, Radio st. - 5.

SOLVING TRAVELLING SALESMAN PROBLEM IN A SIMULATION OF GENETIC ALGORITHMS WITH DNA

Angel Goñi Moreno

Abstract: *In this paper it is explained how to solve a fully connected N-City travelling salesman problem (TSP) using a genetic algorithm. A crossover operator to use in the simulation of a genetic algorithm (GA) with DNA is presented. The aim of the paper is to follow the path of creating a new computational model based on DNA molecules and genetic operations. This paper solves the problem of exponentially size algorithms in DNA computing by using biological methods and techniques. After individual encoding and fitness evaluation, a protocol of the next step in a GA, crossover, is needed. This paper also shows how to make the GA faster via different populations of possible solutions.*

Keywords: *DNA Computing, Evolutionary Computing, Genetic Algorithms.*

ACM Classification Keywords: *I.6. Simulation and Modeling, B.7.1 Advanced Technologies, J.3 Biology and Genetics*

Introduction

In a short period of time DNA based computations have shown lots of advantages compared with electronic computers. DNA computers can solve combinatorial problems that an electronic computer cannot like the well known class of NP complete problems. That is due to the fact that DNA computers are massively parallel [Adleman, 1994]. However, the biggest disadvantage is that until now molecular computation has been used with exact and “brute force” algorithms. It is necessary for DNA computation to expand its algorithmic techniques to incorporate aproximative and probabilistic algorithms and heuristics so the resolution of large instances of NP complete problems will be possible.

On the other hand there are genetic algorithms (or short GA) which are categorized as global search heuristics and use techniques inspired by evolutionary biology [Holland, 1975]. It seems to be perfect to combine DNA computing and GAs.

Previous work on molecular computation for genetic algorithms [J.Castellanos, 1998] show the possibility of solving optimization problems without generating or exploring the complete search space and give a solution to the first step to be done in a GA, the coding of the population and the evaluation of individuals (fitness). A recent work [M.Calviño, 2006] produced a new approach to the problem of fitness evaluation saying that the fitness of the individual should be embedded in his genes (in the case of the travelling salesman problem in each arch of the path). In both cases the fitness will be determined by the content in G+C (cytosine + guanine) which implies that the fitness of an individual will be directly related with the fusion temperature and hence would be identifiable by spectophotometry and separable by electrophoresis techniques [Macek 1997].

In this paper the crossover (also called recombination) of DNA-strands has been resolved satisfactorily by making a crossover operator suitable for DNA computing and its primitive operations. This crossover operator is used in the simulation of the travelling salesman problem (TSP) with both genetic algorithm and DNA computing continuing the work previously done about the coding of information. The Lipton [Lipton, 1995] encoding is used to obtain each individual coded by a sequence of zeros and ones, and when using DNA strands this information is translated into the four different bases that are presented in DNA – adenine (A), thymine (T), cytosine (C), and guanine (G).

Molecular Computing

Leonard Adleman [Adleman, 1994], an inspired mathematician, began the research in this area by an experiment using the tools of molecular biology to solve a hard computational problem in a laboratory. That was the world’s first DNA computer. A year later Richard J.Lipton [Lipton, 1995] wrote a paper in which he discusses, in detail, many operations that are useful in working with a molecular computer. After this moment many others followed them and started working on this new way of computing.

Adleman's experiment solved the travelling salesman problem (TSP). The problem consists on a salesman who wants to find, starting from a city, the shortest possible trip through a given set of customer cities and to return to its home town, visiting exactly once each city. TSP is NP-Complete (these kinds of problems are generally believed cannot be solved exactly in polynomial time. Lipton [Lipton, 1995] showed how to use some primitive DNA operations to solve any SAT problem (satisfiability problem) with N binary inputs and G gates (AND, OR, or NOT gates). This is also a NP-Complete problem.

Here a short description of the tool box of techniques for manipulating DNA is provided so that the reader can have a clear intuition about the nature of the techniques involved.

- Strands separation:
  - Denaturation of DNA strands. Denaturation of DNA is usually achieved by heat treatment or high pH, which causes the double-stranded helix to dissociate into single strands.
  - According to their length using gel electrophoresis. This technique is used to push or pull the molecules through a gel matrix by applying an electric current. The molecules will move through the matrix at different rates depending on their size.
  - According to a determinated subchain using complementary probes anchored to magnetic beads.
- Strands fusing:
  - Renaturation. If the soup is cooled down again, the separated strands fuse again.
  - Hybridization. Originally it was used for describing the complementary base pairing of single strands of different origin (e.g., DNA with RNA).
- Cutting DNA. Using restriction enzymes which destroy internal phosphodiester bonds in the DNA.
- Linking (pasting) DNA. Molecules can be linked together by certain enzymes called ligases.
- PCR mutagenesis. To incorporate the primer as the new (mutant) sequence.

Genetic Algorithms

Genetic Algorithms are adaptive search techniques which simulate an evolutionary process like it is seen in nature based on the ideas of selection of the fittest, crossing and mutation. GAs follow the principles of Darwin's theory to find the solution of a problem. The input of a GA is a group of individuals called initial population. The GA following Darwin's theory must evaluate all of them and select the individuals who are better adapted to the environment. The initial population will develop thanks to crossover and mutation.

John Holland [Holland, 1975] was the first one to study an algorithm based on an analogy with the genetic structure and behavior of chromosomes. Genetic algorithms has been widely studied and experimented. The structure of a basic genetic algorithm includes the following steps. (1) Generate the initial population and evaluate the fitness for each individual, (2) select individuals, (3) cross and mutate selected individuals, (4) evaluate and introduce the new created individuals in the initial population. In that way, the successive generation will become more suited to their environment.

Before generating the initial population, individuals need to be coded. That is the first thing to be done when deal with a problem so that it can be made combinations, duplications, copies, quick fitness evaluation and selection.

Get the solution faster: island model

The next method (island model) is followed in order to get the solution faster. It represents an upgrade of a simple genetic algorithm. In this model, the initial population is duplicated as many times as we want creating a fully connected graph. Each of the populations exchanges a portion of individuals (*m*) with the others. The graph is presented on fig.1.

Each of the populations exchanges a portion of individuals (*m*) with the others. This process is repeated every generation until a common equilibrium frequency is reached. By this method the speed of the GA is increased exponentially. Next figure shows how quickly populations converge on the same allele frequency when 10% (m = 0,1) of each population is made up of immigrants from the other populations.

In the figure 2 we see the change of allele frequency when using five subpopulations exchanging migrants at the rate m= 0,1 per generation. It is reached a final frequency (p) for all subpopulations which depart from the initial frequency (p_0) [Hartl and Clark, 1989].

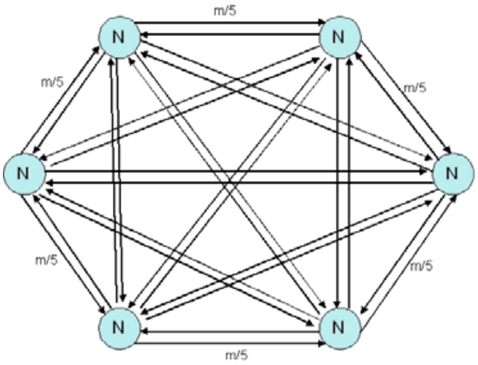


Fig. 1. Fully connected graph
for an initial population of N individuals.
The initial population is cloned six times.

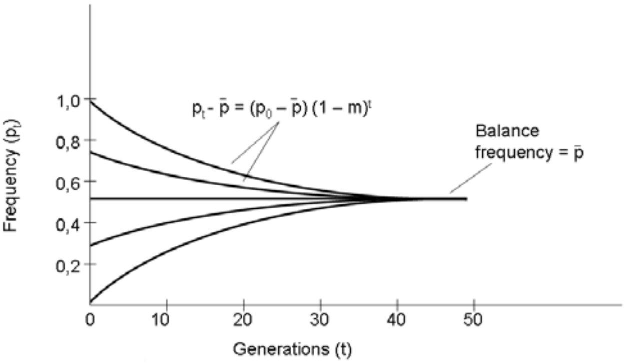


Fig. 2. Change of allele frequency using 5 subpopulations
exchanging migrants at the rate m= 0,1 per generation

Fitness and selection

When solving TSP, each possible solution to the problem (each individual of the initial population) is represented in a single DNA strand [M.Calviño, 2006]. Their form is the next:

PCR-primer Np Rep XY RE0 XY RE1 … REn-1 XY Rep Np-1 PCR-primer
XY (gene) is better evaluated as more C+G content

In this way the individuals are already evaluated. Once they are evaluated we must select them. By isopycnic centrifugation we can select the best suited to their environment. This technique is used to isolate DNA strands basing on the concentration of Cytosine and Guanine they have. The relationship between this concentration and the density (θ) of the strand is:

$$\Theta = 0{,}100[\%(G+C)] + 1{,}658$$

To begin the analysis, the DNA is placed in a centrifuge for several hours at high speed to generate certain force. The DNA molecules will then be separated based primarily on the relative proportions of AT (adenine and thymine base pairs) to GC (guanine and cytosine base pairs), using θ to know that proportion [Gerald Karp, 2005]. The molecule with greater proportion of GC base pairs will have a higher density while the molecule with greater proportion of AT base pairs will have a lower density. In this way the different individuals (different paths or solutions of TSP) are separated and can be easily selected. See figure 3 to understand how centrifugation works.

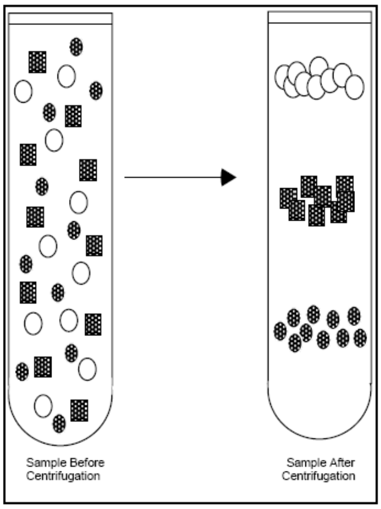


Fig. 3. Isopycnic centrifugation.

Crossover

As it has already been said the first thing to do when a problem is presented is the codification of individuals. In our case the problem is TSP. How can we code the paths? A possible solution was provided in a previous work [J.Castellanos, 1998] giving a representation of individuals like a DNA-strand for each path. This encoding is based on a sequence of genes each one represents an arch between two cities. Here the fitness would be an extra field placed at the beginning of the DNA-strand and its length is proportional to the value it represents (in fact it depends on the problem. For example in the travelling salesman problem, TSP, the length should be inversely proportional to the value of the path). Between the DNA code belonging to the genes a cutting site for a restriction enzyme will be inserted. The final encoding for a path is:

PCR Primer Np REp **Fitness** RE1 gene REn-1 …… RE0 gene REp Np PCR Primer

A recent work approached individual encoding by eliminating the field fitness. In that case, the fitness is embedded in the genes. The advantage of this work is that when all the individuals of the population are generated, there is no need to evaluate them because they have already been evaluated by themselves. After solving the problem of the selection by adding a specific field in each gene which tells the distance between both cities, it is necessary to see if the same format of the strands is valid in the next step of de GA, crossover.

First of all let's try to solve this step using the technique "cut and splice" like it is done in vivo. A single cut point is selected and after cut we splice both ends. An example is shown in figure 4 with two different chromosomes.

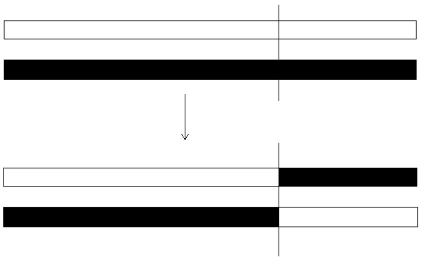
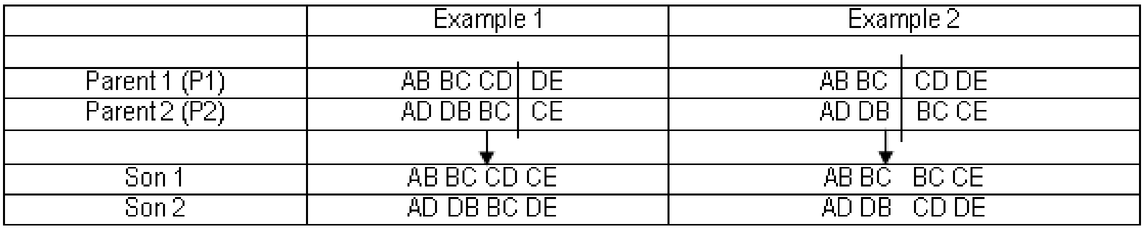


Fig 4

Solving TSP crossover with cut and splice:



	Example 1	Example 2
Parent 1 (P1)	AB BC CD DE	AB BC CD DE
Parent 2 (P2)	AD DB BC CE	AD DB BC CE
Son 1	AB BC CD CE	AB BC BC CE
Son 2	AD DB BC DE	AD DB CD DE

Table 1

The results show that this method must be discarded because all the sons it produces are invalid. Obviously, it has no sense to create a son that contains a specific city twice.

I proceed then to apply a different protocol called "order crossover" (OX). Adjacency information in the total ordering is important and this crossover preserves relative ordering. Two parents are selected between the population then a random mask is selected (with 0's and 1's which are chosen randomly with the same probability both bits). During the first step, the sons are filled with the genes of the parents which the mask allows. To complete the sons, we put the genes missing in S1 in the order they appear in P2 and the same with S2. An example is shown in Table 2.

Suitable mask for order crossover

Let's try to apply OX to TSP. It is remembered that each gene represents an arch between two different cities. Like a first attempt it is used the mask: 1001 (one bit for gene).

As we see in Table 3, the result of the computation of OX using that mask is invalid because the genes CD and DB do not even exist in P2 so S1 cannot be completed. By using this mask we will never get valid individuals. Now, OX is computed with the mask 10 01 01 11, using two bits for each gene. Each bit represents a city. As usually, the mask is chosen randomly (every bit).

Once again the mask is not correct. In the example (Table 4) we can see how in the third gene of P1 (DB) there is missing only one city, city D. That has no sense at all, because the second city of the second gene must be the same as the first city of the third gene. That give us the idea of how the definitive mask should be. Let's try now with the mask 10 01 10 01. In this mask we choose randomly the pair of bits that represents the same city, for example we choose if the second bit of the first gene (1**0**) and the first bit of the second gene (**0**1) are 0 or 1 both of them but not different.

In this example (Table 5) the sons are correct. So that is the suitable mask. In order to find less invalid individuals we force the mask to one last rule: the first bit and the last must be 1 because in TSP the first city we visit and the last one must be always the same.

P1	A C D B E
P2	A D B C E
Mask	1 0 1 0 1
S1 (C, B missing)	A - D - E
S2 (D, C missing)	A - B - E

S1 (B before C in P2)	A B D C E
S2 (C before D in P1)	A C B D E

Table 2

P1	AC CD DB BE
P2	AD DB BC CE
Mask	1 0 0 1
S1 (missing CD, DB)	AC - - BE

Table 3

P1	AC CD DB BE
P2	AD DB BC CE
Mask	10 01 01 11
S1 (missing C(twice),D)	A- -D -B BE

Table 4

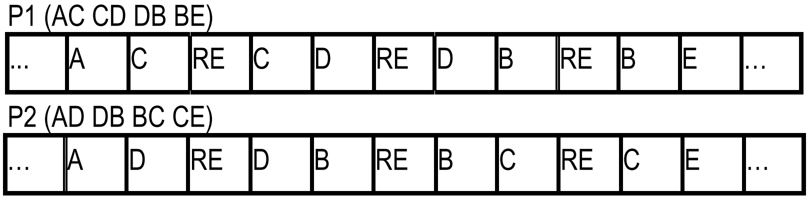
P1	AC CD DB BE
P2	AD DB BC CE
Mask	10 01 10 01
S1 (C, B missing)	A - -D D- -E
S2 (D, C missing)	A - -B B- -E

S1 (B before C in P2)	AB BD DC CE
S2 (C before D in P1)	AC CB BD DE

Table 5

Translating order crossover to DNA computing

How can be translated into DNA computing the previous crossover operator? Firstly, imagine that we have in a test tube the individuals that we had before but in the encoding which is explained above, representing each individual like a sequence of genes in a DNA strand. That is shown in Fig. 5.



P1 (AC CD DB BE)

...	A	C	RE	C	D	RE	D	B	RE	B	E	...

P2 (AD DB BC CE)

...	A	D	RE	D	B	RE	B	C	RE	C	E	...

Fig. 5

When the problem of representing the mask is tackled a different view of the mask is given. This mask is suitable for our problem (TSP) using DNA computing and is obtained by following these steps:

1. Imagine that in our problem we have 5 cities: A B C D E.
2. To be discarded are the initial city and the final city. Then we have: B C D.
3. Randomly we choose one or several cities. For example: C D
4. Introduce in the soup (into the test tube) the following strands:

$\overline{C}$	(*)	$\overline{C}$

$\overline{D}$	(*)	$\overline{D}$

This represents the complementary bases of the cities C and D so that when introduced in the soup they can match the original strands there were in the soup before. See Fig. 6.

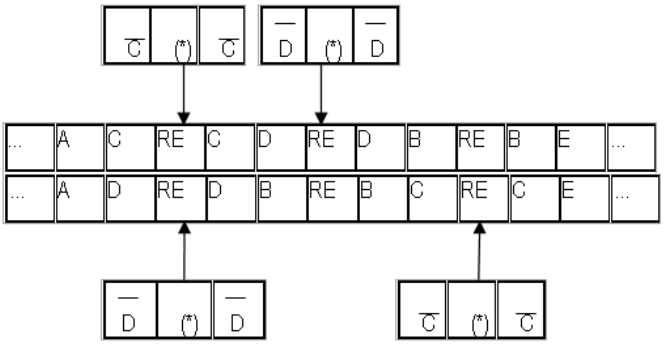

C (*) C
D (*) D
... A C RE C D RE D B RE B E ...
... A D RE D B RE B C RE C E ...
D (*) D
C (*) C

Fig 6

Computing crossover of Fig. 6:

P1	AC CD DB BE
P2	AD DB BC CE
Mask	C and D
S1 (C, D missing)	A - -- -B BE
S2 (D, C missing)	A - -B B- -E

S1 (D before C in P2)	AD DC CB BE
S2 (C before D in P1)	AC CB BD DE

Table 6

As a result of all this steps the crossover operator has changed a lot. Now it is still order crossover but with a very particular way of choosing the genes that must be changed. Instead of the initial mask we saw in Table 2, the mask it is used now consists on a selection of which cities (not genes) of P1 must be changed in the order they are found in P2. In the example shown in Table 5 the mask (10 01 10 01) showed the position of the cities that should be changed by the crossing operator. Now the mask for the same example (C, D) doesn't tell the position but de name of the cities to change.

However, if we try to apply this crossover operator in a genetic algorithm which uses the individual encoding that M.Calviño presented [M.Calviño, 2006] there is a big problem found. Spouse we have the gene AFC, in which F means the fitness between cities A and C. If we try to carry out the crossover operation of Fig 3, city C must be changed by D and then the gene would be AFD. Obviously F is not the fitness between A and D so this crossover operator only works with the strand-format proposed by J.Castellanos [J.Castellanos, 1998] though the other one works much better in the previous step of the GA, evaluation and selection.

Conclusion

The most important problem of DNA computers is resolved in this paper. This problem is the exponentially size algorithms the first DNA computer had. Genetic algorithms allow us to solve NP-Complete problems without exploring all the possible solutions. In GA a population of candidate solutions (individuals) to an optimization problem, like TSP, evolves toward better solutions. The "island model" of population dynamics can make the process faster meanwhile protocols of selection and crossover are presented in the paper.

The problem of crossover in a genetic algorithm using DNA has been resolved satisfactorily. Although the crossover technique might be different depending on the problem to be solved, it has been proved that it is possible to find a suitable crossover for NP-Complete problems such as TSP. This represents a new approach to the simulation of genetic algorithms with DNA.

Since the beginning of DNA computing, the lack of algorithms to be applied to this scientific area has been very large. Until recently, molecular computation has used "brute force" to solve NP-Complete problems. That is why the simulation of concepts of genetic evolution with DNA will help DNA computing to resolve hard computations. The crossover operator I have presented here gives an idea of how important and useful genetic algorithms are for DNA computing.

Bibliography

[Adleman, 1994] Leonard M. Adleman. Molecular Computation of Solutions to Combinatorial Problems. Science (journal) 266 (11): 1021-1024. 1994.

[Adleman, 1998] Leonard M. Adleman. Computing with DNA. Scientific American 279: 54-61. 1998

[Lipton, 1995] Richard J.Lipton. Using DNA to solve NP-Complete Problems. Science, 268:542-545. April 1995

[Holland, 1975] J.H.Holland. Adaptation in Natural and Artificial Systems. MIT Press. 1975.

[J.Castellanos, 1998] J.Castellanos, S.Leiva, J.Rodrigo, A.Rodríguez Patón. Molecular computation for genetic algorithms. First International Conference, RSCTC'98.

[M.Calviño, 2006] María Calviño, Nuria Gómez, Luis F.Mingo. DNA simulation of genetic algorithms: fitness computation.

[Macek, 1997] Milan Macek M.D. Denaturing gradient gel electrophoresis (DGDE) protocol. Hum Mutation 9: 136 1997.

[Dove, 1998] Alan Dove. From bits to bases; Computing with DNA. Nature Biotechnology. 16(9):830-832; September 1998.

[Mitchell, 1990] Melanie Mitchell. An Introduction to Genetic Algorithms. MIT Press, Boston. 1998.

[Lee, 2005] S.Lee, E. Kim. DNA Computing for efficient encoding of weights in the travelling salesman problem. ICNN&B'05. 2005.

[SY Shin, 2005] SY Shin, IH Lee, D Kim, BT Zhang. Multiobjective evolutionary optimization of DNA sequences for reliable DNA computing. IEEE Transactions, 2005.

[Gerald Karp, 2005] Gerald Karp.Cell and molecular biology: Concepts and experiments, 2005, Von Hoffman press

[Ayala, 1984] F.J. Ayala, J.A. Kiger. Modern genetics (2nd edition).

[Crow, 1986] J.F. Crow. Basic concepts in population, quantitative, and evolutionary Genetics. W.H. Freeman and Co. New York.

[Hartl, 1989] D.L. Hartl, A.G. Clark. Genetics of populations. Science Books International. Boston.

[Ford, 1991] T.C. Ford, J.M. Graham. An introduction to centrifugation. Bios Scientific Publishers. Oxford.

[Wilson, 1986] K. Wilson, K.H. Goulding. Principles and Techniques of Practical Biochemistry. Arnold LTD. Suffolk.

Authors' Information

Ángel Goñi Moreno *– Natural Computing Group. Universidad Politécnica de Madrid, Boadilla del Monte, 28660 Madrid, Spain: e-mail: ago@alumnos.upm.es*

A FRAMEWORK FOR FAST CLASSIFICATION ALGORITHMS

Thakur Ghanshyam, Ramesh Chandra Jain

Abstract*: Today, due to globalization of the world the size of data set is increasing, it is necessary to discover the knowledge. The discovery of knowledge can be typically in the form of association rules, classification rules, clustering, discovery of frequent episodes and deviation detection. Fast and accurate classifiers for large databases are an important task in data mining. There is growing evidence that integrating classification and association rules mining, classification approaches based on heuristic, greedy search like decision tree induction. Emerging associative classification algorithms have shown good promises on producing accurate classifiers. In this paper we focus on performance of associative classification and present a parallel model for classifier building. For classifier building some parallel-distributed algorithms have been proposed for decision tree induction but so far no such work has been reported for associative classification.*

Keywords: *classification, association, and data mining.*

1. Introduction

Data mining algorithms task is discovering knowledge from massive data sets. Building classifiers is one of the core tasks of data mining. Classification generally involves two phases, training and test. In the training phase the rule set is generated from the training data where each rule associates a pattern to a class. In the test phase the generated rule set is used to decide the class that a test data record belongs to. Traditionally, greedy search techniques such as decision trees [8] and others are used to develop classifiers. Decision Tree Induction approaches have been preferred to other traditional techniques due to the generation of small rule set and transparent classifiers. Transparent classifier means that rules are straightforward and simple to understand, unlike some opaque classifiers, such as one generated by neural networks, where interpretation of rules is difficult. Greedy techniques in decision tree construction approaches tend to minimize overlapping between training data records to generate small rule sets. However small rule sets have some disadvantages. Greedy techniques may achieve global optimality if the problem has a optimal substructure. A novel technique of associative classification based on association rule mining searches globally for all rules that satisfy minimum support and count thresholds [5].

2. Work already done in the field.

Several methods for improving the efficiency of all approach have been proposed [4,5,7,9,10] based on a recursive method for constructing a decision tree. In associative classification the classifier model is composed of a particular set of association rules, in which consequent of each rule is restricted to classification class attribute. The experiments in [4,5,7] show that this approach achieves higher accuracy than traditional approaches. Many sequential algorithms have been proposed for associative classification [4,5,7,9,10]. However associative classification suffers from efficiency due to the fact that it often generates a very large number of rules in association rule mining and it also takes efforts to select high quality rules from among them [7].

Since data mining is mostly applied on databases, which are very large, to improve the performance parallel algorithms are needed. Many parallel approaches have been given for association rule mining [11] and decision tree classifiers [3], no parallel algorithm has been proposed for associative classification. In this paper a model is proposed for parallel approach of associative classification for significant performance improvement. We propose a parallel model for CBA [5] algorithm.

3. Proposed methodology

Parallel Approaches for Data Mining:

Since data mining is frequently applied over large datasets, performance of algorithms is of concern. Exploiting the inherent parallelism of data mining algorithms provide a direct solution to their performance lift. A classification of different approaches to parallel processing for data mining is presented in [3].

Parallel Approaches:

1.Task Parallelism

    1. Divide & Conquer
    2. Task Queue

Task-parallel algorithms assign portions of the search space to separate processors. The task parallel approaches can again be divided in two groups. The first group is based on a Divide and Conquer strategy that divides the search space and assigns each partition to a specific processor.

2.Data Parallelism

    1. Record Based
    2. Attribute Based

The second group is based on a task queue that dynamically assigns small portions of the search space to a processor whenever it becomes available. Data-parallel, approaches distribute the data set over the available processors. Data-parallel approaches are in two directions. A partitioning based on records will assign non-overlapping sets of records to each of the processors. Alternatively a partitioning of attributes will assign sets of attributes to each of the processors. Attribute-based approaches are based on the observation that many algorithms can be expressed in terms of primitives that consider every attribute in turn. If attributes are distributed over multiple processors, these primitives may be executed in parallel. Many other issues on parallel processing of data mining with respect to Association Rule mining have been presented in [11].

The main challenges include synchronization and communication minimization, workload balancing, finding good data layout, data decomposition, and disk I/O minimization.

The parallel design space spans three main components:

1. The hardware platform,
2. The type of parallelism,
3. The load-balancing strategy.

Two dominant approaches for using multiple processors have emerged:

Distributed memory (where each processor has a private memory)

In distributed-memory (DMM) architecture, each processor has its own local memory and independent hard disk, which only that processor can access directly. For a processor to access data in the local memory of another processor, message passing must send a copy of the desired data elements from one processor to the other. A distributed memory, message-passing architecture cures the scalability problem by eliminating the bus, but at the expense of programming simplicity

Shared memory (where all processors access common memory).

Shared-memory (SMP) architecture has many desirable properties. Each processor has direct and equal access to all the system's memory. Parallel programs are easy to implement on such a system. A different approach to multiprocessing is to build a system from many units, each containing a processor and memory. Although shared memory architecture offers programming simplicity, a common bus's finite bandwidth can limit scalability. Load balancing strategies entail static or dynamic approaches. Static load balancing initially partitions work among the processors using a heuristic cost function, no subsequent data or computation movement is available. Dynamic load balancing seeks to address this by taking work from heavily loaded processors and reassigning it to lightly loaded ones. Computation movement also entails data movement, because the processor responsible for a computational task needs the data associated with that task. Dynamic load balancing thus incurs additional costs for work and data movement, and also for the mechanism used to detect whether there is an imbalance. However, dynamic load a balancing is essential if there is a large load imbalance or if the load changes with time. Dynamic load balancing is especially important in multi-user environments with transient loads and in heterogeneous platforms, which have different processor and network speeds. These kinds of environments include parallel servers and heterogeneous clusters, meta-clusters, and super-clusters (the so called grid platforms that are becoming common today).There are various approaches that can be applied to parallel processing of data mining algorithms. Cost measures for various parallel data mining strategies to predict their computation, data access and communication performance are presented in [6].

I. Associative Classification

Associative Classification is an integrated framework of Association Rule Mining (ARM) and Classification. Focusing on a special subset of association rules whose right-hand-side is restricted to the classification class attribute does the integration. This subset of rules is referred as the Class Association Rules (CARs). CBA (Classification Based on Associations) [5] is a sequential approach of building associative classifier. CBA consists of two parts, a rule generator (called CBA-RG), which is based on algorithm Apriori for finding association rules in [2], and a classifier builder (called CBA-CB). CBA approach is described below. Assuming given dataset is a normal relational table, which consists of N cases described by l distinct attributes. These N cases have been classified into q known classes. For Associative Classification it is assumed that in training data set all continuous attributes (if any) have been discretized as a preprocessing step. For all attributes, all the possible values are mapped to a set of consecutive positive integers. With these mappings, a data case can be treated as a set of (*attribute*, *integer-value*) pairs and a class label. Each (*attribute*, *integer-value*) pair is called an *item.* Let D be the dataset. Let I be the set of all items in D, and Y be the set of class labels. We say that a data case $d \in D$ contains $X \subseteq I$, a subset of items, if $X \subseteq d$. A classification rule (CAR) is an implication of the form $X \rightarrow y$, where $X \subseteq I$, and $y \in Y$. A rule $X \rightarrow y$ holds in D with confidence c if $c\%$ of cases in D that contain X are labeled with class y. The rule $X \rightarrow y$ has support s in D if $s\%$ of the cases in D contain X and are labeled with class y.

Rule Generator CBA-RG

The key operation of CBA-RG is to find all ruleitems that have support above minsup. A ruleitem is of the form: <condset, y>, where condset is a set of items, $y \in Y$ is a class label. The support count of the condset (called condsupCount) is the number of cases in D that contain the condset. The support count of the ruleitem (called rulesupCount) is the number of cases in D that contain the condset and are labeled with class y. Each ruleitem basically represents a rule: condset → y, whose support is (*rulesupCount* / |D|) *100%, where |D| is the size of the dataset, and whose confidence is(*rulesupCount / condsupCount*)*100%. *Ruleitems* that satisfy minsup are called *frequent ruleitems*, while the rest are called *infrequent ruleitems*. For example, the following is a *ruleitem*: <{(A, 1), (B, 1)}, (class,1)>, where A and B are attributes. If the support count of the condset {(A, 1), (B, 1)} is 3, the support count of the ruleitem is 2, and the total number of cases in D is 10, then the support of the ruleitem is 20%, and the confidence is 66.7%. If minsup is 10%, then the ruleitem satisfies the minsup criterion. We say it is frequent. For all the ruleitems that have the same condset, the ruleitem with the highest confidence is chosen as the possible rule (PR) representing this set of *ruleitems*. If there are more than one *ruleitem* with the same highest confidence, we randomly select one *ruleitem*. For example, we have two *ruleitems* that have the same *condset*:

1. <{(A, 1), (B, 1)}, (class: 1)>.
2. <{(A, 1), (B, 1)}, (class: 2)>.

Assume the support count of the *condset* is 3. The support count of the first *ruleitem* is 2, and the second *ruleitem* is 1. Then, the confidence of *ruleitem* 1 is 66.7%, while the confidence of *ruleitem* 2 is 33.3% With these two *ruleitems*, we only produce one PR (assume |*D*| = 10): (A, 1), (B, 1)_(class, 1) [*support* = 20%, *confidence*= 66.7%]. If the confidence is greater than *minconf*, we say the rule is *accurate*. The set of *class association rules* (CARs) thus consists of all the PRs that are both frequent and accurate.

The CBA-RG algorithm generates all the frequent *ruleitems* by making multiple passes over the data. In the first pass, it counts the support of individual *ruleitem* and determines whether it is frequent. In each subsequent pass, it starts with the seed set of *ruleitems* found to be frequent in the previous pass. It uses this seed set to generate new possibly frequent *ruleitems*, called *candidate ruleitems*. The actual supports for these candidate *ruleitems* are calculated during the pass over the data. At the end of the pass, it determines which of the *candidate ruleitems* are actually frequent. From this set of frequent *ruleitems*, it produces the rules (CARs). Let *k-ruleitem* denote a *ruleitem* whose *condset* has *k* items. Each element F_k of this set is of the following form: <(*condset*, *condsupCount*), (*y*, *rulesupCount*)>.

The CBA-RG algorithm is given in Figure 1.

```
$F_1$ = {large 1-ruleitems};
$CAR_1$ = genRules($F_1$);
for ($k$ = 2; $F_{k-1} \neq \phi$ ; $k$++) do
  $C_k$ = candidateGen($F_{k-1}$);
  for each data case $d \in D$ do
    $C_d$ = ruleSubset($C_k$, $d$);
    for each candidate $c \in C_d$ do
      c.condsupCount++;
      if d.class = c.class then c.rulesupCount++;
    end
  end
  $F_k$ = {$c \in C_k$ | c.rulesupCount >=minsup};
  $CAR_k$ = genRules($F_k$);
end
CARs = $\cup_k CAR_k$;
```

Figure 1

II.Building a Classifier CBA-CB

To produce the best classifier out of the whole set of rules, a heuristic approach is used. A total order on the generated rules is defined. For more details on CBA approach readers are referred to [5]. This is used in selecting the rules for classifier.

```
R = sort(R); // sort on precedence $\succ$
for each rule $r \in R$ in sequence do
   temp = $\phi$ ;
   for each case $d \in D$ do
      if d satisfies the conditions of r then
         store d.id in temp and mark r if it correctly
               classifies d;
   if r is marked then
      insert r at the end of C;
      delete all the cases with the ids in temp from D;
      selecting a default class for the current C;
      compute the total number of errors of C;
   end
end
Find the first rule p in C with the lowest total number
of errors and drop all the rules after p in C;
Add the default class associated with p to end of C;
return C (our classifier).
```

Figure 2. The CBA-RG : M1 algorithm

Definition: Given two rules, r_i and r_j, $r_i \succ r_j$ (also called r_i precedes r_j or *ri* has a higher precedence than r_j) if *1)*. the confidence of r_i is greater than that of r_i, or *2)*. their confidences are the same, but the support of r_i is greater than that of r_j, or 3. both the confidences and supports of r_i and r_j are the same, but r_i is generated earlier than r_j;

Let *R* be the set of generated CARs and *D* the training data. The basic idea of the algorithm is to choose a set of high precedence rules in *R* to cover *D*.

Our classifier is of the following format: <$r_1, r_2, r_3, \ldots, r_n$, *default_ class* >, where $r_i \in R$, $r_a \succ r_b$ if *b*> *a*. *default_class* is the default class. In classifying an unseen case, the first rule that satisfies the case will classify it. If there is no rule that applies to the case, it takes on the default class as in C4.5. A pseudo code of algorithm M1 for building such a classifier is shown in Figure 2.

III. Parallel and Distributed Associative Classification

CBA is an associative classification algorithm that uses an Apriori based approach to mine CARs and produces a subset of these CARs after pruning to form a classifier. We adapt here popular CBA algorithm discussed in section 3 to present our approach of parallel associative classification. For both phases of associative classification, rule generation phase and classifier builder phase our approach is based on Distributed Memory Systems, Record Based data parallelism and uses Static Load Balancing. The above configuration of parallel approach suits to the inherent parallel nature of existing serial approach. Three parallel versions of Apriori are given in [1] on shared nothing architecture. We adapt count distribution algorithm of ARM mining for mining of CARs in associative classification and present parallel version of CBA-M1 for classifier building. Count distribution approach has minimized communication among the processors. For CARs mining the training data set is partitioned among *P* processors. Each processor works on its local partition of the database and performs same set of instructions to mine CARs that have global min support and confidence. Later when all CARs are found, same partitions of training set are used in respective nodes and pruning process based on coverage analysis is applied in parallel to generate reduced set of CARs, to form classifier. Our approach simply achieves load balancing if training data sets are sufficiently randomly distributed over different processors to avoid any data skew.

C_k	Set of candidate *k-ruleitems*
F_k	Set of frequent *k-ruleitems*
D	Training Dataset
D_i	Local Training Dataset on i^{th} Processor
P_i	i^{th} Processor
R	Set of generated *CARs*

Figure 3. Notations

```
do in parallel
  k = 2;
  while ($F_{k-1} \neq \phi$) do
    for each processor $P_i$ (i=1..p) do
      $C_k$ = candidateGen($F_{k-1}$);
      for each data case $d \in D$ do
        $C_d$ = ruleSubset($C_k$, d);
        // compute local support for ruleitems
        for each candidate $c \in C_d$ do
          c.condsupCount++;
          if d.class = c.class then c.rulesupCount++
        end
      end
      exchange local $C_k$ counts with all other processors
                                                  and synchoronize;
      for each $c \in C_k$   c.condsupCount=$\sum_{Pi(i=1..p)}$c.condsupCount;
        c.rulesupCount =$\sum_{Pi\ (i=1..p)}$c.rulesupCount;
      end
      $F_k$ = {$c \in C_k$ | c.rulesupCount >=minsup};
      $CAR_k$ = genRules($F_k$);
      k++;
  end while
  CARs = $\cup$ $CAR_k$
End parallel
```

Figure 4

It can simply be inferred

$|D|$ = *N* and number of processors = *p*

$|D_i|$ = $|D|/p$ (approximately), *i=1,2,…p*

The necessary communication among processors is through message broadcasting. There has been no need of dynamic load balancing as the Associative Classifier builder task does not involve multi-user environment with changing training data sets during learning. As in ARM in associative classification too the static load balancing is inherent in the partitioning of the database among processors because training data sets have been made available in a homogeneous environment. Parallel versions adapted from CBA for CAR generation and classifier builder are presented below.

Pseudo codes given in Figure 4 use notations given in Figure 3.

Parallel CAR generation phase

At the time of generating partitions D_i for processors $P_{i,}$ F_1 and hence CAR_1 can be generated and distributed to distributed processors along with data partitions. Hence algorithm begins with a seed of F1. During partitioning *class_distribution* i.e. number of data cases for each of the classes will also be computed and distributed to processors to be used during class builder phase. Pseudo code of parallel rule generation algorithm is presented in figure 4.

In Parallel CAR generation algorithm each processor independently and in parallel generates identical C_k for $k >1$ and calculates local counts and broadcast these to all other processors. At this step processors synchronize and wait for all other processors to compute and broadcast their local counts. Summing local counts of all processors global counts for C_k are computed. Each processor now computes F_k and generates CARs from F_k. Process is iterated for next k till until F_k is empty. At the end of rule generation algorithm each processor has complete set of CARs.

4. Cost Measures For Proposed Model

All this information exchanged is integer valued and its volume is very small.

Cost estimate of sequential CBA approach based on cost models in [6] is as follows. Global structure of both rule generation and class builder algorithms is a loop building more accurate concepts from those of the previous iterations. Suppose loop in rule generation algorithm and classifier builder algorithms executes $k_{s1,}k_{s2}$ times and builds $\Omega_{1,,}\Omega_2$ concepts respectively. Total size D is N and number of attributes in D is $l.$ The cost estimate of the sequential CBA algorithm can be given by formula

$$Cost_{seq} = k_{s1} \left[STEP(N^*l, \Omega_1) + ACCESS\ (N^*l) \right] + k_{s2} \left[STEP(N^*l, \Omega_2) + ACCESS\ (N^*l) \right]$$

Where *STEP* gives the cost of single iteration of the loop, and *ACCESS* is the cost of accessing the data set once.

If the CBA is performed in parallel version of rule generation algorithm and classifier builder algorithms and requires $k_{a1,}k_{a2}$ iterations respectively with number of p processors, formula for the cost can be given by

$$Cost_{par} = k_{a1} \left[STEP(N^*l/p, \Omega_1)+ACCESS\ (N^*l/p) + C_{e1}\right] + k_{a2} \left[STEP(N^*l/p, \Omega_2)+ACCESS\ (N^*l/p)+C_{e2}\right]$$

C_{e1}, C_{e2} are total cost of communication and information exchange between the processors.

It can be reasonably assumed that

$STEP(N^*l/p, \Omega_1) = STEP(N^*l, \Omega_1)/p$

and

$ACCESS\ (N^*l/p) = ACCESS\ (N^*l)/p$

So, we get significant p-fold speedup in executing parallel version except cost of overheads. In our model overhead cost is small as information exchanged is integer valued and its volume is very small.

5. Conclusion

In this paper the focus was on the performance of classifier builder approach known as associative classification. We proposed a model to show that associative classification task can be performed in parallel on distributed memory systems to achieve a significant performance lift. We have presented parallel versions for both ruled generation and class builder phase of sequential CBA algorithm for load balancing we have distributed almost equal number of data sets randomly on each of the local processors to avoid data skewness.

Bibliography

1. R. Agrawal and J. C. Shafer, "Parallel Mining of Association Rules", *IEEE Transactions On Knowledge And Data Engineering*, pages 962–969, 1996.
2. R. Agrawal and R. Srikant, "Fast algorithms for mining association rules", In *Proc. of the Int. Conf. on Very Large Databases*, SanDiago, Chile, pages 487–499, 1994.
3. J. Chattratichat, "Large Scale Data Mining: Challenges and Responses," *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining*, 1997, pp 143-146.
4. W. Li, J. Han and J. Pei, "CMAR: Efficient classification based on multiple class-association rules. In *Proc. of the Int. Conf. on Data Mining*, pages 369–376, 2001.
5. B. Liu, W. Hsu and Y. Ma, "Integrating classification and association rule mining", In *Knowledge Discovery and Data Mining*, pages 80–86, 1998.
6. D. B. Skillicorn, "Strategies for parallel data mining", *IEEE Concurrency*, vol. 7, No. 4,1999.
7. X. Yin and J. Han, "CPAR: Classification based on Predictive Association Rules", In *Proc. of the Int. Conf. on Data Mining*, SDM. SIAM, 2003.
8. F. Thabtah, P. Cowling and Y. Peng, "MCAR: Multi-class Classification based on Association Rule Approach. In *Proceeding of third IEEE International Conference on Computer Systems and Applications*, Cairo, Egypt, pages 1-7, 2005.
9. F. Thabtah, P. Cowling and Y. Peng, "MMAC: A New Multi-class, Multi-label Associative Classification Approach", In *Proceeding of fourth IEEE International Conference on Data Mining* (ICDM '04), Brighton, UK, pages 217-224, Nov. 2004.

Authors' Information

***Thakur S. Ghanshyam, Dr. R.C.Jain** – Department of Computer Application; Samrat Ashok Technological Institute; Vidisha(M.P.), INDIA; e-mail: ghanshyamthakur@gmail.com*

ENHANCING INFORMATION RETRIEVAL BY USING EVOLUTION STRATEGIES

Abdelmgeid Amin Aly

***Abstract:** Similar to Genetic algorithm, Evolution strategy is a process of continuous reproduction, trial and selection. Each new generation is an improvement on the one that went before. This paper presents two different proposals based on the vector space model (VSM) as a traditional model in information Retrieval (TIR). The first uses evolution strategy (ES). The second uses the document centroid (DC) in query expansion technique. Then the results are compared; it was noticed that ES technique is more efficient than the other methods.*

1. Introduction

Since the 1940s the problem of Information Retrieval (IR) has attracted increasing attention, especially because of the dramatically growing availability of documents. IR is the process of determining relevant documents from a collection of documents, based on a query presented by the user.

There are many IR systems based on Boolean, vector, and probabilistic models. All of them use their model to describe documents, queries, and algorithms to compute relevance between user's query and documents.

Information Retrieval (IR) proposes solutions for searching, in a given set of objects, for those replying to a given description. IR tries to make a suitable use of these databases, allowing the users to access to the information which is really relevant in an appropriate time interval [1]. Unfortunately, commercial IR Systems (IRSs), usually based on the Boolean IR model [2], have provided unsatisfactory results. Vector space, probabilistic and fuzzy models, which have been developed to extend the Boolean model [3], as well as the application of knowledge-based techniques, have solved some of these problems, but there are still some lacks [4]. In the last few years, an increasing interest on the application of artificial intelligence (AI)-based techniques to IR has been shown with the aim of solving some of those lacks.

One of the AI areas with a considerable growth in the last decades is evolutionary computation (EC) [5], based on the use of models of evolutionary process for the design and implementation of computer-based problem solving systems. The different models which have been proposed within this philosophy are named in a generic way as evolutionary algorithms (EAs) [5].

In this paper, we introduce two different proposals using our IR model. One is using Evolution Strategies (ES) and the second using the Document Centroid (DC) in Query Expansion (QE) technique. Then the results are compared with our Traditional IR (TIR) model. To this end, we used the ES that presented the best performance, running it with two different fitness functions in the vector space model which is the most commonly used model in this type of application. We applied it to three well-known test collections (CISI, CACM and NPL). This allows us to generalize our earlier results and conclusion.

2. Antecedents

2.1. Evolutionary algorithms

EC [5] uses computational models of evolutionary processes as key elements in the design and implementation of computer-based problem solving systems. There is a variety of evolutionary computational models that have been proposed and studied, which are referred as EAs [5]. There have been four well defined EAs which have served as the basis for much of the activity in the field: Genetic Algorithms (GAs) [6], Evolution Strategies (ES) [7], Genetic Programming (GP) [8] and Evolutionary Programming (EP) [9].

An EA maintains a population of trial solutions, imposes random changes to these solutions, and incorporates selection to determine which ones are going to be maintained in future generations and which will be removed from the pool of trials. There are some important differences between the existing EAs. GAs [6] emphasize models of genetic operators as observed in nature, such as crossover (recombination) and mutation, and these re appllied to abstracted chromosomes with different representation schemes according to the problem being solved. Evolution strategies and evolutionary programming only apply to real-valued problems and emphasize mutational transformations that maintain the behavioral linkage between each parent and its off-spring.

As regards GP [8], it constitutes a variant of GAs, based on evolving structures encoding programs such as expression trees. Apart from adapting the crossover and mutation operators to deal with the specific coding scheme considered, the remaining algorithm components remain the same.

2.2. Evolution Strategies

Evolution strategies (ESs) were independently developed by Rechenberg [10], with selection, mutation, and a population of size one. Schwefel [11], introduced recombination and populations with more than one individual, and provided a nice comparison of ESs with more traditional optimization techniques. Evolution strategies typically use real-valued vector representations. Evolution strategies are similar to genetic algorithms in that both attempt to find a (near-)optimal solution to a problem within a search space (all possible solutions to a problem) without exhaustively testing all solutions.

Evolution strategies are based on the principal of strong causality, which states that similar causes have similar effects. That is, a slight change to one encoding of a problem only slightly changes its optimality. The process of evolution strategy can be summarized by a relatively simple algorithm:

1. Generate some random individuals
2. Select the p best individuals based on some selection algorithm (fitness function)
3. Use these p individuals to generate c children (using mutation or recombination)
4. Go to step 2, until the ending condition is satisfied (i.e. little difference between generations, or maximum number of iterations completed).

2.3. Automatic Query Expansion

The automatic query expansion or modification based on term co-occurrence data has been studied for nearly three decades. The various methods proposed in the literature can be classified into the following four groups:

1. Simple use of co-occurrence data. The similarities between terms are first calculated based on the association hypothesis and then used to classify terms by setting a similarity threshold value [12], [13] and [14]. In this way, the set of index terms is subdivided into classes of similar terms. A query is then expanded by adding all the terms of the classes that contain query terms. It turns out that the idea of classifying terms into classes and treating the members of the same class as equivalent is too naive an approach to be useful [13], [15] and [16].
2. Use of document classification. Documents are first classified using a document classification algorithm. Infrequent terms found in a document class are considered similar and clustered in the same term class (thesaurus class) [17]. The indexing of documents and queries is enhanced either by replacing a term by a thesaurus class or by adding a thesaurus class to the index data. However, the retrieval effectiveness depends strongly on some parameters that are hard to determine [18]. Furthermore, commercial databases contain millions of documents and are highly dynamic. The number of documents is much larger than the number of terms in the database. Consequently, document classification is much more expensive and has to be done more often than the simple term classification mentioned in 1.
3. Use of syntactic context. The term relations are generated on the basis of linguistic knowledge and co-occurrence statistics [19], [20]. The method used grammar and a dictionary to extract for each term *t* a list of terms. This list consists of all the terms that modify *t*. The similarities between terms are then calculated by using these modifiers from the list. Subsequently, a query is expanded by adding those terms most similar to any of the query terms. This produces only slightly better results than using the original queries [19].
4. Use of relevance information. Relevance information is used to construct a global information structure, such as a pseudo thesaurus [21], [22] or a minimum spanning tree [23]. A query is expanded by means of this global information structure. The retrieval effectiveness of this method depends heavily on the user's relevance information. Moreover, the experiments in [23] did not yield a consistent performance improvement. On the other hand, the direct use of relevance information, by simply extracting terms from relevant documents, is proved to be effective in interactive information retrieval [24], [25]. However, this approach does not provide any help for queries without relevance information.

In addition to automatic query expansion, semi-automatic query expansion has also been studied [26], [27] and [28]. In contrast to the fully automated methods, the user is involved in the selection of additional search terms during the semi-automatic expansion process. In other words, a list of candidate terms is computed by means of one of the methods mentioned above and presented to the user who makes the final decision. Experiments with semi-automatic query expansion, however, do not result in significant improvement of the retrieval effectiveness [26]. We use a document centroid (DC) as the basis of our query expansion.

3. System Framework

3.1. Building IR System

The proposed system is based on Vector Space Model (VSM) in which both documents and queries are represented as vectors. Firstly, to determine documents terms, we used the following procedure:

- Extraction of all the words from each document.
- Elimination of the stop-words from a stop-word list generated with the frequency dictionary of Kucera and Francis [29].
- Stemming the remaining words using the porter stemmer that is the most commonly used stemmer in English [3], [30].

After using this procedure, the final number of terms was 6385 for the CISI collection, 7126 for CACM and 7772 for NPL. After determining the terms that described all documents of the collection, we assigned the weights by using the formula (1) which proposed by Salton and Buckley [25]:

$$a_{ij} = \frac{\left(0.5 + 0.5\,\frac{tf_{ij}}{\max tf}\right) \times \log \frac{N}{n_i}}{\sqrt{\left(0.5 + 0.5\,\frac{tf_{ij}}{\max tf}\right)^2 \times \left(\log \frac{N}{n_i}\right)^2}} \longrightarrow \tag{1}$$

where a_{ij} is the weight assigned to the term t_j in document D_i, tf_{ij} is the number of times that term t_j appears in document D_i, n_j is the number of documents indexed by the term t_j and finally, N is the total number of documents in the database.

Finally, we normalize the vectors, dividing them by their Euclidean norm. This is according to the study of Noreault et al. [31], of the best similarity measures which makes angle comparisons between vectors. We carry out a similar procedure with the collection of queries, thereby obtaining the normalized query vectors. Then, for applying ES, we apply the following steps:

- For each collection, each query is compared with all the documents, using the cosine similarity measure. This yields a list giving the similarities of each query with all documents of the collection.
- This list is ranked in decreasing order of similarity degree.
- Make a training data that consists of the top 15 document of the list with a corresponding query.
- Automatically, the keywords (terms) are retrieved from the training data and the terms which are used to form a query vector.
- Adapt the query vector using the ES approach.

3.2. The Evolution Strategy Approach

Once significant keywords are extracted from training data (relevant and irrevelent documents) including weights are assigned to the keywords. We have applied ES to get an optimal or near optimal query vector. Also we have compared the result of the ES approach with both the result obtained of (DC) and the traditional IR system. The (ES) approach will be explained in the following subsections.

Encoding & Fitness Functions

To implement an evolution strategy, the individuals in the population (solutions) need to be represented. Unlike genetic algorithms, which use bit strings, evolution strategies encode these individuals as vectors of real numbers (object parameters). Another vector of parameters, the strategy parameters, affects the mutation of the object parameters. Together, these two vectors constitute the individual's chromosome.

To distinguish whether one solution is more optimal than another, we use the cosine similarity as fitness function (2).

$$\frac{\sum_{i=1}^{t} x_i \cdot y_i}{\sqrt{\sum_{i=1}^{t} {x_i}^2 \cdot \sum_{i=1}^{t} {y_i}^2}} \longrightarrow \tag{2}$$

where X_i is the real representation weight of term *i* in the chromosome, Y_i is the real representation weight of that term in the query vector and *t* is the total number of terms in the query vector as in a given chromosome.

Forming the Next Generation

One key difference from genetic algorithms is that only the *p* most fit individuals in the population survive until the next generation (this form of selection is known as elitist selection). (Genetic algorithms usually use roulette wheel selection to give the fittest individuals a better chance of survival, but don't, like evolution strategies, guarantee that they will survive.) Using the fitness function as the evaluator, the *p* best individuals from the population are selected to be the parents of the next generation. A large value of *p* prevents bad characteristics from being filtered out of the gene pool (since they will persist from generation to generation), while a small value reduces variation in the gene pool, increasing the need for mutation. These *p* parent individuals produce a total of *c* children using mutation and recombination. The parents can be included in the next generation. Producing more children increases the probability of achieving better solutions, Mutation and recombination are used, but there are some Differences in how they are applied.

Mutation

To simulate mutation, random changes to the chromosome are made. These changes are necessary to add new genes to the gene pool; otherwise an optimal solution could not be reached if a necessary gene is absent.

Recombination

Recombination (also known as crossover) is the process where two or more parent chromosomes are combined to produce a child chromosome. Recombination is necessary in cases where each child is to have multiple parents, since mutation provides no mechanism for the "mixing" of chromosomes. We use a single point recombination, exchanges the weights of sub-vector between two chromosomes, which are candidate for this process.

Evolution Process

Figure 1 outlines a typical evolution strategy (ES). After initialization and evaluation, individuals are selected uniformly randomly to be parents. In the standard recombinative ES, pairs of parents produce children via recombination, which are further perturbed via mutation. Survival is deterministic and is implemented in one of two ways.

```
procedure ES; {
  t = 0;
  initialize population P(t);
  evaluate P(t);
  until (done) {
    t = t + 1;
    parent_selection P(t);
    recombine P(t)
    mutate P(t);
    evaluate P(t);
    survive P(t);
  }       }
```

Fig. 1. The evolution strategy algorithm

The first allows the best children to survive and replaces the parents with these children. The second allows the N best children and parents to survive. Like EP, considerable effort has focused on adapting mutation as the algorithm runs by allowing each variable within an individual to have an adaptive mutation rate that is normally distributed with a zero expectation. Unlike EP, however, recombination does play an important role in evolution strategies, especially in adapting mutation.

3.3. The Query expansion approach

After using the vector space model (VSM) to represent the user's query and the documents. Each document d_k in the document database is represented by a document vector $\overline{d_k}$, the system calculates the degree of similarity between the query vector $\overline{Q}$ and each document vector $\overline{d_k}$. Then, the system ranks the document according to their degrees of similarity with respect to the user's query from the largest to the smallest. Based on the relevant degree of relevant documents, get the top 15 documents for each query. The system considers each term appearing in any relevant document from the top 15 documents as a relevant term. The weight of each relevant term in each relevant document is calculated using formula (1). The average weight W_{avg} of each relevant term t_i is calculated as follows:

$$W_{avg} = \frac{\sum_{k=1}^{15} w_{ik}}{15} \quad \rightarrow \qquad \textbf{(3)}$$

where w_{ik} denotes the weight of relevant term t_i relevant document d_k. The result obtained from equation (3) represents the document Centroid (DC). The original and the additional terms together form the expanded query that is, consequently, used to retrieve documents, to get the result of the DC. We have three types of results, one for Traditional IR (TIR), second from adding the Document Centroid (DC), and third from the Evolution Strategies (ES).

4. Experimental Results

The test databases for our approaches are three well-known test collections, which are: the CISI collection (1460 documents on information science), the CACM collection (3204 documents on Communications), and finally the NPL collection (11,429 documents on electronic engineering). One of the principal reasons for choosing more than one test collection is to emphasize and generalize our results in all alternative test documents collections. The Experiments are applied on 100 queries chosen according to each query which does not retrieve 15 relevant documents for our IR system.

CACM Collection Results for 100 Queries

Table (1), and its corresponding graph represented in figure (2) both are using non-interpolated average Recall – Precision relationship. From this table we notice that ES gives a higher improvement than TIR with 21.35% and higher than DC with 39.6 % respectively as average values. The average number of terms of query vector before applying ES is 160.7 terms; these terms are reduced after applying ES to 16.83 terms, and increasing to 167.8 terms when applying DC approach.

CISI Collection Results For 100 Queries

Table (2), and its corresponding graph represented in figure (3) both are using non-interpolated average Recall – Precision relationship. From this table we notice that ES gives a higher improvement than TIR with 17.66% and less than DC with 20.6% respectively as average values. The average number of terms of query vector before applying ES is 509.61 terms; these terms are reduced after applying ES to 358.84 terms, and increasing to 514.9 when applying DC approach.

NPL Collection Results for 100 Queries:

Table (3), and its corresponding graph represented in graph (3) both are using non-interpolated average Recall – Precision relationship. From this table we notice that ES gives a higher improvement than TIR with 19.82% and higher than DC with 99.82% respectively as average values. The average number of terms of query vector before applying ES is 134.14 terms; these terms are reduced after applying ES to 16.8 terms, and increasing to 142.9 terms when applying DC approach.

Table (1): Experimental results on CACM Collection

Average Recall-Precision Relationship			
	Precision		
Recall	TIR	DC	ES
0.1	0.72267	0.774552	0.81134
0.2	0.41646	0.52837	0.50888
0.3	0.36936	0.442366	0.45776
0.4	0.24673	0.267359	0.31126
0.5	0.21268	0.109873	0.2632
0.6	0.15801	0.005216	0.20032
0.7	0.14291	0.005216	0.17607
0.8	0.10728	0.005216	0.141
0.9	0.08965	0.005216	0.12236
Average	0.27397	0.238154	0.33247

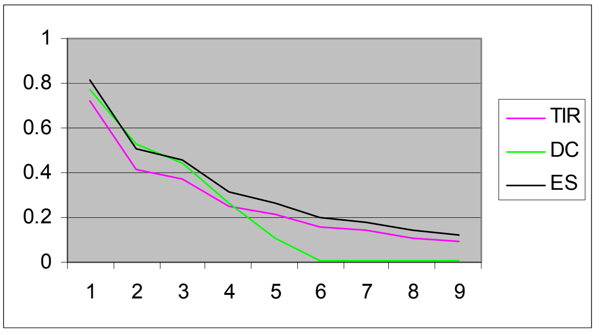


Fig. 2. The relationship between average recall-precision for 100 queries on CACM

Table (2): Experimental results on CISI Collection

Average Recall-Precision Relationship			
	Precision		
Recall	TIR	DC	ES
0.1	0.67935	0.83819	0.84987
0.2	0.55781	0.753699	0.66449
0.3	0.46199	0.732035	0.5962
0.4	0.4007	0.67232	0.46771
0.5	0.34937	0.612855	0.40763
0.6	0.30394	0.454284	0.31125
0.7	0.25167	0.355882	0.27429
0.8	0.19887	0.187812	0.20741
0.9	0.14908	0.150724	0.16604
Average	0.37253	0.528645	0.43832

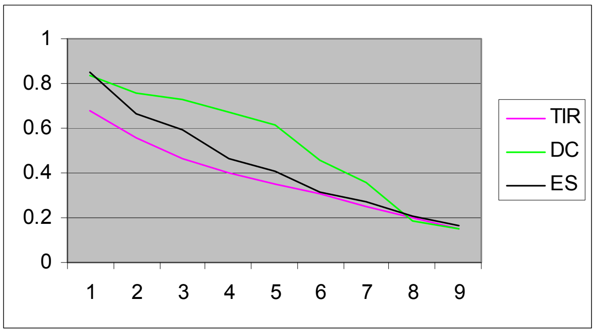


Fig. 3. The relationship between average recall-precision for 100 queries on CISI

Table (3): Experimental results on NPL Collection

Average Recall-Precision Relationship			
	Precision		
Recall	TIR	DC	ES
0.1	0.73292	0.886421	0.80875
0.2	0.50337	0.457369	0.56654
0.3	0.43515	0.303652	0.50423
0.4	0.34047	0.147124	0.4184
0.5	0.31333	0.053247	0.39739
0.6	0.23999	0.00282	0.31045
0.7	0.21539	0.00282	0.27597
0.8	0.17428	0.00282	0.23302
0.9	0.14555	0.00282	0.20027
Average	0.34449	0.206566	0.41278

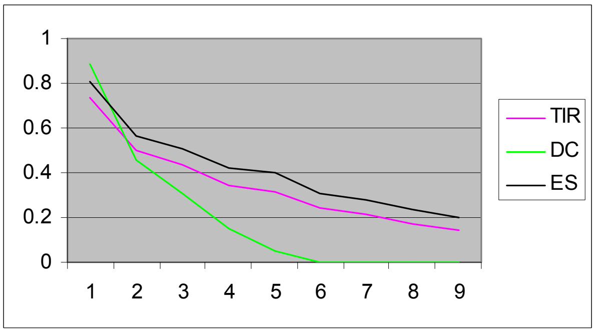


Fig. 4. The relationship between average recall-precision for 100 queries on NPL

5. Conclusion

The goal is to retrieve most relevant documents with less number of non-relevant documents with respect to user's query in information retrieval system using evolution strategies. Our results have been applied on three well-known test collections (CISI, CACM and NPL), and compare the results of three variant methods (TIR, DC and ES). The results demonstrate that evolution strategies are effective optimization technique for Document retrieval.

Bibliography

[1] G. Salton, M.H. McGill, Introduction to Modern Information Retrieval, McGraw-Hill, 1983.

[2] C.J. Van Rijsbergen, Information Retrieval, second ed., Butterworth, 1979.

[3] R. Baeza-Yates, B. Ribeiro-Neto, Modern Information Retrieval, Addison, 1999.

[4] H. Chen et al., "A machine learning approach to inductive query by examples: an experiment using relevance feedback", ID3, genetic algorithms, and simulated annealing, Journal of the American Society for Information Science 49 (8) (1998) 693–705.

[5] T. Bäck, D.B. Fogel, Z. Michalewicz, Handbook of Evolutionary Computation, IOP Publishing and Oxford University Press, 1997.

[6] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, Springer-Verlag, 1996.

[7] H.-P. GenerSchwefel, Evolution and Optimum Seeking, in: Sixth Generation Computer Technology Series, John Wiley and Sons, 1995.

[8] J. Koza, "Genetic Programming", On the Programming of Computers by means of Natural Selection, The MIT Press, 1992.

[9] D.B. Fogel, "System Identification trough Simulated Evolution: A Machine Learning Approach", Ginn Press, USA, 1991.

[10] I., Rechenberg, "Evolutions strategie: Optimierung Technischer Systeme nach Prinzipien der Biologischen Evolution", Frommann-Holzboog, Stuttgart (1973).

[11] H.-P., Schwefel, Numerical Optimization of Computer Models, New York: John Wiley & Sons (1981).

[12] M.E., Lesk, "Word-word association in document retrieval systems", American Documentation, 20(1): 27-38, 1969.

[13] J.,Minker, Wilson, G.A., Zimmerman, B.H., "An evaluation of query expansion by the addition of clustered terms for a document retrieval system", Information Storage and Retrieval, 8(6): 329-48, 1972.

[14] K., Sparck-Jones, E.B., Barber, "What makes an automatic keyword classification effective?", Journal of the ASIS, 18: 166-175, 1971.

[15] H.J.,Peat, P., Willett, "The limitations of term co-occurrence data for query expansion in document retrieval systems", Journal of the ASIS, 42(5): 378-83, 1991.

[16] K., Sparck-Jones, "Notes and references on early classification work". SIGIR Forum, 25(1): 10-17, 1991.

[17] C.J., Crouch, "An approach to the automatic construction of global thesauri", Information Processing & Management, 26(5): 629-40, 1990.

[18] C.J.,Crouch, B.,Yong, "Experiments in automatic statistical thesaurus construction", SIGIR'92, 15th Int. ACM/SIGIR Conf. on R&D in Information Retrieval, Copenhagen, Denmark, 77-87, June 1992.

[19] G., Grefenstette, "Use of syntactic context to produce term association lists for retrieval", SIGIR'92, 15th Int. ACM/SIGIR Conf. on R&D in Information Retrieval, Copenhagen, Denmark, 89-97, June 1992.

[20] G.,Ruge, "Experiments on linguistically-based term associations", Information Processing & Management, 28(3): 317-32, 1992.

[21] G., Salton, "Experiments in automatic thesaurus construction for information retrieval", Information Processing 71, 1: 115-123, 1971.

[22] G., Salton, "Automatic term class construction using relevance-a summary of work in automatic pseudo classification", Information Processing & Management, 16(1): 1-15, 1980.

[23] A.F., Smeaton, C.J., van Rijsbergen, "The retrieval effects of query expansion on a feedback document retrieval system", The Computer Journal, 26(3): 239-46, 1983.

[24] Harman, D., "Relevance feedback revisited", SIGIR'92, 15th Int. ACM/SIGIR Conf. on R&D in Information Retrieval, Copenhagen, Denmark, 1-10, June 1992.

[25] G., Salton, C. Buckley, "Improving Retrieval Performance by Relevance Feedback", Journal of the ASIS, 41(4): 288-297, 1990.

[26] F.C., Ekmekcioglu, A.M., Robertson, P., Willett, "Effectiveness of query expansion in ranked-output document retrieval systems", Journal of Information Science, 18(2): 139-47, 1992.

[27] M., Hancock-Beaulieu, "Query expansion: advances in research in on-line catalogues", Journal of Information Science, 18(2): 99-103, 1992.

[28] S.J., Wade, P., Willett, " INSTRUCT: a teaching package for experimental methods in information retrieval". III. Browsing, clustering and query expansion, Program, 22(1): 44-61, 1988.

[29] H. Kucera, N. Francis. “Computational analysis of present-day American English”. Providence, RD: Brown University Press (1967).

[30] M. F. Porter. “An algorithm for suffix stripping. Program”, 14(3), 130–137 (1980).

[31] T. Noreault, M. McGill and M. B. Koll. “A performance evaluation of similarity measures, document term weighting schemes and representation in a Boolean environment”. Information retrieval research. London: Butterworths (1981).

Author's Information

***A. A. Aly** – Computer Science Department, Minia University, El Minia, Egypt; Email: abdelmgeid@yahoo.com*

AGENT-BASED ANOMALIES MONITORING IN DISTRIBUTED SYSTEMS

Andrii Shelestov

Abstract. *In this paper an agent-based approach for anomalies monitoring in distributed systems such as computer networks, or Grid systems is proposed. This approach envisages on-line and off-line monitoring in order to analyze users' activity. On-line monitoring is carried in real time, and is used to predict user actions. Off-line monitoring is done after the user has ended his work, and is based on the analysis of statistical information obtained during user's work. In both cases neural networks are used in order to predict user actions and to distinguish normal and anomalous user behavior.*

Keywords: *security, distributed systems, agent approach, neural networks.*

ACM Classification Keywords*: K.6.5 Security and Protection – Authentication, I.2.6 Learning - Connectionism and neural nets, I.2.11 Distributed Artificial Intelligence - Multiagent systems.*

1 Introduction

Nowadays it is practically impossible to imagine different areas of human activity without the use of distributed systems, for example, corporate computer networks, Grid systems [1] for complex scientific problems solving, etc. However, it is evident that the work of many organizations (or set of organizations) considerably depends upon effective use of distributed systems resources and the level of their protection. Many problems, such as data storage, data transfer, information processing automation, complex problems solving are entrusted on them. The security level of information used in distributed systems can vary from private and business to military and state secret. The violation of information confidentiality, integrity and accessibility may have significant and undesirable consequences to its owner. Besides, many sources report that the majority (80%) of information security incidents is perpetrated by insiders (Microsoft Encyclopedia of Security, 2003) [2]. This means that internal computer users constitute the largest threat to the computer systems security.

Unfortunately, traditional methods (such as identification and authentication, access restriction, etc.) are not seemed to solve this problem at all. These rigorous and deterministic approaches possess some drawbacks; among them are low ability of internal malicious users detection, inability to process large amounts of information, low productivity, etc. That is why new approaches for users activity monitoring (including those relying on intelligent methods) are applied.

We may consider so called Personal Security Programs that are used by commercial companies to monitor the activity of their employees. The results of such monitoring can be used to reveal malicious users in the case of information leakage, or to find out whether users use computers for their personal purposes. For example, such programs as PC Spy (www.softdd.com/pcspy/index.htm), Inlook Express (www.jungle-monkey.com), Paparazzi (www.industar.net) allow to capture and save screen images (screenshots) showing exactly what was being viewed by users. All screens can be captured, including Web pages, chat windows, email windows, and anything else shown on the monitor. However, these programs have some disadvantages; among them are high volume of stored information and manual configuration of snapshots frequency.

Another example refers to Intrusion Detection Systems (IDS), particularly anomaly detection in computer systems. Usually, a model of normal user behavior is firstly created, so during monitoring any abnormal activity can be regarded as potential intrusion, or anomaly. Different approaches are applied to the development of anomaly detection systems: statistical methods [3], expert systems [4], finite automata [5], neural networks [6-8], agent-based systems [9], etc.

Generally, the development of monitoring system involves two phases: creation of user behavior model (normal or usual) and system implementation. First phase involves the following steps: data collection and data pre-processing, when useful information about user activity is collected from log-files; data processing, when feature extraction is made to data representation and dimension reduction methods are used to reduce the size of the data; application of different techniques to obtain interesting characteristics of users' behavior; interpretation of the results. During the implementation phase it should be taken into account the distributed and heterogeneous nature of distributed systems and a great number of users in it. Therefore, it is advisable to provide an autonomous module for each user behavior model developed within the first phase. Moreover, in some cases this module has to move in the system since the user can work on different workstations (computers). Thus, the monitoring system has to be distributed and scalable, it should enable the work with different operating systems and data formats, it should have independent modules to enable autonomy and mobility. To meet these requirements, agent technology represents the most appropriate way [10-11].

In this paper we present an agent-based approach for anomalies monitoring in distributed systems. This approach envisages on-line and off-line monitoring that enables the detection of anomalies and irregularities in users' behavior. On-line monitoring is carried in real time, and is used to predict user actions. For this purpose, we use feed-forward neural networks [12]. Off-line monitoring is done after the user has ended his work, and is based on the analysis of statistical information obtained during user's work. We use neural network as classifier to distinguish normal and anomalous user behavior. The use of on-line and off-line monitoring allows one to reflect both dynamical and statistical features of user's activity. Considering system implementation, we use Java programming language and Aglets Software Development Kit (ASDK) for the development of mobile agents.

2 Agent Paradigm

The main point about agents is that they are autonomous, i.e. capable of acting independently. An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors [10]. That is, the agent can be characterized by the following set:

$$<S, Prog, Eff, Arch, P, A, G, E> \tag{1}$$

where E defines the environment where agent works; S — sensors through which it perceives information from environment; Eff — effectors through which agent can act on environment; P — what kind of information agent can perceive from its sensors; A — what kind of actions agent can make using its effectors; Prog (program) Prog: P–>A — defines agent's response to its percepts; G — goal the agent trying to reach; Arch — agent's architecture.

Main agents' properties are the following ones [13]: autonomy, reactivity (provides an ongoing interaction with its environment, and responds to changes that occur in it), proactiveness (means goal directed behavior of agent), social ability (ability to interact with other agents via some kind of agent-communication language, and perhaps co-operate with others), mobility (the ability of an agent to move around an electronic network), rationality (agent will act in order to achieve its goals), learning/adaptation (agents improve performance over time).

In this paper, software agents will be used for implementation of intelligent security system. In general, they represent computer programs and act in computer systems. Thus, according to (1) for software agent we have — E=computer system, Arch=program code, S and Eff represent some functions (or, in general case, programs) through which agent can interact with environment.

3 System Architecture and Functionality

The proposed intelligent security system for users' activity monitoring in distributed systems consists of the following components (Fig. 1):

— On-line User Agent that provides on-line monitoring,

— Off-line User Agent that provides off-line monitoring,

— Controller Agent that manages other agents,

— Database.

On-line User Agent. This agent is functioning in real time with aim to detect anomalies and irregularities in computer users' activity. It predicts user actions on the basis of previous ones. For this purpose a neural network is used. The output of the neural network is compared to real actions performed by user. If the relative number of correctly predicted actions larger than specified threshold, then it can be assumed that the user behavior is normal. Otherwise, it is abnormal. Additionally, this agent collects information about user's activity and stores it in database. This type of agents should be constructed for different operating systems used in computer system (e.g. Win2K/XP, Win98, FreeBSD).

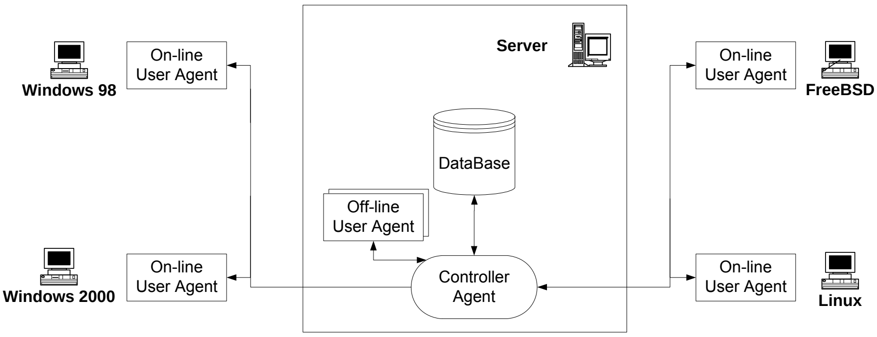


Fig. 1. System architecture

Off-line User Agent. This agent works off-line (i.e. after the user has ended his work) and tries to detect anomalies in the user activity on the basis of statistical parameters (user signature). The following set of characteristics about user behavior were taken as user signature: the set of processes (number of processes started by user), results of on-line agent functioning (number of correctly predicted processes by On-line User Agent), user login host (the set of hosts from which user logs on), user session time (the session duration for the user), user activity time (the time of user session starting). For each user its own Off-line User Agent is created based on feed-forward neural network. The network is trained in order to distinguish normal and abnormal user behavior.

Controller Agent. This agent is responsible for overall system functioning, agents initializing and coordination, and interaction with database.

Database. Contains data that is needed for system functioning.

When the user logs on (that is, begins his work on computer), Controller Agent creates corresponding On-line User Agent and initializes it. On-line User Agent gets data about specified user from a database and moves to the computer where the user works. During the user's session, this agent monitors user's activity by predicting his actions (using neural network) and comparing them to real ones. If the relative number of correctly predicted actions larger than specified threshold, then it can be assumed that user behavior is normal and corresponds to the previously built model. Otherwise, user behavior is assumed to be abnormal. In the case of anomaly detection On-line User Agent informs Controller Agent about suspicious activity. When user finishes his work, On-line User Agent is destroyed.

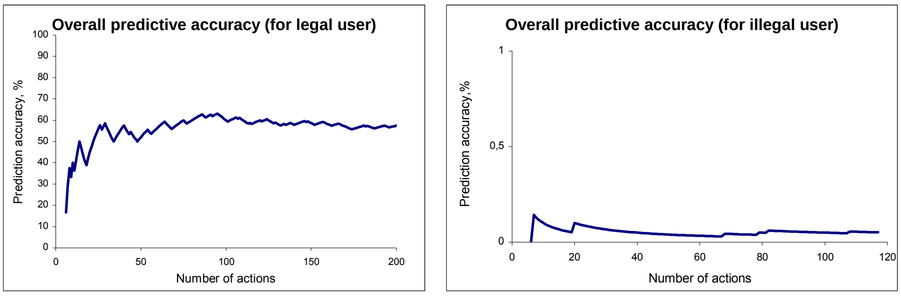
At the end of the day (when the system load is low) Controller Agent initializes Off-line User Agent. On the basis of data obtained from On-line User Agent it detects if the user activity was normal or abnormal. In the case of abnormal activity (i.e. it had anomalies) Off-line User Agent informs Controller Agent about it.

4 Description of Experiments

Different experiments were run to demonstrate the efficiency of both On-line User Agent and Off-line User Agent. Since both types of agents are based on the use of neural networks data needed for neural network training were obtained during real work of users in the Space Research Institute NASU-NSAU. For this purpose special software was developed to get data about users' activity.

For On-line User Agent log files were transformed into format suitable for neural network. That is, for each user an alphabet of actions (processes) was created, and each action was assigned an identifier (decimal number). For neural network input a binary coding was applied (7 bits for each command). Feed-forward neural network trained by means of error back-propagation algorithm [12] was used in order to predict user action on the basis of 5 previous ones. Thus, the dimension of input data space for neural network was 35. In turn, for output data decimal coding was applied, and the dimension of output data space was 1. As to neural network architecture, we used neural network with 3 layers: input layer with 35 neurons, hidden layer with 35 neurons, and output layer with 1 neuron.

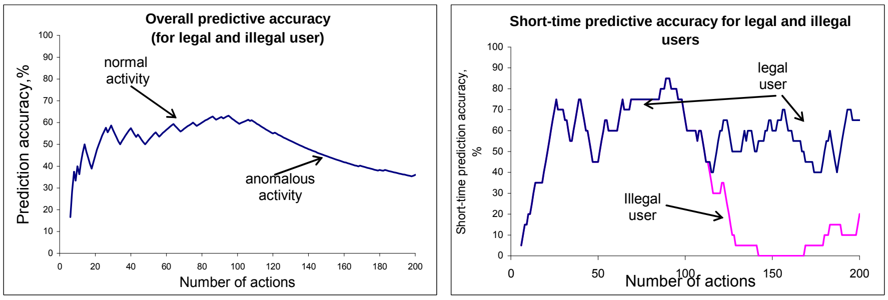
Then all data were randomly mixed and divided into training and test sets (70% for training and 30% for testing). Results of neural network work on test data showed that overall predictive accuracy (that is, the number of correctly predicted commands divided by total number) for different users varied from 33% to 59% (an example of overall predictive accuracy variations within number of user actions is depicted on Fig. 2,a). But the main point in constructing On-line User Agents is to ensure that they differ for different users. That is, the efficiency of On-line User Agent should be viewed not in the term of absolute value of the predictive accuracy for the user, but relative to other users. In order to demonstrate that the neural network was able to distinguish one user from another we run so called cross experiments. Two types of cross experiments were implemented. First one consisted in the following: the data obtained during the work of one user (name him illegal user) were put to neural network that was trained for another (legal user). In such a case, overall predictive accuracy of neural network hardly exceeded 5% (on Fig. 2,b it is shown an example where overall predictive accuracy was 0,05%). That is, the overall predictive accuracy decreased, at least, six times for illegal user. Such experiment modeled the situation when illegal user logged on and begun to work under the account of another user.



(a) (b)

Fig. 2. Overall predictive accuracy for: (a) legal user; (b) illegal user

The second type of cross experiments was carried out by inserting the data of illegal user into the data of legal one. This experiment modeled the situation when an intruder begun to work under the account of another user already logged on. In a such case, the overall predictive accuracy begun to decrease constantly, as shown on Fig.3,a. Another measure that can be used to distinguish normal and anomalous user activity is a short-time predictive accuracy. To estimate the short-time predictive accuracy we took into considerations only last actions performed by the user but not all (for example, twenty last actions). Variations of short-time predictive accuracy for both legal and illegal user are shown on Fig. 3,b. From figure it is evident that the short-time predictive accuracy for illegal user begun to decrease.



(a) (b)

Fig. 3. Predictive accuracy for: (a) overall; (b) short-time

Therefore, experimental results showed the ability of neural networks to distinguish confidently normal and abnormal (anomalous) user behavior.

As with On-line User Agent, all data needed for Off-line User Agent were obtained from the log files. Then the data were encoded, divided into training and test sets, and input to neural network. Results of neural network work on test data gave 80% accuracy of correct user behavior classification. That is, experiments showed that

Off-line User Agent was able to distinguish normal and abnormal (anomalous) user behavior. Additionally, Off-line User Agent can be used to verify the work of On-line User Agent.

5 System Implementation

The proposed agent-based system was implemented using mobile agents. Java language and Aglets Software Development Kit (ASDK) were chosen, respectively, as programming language and environment for mobile agents development. Java offers the set of unique features that allows one to simplify the development of multi-agent systems. The following properties of Java should be mentioned: platform independence; secure code execution; dynamic class loading; multithreading programming; object serialization.

ASDK is a free-ware software, provided by IBM. It enables the development of mobile agents that are called aglets (http://sourceforge.net/projects/aglets/). The following properties of ASDK could be mentioned: the use of special MASIF (Mobile Agent System Interoperability Facility) standard which allows various agent systems to interoperate; the use of ATP (Agent Transfer Protocol) protocol that represents a simple application-level protocol designed to transmit an agent in an agent-system-independent manner; mobility of agents; the use of Java security policy (JDK keytool).

In general, aglets are Java objects that can move from one host on the network to another. That is, an aglet that is run on one host can suddenly halt execution, dispatch to a remote host, and start executing again. When the aglet moves, it takes along its program code as well as the states of all the objects it is carrying. A built-in special security mechanism makes it safe to host untrusted aglets.

Proposed intelligent security system was implemented based on client/server architecture. Server side represented a special platform which was used for the creation of agents and its hosting (all agents used in the system are initiated on server side), for database interaction, requests redirection. Client side is responsible for agent functioning on user computers. Among its functions are support of agents hosting and information logging about user activity. Additionally, special user interface was developed that shows information about user logged on, operating system that is used, client platform parameters, and information about On-line User Agent work.

6 Conclusions

The proposed system takes advantages of both intelligent methods for monitoring of user activity and multi-agent approach. To reflect both dynamical and statistical parameters of user behavior on-line and off-line monitoring is done. The use of neural network provides adaptive and robust approach for the analysis and generalization of data obtained during user activity. The use of multi-agent approach is motivated by the system functioning in heterogeneous environment, and by processing data in different operating systems.

Acknowledgments

This research is supported by INTAS-CNES-NSAU project “Data Fusion Grid Infrastructure”, Ref. No 06-1000024-9154..

Bibliography

[1] Foster, I., Kesselman, C., Tuecke, S.: The Anatomy of the Grid: Enabling Scalable Virtual Organizations. Int. J. Supercomputer Applications 15(3) (2001).

[2] Tulloch, M.: Microsoft Encyclopedia of Security. Redmond, Washington: Microsoft Press (2003) 414 p.

[3] Javitz, H., Valdes, A.: The SRI IDES statistical anomaly detector. In: Proc. IEEE Symp. on Research in Security and Privacy (1991) 316–326.

[4] Dowell, C., Ramstedt, P.: The ComputerWatch data reduction tool. In: Proc. 13th National Computer Security Conf. (1990) 99–108.

[5] Kussul, N., Sokolov, A.: Adaptive Anomaly Detection of Computer System User's Behavior Applying Markovian Chains with Variable Memory Length. Part I. Adaptive Model of Markovian Chains with Variable Memory Length. J. of Automation and Information Sciences Vol. 35 Issue 6 (2003).

[6] Ryan, J., Lin M-J., Miikkulainen, R.: Intrusion Detection with Neural Networks. In: Advances in Neural Information Processing Systems. Cambridge, MA: MIT Press (1998) 943–949.

[7] Reznik, A, Kussul, N., Sokolov, A.: Identification of user activity using neural networks. Cybernetics and computer techniques, vol. 123 (1999) 70–79. (in Russian)

[8] Kussul, N., et al. : Multi-Agent Security System based on Neural Network Model of User's Behavior. Int. J. on Information Theories and Applications Vol. 10 Num. 2 (2003) 184–188.

[9] Gorodetski, V., et al.: Agent-based model of Computer Network Security System: A Case Study. In: Proc. of the International Workshop 'Mathematical Methods, Models and Architectures for Computer Network Security', Lecture Notes in Computer Science, Vol. 2052. Springer Verlag (2001) 39-50.

[10] Russel, S., Norvig, P.: Artificial Intelligence: A Modern Approach. Upper Saddle River NJ: Prentice Hall (1995).

[11] Luck, M., McBurney, P., Preist, C.: Agent Technology: Enabling Next Generation Computing. AgentLink (2003).

[12] Haykin S.: Neural Networks: a comprehensive foundation. Upper Saddle River, New Jersey: Prentice Hall (1999).

[13] Wooldridge, M.: An Introduction to Multi-agent Systems. Chichester, England: John Wiley & Sons (2002).

Author's Information

Andrii Yu. Shelestov – *PhD, Senior Researcher, Department of Space Information Technologies and Systems, Space Research Institute of NASU-NSAU, Glushkov Ave 40, Kyiv-187, 03650 Ukraine, e-mail: inform@ikd.kiev.ua.*

USING THE AGGLOMERATIVE METHOD OF HIERARCHICAL CLUSTERING AS A DATA MINING TOOL IN CAPITAL MARKET[1]

Vera Marinova–Boncheva

Abstract: *The purpose of this paper is to explain the notion of clustering and a concrete clustering method-agglomerative hierarchical clustering algorithm. It shows how a data mining method like clustering can be applied to the analysis of stocks, traded on the Bulgarian Stock Exchange in order to identify similar temporal behavior of the traded stocks. This problem is solved with the aid of a data mining tool that is called XLMiner™ for Microsoft Excel Office.*

Keywords: *Data Mining, Knowledge Discovery, Agglomerative Hierarchical Clustering.*

ACM Classification Keywords: *I.5.3 Clustering*

Introduction

Data mining, the extraction of hidden predictive information from large databases, is a powerful new technology with great potential to help companies focus on the most important information in their data warehouses. Data mining tools predict future trends and behaviors, allowing businesses to make proactive, knowledge-driven decisions. The automated, prospective analyses offered by data mining move beyond the analyses of past events provided by retrospective tools typical of decision support systems. Data mining tools can answer business questions that traditionally are time consuming to resolve. They scour databases for hidden patterns, finding predictive information that experts may miss because it lies outside their expectations. Data mining consists of analysis of sets of supervised data with the aim of finding unexpected dependencies or to be generalized in a new way that is understandable and useful for owners of the data. There is a great deal of data mining techniques but we differentiate two of them like classification and clustering as supervised and unsupervised learning from data. [2]

[1] This work was supported by the Ministry of Education and Science under Project № MU-MI-1601/2006

The Analysis of Clustering

Clustering can be considered the most important unsupervised learning problem. So, as every other problem of this kind, it deals with finding a structure in a collection of unlabeled data. A cluster is therefore a collection of objects which are “similar” between them and are “dissimilar” to the objects belonging to other clusters.

Cluster Analysis, also called data segmentation, has a variety of goals. They all relate to grouping or segmenting a collection of objects (also called observations, individuals, cases, or data rows) into subsets or "clusters", such that those within each cluster are more closely related to one another than objects assigned to different clusters. Central to all of the goals of cluster analysis is the notion of degree of similarity (or dissimilarity) between the individual objects being clustered that depend on the data and the application. Different types of similarity measures may be used to identify classes (clusters), where the similarity measure controls how the clusters are formed. Some examples of values that can be used as similarity measures include distance, connectivity, and intensity. [4]

The main requirements that a clustering algorithm should satisfy are:

- scalability;
- dealing with different types of attributes;
- discovering clusters with arbitrary shape;
- minimal requirements for domain knowledge to determine input parameters;
- ability to deal with noise and outliers;
- insensitivity to order of input records;
- high dimensionality;
- constrained - based clustering;
- interpretability and usability. [7]

Clustering algorithms may be classified as listed below:

- Exclusive Clustering
- Overlapping Clustering
- Hierarchical Clustering
- Probabilistic Clustering

In the first case data are grouped in an exclusive way, so that if a certain datum belongs to a definite cluster then it could not be included in another cluster. On the contrary the second type, the overlapping clustering, uses fuzzy sets to cluster data, so that each point may belong to two or more clusters with different degrees of membership. In this case, data will be associated to an appropriate membership value. A hierarchical clustering algorithm is based on the union between the two nearest clusters. The beginning condition is realized by setting every datum as a cluster. After a few iterations it reaches the final clusters wanted. Finally, the last kind of clustering uses a completely probabilistic approach. [5, 6]

There are a number of problems with clustering. Among them:

- current clustering techniques do not address all the requirements adequately (and concurrently);
- dealing with large number of dimensions and large number of data items can be problematic because of time complexity;
- the effectiveness of the method depends on the definition of “distance” (for distance-based clustering);
- if an obvious distance measure doesn’t exist we must “define” it, which is not always easy, especially in multi-dimensional spaces;
- the result of the clustering algorithm (that in many cases can be arbitrary itself) can be interpreted in different ways.

Clustering is a method that is applicable in many fields like:

- Marketing: finding groups of customers with similar behavior when it is given a large database of customer data containing their properties and past buying records;
- Biology: classification of plants and animals given their features;
- Libraries: book ordering;

- Insurance: identifying groups of motor insurance policy holders with a high average claim cost; identifying frauds;
- City-planning: identifying groups of houses according to their house type, value and geographical location;
- Earthquake studies: clustering observed earthquake epicenters to identify dangerous zones;
- WWW: document classification; clustering weblog data to discover groups of similar access patterns.

Hierarchical Clustering

In hierarchical clustering the data are not partitioned into a particular cluster in a single step. Instead, a series of partitions takes place, which may run from a single cluster containing all objects to N clusters each containing a single object. Hierarchical Clustering is subdivided into agglomerative methods, which proceed by series of fusions of the N objects into groups, and divisive methods, which separate N objects successively into finer groupings. Agglomerative techniques are more commonly used, and this is the method implemented in the free version of XLMiner™ which is the Microsoft Office Excel add-in. [1]

If it is given a set of N items to be clustered and a N*N distance (or similarity) matrix then the basic process of agglomerative hierarchical clustering can be done iteratively following these four steps:

1. Start by assigning each item to a cluster. Let the distances (similarities) between the clusters are the same as the distances (similarities) between the items they contain;
2. Find the closest (most similar) pair of clusters and merge them into a single cluster;
3. Compute distances (similarities) between the new cluster and each of the old clusters;
4. Repeat step 2 and 3 until all items are clustered into a single cluster of size N.

Step 3 can be different because of the varieties in the definition of the distance (or similarity) between clusters:

- Single linkage clustering (nearest neighbor technique) – here the distance between groups is defined as the distance between the closest pair of objects, where only pairs consisting of one object from each group is considered i.e. the distance between two clusters is given by the value of the shortest link between clusters. At each stage the two clusters for which the distance is minimum are merged;
- Complete linkage clustering (farthest neighbor) – is the opposite of the single linkage i.e. distance between groups is defined as the distance between the most distant pair of objects, one from each group. At each stage the two clusters for which the distance is minimum are merged;
- Average linkage clustering – the distance between two clusters is defined as the average of distances between all pairs of objects, where each pair is made up of one object from each group. At each stage the two clusters for which the distance is minimum are merged;
- Average group linkage clustering – with this method, groups once formed are represented by their mean values for each variable, that is their mean vector and inter-group distance is defined in terms of distance between two such mean vectors. At each stage the two clusters for which the distance is minimum are merged. In this case, those two clusters are merged such that the newly formed cluster, on average, will have minimum pairwise distances between the points in it;
- Ward's hierarchical clustering – Ward (1963) proposed a clustering procedure seeking to form the partitions $P_n,...,P_1$ in a manner that minimizes the loss associated with each grouping and to quantify that loss in a form that is readily interpretable. At each step the union of every possible cluster pair is considered and the two clusters whose fusion results in minimum increase in "information loss" are combined. Information loss is defined by Ward in terms of an error sum-of-squares criterion. [3]

Hierarchical clustering may be represented by a two dimensional diagram known as dendrogram which illustrates the fusions or divisions made at each successive stage of analysis. By cutting the dendrogram at a desired level clustering of the data items into disjoint groups is obtained. [1]

Major weakness of agglomerative clustering methods is that:

- they do not scale well and time complexity is at least $O(n^2)$, where n n is the number of total objects;
- they can never undo what was done previously.

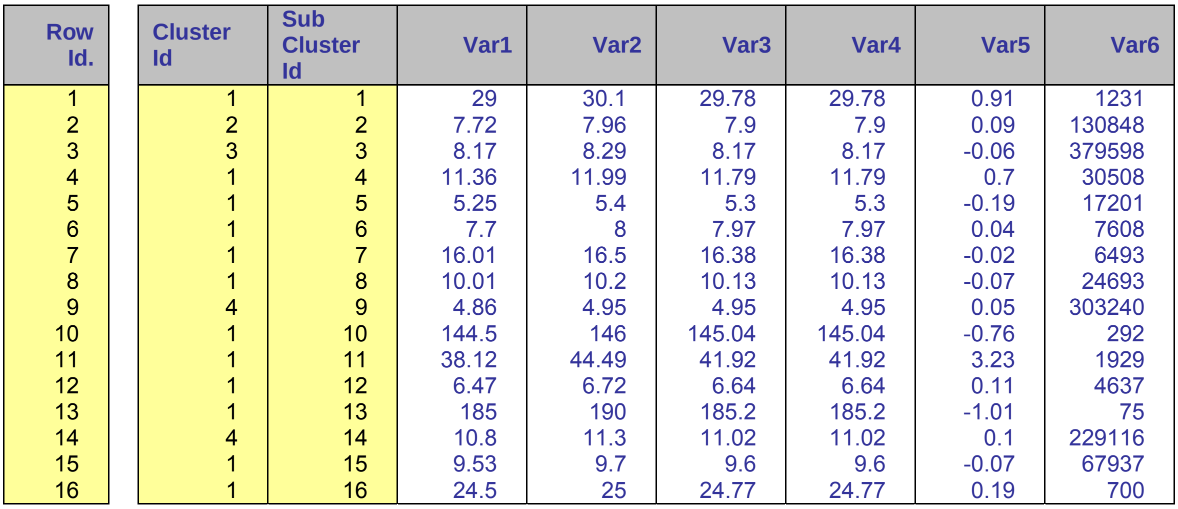
Clustering of Stocks, traded on the Official Market of BSE

As inputs we have taken data for 16 stocks from the Bulgarian Stock Exchange in a single day. (Table 1) These data are listed on the Internet address: http://www.econ.bg/capital.html. It contains information for each stock as the code and the name of the company, the nominal, prices (low, high, last, medium), the change in price in comparison to the previous day and the traded amount of this kind of stock.

Company code	nominal	prices				change	amount
		low	high	last	medium		
CENHL	1	29	30.1	29.78	29.78	0.91	1231
SFARM	1	7.72	7.96	7.9	7.9	0.09	130848
CCB	1	8.17	8.29	8.17	8.17	-0.06	379598
PETHL	1	11.36	11.99	11.79	11.79	0.7	30508
DOVUHL	1	5.25	5.4	5.3	5.3	-0.19	17201
IHLBL	1	7.7	8	7.97	7.97	0.04	7608
ALBHL	1	16.01	16.5	16.38	16.38	-0.02	6493
GAZ	1	10.01	10.2	10.13	10.13	-0.07	24693
PET	1	4.86	4.95	4.95	4.95	0.05	303240
ORGH	1	144.5	146	145.04	145.04	-0.76	292
HVAR	1	38.12	44.49	41.92	41.92	3.23	1929
SEVTO	1	6.47	6.72	6.64	6.64	0.11	4637
ODES	1	185	190	185.2	185.2	-1.01	75
CHIM	1	10.8	11.3	11.02	11.02	0.1	229116
MONBAT	1	9.53	9.7	9.6	9.6	-0.07	67937
KTEX	1	24.5	25	24.77	24.77	0.19	700

Table 1. Information about stocks, traded on the Official Market of Bulgarian Stock Exchange

We use the data mining tool named XLMiner™ for MS Excel. We select the agglomerative method of hierarchical clustering to find clusters of stocks. We experiment on all five variants of agglomerative method of hierarchical clustering and we have founded that the average linkage method will give the best results. We use as a stop rule for the process of clustering the number of clusters which is 4. [1]



Row Id.	Cluster Id	Sub Cluster Id	Var1	Var2	Var3	Var4	Var5	Var6
1	1	1	29	30.1	29.78	29.78	0.91	1231
2	2	2	7.72	7.96	7.9	7.9	0.09	130848
3	3	3	8.17	8.29	8.17	8.17	-0.06	379598
4	1	4	11.36	11.99	11.79	11.79	0.7	30508
5	1	5	5.25	5.4	5.3	5.3	-0.19	17201
6	1	6	7.7	8	7.97	7.97	0.04	7608
7	1	7	16.01	16.5	16.38	16.38	-0.02	6493
8	1	8	10.01	10.2	10.13	10.13	-0.07	24693
9	4	9	4.86	4.95	4.95	4.95	0.05	303240
10	1	10	144.5	146	145.04	145.04	-0.76	292
11	1	11	38.12	44.49	41.92	41.92	3.23	1929
12	1	12	6.47	6.72	6.64	6.64	0.11	4637
13	1	13	185	190	185.2	185.2	-1.01	75
14	4	14	10.8	11.3	11.02	11.02	0.1	229116
15	1	15	9.53	9.7	9.6	9.6	-0.07	67937
16	1	16	24.5	25	24.77	24.77	0.19	700

Table 2. Clusters of stocks taken from table 1

The dendrogram in Figure 1 shows how the numbered stocks are divided into the following four clusters: {1,4,5,6,7,8,10,11,12,13,15,16}, {2}, {3}, {9,14}. (Table 2) The last cluster is composed by two stocks that have the least prices, the greatest amounts traded and positive change. They are the most interesting for the investor. The second and the third cluster consist of only one stock. They have approximately equal prices and high amounts of them are traded but they differ from each other because stock 2 has positive change but stock 3 has negative change. The rest of stocks are grouped in another cluster. So this method is a good way to combine stocks that are preferred by the investors.

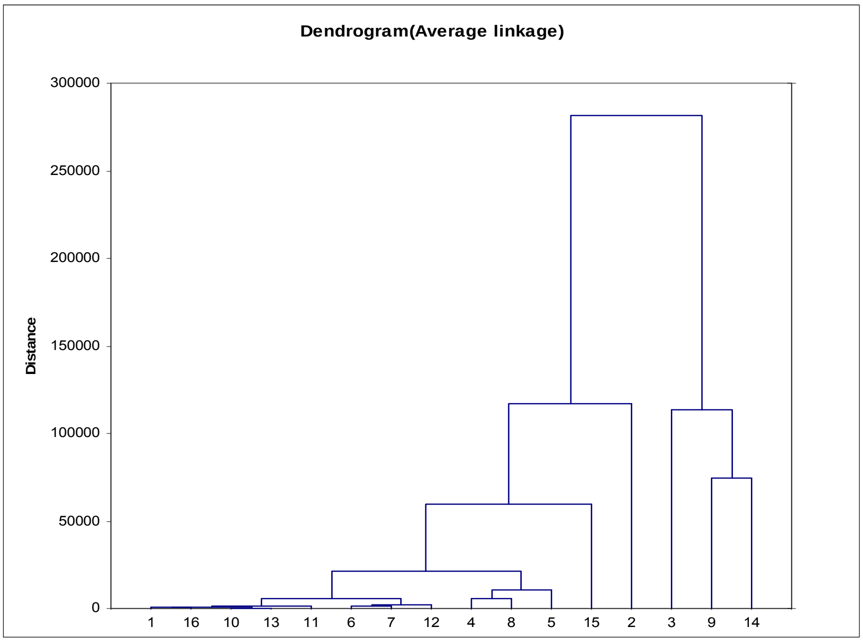


Figure 1. Dendrogram of the clusters from table 1

Conclusion

Data mining software allows users to analyze large databases to solve business decision problems. Data mining is, in some ways, an extension of statistics, with a few artificial intelligence and machine learning. Like statistics, data mining is not a business solution, it is just a technology. In this article it has been shown how a hierarchical clustering method can support an investor decision to choose stocks which can pretend to be participants in an investment portfolio by using a data mining tool. So the identification of clusters of companies of a given stock market can be exploited in the portfolio optimization strategies.

Bibliography

1. G, Nitin, R. Patel, P. C. Bruce. Data Mining for Business Intelligence: Concepts, Techniques, and Applications in Microsoft Office Excel with XLMiner. Hardcover, 2007.
2. Chris Westphal, Teresa Blaxton, Data Mining Solutions, John Wiley, 1998.
3. A.K.Jain, R.C. Dubes. Algorithms for Clustering Data. Englewood Cliffs, NJ: Prentice Hall, 1988.
4. L. Kaufman, P.J. Rousseeuw. Finding Groups in Data: An Introduction to Cluster Analysis. New York: John Wiley&Sons, 1990.
5. J.A. Harigan. Clustering Algorithms. New York: John Wiley&Sons,1975.
6. A.K. Jain, M.N. Murty, and P.J. Flynn. Data clustering: A survey. ACM Comput. Surv., 31:264-323, 1999.
7. J. Han, M. Kamber. Data mining: Concepts and Techniques, Morgan Kaufmann, 2000.

Author's Information

Vera Marinova-Boncheva - *Institute of Information Technologies, Bulgarian Academy of Science, Sofia-1113, Bulgaria; e-mail: vboncheva@iit.bas.bg*

SEMANTIC MODELING FOR PRODUCT LINE ENGINEERING

Mikhail Roshchin, Peter Graubmann, Valery Kamaev

Abstract: *The aim of our work is to present solutions and a methodical support for automated techniques and procedures in domain engineering, in particular for variability modeling. Our approach is based upon Semantic Modeling concepts, for which semantic description, representation patterns and inference mechanisms are defined. Thus, model-driven techniques enriched with semantics will allow flexibility and variability in representation means, reasoning power and the required analysis depth for the identification, interpretation and adaptation of artifact properties and qualities.*

Keywords: *Variability Modeling, Semantic Modeling, Product Line Engineering, MDA.*

Problem Statement

Let us assume that we require a software system that is specifically tailored to rely on our needs; that is valid and consistent within the reality of the environment and involved domains. But the cost issue plays an important role, and the development of specific and generic products is not that cost-effective as we expect. For reduction of costs, software engineering aims of an increasing reuse by collecting and composing artifacts and assets, components and products into complex systems and new applications. Also, the ideas and concepts of families of systems and product lines are formalized for easier way of future artifact implementation.

Behind the system composition process and derivation of new product implementation based on reuse, there is a heavy and massive layer of computing model-based procedures. Therefore models are considered to be interchangeable and valid for particular task and requirements. Model-driven engineering introduces models together with techniques for system design and artifact adaptation into business process and software lifecycle. At the same time, domain engineering provides with deep understanding of the targeted domain and its specifics, and variability modeling specifies commonalities, variants and features, their relations and restrictions, for the whole product family of systems realized and presented as models.

But, due to the high diversity of modeling techniques, distinctions between models of different aspects, domain-dependent and company-specific knowledge and specifications, the reuse is still difficult and non-trivial. The lack of formal semantics for MDAs [Greenfield, 2004], domain and variability models and requirements engineering, affects with the impossibility of pragmatic and cost-effective solution for automated reasoning techniques. The absence of well-established semantic model does not allow us to provide self-configuring techniques, consistency verification procedures and advanced selection of valid artifacts.

Domain engineering has been proved to handle a high priority share in the entire model-driven engineering, but the state of the art shows that the lack of formal semantics and proper tool support for automated reasoning have hindered the development in this area. So far, the knowledge representation techniques based on semantics are being developed in isolation from software engineering activities, in particular from feature and variability modeling. Existing semantic approaches are not aligned with the entire modeling process, and need an advanced review on conceptual level for the proper role and place of formal methods within existing software engineering streams.

No doubts, that model-driven architecture, domain engineering, variability and feature models are perfect approaches themselves. But there is an urgent need to enrich them with formal methods of knowledge representation and benefit from that in the near future [Assmann, 2003].

Here we focus just on variability modeling, assuming that our approach can be used in a wider range, in particular for MDE and domain engineering. It is shown how semantic modeling can handle and support variability modeling, and how software engineering will benefit from that.

Semantic Modeling Approach

The need for variability modeling and its role within the scope of domain engineering in the software development area are obvious and generally accepted. Variability modeling becomes necessary when we derive new specifications for further artifact implementation from the set of commonalities and variants related to particular system family. Also, it is important for describing dynamical behavior of systems. We take a variability model as proposed by [Buehne, 2005]. But still, there are open questions and issues, mentioned by different research institutes and software communities, which have hindered the expected development in the field of knowledge reuse.

The automation in general is based on a set of specific methods and procedures, which allow us to substitute the human participation with some formal algorithms. The design automation needs assistance in making decisions and solving problems in analyzing requirements from customers, and mapping them onto our product family description – variability model. But applying selection procedures to variability model is not sufficient. The project manager has to be aware of existing components, which are ready for reuse. Thus component repository and its participation in a decision procedure play an important role (see 0).

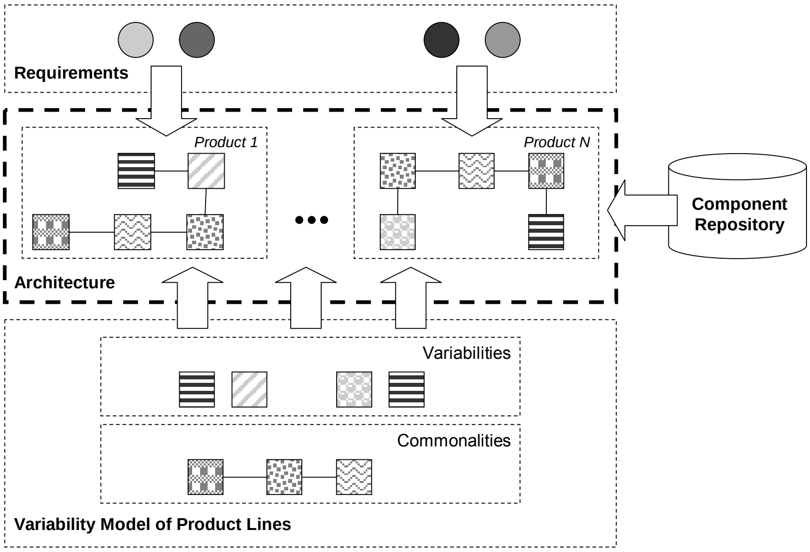


Figure 1. Software Design: from Requirements and Variability to Architecture

Our goal is to provide proper methods and tool support for formally expressing, processing and analyzing models and variants. We need to introduce formal semantics and appropriate automated reasoning techniques. Based on that, we achieve explicit consideration of environmental, behavioral and business model aspects, interoperability of the diversity of components. Semantic modeling allows acquisition, interpretation and adaptation of different variability models into one decision process.

Our Semantic Modeling approach presented in [Graubmann, 2006] is based on two concepts, which are significant for the whole procedure and aligned with requirements to semantics. These concepts are Logic-on-Demand and Triple Semantic Model (see 0).

The Triple Semantic Model Concept

Our Semantic Model is based on the principles of the Triple Semantic Model concept, which aims in defining a distributed computing model for the whole lifecycle of variability model and to provide mechanisms to distinguish between different entities represented within that model. It consists of three levels: the Ontology Level, the Dynamic Annotation Level, and the Annotation Level. The ontologies on the *Ontology Level* are intended to provide a general framework, in most cases based on a specific application domain, to describe any kind of product line and related information. Since ontologies enforce proper definitions of the concepts in the application domain, they also play an essential role in standardizing the definitions of component or service properties [0], requirements and interfaces with respect to their domain. Ontologies hold independently from actual circumstances, the situation in the environment or the actual time. However, such dependencies from actual, dynamically changing circumstances do have an important influence in the compositional approach. Hence, rules determining how to cope with this dynamicity have to be provided if one has to include it into the reasoning. They are specified on the *Dynamic Annotation Level*: Dynamic annotations play the role of mediators between the ontology and the static semantic annotations that describes the artifact variants and features, and in particular its requirements with respect to reuse and composition. It becomes possible to express behavior variants, and options depending on dynamic features, and it enables the reasoning about particular situations and dynamically changing lifecycle conditions. The *Annotation Level* comprises the static descriptions of the properties and qualities of artifacts.

The Logic-on-Demand Concept

Semantic modeling of products and families involves a large variety of information from different application domains and of various categories, like terms and definitions, behavior rules, probability relations, and temporal properties. Thus, it seems to be the obvious to choose the most expressive logical formalism that is capable to formulate and formalize the entire needed information. But, doing so very likely it results in severe decidability problems.

Our semantic modeling approach, based on the concept of Logic-on-Demand (LoD), is supposed to overcome the problems of complexity of formal semantics by accommodating the expressivity of the proposed ontology languages to the varying needs and requirements, in particular with respect to decidability. The main purpose of the LoD concept is to provide an adequate and adaptive way that is based on uniform principles for describing all the notions, relations and rules, the behavior and anything else that proves necessary during the component or service annotation process. To achieve this, LoD means to define a basic logical formalism that is adequate and tailored to the application domain and to incorporate additional logic formalisms and description techniques with further expressivity as optional features that can be used whenever needed. These additional formalisms share notions and terms with the basic formalism which will be grounded syntactically in OWL and semantically in the description logics.

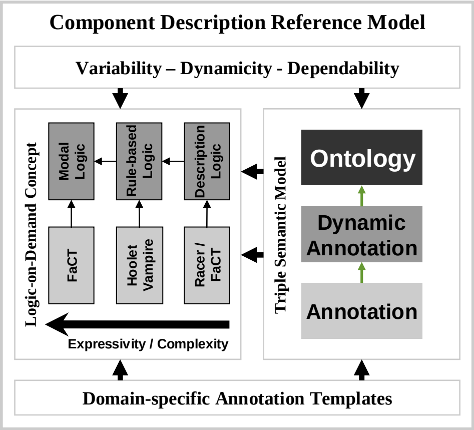


Figure 2. Semantic Model of Components

Thus, semantic modeling is applied for both formal description of Variability Models in Product Line Engineering and software components. The meta-model of variability description can be easily obtained by substitution of nodes and edges on modeling graph by classes and property relations from Description Logic. Instances of the classes represent specific notions and features from product family description.

A brief sketch of the component or service selection and composition process according to the Triple Semantic Model now comprises the following steps:

- Requirements on a component or service to be integrated into the system are collected. They serve as selection criteria when candidates are checked.
- The dynamic annotation and the (static) annotation of the candidate component/service are used to create an annotation that is valid in the given situation and time.
- This annotation is analysed and compared with the initial requirements.
- If the result shows that the component fits, it may be integrated (what may include the generation of data transformations in order to adapt the interfaces).

So, software engineers and system developers have to define their specific view on the concrete component/service and they naturally formulate this information in the terminology of the domain or system family to which the component/service belongs. If the annotating is done properly, we have the complete information about the component/service properties. Due to the Logic-on-Demand concept, this information is available not only for the developers but also presented in a form that is readable for automated acquisition and adaptation tools and thus, it allows reasoning and derivation of additional information.

The validation of the approach includes three aspects:

- Evaluation of the applied formal semantics with respect to sufficiency and decidability. The work on Logic-on-Demand concept is still in progress. Our intention is to avoid complexity issues and to guarantee adequate system response time.

- Feasibility issue. So far, we implement proposed techniques in a prototype tool. It covers the whole lifecycle of semantic modeling – starting out from defining semantic patterns and domain-specific information and eventually providing fully automated composition techniques based on semantic models.
- Estimating the additional cost and time for semantic modeling according an approach. Do a creation of semantic models and an implementation of formal methods and techniques really pay off in software engineering? This question touches a most important issue of our work and will be investigated in concordance the prototype tool development.

Conclusion

Introducing a well-structured semantic modeling procedure for variability modeling provides with flexibility of representation means and methods. It allows correct (self) configuration and composition of different shares among the whole set of domain pieces during the entire modeling process, by taking into account behavioral, environmental and business aspects. Improved acquisition, interpretation and adaptation techniques allow to increase reuse among different domains and system families. Formal methods in modeling support automated derivation of an executable and sufficient model for further system or artifact implementation based on semantic mapping of requirements criteria and the given set of features and variants.

Our approach proposes an annotation process and its semantic extensions through knowledge-based techniques as the basis for semantic modeling. The Component Description Reference Model (semantic model for software components) structures the annotation process and introduces flexibility with respect to the description mechanisms what allows for a trade-off between expressivity and complexity and the selection of the appropriate reasoning tools. It is based on the Logic-on-Demand concept which means to achieve a proper compromise between existing semantic approaches and it proposes a hybrid knowledge-based solution for annotating software components. By introducing the Triple Semantic Model concept we allow an integration of means to adequately express dynamicity and variability into an modeling process.

There are, however, still open questions. We continue to work on automatic mapping of different ontologies from heterogeneous environments and knowledge application domains, on integration of different logic formalisms for component and service description, and on the mutual adaptation of problem solvers based on different logics and inference algorithms, to name but a few of the themes to be tackled in the future. We also will particularly focus on tool support for the proposed techniques to demonstrate the expected benefits, and we will later on integrate the techniques into a software development environment.

Bibliography

[Graubmann, 2006] Peter Graubmann, Mikhail Roshchin, "Semantic Annotation of Software Components", Accepted for 32th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA), Component-Based Software Engineering Track, 2006

[Greenfield, 2004] Jack Greenfield, Keith Short, *Software Factories*. Wiley Publishing, 2004

[Pahl, 2002] Claus Pahl, "Ontologies for Semantic Web Components," ERCIM News No 51, Oct. 2002. http://www.ercim.org/publication/Ercim_News/enw51/pahl.html

[Assmann, 2003] Uwe Assmann, Steffen Zschaler, Gerd Wagner, *Ontologies, Meta-Models and the Model-Driven Paradigm,* 2003

[Buehne, 2005] Stan Buehne, Kim Lauenroth, Klaus Pohl: *Modelling Requirements Variability across Product Lines.* Proceedings of the 2005 13th IEEE International Conference on Requirements Engineering (RE '05), IEEE

Authors' Information

***Roshchin Mikhail** - PhD Student of Volgograd State Technical University, working in collaboration with CT SE, Siemens AG; e-mail: roshchin@gmail.com*

***Graubmann Peter** - Senior Engineer, CT SE, Siemens AG; e-mail: peter.graubmann@siemens.com*

***Kamaev Valery** - Prof., Chair Head for CAD Technologies of Volgograd State Technical University, e-mail: kamaev@cad.vstu.ru*

A METHOD OF CONTEXT-SENSITIVE HELP GENERATION USING A TASK PROJECT[1]

Valeriya Gribova

Abstract. *The article presents a new method to automatic generation of help in software. Help generation is realized in the framework of the tool for development and automatic generation of user interfaces based on ontologies. The principal features of the approach are: support for context-sensitive help, automatic generation of help using a task project and an expandable system of help generation.*

Keywords*: Ontology, task model, context-sensitive help, user interface development*

ACM Classification Keywords*: I.2.2 Artificial intelligence: automatic programming*

Introduction

One of the basic quality criteria of any software is learnability (reducing learning time). A principal characteristic of learnability is intuitive understandability of a user interface; however, a help system in software is also required. Complexity and functionality of software are increasing every year. As a result, the number of users who know all features of an application program (according to statistics, users are familiar with about 10% of application program functions) is decreasing. Therefore availability of a help system for users is important feature software.

Development of a help system is a costly and time-consuming task. Most of software has some context-free help realized as static guidance or a tutor. Static guidance can be realized in the form of an on-line help system. Nevertheless, searching information in large help systems is difficult for users, and cost of maintenance is very high (when an application program is modified, the help system must be modified as well). Tutors also have shortcomings because they teach users only some aspects of using an application program and have high cost of maintenance in the life cycle of an application program. Context-sensitive help realized in some model-based interface development environments [1,2] has a number of advantages over context-free help. The most important of them are automatic generation of a help system and using a current status execution of an application program when an answer in the help system is being generated. Nevertheless, the grave disadvantage of this system is that it cannot be expanded.

In this article an approach to automatic generation of expandable context-sensitive help is proposed. The help generator is a component of the tool for development and automated generation of user interfaces based on ontologies [3]. To provide help the help generator uses a task project. It is a component of the interface project to be used to generate the executive code of the interface. The task project is a tree. To expand the number of context-sensitive help types a script language has been developed. Recently the context-sensitive help generator has been implemented and introduced into the tool for development and automated generation of user interfaces based on ontologies.

Tools for help generation

There are next types of help in software, namely, context-free and context-sensitive. Traditional help systems are context-free and realized either in the form of a static guidance system or of a tutor.

A static guidance system provides help that is defined as a canned text at the development stage. To learn some aspects of software features users have to read a part or parts of the guidance. There are several kinds of this help: books, instruction manuals and on-line static help. On-line help is developed by authoring tools in various formats (HTML help, HTML-based Help, JavaHelp, Oracle Help, Adobe Portable Document Format – PDF, Macromedia Flash, WinHelp, AP Help, and others).

[1] The presented work has discussed on the KDS-2007. It has corrected in compliance with remarks and requests of participants.

Compared with instruction manuals, on-line static help is more convenient. However, it is difficult to find required information for large on-line static help so developers make a particular section called ”help for help”. In some cases it is useful to know a special query language. The help system and the application program are not interrelated.

Tutors simulate some application program behavior; so make the process of learning easy for the user. However, they teach users only the main functions of an application program. To find out other functions of the application program, users have to use instruction manuals and on-line static help. Development and modification of tutors is expensive and time-consuming, because all the alterations in the application program must be represented in the tutor.

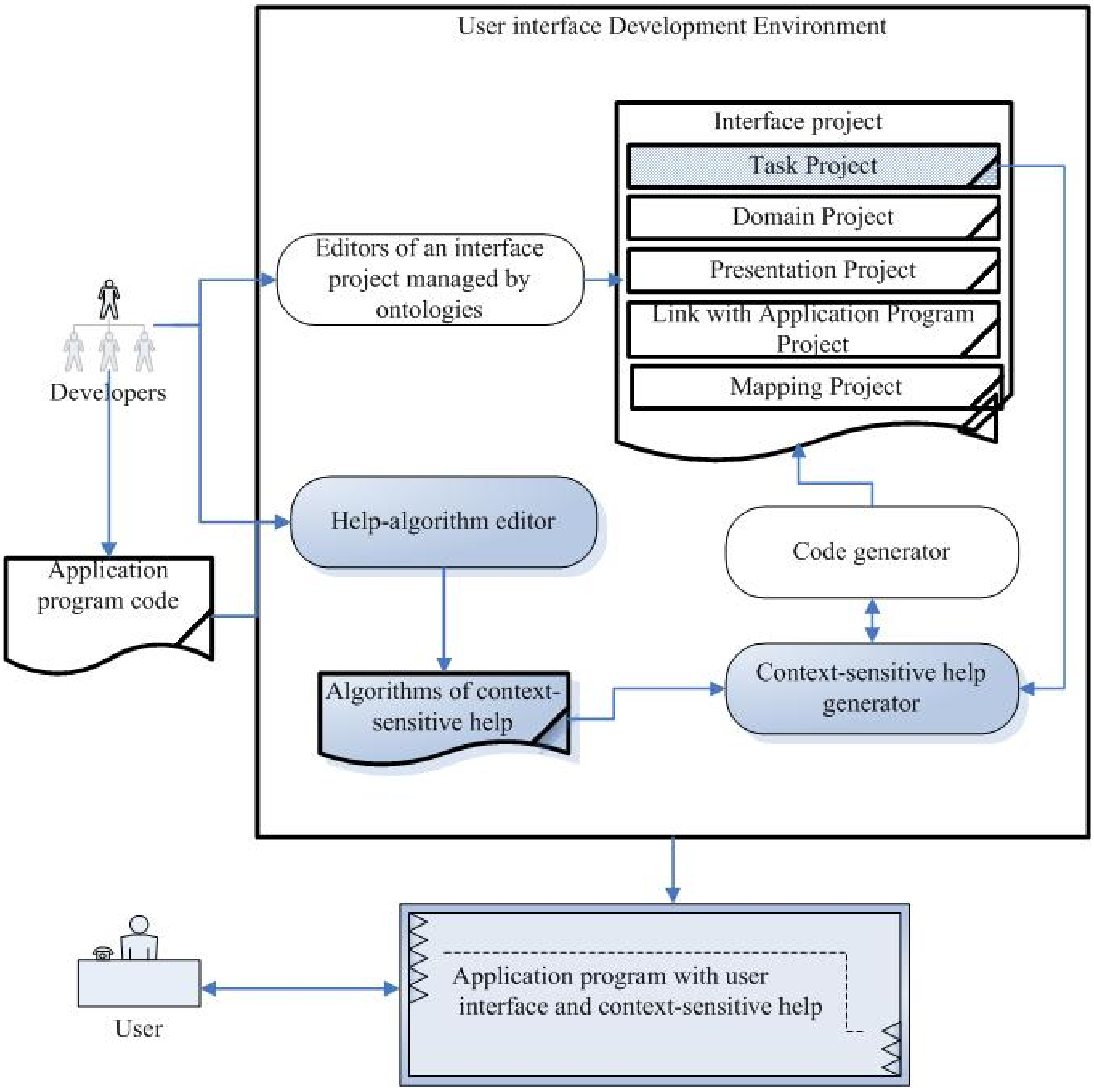

User interface Development Environment
Interface project
Task Project
Domain Project
Presentation Project
Link with Application Program Project
Mapping Project
Editors of an interface project managed by ontologies
Developers
Help-algorithm editor
Code generator
Application program code
Algorithms of context-sensitive help
Context-sensitive help generator
User
Application program with user interface and context-sensitive help

Fig. 1 The basic architecture of the expanding system of context-sensitive help-generation

Context-sensitive help is a kind of on-line help. There are two types of the context-sensitive help: conceptual and procedural. The conceptual help describes a framework of an application program, knowledge required to interact with it and the meaning of interface elements. The procedural help describes functions and operations of an application program. Unlike the conceptual help, it concentrates on user's tasks.

Implementation of the context-sensitive help is an important but very expensive task because it addresses to internal data structures. The most difficult objective is to provide conformity between the help system and the application program during the life cycle.

Some of the model-based user interface development environments (MB-IDE) have facilities for generation of context-sensitive help using an interface model. The principal method of context-sensitive help generation is based on transition networks. The transition network consists of a set of vertexes and arcs. Vertexes describe a set of valid states. Arcs show allowable user's actions. A help generator scans the transition network to form the context-sensitive help. A number of MB-IDE use a task model for help generation. The task model represents the tasks that users need to perform with the application program. It describes tasks by hierarchically decomposing each task into sub-tasks (steps) until leaf tasks become operations supported by the application program. Various specification languages are used for performing task models, for example, LOTOS, CCS [4,5]. Usually help systems can give answers to a set of questions which are task-related (*):

- Why this task is not available?
- How to realize this task?
- How to activate this task?
- What tasks can I perform now?

The basic advantages of context-sensitive help are coupling with the interface model (the task model is a component of an interface model) and automatic generation of help which makes the cost of development and maintenance of context-sensitive lower compared with other types of help.

Nevertheless, help systems support only a fixed number of questions. To add a new question to a system it is necessary to develop a new version of the help system. Tasks do not have an executing status so the quality of the help is lowered.

An approach to help generation

The state of the art in help generation enables us to set following some principle:

- Help should be context-sensitive;

- Help should be automatically generated at the time of user interface generation;

- Help should be invoked in any period of user's interaction with a user interface;

- Help should give answers to the set of questions marked * (see above);

- A help system should be expandable.

Context-sensitive help is realized in the framework of the tool for development and automatic generation of user interfaces based on ontologies (Fig. 1).The main idea of an ontology-based approach [3] to user interface development and generation is to form an interface project using ontology models which describe features of every component of the project and then, based on a component of this project (task project), generate a code of the user interface. The components of the interface project are:

- A domain project,
- A task project,
- A presentation project,
- A project of link a user interface with the application program (An application program project),
- A dialog scenario project,
- A mapping project.

Every component of the interface project is developed by a structural or graphical editor managed by an ontology model.

The domain project determines domain terms, their properties and relations between them. These terms describe output and input data of the application program and information on the intellectual support of the user.

The task project determines the tasks users can implement using the application program.

The presentation project determines a visual component of the interface and provides support for various types of the dialog.

The an application program project determines variables, types of their values shared by the interface and the application program, protocols for communication between the application program and the interface, addresses of servers and methods of messages transfer.

The dialog scenario project determines abstract terms used to describe the response to events (sets of actions executed when an event occurs, sources of events, modes of transfer between windows, methods of the window sample selection, and so on).

The mapping project determines relations between components of the interface project.

A context –sensitive help generator uses the task project of the interface project and a current task (the name of an executed task). The task project is a tree. The root of the tree is marked by the name of an abstract task which represents a set of application program tasks. The task project can be reused to design other interfaces if their application programs have similar features. Nonterminal vertexes of the tree are marked by names of abstract tasks. Terminal vertexes are marked by names of elemental tasks. The elemental tasks are tasks that cannot be divided into sub-tasks. Arcs of the tree do not have any marks; they link vertexes indicating the tasks hierarchy.

Every set Y can be divided into four types of sub-sets. The set Y={Y1, Y2,...,YN} is formed from marks of vertexes which are direct descendants of a vertex marked by X. Every sub-set of the Y set establishes relations among tasks. These relations are: choice, interleaving, synchronization and deactivation. These relations are taken from notations developed for specifying concurrent systems (LOTOS) [4].

- The sub-set called “choice” means that every task from the sub-set can be implemented; nevertheless, users can not start implementing any task until the previous task, i.e. the task which has begun implementation has been completed.
- The sub-set called “interleaving” means that every task from the sub-set can be implemented; users can start implementing any task of the sub-set while the previous task from this sub-set is been executed.
- The sub-set called “synchronization” means that only successful implementation of the sub-set task allows the user to implement other tasks of the sub-set.
- The sub-set called “deactivation” means that as soon as a task from the sub-set has been completed, implementation of the other tasks of the sub-set is broken.

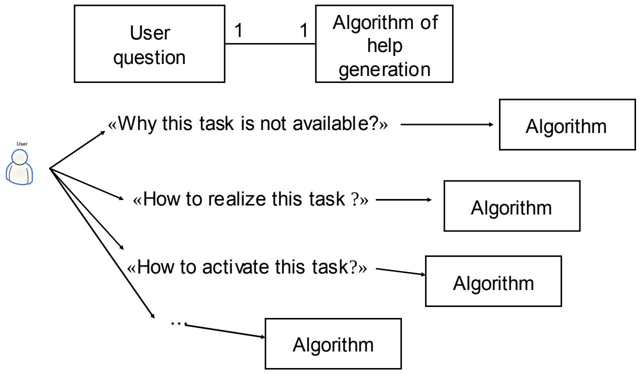

User question
1
1
Algorithm of help generation
User
«Why this task is not available?»
Algorithm
«How to realize this task ?»
Algorithm
«How to activate this task?»
Algorithm
...
Algorithm

Fig. 2. A principle of forming answers on user’s questions

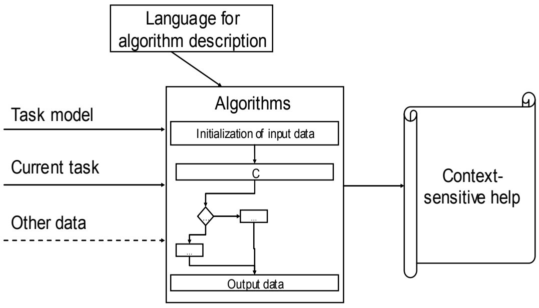


Fig. 3. A sceme of context-sensitive help generation

A process of the interface project design according to the ontology-based approach, in particular, requires that the developer of a user interface in the mapping project determine links between every task from the task project and interface elements from the presentation project. By handling an interface element, the user initiates implementation of a task linked to this interface element. As soon as an event of the interface element occurs, implementation the task begins. It means execution of an action chain defined in the scenario dialog project. For example, a task called “activation of the expert system" is defined in the task project. This task links to a push-button (an interface element) called, "activation of the expert system", in the mapping project. When an event called "keystroke" appears in the interface, i.e. it means that the user presses the push-button called by the name of this task, "activation of the expert system", the following chain of actions defined in the scenario dialog project begins to be executed: "to save data", "to pass data to the application program", "to represent new dialog window", etc.

To realize context-sensitive help, every chain of actions in the scenario dialog project must be extended by a system function. It passes the name of the executing task to the help generator. Inclusion of this function in the scenario dialog project automatically produces a sub-system of the help generator. Then, while interacting with an application program, the user invokes context-sensitive help. The help generator based on the user's query chooses an appropriate algorithm for help generation (Fig 2). Input data for this algorithm are: the name of the executing task and the task project (Fig. 3). To generate context-sensitive help the developer adds a set of interface elements used to call context-sensitive help to the interface project.

The principal requirement to the system of help generation is its expandability. To realize this requirement a script language for describing algorithms of help generation is proposed. This language consists of imperative constructions and a set of system functions which allow a new algorithm to be described. Using a structural editor the developer can add a new algorithm (a new kind of help generation). The added algorithm is transmitted to the XML-file and included in the algorithm base. Recently the main kinds of help generation mentioned in the requirements (see above) have been developed.

Conclusion

In this article an approach to automatic generation of context-sensitive help is proposed. The basic idea of the approach is to add an expanding system of help-generation to the tool for user interface development based on ontologies. The main task of the system is to form answers to user's queries using a name of the executed task and the task project. To date a prototype of the system has been developed at the Intellectual Systems Department of the Institute for Automation and Control Processes, the Far Eastern Branch, the Russian Academy of Sciences.

Acknowledgements

The research was supported by the Far Eastern Branch of Russian Academy of Science, the grants «An expandable system for quality monitoring».

Bibliography

a. Moriyón, R., Szekely, P., Neches, R.: Automatic Generation of Help from Interface Design Models. In C. Plaisant (ed.): Proceedings of CHI'94. New York: ACM Press 1994 (pp. 225-231).

b. Palanque, P., Bastide, R.: Contextual Help for Free with Formal Dialogue Design. In Alty J.L., Diaper D., Guest S. (eds.): Proceedings of HCI'93. Cambridge: Cambridge University Press 1993.

c. Gribova V., Kleshchev A. From an ontology-oriented approach conception to user interface development International //Journal Information theories & applications. 2003. vol. 10, num.1, p. 87-94.

d. Paternó, F., Faconti, G.: On the Use of LOTOS to Describe Graphical Interaction. In Monk A., Diaper D., Harrison M.D. (eds.): Proceedings of HCI'92. Cambridge: Cambridge University Press 1992 (pp. 155-174).

e. Sukaviriya, P., Foley, J.D.: Coupling a UI Framework with Automatic Generation of Context-Sensitive Animated Help. In Proceedings of UIST'90. New York: ACM Press 1990 (pp. 152-166).

Author's Information

Gribova Valeriya – *Ph.D. Senior Researcher of the Intellectual System Department, Institute for Automation & Control Processes, Far Eastern Branch of the Russian Academy of the Sciences: Vladivostok, +7 (4323) 314001 e-mail: gribova@iacp.dvo.ru, http://www.iacp.dvo.ru/is.*

TABLES OF CONTENTS OF IJ ITA – VOLUME 15

Volume 15, Number 1

Volume 15, Number 2

Volume 15, Number 3

Volume 15, Number 4

AUTHORS' INDEX

TABLE OF CONTENTS OF VOLUME 15, NUMBER 4