Bull. Nov. Comp. Center, Comp. Science, 28 (2008), 79–92

On structures of data and ontology of facto-graphical information systems

P. A. Marchuk, F. A. Murzin

Abstract. Structures of data and ontology arising in information systems containing facto-graphical data are considered.

1. The supported data

In this paper, the problems related to information systems containing facto-graphical data [1,2] are considered.

The first forms of data, which a base may contain, are objects of storage, such as photo, video, audio documents, or references to sources of actual publications. These objects are stored in museums, archives, libraries, etc. We will call them initial objects. In the case of references, initial objects are papers to which they refer.

Information about these initial objects is the second form of data that is meta-data. It may be the objects names, description, identifiers, and connections with other objects of the second form. For each selected initial object, the object of the second form should be created, and then this created meta-object is included into a semantic network. Besides, meta-objects are created for the real world objects, which should be represented in the information system: persons, organizations (orgsystems), actions, geographical places (geosystems).

To summarize, we say that, first of all, our interest is facto-graphical data, and the ontology of our system is based on these characteristics. We take into account the following four base types of concepts of meta-objects:

1. A person – this is, as a rule, for people, though it may be another living being.
2. An organizational system – it is an organization, action, department, association, club, etc.
3. A geographical system means a country, region, city (as a place), ocean, sea, river, a local place, etc.;
4. A document – it is some text, photo, video, audio, etc.

Our ontology is described by program systems rather simply for understanding, namely, in the OWL language [3] (Web Ontology Language). The data are stored in the system as XML-files in RDF format [4–6].

2. The structure of ontology

Let us consider new input meta-objects on an example of documentary photographs.

First of all, for a photo we need to find out, who is displayed on it. Representation is the relation in our ontology which means that some object is represented in some document. Using the standard form of the archival card of a photo and taking into account our available ontology, we should quickly answer the following questions concerning the photo:

1. When the photo was taken (date).
2. Where this took place (geosystem)
3. What is displayed on the photo: for example, an action (orgsystem).
4. Who is represented here: persons; may be, it is a building of some organization (orgsystem); or there is a river on a background (geosystem), etc.
5. Who is the author of this photo (person).

Thus, if the essence still is not represented, we should create it at once. If there is such an essence in our system, then we should put association on it instead of adding its duplicate. Of course, it is necessary to find it in the shortest time.

So, we came to an interesting problem of fast association of a new object with other objects already available in the base. Some appearing problems and decisions are offered in our system.

We can mention here the problem of absence of exact formal principles for creation of names. For example, it is typical for actions.

1. For official actions the names of which are yet unknown, for example, a conference may be called symposium or congress.

2. For informal actions that have no names, for example, the word "picnic" may be called 'departure to the country' or 'rest in the open air'. From the factographical point of view, such events may appear quite significant, for example, if it was "departure of VIP-persons to the country" or something like this.

We can help in these cases. First, we give experts the possibilities to enter different names for this action. In our ontology, this feature is given by means of the construction "name". Besides, we can also use the standard search using key words, information about a date and other properties of an object.

Advantages of our ontology are as follows.

1. We search not for an abstract textual document, but for an object with appropriate key words in its definition or with appropriate properties.

2. We can carry out a more flexible search using connections of the required object with other objects from the database (for example, a person working in a certain organization, etc.).

The problem of different names appears not only for actions, but also for people and organizations. For example, a person can change a family name or take a pseudonym. Organizations have short (reduced) names, names in other languages, etc. By means of the construction "name" and appropriate search procedure, we try to find an approach to decide this problem.

Axiomatic properties of notions here considered can be easily formulated.

1. A person, orgsystem and geosystem are objects
$\forall x\,((Person(x) \vee Orgsystem(x) \vee Geosystem(x) \vee Document(x)) \rightarrow$
$\rightarrow Object(x))\,.$
From this formula, it is also clear that introduction of other types of objects is possible.

2. An object may have a name, i.e. the following is true:
$\forall x \forall y\,(Name(x,y) \rightarrow Object(y))$.

3. An object may have a description, i.e.
$\forall x \forall y\,(Description(x,y) \rightarrow Object(y))$.

4. It is possible to use synonyms
$\forall P \forall Q\,(TheSame(P,Q) \leftrightarrow \forall \mathrm{x}\,(P(x) \leftrightarrow Q(x)))$.
Note that here we use the formula of the second order logic.

5. The property of the reflection relation can be written as
$\forall x \forall y\,(Rflection(x,y) \rightarrow$
$\rightarrow ((Person(x) \vee Orgsystem(x) \vee Geosystem(x)) \wedge Document(y)))\,.$

6. Similarly, for relations of participation and location, we have
$\forall x \forall y\,(Participation(x,y) \rightarrow (Person(x) \wedge Orgsystem(y)))\,,$
$\forall x \forall y\,(Location\ (x,y) \rightarrow$
$\rightarrow ((Person\ (x) \vee Orgsystem\ (x) \vee Geosystem\ (x) \vee Document\ (x)) \wedge$
$\wedge Geosystem\ (y)))\,.$

In an information system, there also are various notions that can be formulated only with the help of a modal or temporal logic, such as, for example "it is possible", "it is necessary", "it is true at a certain moment of time", and "it is true in a time interval. There are several time scales allowing us to work with the notions like centuries, years, months, days, hours and minutes. All these questions should be investigated.

3. The problem of analysis of a database as a whole for searching objects for association

Certainly, the process of search through a database as a whole takes much time, especially if the query for this search contains not only key words and direct properties, but also connections with other objects. Besides,

documents are frequently entered in a series. Therefore, for simplification of input of new documents, the system can help a user by checking the context of newly entered documents.

Let us consider some possible variants of our contextual control. The simplest one is the use of static parameters. For example, it may be the author of a document, or the source of a receipt. Less often it may be the date of a document, a selected event or action reflected in this series of documents.

Although we speak about the context of our series of documents, the associated context is not a physical folder for storage of our documents. Also, this contextual information not necessarily specifies virtual folders connected with these series of objects. The term "virtual folder" is frequently used in modern structures of data storage, however, it is only one direction of structural analysis. In our ontology, we establish an appropriate associative connection with each object (for example, a connection with the author of this document), or we indicate some selected property (for example, a date). But in some areas, these documents may be grouped according to the preferences of the information expert, and in this case they will be represented as virtual folders ("collections" in our ontology).

Although storage of a context is implemented by updating properties or establishing the connections for each object, nevertheless, for acceleration of input, we should define the context for all entered series of documents at once.

In addition to static parameters named above, there are dynamic parameters, i.e. such that they are very similar in these series of documents, but there is no complete coincidence between them. For example, it may be a list of persons represented in a given document. If we have series of documents with frequently repeating persons, then we use the notion of context.

As an elementary variant of decision of this problem, we can output a list of recently used objects in a given form, for example, a list of persons with whom associative connection was recently established. Thus, if we output them in a decreasing order of time of their last consideration, then we often have minimal costs, and it is essential to increase the speed of creating associations between documents and persons.

Here we briefly mention more complex possibilities to select contextual dynamic parameters. Besides their recent use in our input, we have an available semantic network between other documents and persons, between organizations and persons. In addition, we have a chronological distribution of objects, their geographic distribution, and so on. With the help of more complex algorithms, we can find out the necessary context and accelerate input of new documents.

4. Distributed editing of data

In this section, we consider meta-objects contained in our information system. Some of them can be integrated in our system, others may be in the removed position in the format necessary for us. Also, some of them may be on the other removed servers and, in addition, in the other format, but we have import-filter for their use. In general, from initial sources, we receive RDF-documents in which meta-objects are stored. In aggregate these RDF-documents give us a uniform information field with which we should work.

Work with these meta-objects in the reading mode is rather simple, if we have no problems with identifier duplication. For example, we should find an object to look at its properties, or we should find objects which refer to this object to display their properties. The methodology of this search of objects, parts of our associative graph, is described, for example, in the search language SPARQL Query Language for RDF [4].

In this section the case is considered when we need to edit information in our information field, but we have sources which are closed for editing. First, we consider the basic feature of our ontology. In the standard concept of RDF graphs, associative connections between objects are fixed with the help of RDF triples. The subject, predicate and object compose a triple (see Figure 1, the upper part). As the basic scheme of establishment of associative connections between objects in our ontology, we use the following approach. The associative connection which we establish is stored as a meta-object in the database and it refers to the objects of this connection (see Figure 1, the lower part). Actually, one connection is represented by two triples.

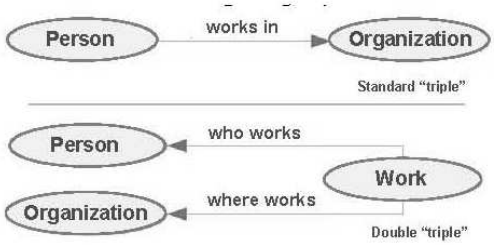

Figure 1. Two forms of associative connections between objects

Now we will define the working context. Let us consider an elementary model with only 2 sources in our uniform distributed information system. Our source, where we may edit data, and another source, where we may not do this, but we may look it through. And it cannot look through or edit our data. Thus our area of viewing is the whole distributed system, but the context of editing is only our source.

Proceeding from this model and our ontology, we can add to our context new meta-objects, edit their properties and establish connections between them. Besides, we may establish connections between objects of different sources (see Figure 2). Thus, having all sources as an area of viewing, we receive a full information picture. The second source cannot receive a full picture, because, by our definition, its area of viewing does not include our source, but our added objects will not generate contradictions.

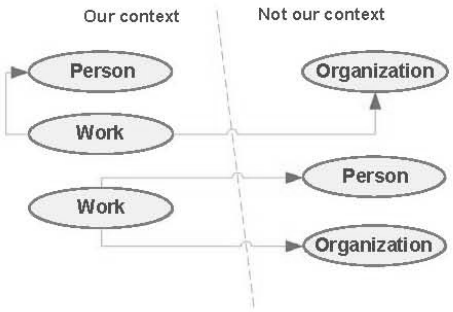

Figure 2. Connections between objects of different sources

In some sense, the basis of a semantic network is connections between objects, but not the selected properties of objects. For example, if we create a lot of new meta-objects, then the properties of these objects begin to play an appreciable role in view of their plurality. We may change a property already available, for example, a wrong name of an organization or add a new property, for example, the date of foundation. Besides, we may consider a situation, when, for example, a person had never worked in a given organization, but there is an appropriate mention in the database. Thus we should remove this meta-object. It is clear that it is possible to remove it from its own context, however it becomes a problem in the case of another context.

In the case of the main centralized database, this problem is solved at the level of authorization. The right of editing is given to the user only in a certain area of data. It is possible that editing should be confirmed by a moderator. In this case, even after changes, we may cancel them and keep the previous unchanged version. Basically, we can transfer this scheme onto the distributed model of data storage. Then each source may determine priorities of users of information system and supervise some "bush" of data. A disadvantage of this approach is, for example, obligatory availability of a server part tracing the priorities of all users. Thereby, the properties of these objects become a part of a bush of data.

Let us consider the case when there is no server part determining which objects may be edited. For example, we simply get other bushes by http-

protocol. In this case, we still have to make changes, however we should make them only in their own context of editing. Thus, at visualization, changes should be shown correctly, i.e., at least in our part of the system, they should be shown as we want to see them.

So, the following variant of a decision is offered to make changes of properties of meta-objects placed in a remote bush. We create a new essence in the context of editing and allocate it with the same properties, as in the remote object. Now this meta-object is in our bush, and it is possible to make all changes necessary for us. However, the semantic network still has references to the old object.

In principle, in the context of editing, we may replace all references to old objects by references to new added objects, however, the references to old objects remain in all other sources. Therefore we add an instruction for looking through the system that there was a change of identifiers, and new objects replaced old ones. Then the information system should automatically replace old identifiers by new ones.

We act similarly in the case when we have more than 2 sources and each of them has all sources as its area of viewing and we can edit properties of all meta-objects that we can see. In this case, we need to replace the identifier as many times, as the number of those copying of meta-objects from one source to another.

Slightly changing this scheme, we can similarly delete meta-objects from other bushes, indicating in our context that this meta-object does not exist (for us).

5. Distributed file systems

This section presents some of the most significant software systems for distributed file systems [7] which are very important for our research.

DNS and GNS. The Domain Name System (DNS) has a long history. The Global Namespace Service (GNS) is still at the initial stage of development. The architecture and existing implementations used by DNS may be suitable for holding GNS data, especially for naming organizations close to the root of the namespace.

LDAP. Lightweight Directory Access Protocol (LDAP) is perhaps the most prominent directory service available today. LDAP defines a communication protocol for access to and search in a database of entries annotated with attributes. Accordingly, LDAP defines not a directory service, but rather the transport and format of messages used by clients to access data in a directory. These messages deal with directory entries, whose structure and relationships are described by a schema.

RDBMS. Relational database management systems (RDBMS) generally support transactional operations. Most RDBMSs incorporate optimized

mechanisms for data replication, with transaction-based features, in a distributed environment. In fact, certain products, such as IBM's eNetwork LDAP, use a relational database for their implementation.

CIFS and NFS. Traditional distributed file systems are Network File System (NFS) and Common Internet File System (also called SMB [Server Message Block]), collectively called NAS file systems.

AFS and DFS. In principle, experience with AFS (Andrew File System) and DFS (Distributed File System) has demonstrated many of the benefits of a global, uniform, hierarchical namespace. These file systems feature excellent security and good performance over a wide range of geographies, numbers of users, servers, clients, and data capacities.

Avaki. The Avaki Data Grid product provides a uniform namespace built from heterogeneous file systems using a proprietary solution. The system-level architecture at a domain level consists of a network of share, a high speed grid, and access servers, as well as a grid domain controller, which collectively manage a global namespace.

SRB. Storage Resource Broker (SRB) from the SDSC (San Diego Supercomputing Center) is "middle" software that provides a comprehensive distributed data-management solution, with features to support the management, collaborative (and controlled) sharing, publication, and preservation of distributed data collections. It has a rich set of APIs (application programming interfaces) to the management layer providing a uniform interface for connecting to heterogeneous data resources over a network and accessing replicated data sets. The APIs allow us to create higher-level applications which can be on top of a wide variety of storage systems.

Globus. The Globus Toolkit contains key features needed for a distributed file-system. It provides strong security based on a public key infrastructure using server certificates and user proxies.

Plan 9. The Plan 9 operating system uses the file system namespace to represent, control, and monitor virtually all system resources. The file system's message-based interface, called 9P, provides a uniform access method for all these objects.

6. Image processing and further research directions

In this paragraph, we consider the possibilities of using the program IMan [8] for face recognition, as an additional service of a facto-graphical system.

Sometimes it is important to find quickly an image on another one. A simple example is a machine vision system intended to recognize parts of a certain shape on an assembly belt. Digital images are usually raster and have a rectangular shape, so we presume that all images are rectangular. That is, if we have a part of an image (pattern), the task is to find this part on the whole image (source). The pattern may be rotated with respect to

its position on the source or it can have another scale.

The program IMan takes a source and a list of patterns, finds out if each pattern is fully included in the source and determines the rotation angle and the scale coefficient for them. The search results are placed in the search report which is generated as an html-file and can be viewed using the internal IMan report viewer or in a web browser installed in your system (MS Internet Explorer 5.0 or later is recommended).

Before the search, all images are preprocessed, which takes a lot of time, but this preprocessing should be performed only once. The search stage is time-critical and this is the most important factor of IMan effectiveness. In practice, this time is limited by the requirements of the technological process.

The research results can be used in the following areas.

1. Satellite object recognition, aerial photography and terrain conjunction.
2. Photo images can be stored in an archive and then this algorithm can be employed for search in the archive.
3. On a production line, the algorithm can select the components and act as machine vision for automated assembling machines. Thus, it can be used in complicated robotic productions.

Figure 3. The main window of the program IMan

The basic principles of the "Algorithm Based on Support Points" are described below.

On the first stages of the algorithm, we apply Low–Frequency and Median Filtration. These transformations are done on Control and Source Bitmaps.

Usually Low–Frequency Filtration is called the smoothing techniques. It makes the graph of luminosities smoother. The smoothness is useful in the situations when rotated images are processed. This fact is well known, but it is not justified theoretically. We can say that the fact is experimental. Two-dimensional Median Filtration allows us to suppress the impulse noise, i.e. exclude the individual defective pixels. Simultaneously we reject (erase) points with high-frequency color characteristics. It makes the algorithm more stable and decreases the number of considered pixels.

The next stage is Contour Separation. It allows us to perform further image simplification. This approach is also well known.

The main idea is the use of support points. We select support points on Control Bitmap and then search for analogous points on Source Bitmap. The problem of selection of support points on Control Bitmap is the most interesting and still poorly investigated. Usually the luminosity characteristics are used. Also, one may use detectors of various properties, topological properties analysis, gradient methods and space frequency filtration methods. The method we used is heuristic, but it showed good results. We select points maximally far from each other in some metric which is a linear combination of the color and Euclidean metrics.

In the process of searching support points on Source Bitmap, a special structure associated with Source Bitmap is used.

We spend some time for building the structure. The time is proportional to the image size $n \times m$. But after this procedure, we obtain the possibility to find support points in the logarithmic time, i.e. in $C \cdot \log(n \times m)$ steps.

The structure contains information about various partitions of Source Bitmap onto rectangles. Each following partition is finer than the previous one. On the first step, there is no partition. The finest structure has rectangles consisting of one pixel. A special array is associated with every partition. In this array, information about colors is written.

The previous (more coarse) structure in this sequence will be easily constructed from the following finer one. Note that all these partitions can be represented in the form of a tree.

We check if there exist points with the same colors as support points on Source Bitmap within the precision of a coarse palette. The coarsest structure (when there is no partition) allows us to do it immediately, because existence of the corresponding colors is coded in the structure.

On the finer partition, we try to find a distribution of support points inside the corresponding rectangles selected at the previous stage. This is performed recursively, which means that we should pass a tree describing an embedding of rectangles into each other. Thus, using more and more fine

structures, the support points will be fixed.

The simplest method of passing a tree is lexicographic. Here different optimizations are possible, for example we can take into account the distances between support points and other information.

After fixing the support points, we check whether their position is in the list of bad positions. This list enlarges step by step, and it is empty at the beginning.

If the support points are in the list of bad positions, then we return to the algorithm of searching support points (this means that the final compare failed, and we should continue the searching process), otherwise we calculate the coordinates for corners of the rectangle and rotate it. Simultaneously we verify whether the corners of the rectangle are placed inside Source Bitmap.

So, the algorithm called Quick Rotation Algorithm is used to map the rotated image placed inside Source Bitmap onto the image not rotated and placed in the origin of coordinates.

At the stage of final compare, we take Control and Source Bitmaps from different scanners. Therefore, Correction of Brightness and Contrast is necessary.

In addition, we use Refinement of Threshold in the Measure of Nearness of images.

A natural question arises from which criteria the threshold for the measure of nearness should be taken. We must be sure that if its value is less than some fixed threshold, then the images are almost identical. Such a threshold is called admissible.

The following method is proposed. We slightly vary the position of Control Bitmap and calculate the threshold on the intersection of Control Bitmap with its variant obtained after varying. Collecting the values of threshold for various positions, we can study this information and suggest the correct threshold.

At the stage of final decision, we should decide whether two Bitmaps are identical or not. The distance between them in some metric should be less than the admissible threshold. In this case we admit that Bitmaps are identical, otherwise we return to the process of searching the support points on Source Bitmap.

Now let us discuss the possibilities of using the program IMan for face recognition, which can be very useful for facto-graphical systems.

The natural area of application of the program IMan is robotics, i.e. robot-technical systems. In this case, the typical situation is that there are many samples (Patterns) to be found inside one large image (Source).

In face recognition, the situation is inverse in some sense. One sample, for example, an eye, can be given and there are a lot of source images-faces.

The program IMan is very good among programs of a similar type. The typical time of fragments search is tens milliseconds. And it's not a problem

to transform it for the case when we have a lot of images-sources.

The program IMan finds samples up to parallel transfer, rotation and zoom transform (scaling). In the case of face recognitions, it is necessary to provide some other conversions (transformations) according to specificity of the given area. For example, an eye can be more or less open.

In addition, morphing is to be connected, and we should search for patterns with accuracy up to morphing. It is impractical to consider arbitrary affinities. It is known that in this case we can obtain unacceptable run time.

Taking into account our experience with the images in infrared area, we can conclude, that IMan will work effectively in this case. An example of application of IMan is given below.

Figure 4. The result of search for a person with a given eye

6.1. Implementation

The work was carried out within the framework of the project "Electronic photo-archive of the Siberian Branch of the Russian Academy of Sciences". The approaches considered above were implemented as modules included into information system. In addition to the interface for input, editing and internal information search, there exists a public interface accessible at: `http://soran1957.ru/`.

There is a difficult problem of different data formats. For our purpose, we use an updating standard of metadata Dublin Core [9].

References

[1] Marchuk A.G. Distributed electronic archives, libraries and databases. – Novosibirsk, 2004. – 25 p. – (Prepr. / IIS SB RAS; N 122). (In Russian)

[2] Marchuk P.A. A new class of technologies for creation of personal and distributed information systems // Proc. of Conf. "Microsoft Technologies in Informatics and Programming". – Novosibirsk, 2008. – P. 143–145. (In Russian).

[3] Web Ontology Language (OWL). – `http://www.w3.org/2004/OWL`

[4] SPARQL Query Language For RDF. –
Available at `http://www.w3.org/TR/rdf-sparql-query`

[5] Resource Description Framework (RDF). – Available at
`http://www.w3.org/RDF`

[6] Sesame, open source RDF framework. – Available at `http://openRDF.org`

[7] Anderson O.T., Luan L., Everhart C. et al. Global namespace for files. – Available at `http://www.ibm.com/journal/sj/434/anderaut.html#Anderson`

[8] Dunaev A.A., Lobiv I.V., Mekhontsev D.Yu. et al. Algorithms of fast search for fragments of photoimages // Modern problems of program construction. – Novosibirsk, 2002. – P. 88–109. (In Russian)

[9] The Dublin Core Metadata Initiative (DCMI). – Available at `http://dublincore.org`