Simple Method for Hierarchical Conceptual Indexing of Documents Using Relational Data Model

A. GELBUKH, G. SIDOROV, and A. GUZMÁN-ARENAS
Natural Language and Text Processing Laboratory,
Center for Computing Research, National Polytechnic Institute,
Av. Juan Dios Batiz s/n, Zacatenco 07738, DF
MEXICO

Abstract: - A method for indexing and retrieval of textual documents is presented. The method allows for a powerful query language with Boolean and proximity expressions (the latter refers to the possibility to search for two or more keywords within the same paragraph or sentence or within a certain distance from each other). Specifically, the method is designed for search with generalization (by the query *mathematics*, to find the documents mentioning *function*, *matrix*, *theorem*, etc.) and to control morphological normalization (let the user decide of the query *matrix* implies *matrices*). Another advantage of the method is the possibility to mix textual (keywords) and relational (field values) data in one query. The query language is internally translated into SQL. A system implemented by the authors according to the proposed method is described. Advantages and disadvantages of the method as compared with the query enrichment method are discussed.

Key-Words: - Information retrieval, hierarchical indexing, concept tree, relational model, automatic morphological analysis.

1 Introduction

Traditional keyword-based information retrieval systems [1] lack flexibility in at least three important aspects:

1. They do not allow performing search with generalization. For example, to find the documents on mathematics, the user has to type all words such as *function*, *vector*, *matrix*, *differential*, *theorem*, etc., since it is unlikely for a paper on mathematics to explicitly mention the word *mathematics*.
2. They do not give the user control over matching of inflectional forms. The system either always or never matches the query *matrix* with the word form *matrices* occurring in the text, regardless of the user's preference. Though this problem is not very important for English, it *is* important for many major languages such as Spanish, French, or Russian, where a lexeme can have many (for some languages, hundreds) of inflectional forms.
3. They do not allow the user to control how close two (or more) given keywords, say, *European tours*, should occur in the text. Though most information retrieval systems do distinguish between the keywords occurring anywhere in the text (e.g., a newspaper mentioning *tours to Australia* in one article and *European industry* in another one) and a contiguous phrase (e.g., *We offer European tours for only $650*), such systems usually do not allow to search for the words occurring within a certain distance, sentence, or paragraph (e.g., *We offer tours to all European countries*).

In this paper we describe a system that does not present such drawbacks.

1.1 Related work

1.1.1 Database issues

A partial solution to the first of these problems is a system that permits navigation in a topic hierarchy, such as, for example, Yahoo. Usually such systems require manual document indexing, which is expensive, imprecise, and subjective. Also, existing systems such as Yahoo lack a powerful query language that would allow for combination of different search conditions. However, the most important drawback of such document hierarchies is that a document is placed at some specific node of the tree and thus cannot be retrieved at a more general node (i.e., a document placed at *algebra* cannot be retrieved at

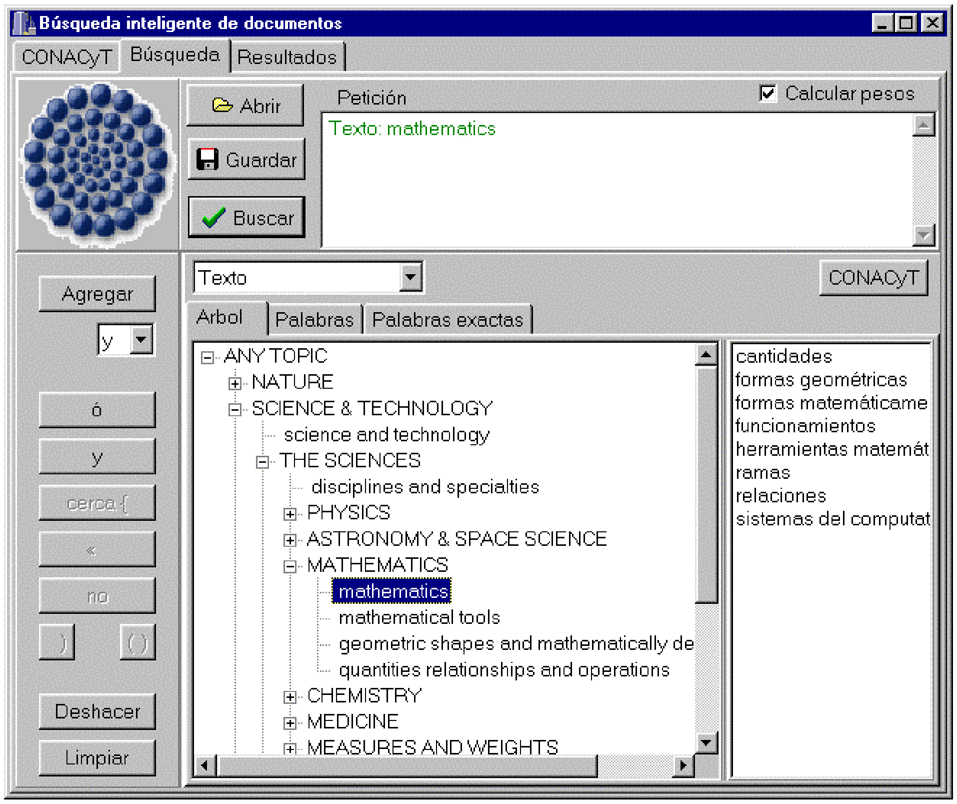


Fig. 1. Query using the concept tree.

mathematics). Thus, such document hierarchies do not provide a solution for the problem of retrieval of specific documents by a general query.

The problem of manual indexing can be solved with automatic document indexing using a concept hierarchy dictionary (concept tree) [4]; in this paper we assume such indexing method.

One can suppose that for handling hierarchical queries, a hierarchical database model is necessary. However, conventional relational databases are more attractive because of their availability and, most important, availability of the corresponding search tools and languages (specifically, SQL).

In [2] we have explored the possibility to use a very simple relational database model (using only simple Boolean expressions) for the task of keyword search with generalization. There we proposed a query expansion method. With this method, the document-indexing procedure does not use the concept hierarchy; instead, the concept tree is used at the query time (thus, changes to the concept tree do not require re-indexing). Though this method has important advantages, they are at the cost of very large query expressions used internally by the system. In this paper, we explore a possibility to considerably reduce the size of the query built internally by the system, though at the cost of slightly larger index and the necessity to know the concept tree at the indexing time.

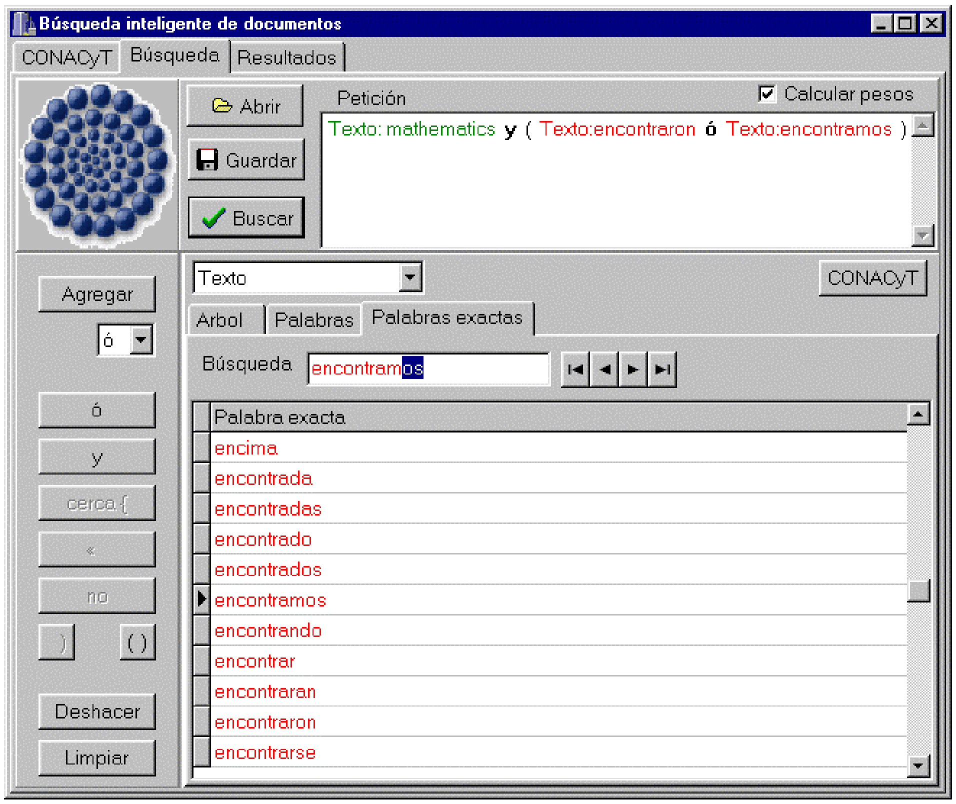
1.1.2 Linguistic issues

There are several linguistic techniques that allow for more precise and effective document processing for further retrieval, see [3, 5], such as morphological analysis, synonymy, etc. For example, morphological analysis permits to reduce several wordforms to only one lexeme (which may be not so important for the analytical language as English, but is absolutely necessary for many other languages).

For example, with this technique the query *stand* would retrieve the documents containing the wordforms *stand*, *stands*, *stood*, and *standing*.

In this paper we propose to index the documents using the concept tree and also applying automatic morphological analysis.

Since the system has to be able to search by a specific wordform (say, *matrices* but not *matrix*), information on the location of individual word forms within documents is to be preserved, i.e., like in [2], what is to be indexed are wordforms rather than lexemes. However, unlike [2], here we propose an indexing scheme that allows naturally expressing in SQL the queries concerning lexemes or concepts.



Petición [✓ Calcular pesos]

Texto: mathematics **y** (Texto:encontraron **ó** Texto:encontramos)

Texto

Búsqueda: encontramos

Palabra exacta
encima
encontrada
encontradas
encontrado
encontrados
encontramos
encontrando
encontrar
encontraran
encontraron
encontrarse

Fig. 2. Query using specific wordforms and Boolean operators.

The main idea of the proposed method is to give consecutive numbers to all wordforms corresponding to the same lexeme, and similarly to the words that correspond to each node in concept tree.

Currently we use relational data model for the index (though it may not the best model for our purposes). Namely, we use one of the standard database engines. Our system implements a rather sophisticated query language internally translated into SQL.

The system was developed for Spanish language, though the only language-specific components of the system are the morphological table (a table with two columns relating wordforms to lemmas: *stood* – *stand*) and the word lists of the thesaurus (relating a concept with the corresponding words: *mathematics* – *formula*, *matrix*, *theorem*, etc.). Thus, it is easy to adapt the system for different languages by replacing this table.

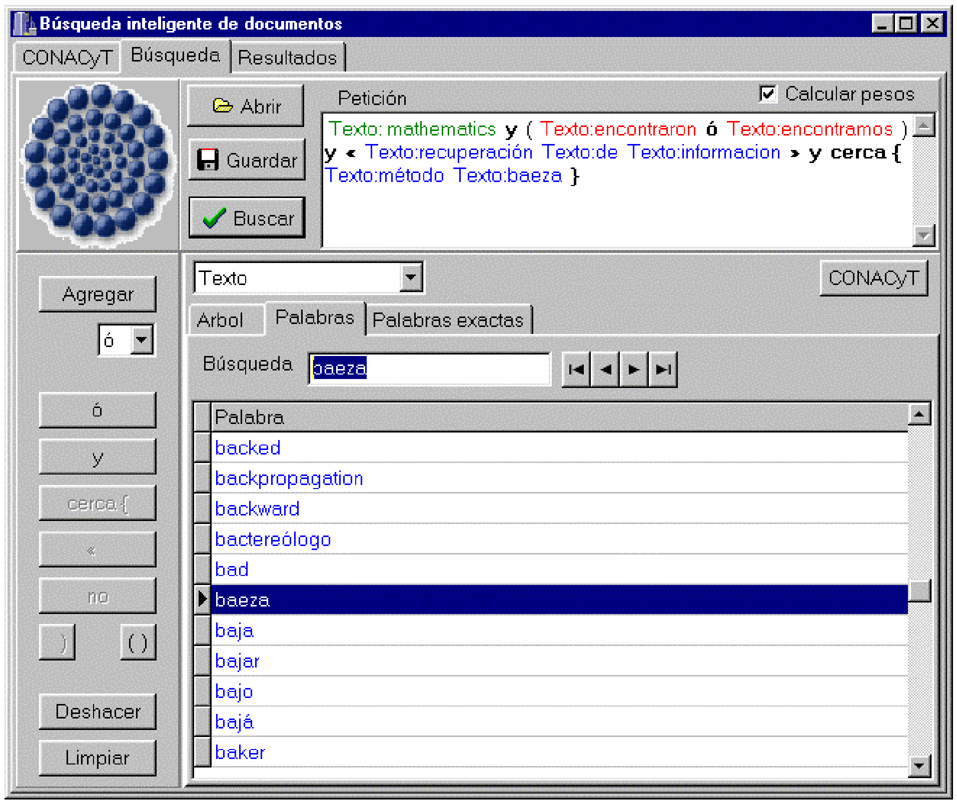
In the rest of the paper, we describe in detail the procedure of hierarchical indexing and then briefly describe the system and its query language.

2 Hierarchical Indexing

To find documents about *science*, one needs to search for documents about *mathematics*, *physics*, *chemistry*, etc. Specifically, to search for *mathematics*, one needs to retrieve the documents that contain such words as *function*, *theorem*, *matrix*, etc. Finally, to find the documents mentioning a *matrix*, one needs to look for the letter strings (wordforms) *matrix* or *matrices*. This can be done with a thesaurus that organizes concepts and words in a tree: concepts (e.g., *science*) include other concepts (*mathematics*) or specific words (*matrix*), the latter being further subdivided into specific wordforms (*matrix*, *matrices*).

The concept tree has the following structure: the terminal concept nodes are supplied with lists of lexemes that correspond to them (each lexeme being supplied with a list of its wordforms); the non-terminal concept nodes join several terminal or non-terminal nodes (see the example below). Each node has only one parent node except the uppermost node that does not have parent. (In fact, some nodes of this hierarchy should have belonged to several different parent nodes—say, the string *well* belongs to both *good* and *oil drilling*; however, for sake of simplicity of description we will ignore this complication, duplicating the corresponding elements when necessary.) Thus, the terminal nodes of this structure are specific wordforms.

The main idea of our indexing scheme is to establish a correspondence between each node in this tree



Petición: Texto: mathematics **y** (Texto:encontraron **ó** Texto:encontramos) **y** « Texto:recuperación Texto:de Texto:informacion » **y cerca** { Texto:método Texto:baeza }

Búsqueda: baeza

Palabra
backed
backpropagation
backward
bactereólogo
bad
baeza
baja
bajar
bajo
bajá
baker

Fig. 3. Query using lexemes and proximity.

(not necessarily the terminal one) and the words (wordforms) that are below this node in the way that the words are indexed with contiguous intervals of numbers.

To assign the correct numbers (identifiers) to the nodes (both terminal—wordforms—and non-terminal), we traverse the tree in the depth-first order and enumerate all terminal nodes (wordforms) starting from 0. In this process, the morphological analysis is applied to generate all wordforms of each lexeme node.

Non-terminal nodes are assigned identifier ranges (i.e., pairs of numbers) corresponding to the span covered by their sub-nodes, i.e., the first and the last node below the given one. This is done in a one-pass process: the initial number is assigned to a non-terminal node when it is entered and the terminal number when the enumeration process leaves the node.

After this processing, the tree looks like the following (only the concept nodes are shown; terminal concepts are shown in lowercase):

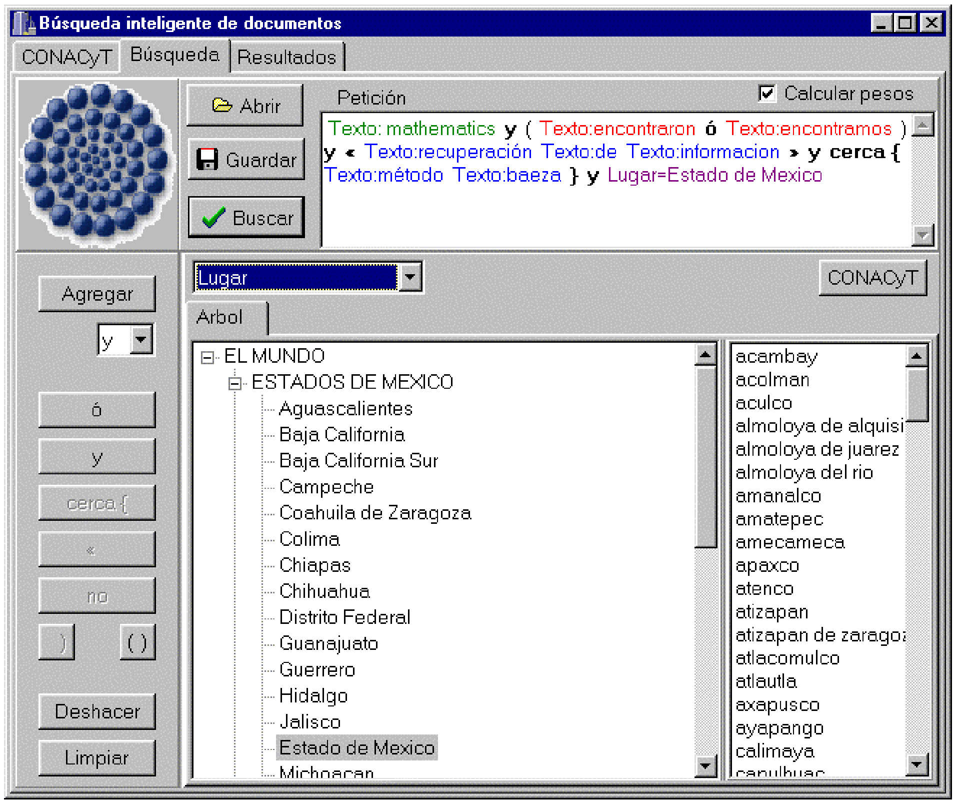
```
0 ANY TOPIC: 0–4624977
1     NATURE: 0–1282748
2         nature: 0–4
2         HUMAN BODY: 5–368954
3             the human body: 5–80
3             ANATOMY: 81–31261
4                 anatomy: 81–162
4                 reproductive system: 163–221
4                 respiratory system: 222–252
4                 skeleton: 253–15288
etc.
```

with, say, the wordform *crania* being assigned the identifier 262 (below *skeleton*).

As one can see, currently the total number of wordforms present in our hierarchy is 4,624,978—the higher value of the *ANY TOPIC* node (the root node).

In case if a word belongs to two or more different terminal nodes it is indexed several times with different identifiers (with all its wordforms); if a wordform is ambiguous, it is also assigned as many identifiers as necessary. Later, at the stage of indexing the documents, the text containing such a wordform is indexed as if there were several different wordforms in the text: e.g., *well* is indexed as if the text contained "*well*$_1$ *well*$_2$" with different identifiers, one below the node *good* and another *oil drilling*).

Having the concept tree enumerated, it is possible to begin indexing the documents. If the indexing process finds in the documents a wordform that is not in the tree, it gives them (with all their grammatical forms) identifiers continuing the enumera-



Texto: mathematics **y** (Texto:encontraron **ó** Texto:encontramos)
y « Texto:recuperación Texto:de Texto:informacion » **y cerca** {
Texto:método Texto:baeza } **y** Lugar=Estado de Mexico

Fig. 4. Query mixing textual and relational data.

tion of the concept tree (i.e., the next available number). Currently, we consider all such wordforms as belonging to a special concept *unknown word*.

After the documents have been indexed, we obtain a relational database that for each running word of each document contains the following data fields:

- the identifier of the wordform,
- the position of the running word in the document (starting from the first word of the document),
- optionally the number of sentence or paragraph.

With these data, it is easy to search using a rich language combining in the same query individual wordforms, lexemes, or concepts. An example of a query is given in the next section.

3 Implementation

We have developed the system that implements the indexing method described above. The system implements a sophisticated query language with Boolean operations and proximity criteria, translated internally into SQL. The main advantage of the system is the possibility to search mixing concepts, lexemes, wordforms, and other available information about the documents (for example, for a scientific project, the initial and final date, funding, responsible person, institution, etc. can be available).

The queries are translated into SQL in an obvious way:

- An individual wordform, say, *cranea*, is translated into an expression "*x.id* = 262" (where 262 is its identifier and *id* is the name of the identifier field);
- A lexeme, say, *cranium*, is translated into an expression "*x.id* >= 261 and *x.id* <= 262" (where the numbers correspond to the range covered by the lexem); this is especially useful for lexemes ðôìø òï hundreds of wordforms, as is true for Spanish or Russian verbs;
- A concept, say, *ANATOMY*, is translated in a similar way: "*x.id* >= 81 and *x.id* <= 31261".
- Proximity, say, within 8 wrds is translated into "*x.document* = *y.document* AND mod (*x.position* – *y.position*) < 8"; proximity within, say, 2 paragraphs is treated similarly.

Such expressions can be combined using Boolean operators. As the result, the value of *x.document* field is used to compile the list of found documents.

The system implemented for Spanish language was tested on the descriptions of research projects developed in our Center, so that the database has some traditionally relational fields mentioned above.

Sample screens of the system are shown in the figures. In Fig. 1, a query using concept tree is shown. The query is shown near the top of the window. In the left-hand part of the window, the buttons corresponding to the operators of the query language are located. The central part of the window contains the concept tree (we use English for concept labels), from which the user can choose the necessary node. Individual lexemes corresponding to a terminal concept node are shown in the right-hand part of the window (in our implementation, the lexemes are Spanish). A dropdown control in the center of the window indicates that the data are to be searched in the text of the document (see below the discussion of Fig. 4); the same is indicated in the query area.

In Fig. 2, specific wordforms of the Spanish word *encontrar* 'find' are added: *encontraron* 'they have found' and *encontramos* 'we have found', while, say, *encontraste* 'you have found' is not included. The wordforms are chosen from the list.

In Fig. 3, an exact phrase (a sequence of lexemes *recuperación de información* 'information retrieval' in this case) and a proximity expression *método Baeza* 'method ... Baeza' are added. The latter corresponds, say, to a phrase *The method suggested by Ricardo Baeza Yates.*

Finally, in Fig. 4 another type of the information is added: place (*lugar* in Spanish) where the project was proposed, in this case the Mexico state (the right-hand part of the window shows specific cities in the Mexico state; we use lowercase for all words).

4 Conclusions

We presented a simple method for document indexing and search with hierarchical generalization, control of morphology, and with combination of textual and relational (tabular) data using a powerful query language with Boolean and proximity expressions. The method is based on the use of traditional relational data model; the queries are internally translated into SQL.

Technically, the method consists in enumeration of all possible (used) strings (wordforms) in the depth-first order in the tree, and using the obtained identifiers in SQL expressions for the nodes of the tree: a node is associated with the first and the last identifier of a string below the node. In case of ambiguity a string is assigned several different identifier, one for each node it belongs to. The documents are indexed using these identifiers for each string found in the text.

An advantage of the method as compared with [2] is much smaller internal queries (thus faster search). A disadvantage is a large index and the necessity to re-index the documents if the concept tree or morphology table is significantly changed (for slight changes, holes can be left in advance in the enumeration of the identifiers).

The method can be easily adapted (this is also implemented in our system) to assign relevance weights to the documents according to the frequencies of the keyword found in the text.

References:

[1] Baeza-Yates, Ricardo, and Berthier Ribeiro-Neto. *Modern information retrieval*. Addison-Wesley Longman, 1999.

[2] Gelbukh, Alexander. Lazy Query Enrichment: A Simple Method of Indexing Large Specialized Document Bases. Proc. *DEXA-2000, 11th International Conference and Workshop on Database and Expert Systems Applications*, Greenwich, England, September 4-8, 2000. Lecture Notes in Computer Science N 1873, ISSN 0302-9743, ISBN 3-540-67978-2, Springer-Verlag, pp. 526–535.

[3] Gelbukh, Alexander, and Grigori Sidorov. Intelligent system for document retrieval of the Mexican Senate. Proc. *CIC-2000, Congreso Internacional de Computación*, November 15–17, 2000, CIC, IPN, Mexico D.F., ISBN 970-18-5540-X, pp. 315-321.

[4] Gelbukh, A., G. Sidorov, and A. Guzman-Arenas. Use of a weighted topic hierarchy for text retrieval and classification. In Václav Matoušek et al. (Eds.). *Text, Speech and Dialogue. Proc. 2nd International Workshop TSD-99*, Plzen, Czech Republic, September 13–17, 1999. Lecture Notes in Artificial Intelligence, No. 1692, Springer (http://www-kiv.zcu.cz/ events/ tsd99/ abstract.html), pp. 130–135.

[5] Sidorov, G. O. Lemmatization in automatized system for compilation of personal style dictionaries of literature writers. In: "*Word of Dostoyevsky*", Moscow, Russia, Russian Academy of Sciences, 1996, pp. 266–300.