Automatic acquisition of lexico-semantic knowledge from corpora

Nadezhda A. Stepanova
Novgorod State University after Yaroslav the Wise
Veliky Novgorod, Russia
StepanovaNadya@gmail.com

Abstract. This paper presents a new concept-based approach to extract lexico-semantic knowledge. Genitive constructions of Russian language are derived from parsed corpora. Formal Concept Analysis is employed to build lexicon structure on the basis of genitive constructions. Next, we propose similarity measure and special algorithm to derive lexical classes from concept lattice. This class hierarchy forms lexical database which can be used in various natural language applications, the example of Question Answering systems is given. In the end we compare concept-oriented lexicon with other lexical database and give the implementation details.

1. Introduction

Most of the state-of-the-art Natural Language Processing (NLP) systems employ lexical databases at their work. This article focuses on the needs of Question Answering (QA) systems. QA-systems process questions of a variable degree of complexity and find short and precise answers. Modern QA-systems focus on open-domain questions and use real world documents, especially the World Wide Web, to find answers. In spite of decades of researches and considerable progress QA-systems still have room for improvement of response time and accuracy rates.

The following QA tasks usually need lexical resources: definition of the question type, detecting the hypernyms of key words for WH-questions, question expansion, and the redundancy removal in answers.

Nowadays WordNet-like lexical resources [5] are most popular in NLP but still the coverage problem remains for languages other than English. For example the current version of RussNet (v1.3.35) [1] includes only 15000 word-meaning pairs organized in 5500 synsets. In addition QA-systems need proper names which are typically not a part of WordNet-like resources. According to [8] there were only 9% of proper names in one hundred of the accidental synsets in WordNet.

In that way lexico-semantic knowledge should be derived automatically from corpora to avoid an acquisition bottleneck and to include proper names and subject-oriented terms. Lexicon items should be organized in a hierarchical structure and form classes of different levels of granularity. To achieve these tasks we use methods of Formal Concept Analysis (FCA) [7]. In [13] Priss proved the application of FCA to computational linguistics problems. In [4] FCA was used in a way similar to our purpose – concept hierarchy acquisition for ontology construction but its methods are different from ours. We suggest using Genitive Constructions of the Russian language as a source of the lexicon and show how formal concepts form lexical classes.

The remainder of this paper is organized as follows. In Section 2 we compare different methods of corpora processing and give the formalization of the semantic of the Genitive Construction. In Section 3 we develop the concept-oriented lexicon model on the basis of FCA methods. In Section 4 the concept's clustering method of forming lexical classes is described. In Section 5 we give an example of the usage of a concept-oriented lexicon in QA-systems. The results evaluation is given in Section 6. Section 7 contains our conclusions and suggestions for future research.

2. The source of the lexicon

2.1 Text processing method

First of all the text corpora processing method is needed to derive new lexical knowledge. There are two widespread text processing methods regarding the target word's context definition:

- word context viewed as unstructured text on the left and right of the target word (n-word window);
- context is a set of word which are connected with target word by syntactical relations.

First approach has low precision. The solid numbers of manually defined syntactical patterns are needed for the second approach. In [15] Yarowsky describes word collocation as a precise and simple method of the word sense extraction from a text. In [16] the unity of the sense of the word in collocation is proved (one sense per collocation approach). We suggest using the Genitive Construction (GC) of the Russian language as a basis of the text processing in relation to the target word. GC helps to avoid lacks of two standard approaches and manual definition of the Russian collocations is not needed.

Lexical knowledge can be derived from GC and then it can be inserted into lexicon. We argue that in the similar GCs their parts have a common meaning and this meaning can be extracted using FCA. In the next two subsections we'll give the formalization of GC's semantics to prove the common meaning availability. We suggest using the Intensional Logic (IL) [10] to formalize GC's semantics.

2.2 Intensional Logic

Now we give short introduction into the IL. IL has a rich system of types. There are two basic types: *e* – entities and *t* – truth values. All other types are function types. Lambda-abstraction operator (λ) is a main instrument for the expression's building. In the IL the semantic expressions are close to syntactic structures of the natural language. To apply IL to the description of the lexical semantic we are using Partee approach [11] based on meaning postulates.

Meaning postulate is the notion that lexeme can be defined in terms of relations with other lexemes. For formalization purpose let's consider meaning postulate as axiom (formula) which associates word with other words of the language. Words are just labels in the IL. For the each set Σ of closed formulas there is corresponding class Σ^* of all models in which all formulas from Σ are true. The class Σ^* is called an axiomatizable class of models, and the set Σ is called the set of its axioms. But in Σ^*, not only the axioms of Σ may be true. The set Σ^{**} of all closed formulas which are true in Σ^* is called a theory, and the formulas of Σ^{**} are called the theorems of the theory Σ^{**}. The axioms are a subset of the theorems. For each word w_i let *form(w_i)* be a form and *mng(w_i)* - a meaning.

Sorts [2, 12] are elements of the 'naive ontology' [3] of the language. It is a way to semantically classify nominal predicates. In [12] Partee showed how sorts can be used to form restrictions to GC accuracy. Therefore GC restricts and specifies lexical meaning of GC's components. The word meaning defined by the theory includes meaning postulates from the theories of all sorts the word belongs to and additional postulates which are specific only to this word.

2.3 Semantic of Genitive Construction

We adapted the Partee approach from [12] for the GC's formalization. The GC (for example *John's brother*) consists of three components: Head Noun (HN) (for example, *brother*), Genitive Phrase (GP) (for example, *John's*) and genitive relation. Outside the GC the HN can be defined by expression $\lambda x[S(x)]$ (for non-relational nouns) or $\lambda xy[S(x)(y)]$ (for relational nouns). A noun phrase has a type of <<*e*, *t*>, *t*> and is defined by $\lambda P[P(c)]$ function, where *c* is individual constant of type <*e*>. These predicate set described by function can be interpreted as a set of properties of the individual constant of the type <*e*>. The formula $w \to \lambda P_i[\bigvee_i P_i(x)]$ defines the set of meaning postulates of the non-relational HN (word *w*), where *w* has a type of <*e*>, P_i – predicate of a type of <*e*, *t*>. The word has always the same type as the sort this word belongs to. The theory of a sort (T_s) consists of properties related by the logical operators. If word the w ($mng(w) = \lambda P[P(x)]$) belongs to the sort s then the following expression is true:

$$\exists T_s(\lambda P[\exists z(P(z) \wedge (\forall x(P(x) \rightarrow T_s(z)(x))))]). \qquad (1)$$

Let $fs_s(w)$ be a set of properties of the word w which belongs to the sort s. According to [12] the GP is always defined by the following expression:

$$Gg = \lambda y \lambda R[\lambda x[R(y)(x)]], \qquad (2)$$

where y – the meaning of the GP, x – argument variable, R – predicative variable.

If the HN is the non-relational noun of type $<e, t>$ then inside the GC's special modifier makes a type-shift from the type $<e, t>$ to the relational type $<e, <e,t>>$. In addition the modifier connects the HN with GP to form the accurate GC. Sorts $s1$ and $s2$ satisfy the selective restrictions of the same modifier *Sft* if $T_{s1} \leftrightarrow T_{s2}$ or there is a sort s which is also acceptable for the modifier *Sft* and $T_{s1} \rightarrow T_s \wedge T_{s2} \rightarrow T_s$ (where T_{s1}, T_{s2}, T_s, are the theories of the sorts $s1$, $s2$, s). According to compositional approach we use, the word meaning cannot be decomposed into the elementary primitives. So that we can not demand that sorts $s1$ and $s2$ have the common elementary property. The common property P has to follow from the sort's theories T_{s1} and T_{s2}:

$$\lambda P[\exists T(\exists z(P(z) \wedge \forall x(T(x) \rightarrow P(x)) \wedge \forall x1(T_{s1}(x1) \rightarrow P(x1)) \wedge \forall x2(T_{s2}(x2) \rightarrow P(x2))))]. \qquad (3)$$

The genitive relation connects HN and GP. The genitive relation is also defined by the meaning postulates. The sort of the genitive relation is always equal to the sort of the GC. The genitive relation has a type of $<e, <e,t>>$.

The modifier consists of the HN's theory and the theory of the genitive relation. The modifier has a type of $<e,<e,t>>$:

$$Sft = \lambda a \lambda b[R_{gen}(a)(b) \wedge (fs_s(w))], \qquad (4)$$

where w – the word which belongs to the sort s ($w \in Sort_s$), R_{gen} – the theory of the genitive relation.

Applying the modifier function to the definition of the GP (2) we'll get the GC definition:

$$Gc = \lambda y \lambda R[\lambda x[R(y)(x)]](mng(w_{gg}))(\lambda a \lambda b[R_{gen}(a)(b) \wedge fs_s(w_s)]), \qquad (5)$$

where w_{gg} – GP, w_s – HN. Applying lambda conversion rules we'll get the final GC definition:

$$Gc = \lambda x[\lambda a \lambda b[R_{gen}(a)(b) \wedge (fs_s(w_s))](mng(w_{gg}))(x)] \qquad (6)$$

According to (6) the GC always includes at least one axiom from the theory of the HN and meaning postulates from the genitive relation. The GP is included into the GC's expression as the individual constant.

The meaning of the GC's components can be derived from the GC if for each GC found in corpora we have an expression according to (6). To get this expression for each GC's sort the modifier expression according to (4) is needed. The modifier expression needs meaning postulates which can be constructed manually by lexicographers. Our purpose is different; we want to derive lexical knowledge automatically from the corpora and the manual expressions cannot be used. From the corpora's parsing results we get only the individual genitive constructions and know where HN and GP are. Thus the meaning (property $\lambda P[P(x)]$) of HN and GP cannot be directly derived from corpora automatically and the additional steps are needed.

Let $Gc1$ and $Gc2$ be GCs which belong to the same sort ($Gc1 \in Sort_k$ and $Gc2 \in Sort_k$). So the corresponding genitive relations ($R1_{gen}$ and $R2_{gen}$) are completely defined by the GC's sorts. Therefore $R1_{gen}=R2_{gen}$ and the same selective restrictions are applied to the GC's components ($Gc1_s$/ $Gc1_{gg}$ and $Gc2_s$/$Gc2_{gg}$). In this case according to (3) the theories of HN or GP include the common property P. Let *w1* and *w2* be the definitions for the appropriate HNs of *Gc1* and *Gc2* which belong to the sorts *s1* and *s2* ($w1 \in Sort_{s1}$ and $w2 \in Sort_{s2}$). The property set *P1* is the *w1* word's theory ($mng(w_1) = \lambda P1[P1(x)]$) and *P2* is the *w2* word's theory ($mng(w_2) = \lambda P2[P2(x)]$). The theories *P1* and *P2* are not completely the same in the general case but they must include the common property *P* according to (3). If we consider *w1's* and *w2's* theories only as a common property *P* then from (1) we get expression for two head nouns used in the genitive constructions of the same sort:

$$\begin{aligned}&\lambda P1[\exists z1(P1(z1) \wedge (\forall x1(P1(x1) \rightarrow P(z1)(x1))))] \wedge \\ &\lambda P2[\exists z2(P2(z2) \wedge (\forall x2(P2(x2) \rightarrow P(z2)(x2))))]\end{aligned} \qquad \textbf{(7)}$$

Inside the genitive constructions of the same sort derived from the corpora according to (7) the HNs have the common property. The common property *P* can be acquired by comparing these HNs' meanings. The same is true for the GPs.

We suggest the following approximation to recognize if GCs obtained from the corpora belong to the same sort. With the certain probability two GCs belong to the same sort if their HNs or GPs are the same. To acquire common property *P* we can not use meaning postulates directly so we are using forms of the HNs and GPs.

3. Concept-oriented lexicon model

3.1 Formal Concept Analysis

In this subsection basic FCA definitions are given according to [7] to clarify further reasoning. Let *G* and *M* be sets called object's and attribute's sets respectively and $I \subseteq G \times M$ is a binary relation. If $g \in G$ and $m \in M$ then *gIm* is interpreted as "the object *g* has the attribute *m*". A triple *(G, M, I)* is called a formal context. For $A \subseteq G$ and $B \subseteq M$ the prime-operator is defined as

$$A' = \{m \in M \mid \forall g \in A : gIm\},\ B' = \{g \in G \mid \forall m \in B : gIm\}. \qquad \textbf{(8)}$$

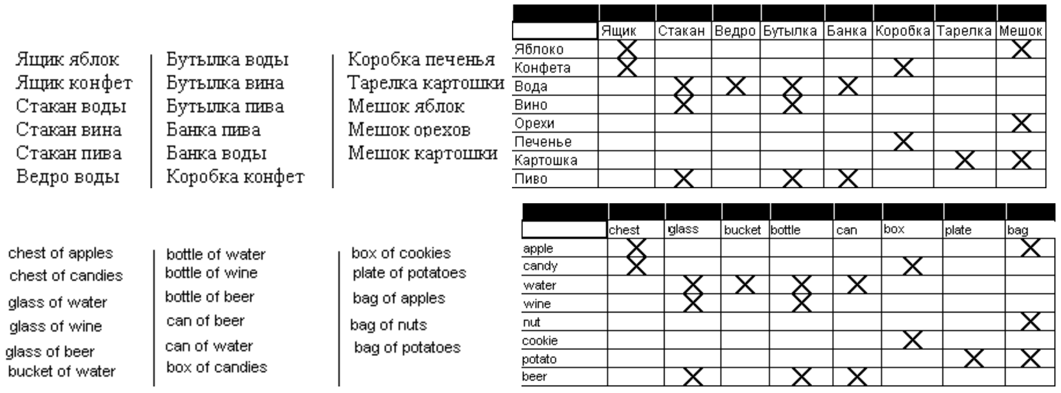
A pair *(A,B)* is formal concept of context *(G, M, I)* if and only if $A \subseteq G, B \subseteq M$ и $A' = B, B' = A$. *A* is called the extent and *B* the intent of the concept *(A, B)*. The concepts *(A₁, B₁)* and *(A₂, B₂)* of a given context are ordered by the subconcept-superconcept relation if $A_1 \subseteq A_2$ ($B_2 \subseteq B_1$) and we write $(A_1, B_1) \leq (A_2, B_2)$. The ordered set of all formal concepts of *(G, M, I)* is denoted by $\mathfrak{B}(G, M, I)$ and is called the concept lattice of *(G, M, I)*. A set of formal concepts is called chain if any two of its elements are comparable ($C_1 \leq C_2$ or $C_1 > C_2$).

3.2 Lexicon structure

Let V_s be the set of HNs ($v_s \in V_s$) and V_{gg} is the set of GPs ($v_{gg} \in V_{gg}$). The pair (v_{gg}, v_s) is the accurate GC if there exists the modifier function *Sft* and sorts of v_s and v_{gg} satisfy the selective restrictions of the *Sft*. The binary relation *I* is the set of pairs (v_{gg},v_s) of the accurate GCs and $I \subseteq V_{gg} x V_s$. If $v_{gg} I v_s$ then the GC was derived from corpora with v_s as HN and v_{gg} as GP and the substitution of v_s and v_{gg} to the (6) gives the accurate GC.

The relation *I* can be presented as the formal context $K=(V_{gg},V_s,I)$. Head nouns are the attributes of objects (GPs) which mean that objects have the common properties. The formal context can be also defined as $K=(V_s, V_{gg}, I)$ then the common properties of the HNs are derived by common attributes (GPs). The complete lattice $\mathfrak{B}(V_{gg},V_s,I)$ can be build upon formal context *K* with order relation.

The example of the formal lattice for the set of genitive constructions derived from Moshkov's library (www.lib.ru) [17] is given in the Figure 2 according to the formal context in the Figure 1.



Ящик яблок
Ящик конфет
Стакан воды
Стакан вина
Стакан пива
Ведро воды

Бутылка воды
Бутылка вина
Бутылка пива
Банка пива
Банка воды
Коробка конфет

Коробка печенья
Тарелка картошки
Мешок яблок
Мешок орехов
Мешок картошки

	Ящик	Стакан	Ведро	Бутылка	Банка	Коробка	Тарелка	Мешок
Яблоко	X							X
Конфета	X					X		
Вода		X	X	X	X			
Вино		X		X				
Орехи								X
Печенье						X		
Картошка							X	X
Пиво		X		X	X			

chest of apples
chest of candies
glass of water
glass of wine
glass of beer
bucket of water

bottle of water
bottle of wine
bottle of beer
can of beer
can of water
box of candies

box of cookies
plate of potatoes
bag of apples
bag of nuts
bag of potatoes

	chest	glass	bucket	bottle	can	box	plate	bag
apple	X							X
candy	X					X		
water		X	X	X	X			
wine		X		X				
nut								X
cookie						X		
potato							X	X
beer		X		X	X			

Fig. 1. The genitive constructions and the formal context.

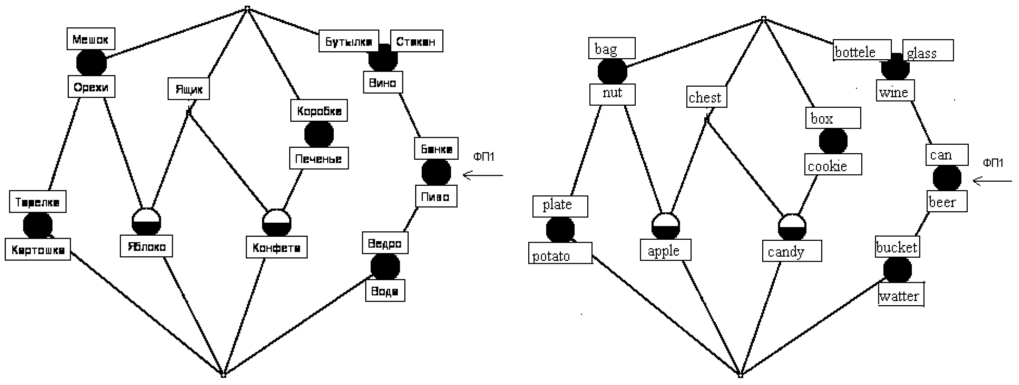


Fig. 2. The formal lattice of genitive constructions.

The formal concept (A, A') has extent and intent. For example the formal concept marked as 'ФП1' at the Figure 2 has extent A=*{Пиво, Вода}* and intent A' =*{Банка, Бутылка, Стакан}*.

All objects at the extent of the formal concept have the same set of common properties defined as formal concept intent A'. The set of attributes can be viewed as the gloss of the objects and presents the lexical knowledge acquired from the text.

Using the concept lexicon (lattice) the accurate genitive constructions can be read. The order relation ($\leq$) in the lexicon defines the word hierarchy. Thus the formal lattice is the lexicon that can be used for question answering.

The longer the highest possible chain in the lattice is the better the lexicon satisfies the NLP needs. So we enlarge the formal context *Kg=(Vgg,* $Vs \cup Vg$ *,I),* where *Vg* – is the set of verbs which use corresponding GC as verb's argument in corpora. For example, *drink glass of water*, $v_g = drink$ and $v_g \in V_g$.

4. Concept's classes acquisition

In the Section 3 we present the lexicon based on the lattice. The lexicon includes sorts (lexeme classes) identical to formal concepts. The number of formal concepts in the lattice is usually bigger than number of input events. So the method to get classes of formal concepts is needed. The class of formal concept (sort) is a higher level of abstraction than a separate formal concept from initial concept lattice *L*. In this section we present the lattice segmentation algorithm to extract classes from initial lexicon (concept lattice).

Any subset of the formal concepts always has the Least Common Superconcept (LCS). Let area of the concept lattice be a set of formal concepts which are related with one LCS. The segmentation algorithm for the initial formal lattice *L* produces a set of formal lattices *{L'}* that the following is true:

- each lattice $L_i \in \{L'\}$ partly corresponds to one of the area of initial lattice *L*;
- each formal concept (except top and bottom concepts) from the initial lattice belongs to one and only one lattice from *{L'}*.

The areas in the initial lattice can overlap. Thus we require only partial correspondence between the lattices from *{L'}* and areas of the initial lattice *L*. The ambiguous formal concept is the formal concept *C* of the initial lattice *L* that belongs to the different areas of the lattice *L* and LCSs of these areas are incomparable.

We require the resulted classes to include maximum number of the formal concepts. Hence the algorithm has to use immediate subconcepts of the top concept of lattice *L* as LCS of the areas to form the lattices $L_i \in \{L'\}$.

The proposed algorithm uses the following criteria to derive lattices $L_i \in \{L'\}$ from the lattice *L*: each formal concept $C \in L_i$ has to be more similar to the formal concepts from the area corresponding to the lattice L_i than to the formal concepts from the other areas. We suggest calculating the similarity measure between the formal concepts as:

$$spc(C_i, C_j) = -\log(1 - \frac{D_c}{path_c}) \times \frac{|B|}{|B_i \setminus B| + |B_j \setminus B| + |B|}, \qquad \textbf{(9)}$$

where formal concept *C=(A,B)* is the LCS of the formal concepts C_i and C_j; D_c is the number of formal concepts in the chain in which top concept and formal concept *C* are maximal and minimal formal concepts; $path_C$ is the minimal number of formal concepts in the chain which includes top and bottom concepts and also formal concept *C*. The similarity measure take into account the volume of common information for two concepts ($|B|$), the volume of the information specific for each individual concept C_i and C_j ($|B_i|$ and $|B_j|$), the specificity of the common information (D_c), and the different length of the lattice hierarchies ($path_C$). The similarity measure is only calculated for ambiguous concept and it's immediate superconcepts.

The segmentation algorithm includes each immediate subconcept C_i of the top concept of lattice *L* into the corresponding resulted lattice L_i. Further all subconcepts of the formal concept C_i in the lattice *L* are included into lattice L_i if for the subconcept C_j each its immediate superconcept is also subconcept of the formal concept C_i and C_j does not have other superconcepts which are incomparable with C_i or coincides with C_i. Otherwise the formal concept C_j is the ambiguous formal concept. It has to be placed into the same classes with its immediate superconcept with which it has the maximal similarity measure value according to (9). If the maximal similarity measure value is the same for several superconcepts then C_j is included into the class with the biggest number of items.

In the Figure 3 the example of concept classes' acquisition is given. Figure on the left is the initial lattice *L*. In the right figure there are three classes L_1, L_2, L_3. Concepts from the class L_1 are more similar to each other than to the concepts from L_2 and L_3 classes. The same is true for concepts of L_2 and L_3 classes. Why does, for example, C_3 concept belong to L_1 class, instead of L_2 class? C_3 concept has two super-concepts – C_2 and C_5. C_3 concept is more similar to C_2 concept than to C_5 concept because $spc(C_3, C_2) = -\log(1 - \frac{2}{5}) \times \frac{3}{1+0+3} = 0{,}5527$ and $spc(C_3, C_5) = -\log(1 - \frac{2}{4}) \times \frac{1}{3+0+1} = 0{,}2500$ and $spc(C_3, C_2) < spc(C_3, C_5)$.

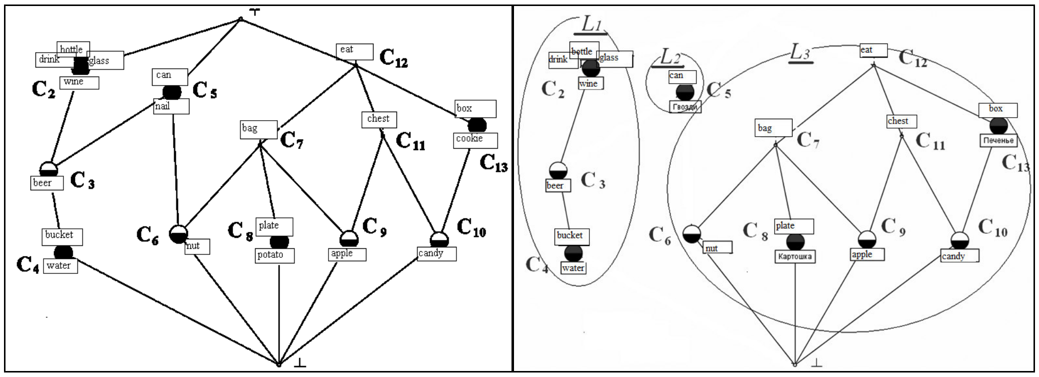


Fig. 3. Concept classes' acquisition.

5. QA tasks and concept-oriented lexicon

The standard QA process is described in the Figure 4. The question type recognition process is the first step in the QA. We use three basic question types: simple Yes/No questions, definition question, WH-questions, and also its subtypes. The lexicon is used to distinguish between different types of definition questions (function, type, etc.) and WH-questions (for example "Which county … ?" asks for country name). The question is parsed and key question words are recognized using syntactic patterns. To detect question subtype we first find the concept which has main key word in its extent. Next we look for its super-concepts. Objects from its extent define question subtype. Key words form query. The lexicon also helps to make query expansion by adding objects from the sibling concepts.

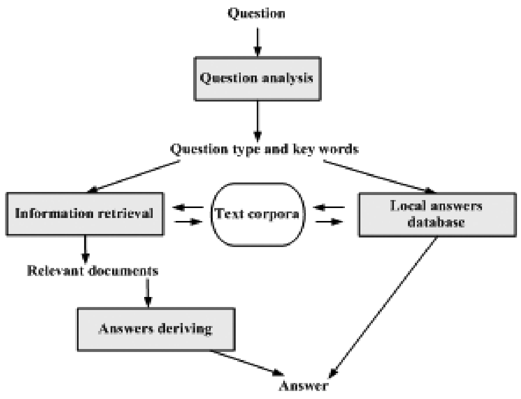


Fig. 4. Question answering process.

The answer can be found in the local answers database or using the information retrieval engine. Information retrieval engine returns text paragraphs with potential answers. Each paragraph is parsed and key answer words are recognized using syntactic patterns.

Next the concept lexicon is used to evaluate found paragraphs. We suggest semantic score metric to compare query and answers from the paragraphs:

$$Sem\_score = \begin{cases} 1, if C_a = C_q \\ Spc_{L_v}(C_a, C_q), if C_a < C_q or C_a > C_q \\ \min(Spc_{L_v}(C_a, C), Spc_{L_v}(C_q, C)), if \exists C \in L_v \mid \\ C_q < C \,\&\, C_a < C \,\&\, \overline{\exists} C_h \mid C_k < C \,\&\, C_q < C_k \,\&\, C_a < C_k \end{cases} , \qquad \textbf{(10)}$$

where q – key question word, a – key answer word from a paragraph, $C_q = (q'', q')$, $C_a = (a'', a')$, L_v – lexicon lattice, $Spc_{L_v}(C_x, C_y)$ – similarity measure between C_x and C_y concepts normalized to maximum similarity measure of the lattice L_v, C – LCS for C_q and C_a concepts.
The final answer is formed from the paragraph with the highest semantic score.

6. Evaluation

6.1 Implementation

To evaluate proposed approach we use text collection from [17] of 85 millions words. From parsed corpora objects (GPs) and attributes (HNs) are derived. The context parameters are given in the Table 1.

Parameter	Value
Number of objects, $(\|G\|)$	5974
Number of attributes, $(\|M\|)$	193580
Context size, $(\|I\|=\|G\| \times \|M\|)$	1156446920
Average number of attributes per object	32,4037
Maximum number of attributes per object	6473

Table 1. Formal context parameters.

Before giving the results we will describe the lattice generation method. We have to choose the best algorithm for lexicon lattice generation in term of performance because number of objects and attributes is big (see Table 1). In addition it should be an incremental algorithm because new GCs appear constantly. The choice of algorithm depends on formal context type. Table 2 gives the distribution analysis of the number of attributes per object.

Number of attributes per object	Number of objects	Number of objects, %
> 1000	18	0,30
500 .. 999	49	0,82
100 .. 499	280	4,69
50 .. 99	345	5,78
40 .. 49	126	2,11
30 .. 39	199	3,33
20.. 29	331	5,54
< 20	4626	77,44

Table 2. Distribution of the number of attributes per object.

Most of the objects (77%) have less than 20 attributes. According to [14] and [6] Ferre algorithm shows the best performance in this situation. But in practice for our context the performance was very pure. The reason of it is that other 23% of objects have huge number of attributes. According to [14] the Norris [9] algorithm shows the best performance on the context with huge number of attributes per object. So we suggest combining Ferre and Norris algorithm: use Ferre for objects with small number of attributes and Norris for objects with huge number of attributes. To choose the switching threshold we analyze the time for adding new object for Ferre (11) and Norris (12) algorithms.

$$O(n \cdot k^2 \cdot l), \qquad (11)$$

$$O(n \cdot m \cdot l), \qquad (12)$$

where n – number of objects, m – number of attributes, k – maximum number of atributes per one object, l – number of concepts in the lattice.

So from (11) and (12) follows that if (13) is true

$$k^2 < m, \tag{13}$$

than Ferre algorithm should be used, otherwise Norris algorithm.

k	n	m	l	O, Ferre	O, Norris	Generation time, sec
120	12	1 619	514	88 819 200	9 985 992	112
100	23	2 085	2 303	529 690 000	110 440 365	106
80	36	2 451	5 186	1 194 854 400	457 591 896	96
70	52	2 783	9 097	2 317 915 600	1 316 481 452	89
60	70	3 108	13 679	3 447 108 000	2 976 003 240	83
50	111	3 621	23 009	6 384 997 500	9 248 030 379	81
40	149	3 964	29 660	7 070 944 000	17 518 263 760	89
30	206	4 306	36 683	6 801 028 200	32 539 141 588	111

Table 3. Switching threshold's test.

The results of experiments are given in the Table 3 for the text of 4 millions words. Objects were sorted by number of attributes in the descending order. One by one each object was added into formal context. Switching threshold (*k*) was changing from 120 to 30. Each row in the table corresponds to the particular switching threshold (*k*) and *n, m, l* parameters correspond to the lattice in the moment of switching between Norris and Ferre. Generation time is given for the complete lattice building process. The threshold equal to 50 corresponds to the best generation time – 81seconds. It proves the condition in (13).

6.2 Comparison

In order to evaluate our approach we compared concept-oriented lexicon with Abramov's Dictionary of Synonyms (ADS). We used this dictionary as a "gold standard" because it is freely available and for Russian language it has largest coverage area (19108 articles). Recall and Precision are calculated according (14) and (15):

$$Recall = \frac{|A|}{|A|+|B|}, \tag{14}$$

$$Precision = \frac{|A|}{|A|+|C|}, \tag{15}$$

where *A* – set of lexemes recognized as synonyms in ADS and concept-oriented lexicon, *B* - set of lexemes recognized as synonyms in ADS but not in concept-oriented lexicon, *C* - set of lexemes recognized as synonyms in concept-oriented lexicon but not in ADS.

In the experiment the concept-oriented lexicon was built upon the corpora of about 17 millions words. The experiment was conducted for the 50 most frequent lexemes of the Russian language. The results were the following: *Recall=24,36%* and *Precision=9,78%*. Low Precision is explained by the fact that concept-oriented lexicon has much bigger coverage than ADS. Also ADS includes very specific synonyms that decrease Recall. So the better "gold standard" is needed.

7. Conclusions and further work

In this paper we have proposed the novel approach to lexical resources to boost the QA-system performance. Our approach is based on the lattice theory and the genitive constructions. It derives new lexical information automatically from corpora and adds lexemes into the lexicon. The lexicon has hierarchical structure and is presented by a lattice. A formal concept is the basic item of the lexicon. The formal concept joins objects from its extent with interpretation (gloss) from its intent. This paper presents similarity measure and segmentation algorithm to derive classes with the several formal concepts from the initial lexicon. These classes correspond to the lexical sorts of different degree of granularity.

The suggested lexicon is applied to the several problems of the question answering. It has been shown how to use concept-oriented lexicon in the question type detection, and the answers finding. In the further researches we would like to apply our approach to the languages other than Russian language and integrate concept-oriented lexicon with the industrial QA-system.

References

1. И. В. Азарова, О. А. Митрофанова, А. А. Синопальникова. Компьютерный тезаурус русского языка типа WORDNET. Труды международной конференции Диалог'2003, Протвино, 2003.
2. Борщев В.Б., Кнорина Л.В. Типы реалий и их языковое восприятие. В сб. "Вопросы кибернетики. Язык логики и логика языка". М.: 1990. С.106-134.
3. Борщев В.Б. Естественный язык - наивная математика для описания наивной картины мира. Московский лингвистический альманах, вып. 1, 1996. С.203-225.
4. Cimiano P., Hotho A., Staab S. Learning Concept Hierarchies from Text Corpora using Formal Concept Anaylsis. Journal of Artificial Intelligence Research. Volume 24, pages 305-339 August 2005.
5. Fellbaum C. WordNet: An Electronic Lexical Database. Cambridge, 1998.
6. Ferré S. The Use of Associative Concepts for Fast Incremental Concept Formation in Sparse Contexts. In B. Ganter and A. de Moor editors, Using Conceptual Structures, Contributions to ICCS 2003, 2003.
7. Ganter B. and Wille R., Formal Concept Analysis – Mathematical Foundations. Berlin: Springer-Verlag, 1999.
8. Mann G.S., Fine-Grained Proper Noun Ontologies for Question Answering, SemaNet'02: Building and Using Semantic Networks, 2002.
9. Norris E. M., An algorithm for computing the maximal rectangles in a binary relation // Revue Roumaine de Mathermatiques Pures et Appliqueres, 23 (2), 1978. – pp. 243-250.
10. Gerasimova I. A.. Formal grammar and intensional logic. – Moscow, 2000.
11. Partee B.H. Formal Semantics, Lectures. RGGU, February 14, 2003.
12. Partee B.H., Borschev V.B. Genitives, types, and sorts. In Possessives and Beyond: Semantics and Syntax, eds. Ji-yung Kim, Yury A. Lander and Barbara H. Partee. Amherst, MA: GLSA Publications, 2004. - PP 29-43.
13. Priss, Uta. Linguistic Applications of Formal Concept Analysis. In: Ganter; Stumme; Wille (eds.), Formal Concept Analysis, Foundations and Applications. Springer Verlag. LNAI 3626, 2005, p. 149-160.
14. Kuznetsov S.O., Obiedkov S.A. Comparing performance of algorithms for generating concept lattices // Journal of Experimental & Theoretical Artificial Intelligence, Volume 14, Issue 2 & 3, 2002. – pp. 189-216.
15. Yarowsky D. Unsupervised word sense disambiguation rivaling supervised methods. Proceedings of the 33rd annual meeting on Association for Computational Linguistics, Association for Computational Linguistics, Morristown, NJ, USA, 1995, P.189 - 196.
16. Yarowsky D. One sense per collocation. In the Proceedings of ARPA Human Language Technology Workshop, Association for Computational Linguistics, Morristown, NJ, USA, 1993. P.266 - 271.
17. www.lib.ru