АНАЛИЗАТОР РУССКОГО ЯЗЫКА SYNTAUTOM ДЛЯ СОРЕВНОВАНИЯ СИНТАКСИЧЕСКИХ ПАРСЕРОВ (ДИАЛОГ-2012)

Антонова А. А. (antonova@yandex-team.ru),
Мисюрев А. В. (misyurev@yandex-team.ru)

Yandex

Ключевые слова: синтаксический анализ, разбор на основе правил, грамматика зависимостей, синтаксис русского языка.

RUSSIAN DEPENDENCY PARSER SYNTAUTOM AT THE DIALOGUE-2012 PARSER EVALUATION TASK

Antonova A. A. (antonova@yandex-team.ru),
Misyurev A. V. (misyurev@yandex-team.ru)

Yandex

In this paper we describe the SyntAutom parser submitted at the Dialogue-2012 parser evaluation task. It is a rule-based system, which performs syntactic and morphological ambiguity resolution in a unified framework. The underlying grammar formalism is Dependency Grammar. The segmentation, shallow parsing and deep parsing are based on a special form of finite-state pushdown automatons, implemented as a sets of recursive functions. The input of the automaton is a lattice of objects — these may be words or shallow trees. Within the functions we have implemented a mechanism of branching and multiple return — a possibility for a function to generate a bunch of states.

We discuss the system architecture, the output parsing trees structure and the common types of incorrect analyses. The distinctive feature of the system is that it tends to directly connect meaningful words, while auxiliary words are demoted to the lower tree levels.

Although the system experiences the common problems of most rule-based systems, it has proven its applicability in a range of real-world applications.

Keywords: natural language parsing, rule-based parser, dependency grammar, Russian syntax

1. Introduction

This paper briefly describes the Russian dependency parser SyntAutom presented at the Dialogue-2011 parser evaluation task. It is a rule-based system, which performs syntactic and morphological ambiguity resolution in a unified framework. The previous versions of the parser are described in [1, 2].

The underlying grammar formalism is Dependency Grammar[3, 5, 6, 8]. The search algorithm is bottom-up and depth-first. The parsing strategy is based on a special form of finite-state pushdown automaton, implemented as a set of recursive functions. The input of the automaton is a lattice of objects — these may be words or shallow trees. An automaton state has access to the current position in the sentence, the global parameters(e. g. the closeness of the sentence boundary, the syntactic class of the previous independent tree), the parameters of the last read object and the parameters of all objects in the stack. Possible transitions to other states are specified by parsing rules, based on whether the current state parameters satisfy the certain conditions.

We have developed a native formalism for writing parsing rules in form of functions. Each function has a predefined scope — objects and parameters that the function can manipulate. All functions also have access to global parameters. Within functions we have implemented a mechanism of branching and multiple return — a possibility for a function to generate a bunch of states. All newly created states are then processed independently.

An automaton path is a sequence of states, which corresponds to a contiguous fragment of the input sentence. Each path reveals dependencies between words. Due to the morphological and syntactic ambiguity many different paths can be associated with the input sentence or its fragment. When all paths are explored, the parsing tree is reconstructed according to the dependencies found along the best path.

The automaton does not allow to skip any words. Obviously one cannot expect to find a full parse tree for all the unrestricted variety of natural sentences. So the common convention is to allow the sentence to break on several parts and look for the best partial parses. The pushdown automaton can split the sentence every time its stack is empty. Then it begins parsing from the next word as if there was a sentence boundary before it. Different paths can put partial sentence boundaries in different positions.

We describe the overall structure of the system in Section 2. In Section 3 we specify the parser output format and explain representation of some specific syntactic structures. We discuss the limitations of our parsing approach in Section 4. Section 5 describes applications of the system. We conclude in Section 6.

2. System architecture

As a rule-based system, SyntAutom relies on hand-built parsing rules and morphological dictionary, as well as some additional data sources. The parsing process is represented in Figure 1.

2.1. Segmentation and morphological analysis

SyntAutom has a sophisticated segmentation procedure, which is responsible for the detection of:

- sentence boundaries (if the input is not sentence-splitted);
- phrases that function as single words;
- complex tokens (e. g. e-mail, url);
- proper names;
- compound cardinal numbers;
- hyphenation.

The segmentation procedure also represents a finite-state automaton, implemented as a set of functions. The automaton receives a string of low-level textual tokens as input. The output of this stage is a lattice, representing different segmentation options, morphological and structural ambiguities, including possible multiword expressions. Due to morphological ambiguity a surface word can be associated with different lattice nodes, having different lemmas or different sets of morphological features. We call such nodes morphological interpretations. Each morphological interpretation is assigned a set of lexical features (e. g. from morphological dictionary or valence list). Unknown words are analyzed heuristically, by finding similar words in morphological dictionary.

The texts provided by the Dialogue-2011 parser evaluation task had been sentence-splitted and tokenized by the organizers. That probably reduced the number of certain segmentation mistakes.

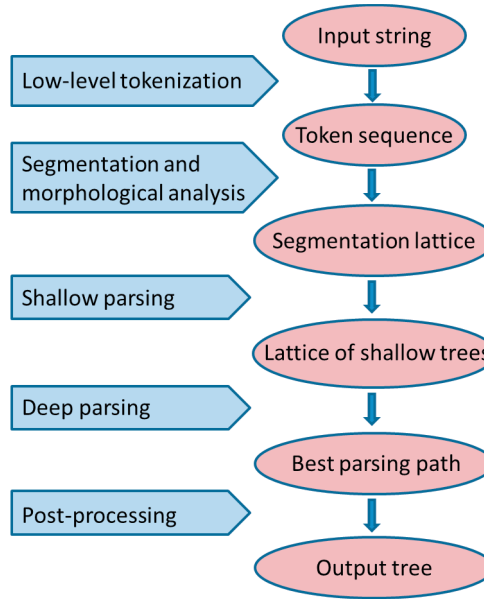

Figure 1. The system architecture

2.2. Verb valences

Typically, morphological dictionaries do not contain information about verb valences. At the same time good-quality valence information is crucial for the performance of a rule-based system. We manually created lists of verb valences for more than 12'000 Russian verbs. The following valences for a verb are indicated:

1) a subject in nominative case;
2) an object in accusative case;
3) an object in dative case;
4) an object in genitive case;
5) an object in instrumental case (optional);
6) a subordinate clause, beginning with "что"("that");
7) a subordinate clause, beginning with question words ("где"(*"where"),* "когда"("*when*"), "как"("*how*"), "почему"("*why*"), "какой"("*which*"), "чей"("*whose*"));
8) possibility to be auxiliary for another verb (see section 3.4)

Some valence combinations are mutually exclusive. We provide each verb with possible combinations of the 8 valence types, instead of indicating each valence independently. For example, possible combinations for the verb "жалеть" ("be sorry") are [1,2], [1,4], [1,6]; "угрожать" (threaten) — [1,3,5], [1,3,6]. The same logic can be applied to adjectives and nouns, some of which can also have special valences.

2.3. Shallow parsing

At the shallow parsing stage the system tries to detect simple phrasal groups that cannot have recursive structure. We do not use pushdown operations on this stage. The shallow parsing strategy can be described as a simple finite-state automaton with right-to-left expansion[1].

Examples of shallow trees that can be constructed on this stage:

1. Noun phrase with left modifiers (except for left participle when it has its own complements).
   "большой дом" ("*big house*");
   "большой красивый дом" ("*big beautiful house*");
   "очень большой дом" ("*very big house*");
   "двадцать два больших дома" ("*twenty two big houses*");
   "с большим домом" ("*with a big house*");
2. Adjectival or verb phrase with left modifiers.
   "очень быстрый" ("*very fast*");
   "быстро бежал" ("*ran fast*", literally "*fast ran*");
   "очень быстро бежал" ("*ran very fast*", literally "*very fast ran*");

The output of the shallow parsing is a lattice of shallow trees, and it is the input to the deep parsing automaton. The shallow parsing finds rather simple phrases, but it helps to save time for the computationally heavy task of deep parsing.

2.4. Deep Parsing

Deep parsing deals with long-distance dependencies and embedded structures - that is why pushdown operations(stack) are necessary here. The deep parser strategy is bottom-up, depth-first, with left-to-right expansion. The parser performs non-deterministic, exhaustive search. It does not attempt to resolve each decision as reached, but rather pursues all alternatives. The parser cannot skip words. Even punctuation and unknown words must have their place in the tree.

An important optimization, that allows to avoid repeated analyses and reduce combinatorics, is caching the function calls together with the parameters that belong to the scope of this function. If another function call happens to have exactly the same parameters, the parser simply retrieves the resulting bunch of states from the cache, without actually running the function. A similar approach, called «chart parsing» is described in [7].

At the end of the deep parsing stage, the path with the best weight is chosen. The evaluation of an automaton path weight is performed based on the following factors:

1) Frequencies of morphological interpretations of words.

---

[1] Since we mostly explore left modifiers, at this stage the automaton reads words from right to left.

2) Frequencies of binary lexicalized dependency relations.
3) Empirical weights that are added when the path crosses some state in the automaton.

As in most rule-based systems, empirical weights are important for the system performance. They allow to penalize or promote some automaton states according to the linguistic intuition. But the roughness and inaccurateness of the empirical weights limits the usefulness of any statistical factors. Thus, we used the simple add-one smoothing when evaluating the frequencies of different morphological interpretations of words and dependency relations.

The frequencies of binary lexicalized dependencies were extracted from large automatically parsed corpus. Though there definitely exist many parsing errors in the corpus, the overall performance slightly improved. For example, in some cases such statistics helps to resolve the problem of prepositional phrase attachment.

Мне нравилось смотреть на улицу через стекло
(I liked to look at the street through the glass)

0	*Top*	*Top*	0	_	/
1	мне	я	3	subj	/prn/sg/fem/msc/neu/dat/fst/
2	нравилось	нравиться	3	auxd	/vrb/sg/neu/fin/fst/sec/trd/pst/ind/act/
3	смотреть	смотреть	0	fin	/vrb/sg/neu/inf/fst/sec/trd/pst/act/
4	на	на	5	prep	/prp/acc/
5	улицу	улица	3	prepnp	/nn/sg/fem/acc/trd/
6	через	через	7	prep	/prp/acc/
7	стекло	стекло	3	prepnp	/nn/sg/neu/acc/trd/

2.5. Post-Processing

A post-processing stage was added to improve parser performance for the evaluation task.

Adverbial participles were made dependent on the head of neighboring clause. Clauses beginning with a conjunction were made dependent on the head of neighboring clause.

We have also tried to redirect certain types of dependency relations, to avoid disagreement with the gold standard parses. That included prepositions, cardinal numbers and modal verbs. We did not redirect dependency relations in cases when the dependent word had its own dependents.

3. Output format and tree structure

In the rest of the paper the parser results are represented as a table. Each line corresponds to one word of the input sentence and consists of:

- word id;
- word form;
- lemma;
- id of the head word;
- tag of the dependency relation;
- morphological attributes.

The words are enumerated according to the order in the input sentence.

An artificial word "*Top*" precedes each sentence. It acts as a head word for words which have no other parent, enforcing a single-tree structure even for partial parses. In some cases the segmentation module inserts additional "*Top*" in the middle of the sentence.

в свои родные края, к своей работе, завести детей,
прожить долгую полноценную жизнь.
(To his homeland, to his work, to have children,
to live a long full life)

0	*Top*	*Top*	0	_	/
1	в	в	4	prep	/prp/acc/
2	свои	свой	4	adj	/prn/pl/msc/nom/trd/
3	родные	родной	4	adj	/adj/pl/msc/nom/trd/
4	края	край	0	prepnp	/nn/pl/msc/acc/trd/
5	,	,	0	misc	/pnt/
6	к	к	8	prep	/prp/dat/
7	своей	свой	8	adj	/prn/sg/fem/loc/trd/
8	работе	работа	0	prepnp	/nn/sg/fem/dat/trd/
9	,	,	0	misc	/pnt/
10	завести	завести	0	inf	/vrb/inf/act/
11	детей	ребенок	10	acc	/nn/pl/msc/anm/acc/trd/
12	,	,	10	conj	/pnt/
13	прожить	прожить	10	homo	/vrb/inf/act/
14	долгую	долгий	16	adj	/adj/sg/fem/acc/trd/
15	полноценную	полноценный	16	adj	/adj/sg/fem/acc/trd/
16	жизнь	жизнь	13	acc	/nn/sg/fem/acc/trd/
17	.	.	0	misc	/pnt/

There are several constraints on the general tree structure. Every word, even punctuation and unknown words, must find their place in the tree. In case of a partial parse tree, no dependencies are allowed across an independent part of the sentence. Punctuation at the end of the sentence is usually dependent on "*Top*", except when it is recognized as a part of an abbreviation.

A dependency relation tag can be considered to be a syntactic role of the word with respect to its head word. If a word is the head of an independent tree, then its tag corresponds to the syntactic class of the tree. The set of syntactic roles and classes is described in Appendix.

Our system follows the common practical convention that subject, object and other complements are dependent on the verb.

моя сестра подарила мне эти жемчужины

(My sister gave me these pearls)

0	*Top*	*Top*	0	_	/
1	моя	мой	2	adj	/prn/sg/fem/nom/trd/
2	сестра	сестра	3	subj	/nn/sg/fem/anm/nom/trd/
3	подарила	подарить	0	fin	/vrb/sg/fem/fin/trd/pst/ind/act/
4	мне	я	3	dat	/prn/sg/fem/msc/neu/dat/fst/
5	эти	этот	6	adj	/prn/pl/fem/nom/trd/
6	жемчужины	жемчужина	3	acc	/nn/pl/fem/acc/trd/

Still there are some constructions for which our analysis may differ from analyses of other systems. Although we do not pretend to explore semantic relations in the sentence, there is a number of situations when we prefer to mark a semantic dependency instead of a syntactic one.

3.1. Prepositions

We consider preposition to be a dependent on the noun phrase rather than its head. Prepositions often convey little or no meaning, whereas in many tasks it is helpful to have a direct dependency between meaningful words.

он был женат на креолке из евангелического прихода

(he was married to a creole from the evangelical parish)

0	*Top*	*Top*	0	_	
1	он	он	3	subj	prn/sg/msc/nom/trd
2	был	быть	3	auxs	vrb/sg/msc/fin/fst/sec/trd/pst/ind/act
3	женат	женатый	0	fin	adj/sg/msc/trd/pst/sht
4	на	на	5	prep	prp/loc
5	креолке	креолка	3	prepnp	nn/sg/fem/anm/loc/trd
6	из	из	8	prep	prp/gen
7	евангелического	евангелический	8	adj	adj/sg/msc/gen/trd
8	прихода	приход	5	prepnp	nn/sg/msc/gen/trd

3.2. Coordination

First coordination member is always the representative of the coordination group. Each successive coordination member is attached to the previous member with the dependency relation “homo”. The coordinating conjunction or comma is also attached to the previous member with the dependency relation “conj”.

чтобы чинить, оказывать помощь и спасать

(to repair, to render assistance and rescue)

0	*Top*	*Top*	0	_	/
1	чтобы	чтобы	0	conj	/cnj/
2	чинить	чинить	0	inf	/vrb/inf/act/
3	,	,	2	conj	/pnt/
4	оказывать	оказывать	2	homo	/vrb/inf/act/
5	помощь	помощь	4	acc	/nn/sg/fem/acc/trd/
6	и	и	4	conj	/cnj/
7	спасать	спасать	4	homo	/vrb/inf/act/

3.3. Auxiliary and modal verbs

We consider as auxiliary almost all instances of verb usage when two verbs are syntactically related and subject of both verbs is identifiable. The parser does not distinguish between auxiliary and modal verbs, but rather makes a distinction based on whether the semantic subject of the verbs is the same or different.

я хотела бы оказаться там

(I would like to be there)

0	*Top*	*Top*	0	_	/
1	я	я	4	subj	/prn/sg/fem/nom/fst/
2	хотела	хотеть	4	auxs	/vrb/sg/fem/fin/fst/sec/trd/pst/ind/act/
3	бы	бы	2	by	/pt/
4	оказаться	оказаться	0	fin	/vrb/sg/fem/inf/fst/pst/act/
5	там	там	4	adv	/adv/

Here the semantic subject “я”(“*I*”) is the same for both the main and the modal verb, and the modal verb “хотеть”(“*want*”) has the syntactic role “auxs”. In this case the subject is made dependent on the main verb, rather than on the modal verb.

она научила меня играть на пианино

(She taught me to play the piano)

0	*Top*	*Top*	0	_	/
1	она	она	2	subj	/prn/sg/fem/nom/trd/
2	научила	научить	4	auxd	/vrb/sg/fem/fin/trd/pst/ind/act/
3	меня	я	4	subj	/prn/sg/fem/msc/neu/acc/fst/
4	играть	играть	0	fin	/vrb/sg/fem/inf/fst/sec/trd/pst/act/
5	на	на	6	prep	/prp/acc/loc/
6	пианино	пианино	4	prepnp	/nn/sg/neu/acc/trd/

Here the semantic subject of the main verb “играть”(“*to play*”) is different from the subject of the modal verb “научить”(“*teach*”), and the modal verb has the syntactic role “*auxd*”. In this case both semantic subjects are dependent on the corresponding

verbs with the role "*subj*". It is worth to note that other arguments are usually dependent on the main verb, rather than on the modal verb.

3.4. Subordinate clauses

The head predicate of a subordinate clause is made dependent of the head predicate of the main clause with the dependency relation "sent".

он рассказывал что там играл духовой оркестр
(He told that there were a brass band)

0	*Top*	*Top*	0	_	/
1	он	он	2	subj	/prn/sg/msc/nom/trd/
2	рассказывал	рассказывать	0	fin	/vrb/sg/msc/fin/trd/pst/ind/act/
3	что	что	5	conj	/cnj/
4	там	там	5	adv	/adv/
5	играл	играть	2	sent	/vrb/sg/msc/fin/trd/pst/ind/act/
6	духовой	духовой	7	adj	/adj/sg/msc/nom/trd/
7	оркестр	оркестр	5	subj	/nn/sg/msc/nom/trd/

3.5. Cardinal numbers

We consider a cardinal number to be a dependent on the noun phrase.

У нее было двадцать две кошки.
(She had twenty two cats)

0	*Top*	*Top*	0	_	/
1	у	у	2	prep	/prp/gen/
2	нее	она	3	prepnp	/prn/sg/fem/gen/trd/
3	было	быть	0	fin	/vrb/sg/neu/fin/fst/sec/trd/pst/ind/act/
4	двадцать	двадцать	5	card	/num/pl/fem/msc/neu/nom/trd/
5	две	два	6	card	/num/pl/fem/nom/trd/
6	кошки	кошка	3	subj	/nn/pl/fem/anm/nom/trd/

4. Robustness vs. coverage

In the real-world applications the parser robustness is more important than its "grammatical coverage" – the system's theoretical ability to build certain types of parse trees. The more permissive is the grammar, the bigger is the chance that the system gets confused with all the possible parses. For example, in a machine-translation application it is often better to translate a fragment word-by-word than to reorder it based on a wrong parse.

We intentionally do not attempt to interpret sentences without predicates: “у меня температура” (“*I <have> a fever*”), “это собака” (“*this <is> a dog*”), “она адвокат” (“*she <is> a lawyer*”) “сегодня ты один” (“*today you <are> alone*”), “как ее дыхание?” (“*How <is> her breath?*”) “мы больше не друзья” (“*we <are> not friends anymore*”). Sentences without predicates are parsed disconnectedly.

У меня температура
(I have a fever)

0	*Top*	*Top*	0	_	/
1	у	у	2	prep	/prp/gen/
2	меня	я	0	prepnp	/prn/sg/fem/msc/neu/gen/fst/
3	температура	температура	0	np	/nn/sg/fem/nom/trd/

Here we obviously sacrifice some of the potential recall, but eliminate spurious analyses and prevent increase in combinatorics: e. g. the sentence “У меня температура зашкаливает” (“*My temperature is very high*”).

The parser does not find connections for some phrases that are not structural parts of the predications, like interjections or direct address construction.

ага, я так и знал
(yeah, I knew it)

0	*Top*	*Top*	0	_	/
1	ага	ага	0	np	/nn/sg/msc/nom/trd/
2	,	,	0	misc	/pnt/
3	я	я	6	subj	/prn/sg/msc/nom/fst/
4	так	так	6	adv	/adv/
5	и	и	4	misc	/cnj/
6	знал	знать	0	fin	/vrb/sg/msc/fin/fst/pst/ind/act/

ты немного староват для войны, бенджамин
(You are a bit old for the war, benjamin)

0	*Top*	*Top*	0	_	/
1	ты	ты	3	subj	/prn/sg/msc/nom/sec/
2	немного	немного	3	adv	/adv/
3	староват	староватый	0	krat	/adj/sg/msc/sec/prs/sht/
4	для	для	5	prep	/prp/gen/
5	войны	война	3	prepnp	/nn/sg/fem/gen/trd/
6	,	,	0	misc	/pnt/
7	бенджамин	бенджамин	0	np	/nn/sg/msc/anm/nom/trd/cap/

Nevertheless, the big number of possible analyses is still an important problem. There exist constructions the addition of which can increase combinatorics dramatically. For example, constructions with emphatic “и”(*and*), the addition of which can complicate the recognition of coordination constructions. The

constructions with emphatic “и”(*and*) are parsed disconnectedly in the current version of the parser.

Пройдет и это
(This will pass too)

0	*Top*	*Top*	0	_	/
1	пройдет	пройти	0	fin	/vrb/sg/fem/msc/neu/fin/trd/prs/ind/act/
2	и	и	0	conj	/cnj/
3	это	этот	0	np	/prn/sg/neu/acc/trd/

As a rule-based system our parser relies heavily on the word valence information. We do not use prepositional valencies - any prepositional group can depend on any previous noun phrase or verb phrase. The system allows any verb to have a complement in instrumental case, though verbs that have a predefined instrumental valence are encouraged. The ability to govern other non-prepositional complements(genitive, dative, accusative), subject and subordinate clauses is strictly controlled by the predefined valence information. For that reason the parser is unable to recognize a dependency if it is not permitted in its valence lists. This kind of mistakes can be divided into two subtypes:

1. Mistakes that can be corrected easily if we add a missing valence to the valence lists.

вещают мортимеру про конец
(They prophecy to Mortimer about the end)

0	*Top*	*Top*	0	_	/
1	вещают	вещать	0	fin	/vrb/pl/fem/msc/neu/fin/trd/prs/ind/act/
2	мортимеру	мортимер	0	np	/nn/sg/msc/anm/dat/trd/
3	про	про	4	prep	/prp/acc/
4	конец	конец	2	prepnp	/nn/sg/msc/acc/trd/

Here we can add a missing dative valence to the verb “вещать” (“*to prophecy*”) and improve the parser performance.

2. Mistakes in which a missing valence is lexically-dependent, for example, Russian construction with dative possessor.

он поцеловал невесте руку.
(He kissed the bride’s hand)

0	*Top*	*Top*	0	_	/
1	он	он	2	subj	/prn/sg/msc/nom/trd/
2	поцеловал	поцеловать	0	fin	/vrb/sg/msc/fin/trd/pst/ind/act/
3	невесте	невеста	0	np	/nn/sg/fem/anm/loc/trd/
4	руку	рука	0	np	/nn/sg/fem/acc/trd/

Here we cannot add a dative valence to the verb «поцеловать», because it depends on the semantics of the other complement (part of body). The parser usually cannot parse this type of constructions correctly.

We allow contextual substantivation of adjectives - i.e. each adjective can act as noun phrase in many contexts.

коричневый идет вашим глазам

(The brown suits your eyes)

0	*Top*	*Top*	0	_	/
1	коричневый	коричневый	2	subj	/adj/sg/msc/nom/trd/
2	идет	идти	0	fin	/vrb/sg/msc/fin/trd/prs/ind/act/
3	вашим	ваш	4	adj	/prn/pl/msc/dat/trd/
4	глазам	глаз	2	dat	/nn/pl/msc/dat/trd/

Accusative-genitive case transformation in negative sentences. “Я вижу собаку” *(I see the dog)*, “Я не вижу собаки” *(I don’t see the dog*. We assign grammatical role “acc” in both cases.

Я не вижу собаки

(I don’t see the dog)

0	*Top*	*Top*	0	_	/
1	я	я	3	subj	/prn/sg/fem/msc/neu/nom/fst/
2	не	не	3	pt	/pt/
3	вижу	видеть	0	fin	/vrb/sg/fem/msc/neu/fin/fst/prs/ind/act/
4	собаки	собака	3	acc	/nn/sg/fem/anm/gen/trd/

5. Practical applications

Initial application of our parser was within a rule-based machine translation system. The parser has already been used as a tool for preparation and analysis of linguistic corpora. For example, the automatically created treebank of 70 million sentences had been used for the lexicographic purposes.

1) Automatic creation of bilingual dictionary. The parsing information was used to filter out ungrammatical phrases and find words and phrases in canonical form.
2) Automatic extraction of synonyms. The parsing information was used to compute distributional similarity between words.

Other applications include distributional semantic clustering, context classification, text comparison (in particular [9, 10]). The parser was also used in student works of Moscow State University (in particular [11]).

Conclusion

We have described the dependency parser submitted at the Dialogue-2011 parser evaluation task. The details of the output tree structures and the common types of incorrect analyses were discussed. The distinctive feature of the system is that it tends to directly connect meaningful words, while auxiliary words are demoted to the lower tree levels.

The advantages of our formalism are the following:

1) The syntactic and morphological ambiguity is resolved simultaneously within a unified framework.
2) The explicit description of automaton transitions provides a flexible way to control parsing process.
3) The automaton state usually provides more information than a context-free rule can provide.
4) It is easy to add "local" functions that are called only in specific conditions.

The system experiences the common problems of most rule-based systems. First, it is difficult to reconcile empirical weights with the weights provided by statistical models. Second, it is hard to increase grammatical coverage beyond certain extent, because of the increase in combinatorics and the drop in precision.

In spite of existing limitations, the system has proven its applicability in a range of real-world applications.

References

1. *Antonova A. A., Misyurev A. V.* The development of a syntactic parser for Russian and English [Realizatsija sintaksicheskogo razbora dlja russkogo i anglijskogo jazykov], Pervaja Mezhdunarodnaja konferentsija "Sistemnyj analiz i informatsionnye tehnologii" [The first international conference "System analysis and information technologies"], Pereslavl'-Zalesskij, 2005, pp. 245–249.
2. *Antonova A. A., Misyurev A. V.* (2008), About applications of the syntactic parser Cognitive Dwarf 2.0. [Ob ispol'zovanii sintaksicheskogo analizatora Cognitive Dwarf 2.0.], Sbornik trudov ISA RAN [Proceedings of ISA RAS], no. 38, pp. 91–109.
3. *Jurij Apresian, Igor Boguslavsky, Leonid Iomdin, Alexander Lazursky, Vladimir Sannikov, Victor Sizov, Leonid Tsinman* (2003) ETAP-3 Linguistic Processor: a Full-Fledged NLP Implementation of the MTT, MTT 2003. First International Conference on Meaning-Text Theory Paris, Ecole Normale Superieure, pp. 279–288
4. *Gershenzon L. M., Nozhov I. M., Pankratov D. V., Sokirko A. V.* Syntactic analysis in RML system. [Sintaksicheskij analiz v sisteme RML], available at http://www.aot.ru/docs/synan.html

5. *David Hays G.* (1964). Dependency theory: A formalism and some observations. Language, 40: P. 511–525.
6. *Hudson Richard* (1991) English Word Grammar. Basil Blackwell, Cambridge, MA.
7. *Kay, Martin.* (1986). Algorithm schemata and data structures in syntactic processing. Readings in Natural Language Processing, pp. 35–70. Los Altos, CA: Morgan Kaufmann.
8. *Mel'ˇcuk Igor A.* (1987) Dependency Syntax: Theory and Practice. State University of New York Press, Albany.
9. *Mihajlov D. V., Emeljanov G. M.* (2010). Theoretical basis of the development of open-source question-answering systems. Semantic equivalence of texts and recognition models [Teoreticheskie osnovy postroenija otkrytyh voprosno-otvetnyh sistem. Semanticheskaja èkvivalentnost' tekstov i modeli ih raspoznovanija]. NovSU, Velikij Novgorod.
10. *Mihajlov D. V., Emeljanov G. M.* (2011). The analysis of formal concepts and the compression of textual information in the task of automated knowledge control [Analiz formal'nyh ponjatij i szhatie tekstovoj informatsii v zadache avtomatizirovannogo kontrolja znanij]. Matematicheskie metody raspoznavanija obrazov "MMPR-15" [Mathematical methods of pattern recognition], available at http://www.machinelearning.ru/wiki/images/4/4c/Mmpr15_mdv_report.pdf
11. *Tretjakov D., Il'jushina E. A., Puchkova E. M.* (2009) Automated analysis of syntactic relations in Russian texts using Markov Models [Avtomatizirovannyj analiz sintaksicheskih otnoshenij v russkom tekste s ispol'zovaniem tsepej Markova]. Sovremennye problem gazovoj i volnovoj dinamiki [Modern problems of Gas and Wave Dynamics], MSU, Moscow, p.107.

Appendix

Russian syntactic roles

subj	subject
acc	direct object
dat	dative-case object
ins	instrumental-case object
gen	genitive-case object
prepnp	prepositional phrase
adj	adjectival modifier with no heavy dependent nodes
ptp	adjectival modifier having its own dependent nodes or located after noun
adv	adverb
prep	preposition (depends on a noun)
conj	conjunction
digit	number
card	cardinal numeral

auxs	auxiliary with the same subject
auxd	auxiliary with different subject
inf	infinitive (depends on a verb)
sent	subordinate clause
by	particle “бы”
li	particle “ли”
pt	particle
emph	emphatic conjunction “и”
homo	conjunct
sharp	part of a compound word
hyph	part of a hyphenated word
quoml	left quotation mark
quomr	right quotation mark
misc	other

Top-level syntactic classes

sent	sentence
np	noun phrase
prepnp	prepositional phrase
prep	preposition
adj	adjective/participle phrase
adv	adverbial phrase
fin	finite verb phrase
krat	short-form adjective/participle phrase
inf	infinitive phrase
dee	adverbial participle phrase
imper	imperative verb phrase
misc	other