UDK 004.912

Davydov M. V.[1], Lozynska O. V.[2], Pasichnyk V. V.[3]

[1]PhD., Associate Professor of Information Systems and Networks Department, Lviv Polytechnic National University, Lviv, Ukraine
[2]PhD., Assistant of Information Systems and Networks Department, Lviv Polytechnic National University, Lviv, Ukraine
[3]Dr.Sc., Professor, Professor of Information Systems and Networks Department, Lviv Polytechnic National University, Lviv, Ukraine

EFFECTIVE ALGORITHM FOR PARSING SENTENCES USING SEMANTICALLY ATTRIBUTED WEIGHTED AFFIX CONTEXT FREE

Context. The problem of increasing efficiency of affix grammars over a finite lattice (AGFL) is considered. AGFL is a context-free grammar with flexible and compact form of productions for parsing texts in natural languages.

Objective. The goal of the work is to increase efficiency of parsing sentences by means of AGFL with a modification that adds semantical attributes to the productions and introduces a new form of production called the “template production”. This modification helps to decrease the number of productions that are required to describe a language and lets reduce the computational complexity of the parsing algorithm.

Method. A mathematical model of the template production is developed and the theorem is proved that claims that the normal form of the template production exists and the normalization procedure produces an equivalent grammar. The normal form is utilized to increase efficiency of parsing Ukrainian sentences. The template production helps to represent ontology-based rules in a short and computationally inexpensive way. The normal form of template production is studied, and an effective algorithm for parsing sentences is proposed. The worst-case complexity of the proposed algorithm is $O\left(n^3 \cdot m_p^3 \cdot m_r\right)$, where n is the length of input string of terminals, m_p is the maximum number of combinations of symbol and attributes that can produce the same string of terminals, and m_r is the maximum number of productions that have the same starting non-terminal symbol in the right part. The growth of parsing time turned out to be almost linear function of the number of words in a sentence when parsing of sentences from the test database of Ukrainian fiction literature.

Results. The developed method has been implemented in the UkrParser software that is available open-source on GitHub.

Conclusions. The developed algorithm was tested on the database of Ukrainian sentences and demonstrated ten times faster parsing speed than Stanford parser. The future research can be focused on the development of grammatically attributed ontologies for wider set of topics that should improve results of semantical sentence parsing.

Keywords: weighted affix context-free grammar, semantic parsing, ontology-driven sentence parsing, template productions.

NOMENCLATURE

A – a set of all affixes;

$A(D_i)$ – a set of affixes that constitute domain D_i;

2^A – a power set of A;

D – a set of disjoint affix domains D_i, $D=\{D_i\}$;

D_{GENDER} – a domain for gender;

D_{NUMBER} – a domain for number;

D_{CASE} – a domain for case;

D_{SEM} – a domain for semantic affixes;

G – a weighted affix grammar over a finite lattice $G=(T,V,S,D,P)$;

n – a length of input sequence of terminals. It is equal to the number of words in input sentence;

P – a set of template and regular productions;

S – a starting symbol, $S \in V \setminus T$;

T – a set of all terminal symbols t_i;

V – a set of all symbols;

$w \in R^+$ – a multiplicative weight of production. The weight symbol is omitted where it is equal to 1.

INTRODUCTION

The problem of natural text parsing arises in such areas of computer applications as text summarization, machine translation, information retrieval, document classification, human-computer interaction, question answering systems, social networks monitoring, expert systems, etc.

The task of semantic parsing is a complex problem of artificial intelligence because its comprehensive solution requires the construction of a complete human knowledge model. Although such models are currently under development [1], no viable solution is available yet.

We propose an approach for partial syntactic and semantic parsing by means of weighted affix grammar over a finite lattice (WAGFL). WAGFL uses benefits of probabilistic context-free grammars (PCFG) [2] and affix grammars over a finite lattice (AGFL) developed by C.H.A. Koster [3]. Weighted and stochastic grammars are known to be equally expressive [4], but the approach based on weights is less restrictive and thus more flexible.

This article supports an approach where semantic analysis is integrated into the syntax parsing algorithm. This approach helps to decrease the number of intermediate constructions that have to be considered. It is especially important for flexible word order languages like Ukrainian and other slavonic languages.

The main contribution of this work is an approach to effective representation of weighted affix context-free grammar using a special form of “template productions”. A small review of the existing methods is given in Section 2, “template productions” and the algorithm for parsing sentences are introduced in Section 3, experiments are provided in Section 4, parsing results are given in Section 5, and the results are discussed in Section 6.

1 PROBLEM STATEMENT

The problem is to develop effective methods for integrating semantic attributes into productions of weighted

DOI 10.15588/1607-3274-2017-4-14

affix context-free grammar and to develop computationally effective algorithm for parsing sentences. The sentence is considered as a list of words $w_1 w_2 \ldots w_n$ that is converted to a sequence of terminal symbols $t_1 t_2 ... t_n$ of the WAGFL grammar. The sentence parsing is formulated as a problem of finding a sequence of productions that has the maximum weight and can be applied sequentially to the starting symbol S to produce the given sequence of terminals $t_1 t_2 ... t_n$. The weight of the sequence is calculated as a multiplication of weights of all contained productions.

2 REVIEW OF THE LITERATURE

The problem of syntactic sentence parsing has been studied for a long time. Among many methods of parsing sentences, the approach based on generative grammars introduced by Noam Chomsky [5] is one of the most studied. Extended affix grammars (EAG) [6] and probabilistic context-free grammars (PCFG) [2] are generative grammar fundamental extensions widely used in linguistic applications nowadays.

Affix grammars, which belong to the family of two-level grammars, are a subset of augmented grammars. Productions of affix grammars are the productions that are extended with attributes. The domain of attributes is defined by a meta-grammar.

Efficient affix grammars over a finite lattice (AGFL) formalism and its parsing algorithm were developed by C. H. A. Koster [3]. The formalism imposes restrictions on a set of productions and attributes to make the parsing computationally inexpensive. However, it still leaves it expressive enough to parse most of the natural sentence structures. AGFL extensions that are based on probabilities were also studied by T. C. Smith and J. G. Cleary [7].

Our approach is based on weighted affix grammar over a finite lattice. It is close to the method introduced by C.H.A. Koster. However, we formulate lattice grammar and productions in a slightly different way what leads to a shorter form of productions and a more compact sentence parsing algorithm.

3 MATERIALS AND METHODS

For the purpose of partial semantic-syntactic parsing of sentences, a new parser was developed. It is based on the weighted affix grammar over a finite lattice. This grammar extends symbols of generative grammar with affixes what can be used to decrease the number of productions required to describe a language. Our definition of the affix grammar over a finite lattice is slightly different from the original given by C.H.A. Koster, but it has the same idea. This new definition was used to prove that some transformation rules can be applied to the grammar to speed up the process of parsing.

The weighted affix grammar over a finite lattice G is defined as a 5-tuple (T,V,S,D,P).

Regular productions have the form $\left\langle \left(V \times 2^A\right)^* \overset{w}{\to} \left(V \times 2^A\right)^* \right\rangle$, where $A = A(D) = \bigcup\limits_{D_j \in D} A\left(D_j\right)$ and $\left(V \times 2^A\right)^*$ represents all non-empty strings of attributed symbols $s_1 s_2 \ldots s_k$, with $k > 0$, $s_j = \left(v_j, A_j\right)$, $\left(v_j, A_j\right) \in V \times 2^A$.

Terminal symbols $t_i \in T$ do not have attributes. They usually represent words of parsed sentences. For example, the word “student” can be a male or female singular noun until it is known from the context. In terms of generative grammar, it can be written in the following way:

$$\left(noun,\{a_{FEMALE}, a_{SINGULAR}, a_{STUDENT}\}\right) \to \left(student,\varnothing\right),$$

$$\left(noun,\{a_{MALE}, a_{SINGULAR}, a_{STUDENT}\}\right) \to \left(student,\varnothing\right).$$

An alternative form is $\left(noun,\{a_{FEMALE}, a_{MALE}, a_{SINGULAR}, a_{STUDENT}\}\right) \to \left(student,\varnothing.\right)$ It represents both cases given above. Productions that generate terminal symbols are added by a morphological parser. If some word is a homograph, the morphological parser generates one production for every meaning of the word. The weight of each production represents the admissibility of this meaning in the parsed context.

In the example above, $a_{FEMALE}, a_{MALE}, a_{SINGULAR}$ are grammatical attributes, and $a_{STUDENT}$ is a semantical attribute. Semantical attributes are elements of domain D_{SEM}.

Providing regular productions for all possible combinations of affixes can be inefficient. Thus, a template form of production is introduced. This form is tailored for the needs of computationally efficient language processing.

The template production has the form $\left(v_1, D_{inh\,1}, A_{set\,1}\right)\ldots\left(v_k, D_{inh\,k}, A_{set\,k}\right)\overset{w}{\to}\left(v'_1, D_{uni\,1}, A_{req\,1}\right)\ldots\left(v'_m, D_{uni\,m}, A_{req\,m}\right)$, where $D_{inh\,1}, D_{inh\,2}, \ldots, D_{inh\,k} \subset D$ are domains which affixes are inherited by symbols $v_1, v_2, \ldots, v_k$; $D_{uni\,1}, D_{uni\,2}, \ldots, D_{uni\,m} \subset D$ are domains which affixes should be common for symbols v'_1, v'_2, ..., v'_m; $A_{set\,1}$, $A_{set\,2}$, …, $A_{set\,k} \subset A$ are additional affixes for symbols in the left part of the production, and $A_{req\,1}$, $A_{req\,2}$, …., $A_{req\,m} \subset A$ are required affixes for symbols in the right part of the production; and w is a multiplicative weight of the production.

The template form is equivalent to a set of regular productions by definition. Consider the following template and regular productions (1) and (2):

$$q = \Big\langle \left(v_1, D_{inh\,1}, A_{set\,1}\right)\ldots\left(v_k, D_{inh\,k}, A_{set\,k}\right)\overset{w}{\to},$$

$$\overset{w}{\to}\left(v'_1, D_{uni\,1}, A_{req\,1}\right)\ldots\left(v'_m, D_{uni\,m}, A_{req\,m}\right)\Big\rangle, \qquad (1)$$

$$p = \Big\langle \left(v_1, A_1\right)\ldots\left(v_k, A_k\right)\overset{w}{\to}\left(v'_1, A'_1\right)\ldots\left(v'_m, A'_m\right)\Big\rangle. \qquad (2)$$

Let $A_{uni}(p,q)$ denote the intersection of all attributes that should be uniform in the right part of regular production p in order to conform to template production q:

$$A_{uni}\left(p,q\right) = \bigcup_{i=1}^{m}\left(A'_i \cup \overline{A}\left(D_{uni\,i}\right)\right) \cap A\left(D_{uni\,1..m}\right),$$

$\overline{A}(D_{uni\,i}) = A \setminus A(D_{uni\,i})$, $D_{uni\,1..m} = D_1 \cup D_2 \cup \ldots \cup D_m$.

We say that regular production p conforms to template production q if requirements R1–R3 are met:

R1. $(\forall j \in 1\ldots n) D_j \in D_{uni\,1..m} \Rightarrow A_{uni}(p,q) \cap A(D_j) \neq \varnothing$;

R2. $(\forall i \in 1\ldots m) A_{req\,i} \subset A'_i$;

R3. $(\forall i \in 1\ldots k) A_i = A_{set\,i} \cup (A_{uni}(p,q) \cap A(D_{inh\,i}))$.

Requirement R1 assures that for each unified domain there is at least one common affix. Requirement R2 describes how required attributes are treated, and requirement R3 states how attributes of symbols in the left part of the production are obtained.

For instance, the Ukrainian equivalent of the English noun phrase "BEAUTIFUL STREET OF THE CITY" is "ГАРНА ВУЛИЦЯ МІСТА". In this noun phrase, the case, gender, and number of the adjective (ГАРНА) is coordinated by the case, gender, and number of the first noun (ВУЛИЦЯ), and the case of the second noun (МІСТА) should be GENITIVE. Semantical attribute for the whole phrase is taken from the word "STREET". The template production for this phrase in Ukrainian is

$$(NP,\{D_{GENDER},D_{NUMBER},D_{CASE},D_{SEM}\},\varnothing) \rightarrow$$
$$\rightarrow (ADJ,\{D_{GENDER},D_{NUMBER},D_{CASE}\},\varnothing)$$
$$(NP,\{D_{GENDER},D_{NUMBER},D_{CASE},D_{SEM}\},\varnothing)(NP,\varnothing,\{a_{GENITIVE}\}),$$

and the English equivalent is

$$(NP,\{D_{NUMBER},D_{SEM}\},\varnothing) \rightarrow (ADJ,\varnothing,\varnothing) \times$$
$$\times \{D_{NUMBER},D_{SEM}\},\varnothing)(prep,\varnothing,\{a_{OF}\})(NP,\varnothing,\varnothing),$$

where NP stands for noun phrase, ADJ stands for adjective, D_{GENDER}, D_{NUMBER}, D_{CASE}, D_{SEM} are domains for gender, number, case, and semantic affixes, respectively.

The Normal Form of Template Productions. The length of the right-hand side of a production is called its rank. Effective parsing of sentences using generative grammars can be achieved when the grammar is in Chomsky normal form (CNF) – the form that ensures that all productions of the grammar have the rank not more than 2. Template productions can also be converted to a form that has at most two symbols in the right part. This conversion is performed by applying simplification steps to all productions that have the rank greater than 2. Every step takes one template production with the rank $m > 2$ and produces two template productions – one with the rank 2 and one with the rank $m-1$. The process stops when there are no more productions with the rank 3 and above.

The simplification step takes one template production q of the form (1) and produces 2 template productions:

$$q_1 = \Big\langle (v_1, D_{inh\,1}, A_{set\,1})\ldots(v_k, D_{inh\,k}, A_{set\,k}) \overset{w}{\rightarrow}$$
$$\overset{w}{\rightarrow} (v'_1, D_{uni\,1}, A_{req\,1})(v'_{2..m}, D_{uni\,2..m}, \varnothing) \Big\rangle$$

and

$$q_2 = \Big\langle (v'_{2..m}, D_{uni\,2..m}, \varnothing) \overset{1.0}{\rightarrow}$$
$$\overset{1.0}{\rightarrow} (v'_2, D_{uni\,2}, A_{req\,2})\ldots(v'_m, D_{uni\,m}, A_{req\,m}) \Big\rangle,$$

where $D_{uni\,2..m} = D_{uni\,2} \cup D_{uni\,3} \cup \ldots \cup D_{uni\,m}$, and $v'_{2..m}$ is a new non-terminal symbol.

Theorem 1: The grammar obtained from original grammar G by the replacement of template production q with template productions q_1 and q_2 produces the same language.

In order to prove this, it is sufficient to show that:

1) all regular productions of form (2), which conform to template production (1), can be split into 2 productions p_1 and p_2 that conform to template productions q_1 and q_2, respectively;

2) all productions that conform to template productions q_1 and q_2 define the same grammar as productions that conform to template production q.

1) The First Part of the Proof: Given that production p conforms to template production q. It should be proven that there exists a split of p into 2 productions p_1 and p_2 such that p_1 conforms to q_1 and p_2 conforms to q_2.

This split can be found by the assignment

$$p_1 = \Big\langle (v_1, A_1)\ldots(v_k, A_k) \overset{w}{\rightarrow} (v'_1, A'_1)(v'_{2..m}, A'_{2..m}) \Big\rangle,$$

$$p_2 = \Big\langle (v'_{2..m}, A_{2..m}) \overset{1.0}{\rightarrow} (v'_2, A'_2)\ldots(v'_m, A'_m) \Big\rangle,$$

$A_{2..m} = A'_{2..m} = A_{uni}(p_{2,}q_2)$. In this case, production p_2 conforms to production q_2 because

$$A_{uni}(p,q) = ((((A'_1 \cup \overline{A}(D_{uni\,1}))\cap(A_{uni}(p_2,q_2) \cup$$
$$\cup \overline{A}(D_{uni\,2..m}))))\cap A(D_{uni\,1..m}).$$

Therefore, $A_{uni}(p,q) \cap A(D_{uni\,2..m}) \subset A_{uni}(p_2,q_2)$ what means that if $A_{uni}(p,q)$ satisfies requirement R1, then $A_{uni}(p_2,q_2)$ also satisfies it. Requirement R2 is satisfied because $A_{req\,2},\ldots,A_{req\,m}$ and $A'_2,\ldots,A'_m$ are taken from productions q and p, respectively, and p conforms to q by the assumption of the theorem. Requirement R3 is satisfied due to the fact that $A'_{2..m} = \varnothing \cup (A_{uni}(p_2,q_2) \cap A(D_{uni\,2..m})) = A_{uni}(p_2,q_2)$.

2) The Second Part of the Proof: Given that p_1 conforms to q_1 and p_2 conforms to q_2, it can be proved that they can be composed into a single production that conforms to q.

First of all, it should be noted that symbol $v'_{2..m}$ is a new non-terminal symbol, and thus it can't be found in any other production. Productions p_1 and p_2 can be applied sequentially only when $A_{2..m} = A'_{2..m}$. Due to the fact that p_2 conforms to q_2, requirement R3 ensures that

$$A_{2..m} = \varnothing \cup \left(A_{uni}(p_2,q_2) \cap A\left(D_{uni\,2..m}\right)\right) = A_{uni}(p_2,q_2).$$

So, $A_{uni}(p_1,q_1)$ can be calculated using the formula

$$A_{uni}(p_1,q_1) = \left(\left(\left(A'_1 \cup \overline{A}\left(D_{uni\,1}\right)\right)\cap\right.\right.$$

$$\left.\left.\cap\left(A_{uni}(p_2,q_2) \cup \overline{A}\left(D_{uni\,2..m}\right)\right)\right)\right) \cap A\left(D_{uni\,1..m}\right) =$$

$$= \left(A'_1 \cup \overline{A}\left(D_{uni\,1}\right)\right) \cap \left(\bigcap_{i=2}^{m}\left(A'_i \cup \overline{A}\left(D_{uni\,i}\right)\right)\right) \cap A\left(D_{uni\,1..m}\right) = A_{uni}(p,q).$$

Therefore, requirements R1 and R3 are satisfied for productions p and q because they are satisfied for p_1 and q_1; requirement R2 follows from the fact that p_1 and p_2 conform to q_1 and q_2, respectively. Thus, production p, that is obtained from p_1 and p_2, conforms to template production q. This concludes the proof of Theorem 1.

Algorithm for Parsing Sentences. The problem of sentence parsing is formulated as a problem of finding a sequence of productions that has the maximum weight and can be applied sequentially to some starting attributed symbol (S, A_s) to produce a given sequence of terminals $t_1 t_2 ... t_n$. The weight of the sequence is calculated as a multiplication of weights of all contained productions. If the right part of a production contains only one symbol, the weight of the production should not exceed 1 in order to avoid cyclic productions that can increase weight of non-terminal symbols during the bottom-up parsing procedure.

The developed algorithm for parsing sentences is based mostly on probabilistic CYK algorithm. The main difference is that symbols are compared not only by weight but also by the set of affixes. The algorithm uses the notion of weighted attributed symbol – a 3-tuple (w, v, A_v) that contains weight w, symbol v, and set of affixes $A_v \subset A(D)$. We say that weighted attributed symbol (w_1, v_1, A_1) dominates weighted attributed symbol (w_2, v_2, A_2) if $w_1 \geq w_2$, $v_1 = v_2$, and $A_2 \subset A_1$.

In the worst-case scenario, the computational complexity of the algorithm is $O(n^3 \cdot m_p^3 \cdot m_r)$, where n is the length of input string of terminals, m_p is the maximum number of combinations of symbols and attributes that can produce the same string of terminals (this value can be treated as the ambiguity of the language being parsed), and m_r is the maximum number of productions that have the same starting non-terminal symbol in the right part.

The parsing algorithm can be described by the following pseudocode:

Algorithm $ParseSentense(s)$.

Input. String of terminals $s = t_1 t_2 ... t_n$.

Output. Sequence of productions that produce string s.

Let $P[1..n, 1..n] = \varnothing$ be an array, each element of which is a set of weighted attributed symbols.

```
Initialize $P[i,1] = \{(1, t_i, \varnothing)\}$, $i = 1,2, ..., n$.
for $j = 1,2,...,n$ do // $j$ is a length of substring of terminals
  for $k = 1,2,...,n-j+1$ do // $k$ is a start of substring
    for $s = 1,2,..., j-1$ do // $s$ is a split of the substring
      for all $(w_1, v_1, A_1) \in P[k,s]$ do
        for all productions of type
        $(v, D_{inh}, A_{set}) \overset{w}{\to} (v_1, D_{uni\,1}, A_{req\,1})(v_2, D_{uni\,2}, A_{req\,2})$ do
          for all $(w_2, v_2, A_2) \in P[k+s, j-s]$ do
            if( $A_{req\,1} \subset A_1 \wedge ((\forall D_i \in D_{uni\,1}) A_1 \cap A(D_i) \neq \varnothing)$) then
            and $((\forall D_i \in (D_{uni\,1} \cap D_{uni\,2})) A_1 \cap A_2 \cap A(D_i) \neq \varnothing)$) then
              Let
              $t = (w \cdot w_1, v, A_{set} \cup ((A_1 \cup \overline{A}(D_{uni1})) \cap (A_2 \cup \overline{A}(D_{uni2})) \cap A(D_{inh})))$
              if $t$ is not dominated by any element of $P[k, j]$,
              then add $t$ to $P[k, j]$ and remove elements dominated by $t$
    Let list $L := allelementsofP[k, j]$
    for all $(w_1, v_1, A_1) \in L$ do// process productions of rank 1
      for all productions of type $(v, D_{inh}, A_{set}) \overset{w}{\to} (v_1, D_{uni}, A_{req})$ do
        if( $A_{req} \subset A_1 \wedge ((\forall D_i \in D_{uni}) A_1 \cap A(D_i) \neq \varnothing)$ then
          let $t = (w \cdot w_1, v, A_{set} \cup (A_1 \cap A(D_{inh})))$
          if $t$ is not dominated by any element of $L$, then append $t$ to $L$.
    Add elements of $L$ to $P[k, j]$.
```

If $P[1,n]$ doesn't contain any triple (w, S, A_s), where S is a starting symbol of the grammar, the parsing is impossible. If it does, select a triple with the maximum weight w among them and reconstruct all productions that are required to produce string $t_1 t_2 ... t_n$.

4 EXPERIMENTS

The algorithm for parsing sentences was implemented in UkrParser[1] open source software project. This project contains classes for morphology analysis and sentence parser. The morphology for Ukrainian language is implemented in com.langproc. UkrainianISpellMorphology and com.langproc.UkrainianGrammarlyMorphology clases. The first class is based on open source project iSpell-uk[2] by Andriy Rysin and the second is based on Ukrainian morphology database gracefully provided by Mariana Romanyshyn from Grammarly. The algorithm for parsing sentences is implemented in com.langproc.APCFGParser class and productions for parsing Ukrainian sentences are placed in com.langproc.APCFGUkrainian class.

[1] https://github.com/mdavydov/UkrParser
[2] https://sourceforge.net/projects/ispell-uk/

Computational efficiency of the developed algorithm was tested on database of 500 sentences from “Fata Morgana” story by Michael Kotsyubynsky. The average sentence parsing time depending of the sentence length is depicted in Fig. 1. These results were obtained on computer with 2.4 GHz Intel Core i5 CPU. The parsing time grows turned out to be almost linear notwithstanding the worst-case cubic estimate provided in Section 3.

5 RESULTS

The developed approach for mixed semantic and syntactic sentence parsing was used for parsing and translation of the annotated Ukrainian Sign language and the Ukrainian Spoken language [8], where the translation based on the parser that utilized productions generated from ontologies outperforms the parser that utilized only syntactic productions by 25% (90% of correct translations as compared to 65% correct translations obtained when using only syntactical productions).

An example of parsing sentence “Моя донька ходить у дитячий садок” (My daughter attends nursery school) by means of the developed method is shown in Figure 2.

In this example the following rules were added from subject area “Education”:

NG(персона)[=] -> NG(донька)[=];

VP(ходити-відвідувати)[=] -> VP(ходити)[=];

NG(дошкільний-заклад)[=] 1.1-> adj(дитячий)[=] AN(садок)[=];

DS(ходити-відвідувати)[=]1.1->

-> NP(персона)[=] VP(ходити-відвідувати)[=] V DNP(дошкільний-заклад)[c4];

where NG stands for Noun Group, NP – Noun Phrase, VP – Verb Phrase, adj – adjective, AN – annotated noun, DS – Declarative Sentence, V – preposition of place, “=” means default grammar attributes for current phrase, c4 stands for “Casus 4”. The weights of ontology-driven rules are intentionally made greater than 1 to supersede default syntactical rules.

6 DISCUSSION

The experimental results on database of Ukrainian sentences show significant speed-up in comparison with well-known context-free grammar parsers. This result was achieved by using compact form of production with syntactical and semantical attributes. In comparison with Stanford Parser[3] the average sentence parsing time was decreased in about 10 times.

CONCLUSIONS

This article demonstrated an efficient algorithm for parsing sentences by means of weighted affix context-free grammar with semantical attributes. The developed algorithm is based on the normal form of “template productions” that were introduced. The algorithms has worst-case cubic complexity, that turned out to be almost linear in real example.

The obtained sentence parsing trees are more semantically rich than the parsing trees obtained by means of regular syntactic parser. Additional computational cost for that is not very high because only hypernyms of words that are present in the sentence and corresponding expressions are included into the grammar.

The future research will be focused on optimal weight assignment and automatic extraction of productions that are specific for particular subject area.

ACKNOWLEDGEMENTS

The work was supported by the state budget scientific research project of Lviv Polytechnic National University and Ministry of Education and Science of Ukraine “Mathematical and algorithmic modeling of translation processes from gestures into text for specialized computer systems” (state registration number 0111U001222).

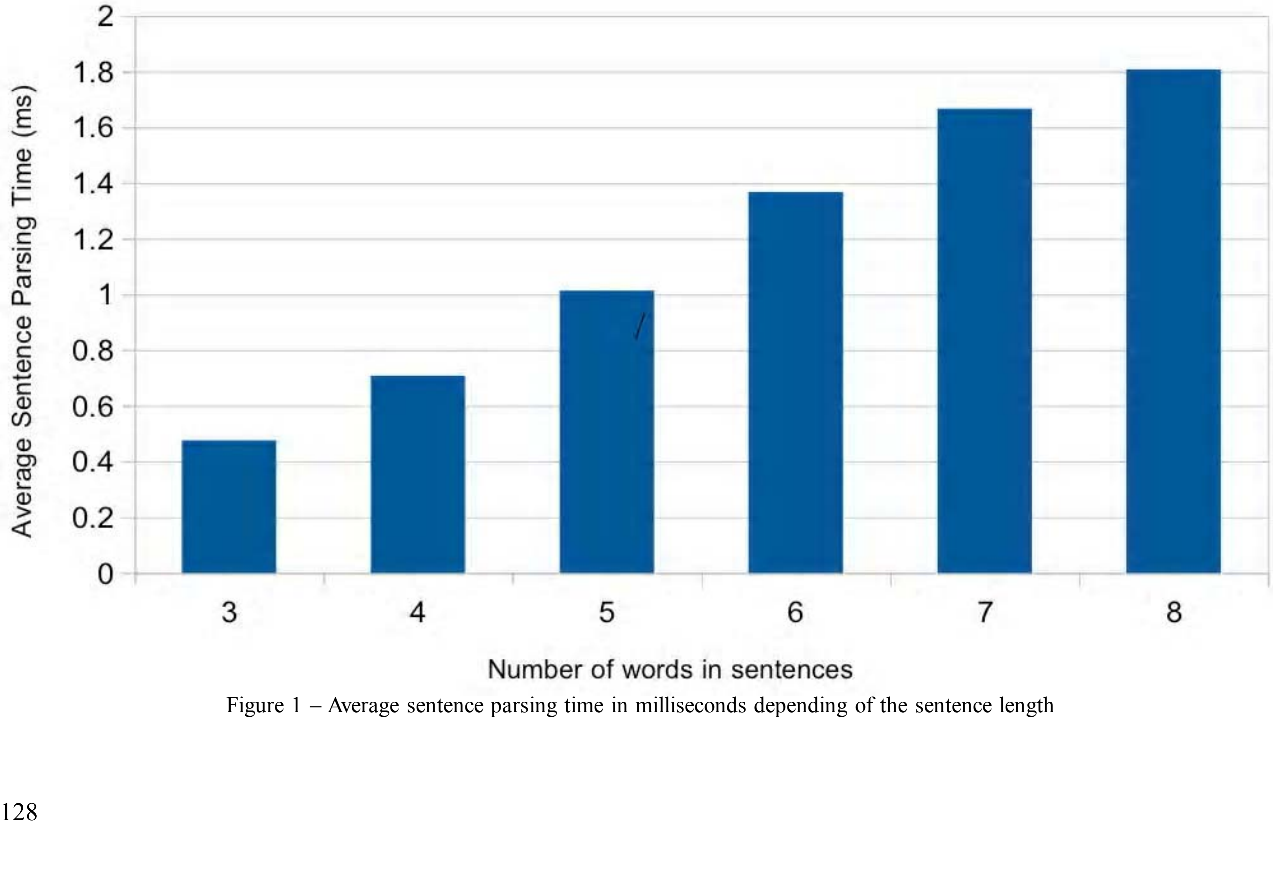

Figure 1 – Average sentence parsing time in milliseconds depending of the sentence length

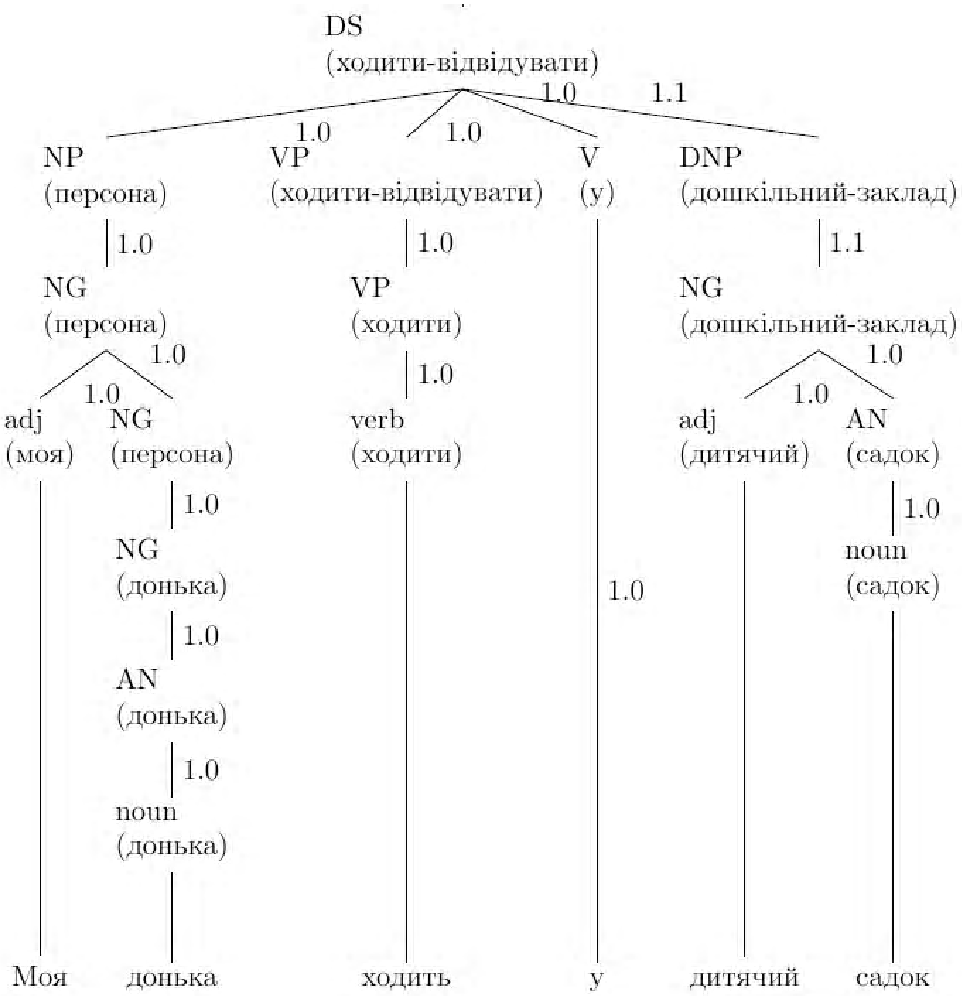

Figure 2 – The result of parsing of the Ukrainian sentence “Моя донька ходить у дитячий садок” (My daughter attends nursery school)

REFERENCES

1. ConceptOnto: An upper ontology based on ConceptNet / [E. Najmi, K. Hashmi, Z. Malik et al.] // Computer Systems and Applications (AICCSA), 2014 IEEE/ACS 11th International Conference on Computer Systems and Applications (AICCSA), November 10–13, Doha, Qatar. – 2014. – P. 366–372. DOI: 10.1109/AICCSA.2014.7073222.
2. Eddy S.R. RNA sequence analysis using covariance models / S. R. Eddy, R Durbin // Nucleic Acids Research. – 1994. – Vol. 22. – № 11. – P. 2079–2088.
3. Koster C.H.A. Affix Grammars for natural languages / C.H.A. Koster // In: Attribute Grammars, Applications and Systems, International Summer School SAGA, Lecture Notes in Computer Science, Prague, Czechoslovakia. – 1991. – Vol. 545. – P. 469–484.
4. Smith N. A. Weighted and Probabilistic Context-Free Grammars Are Equally Expressive / N. A. Smith, M. Johnson // Computational Linguistics. – 2007. – Vol. 33, № 4. – P. 477–491. DOI:10.1162/coli.2007.
5. Chomsky N. Three models for the description of language / N. Chomsky // IRE Transactions on Information Theory – № 2 (3). – 1956. – P. 113–124. DOI:10.1109/TIT.1956.1056813.
6. Oostdijk N. An Extended Affix Grammar for the English Noun Phrase / N. Oostdijk // In: Jan Aarts and Wim Meijs (eds), Corpus Linguistics. Recent Developments in the Use of Computer Corpora in English Language Research, Amsterdam: Rodopi. – 1984.
7. Smith T.C. Probabilistic Unification Grammars / T. C. Smith, J. G. Cleary // In Australasian Natural Language Processing Summer Workshop. – 1997. – P. 25 32.
8. Lozynska O. Information technology for Ukrainian Sign Language translation based on ontologies / O. Lozynska, M. Davydov // ECONTECHMOD: an international quarterly journal on economics of technology and modelling processes. – 2015. – Vol. 04, No. 2. – P. 13–18.

Article was submitted 22.09.2017.
After revision 23.10.1017.

Давидов М. В[1], Лозинська О. В.[2], Пасічник В. В.[3]

[1]Канд. техн. наук, доцент кафедри «Інформаційні системи та мережі», Національний університет «Львівська політехніка», Львів, Україна

[2]Канд. техн. наук, асистент кафедри «Інформаційні системи та мережі», Національний університет «Львівська політехніка», Львів, Україна

[3]Д-р техн. наук, професор, професор кафедри «Інформаційні системи та мережі», Національний університет «Львівська політехніка», Львів, Україна

ЕФЕКТИВНИЙ АЛГОРИТМ ДЛЯ СИНТАКСИЧНОГО АНАЛІЗУ РЕЧЕНЬ З ВИКОРИСТАННЯМ СЕМАНТИЧНО ПОЗНАЧЕНИХ ЗВАЖЕНИХ АФІКСНИХ КОНТЕКСТНО-ВІЛЬНИХ ГРАМАТИК

Актуальність. Розглядається задача підвищення ефективності афіксних граматик над скінченною граткою (AGFL). AGFL – це контекстно-вільна граматика з гнучкими і компактними формами для розбору текстів на природних мовах.

Мета роботи. Метою роботи є підвищення ефективності розбору речень за допомогою модифікації AGFL, яка додає семантичні атрибути в продукції граматики і вводить нову форму продукцій під назвою «шаблонна продукція». Ця модифікація допомагає зменшити кількість продукцій, необхідних для опису мови, і дозволяє зменшити обчислювальну складність алгоритму синтаксичного аналізу.

Метод. Розроблено математичну модель шаблонної продукції і доведено теорему про те, що існує нормальна форма шаблонних продукцій, а процедура нормалізації породжує еквівалентну граматику. Нормальна форма використовується для підвищення ефективності розбору українських речень. Шаблонні продукції допомагають описувати правила на основі онтології в короткій і обчислювально ефективній формі. Вивчається нормальна форма шаблонних продукцій і пропонується ефективний алгоритм для розбору речень. У найгіршому випадку обчислювальна складність запропонованого алгоритму становить $O\left(n^3 \cdot m_p^3 \cdot m_r\right)$, де n – довжина вхідного рядка терміналів, m_p – максимальне число комбінацій символів і атрибутів, які можуть породжувати один і той самий рядок терміналів, m_r – максимальне число продукцій, які мають той самий стартовий нетермінальний символ в правій частині. Час синтаксичного аналізу виявився майже лінійною функцією від кількості слів у реченні при розборі тестової бази речень української художньої літератури.

Результати. Розроблений метод був реалізований в програмному забезпеченні UkrParser, яке доступне з відкритим вихідним кодом на GitHub.

Висновки. Розроблений алгоритм був протестований на базі даних українських речень і продемонстрував в десять разів більшу швидкість розбору, ніж аналізатор «Stanford Parser». Майбутні дослідження можуть бути сфокусовані на розробці граматично доповнених онтологій для більш широкого набору предметних областей, що має поліпшити результати семантичного аналізу речень.

Ключові слова: зважена афіксна контекстно-вільна граматика, семантичний розбір, розбір речень з використанням онтологій, шаблонна продукція.

Давыдов М. В.[1], Лозинская О. В.[2], Пасечник В. В.[3]

[1]Канд. техн. наук, доцент кафедры «Информационные системы и сети», Национальный университет «Львовская политехника», Львов, Украина

[2]Канд. техн. наук, ассистент кафедры «Информационные системы и сети», Национальный университет «Львовская политехника», Львов, Украина

[3]Д-р техн. наук, профессор, профессор кафедры «Информационные системы и сети», Национальный университет «Львовская политехника», Львов, Украина

ЭФФЕКТИВНЫЙ АЛГОРИТМ ДЛЯ СИНТАКСИЧЕСКОГО АНАЛИЗА ПРЕДЛОЖЕНИЙ С ИСПОЛЬЗОВАНИЕМ СЕМАНТИЧЕСКИ ОБОЗНАЧЕННЫХ ВЗВЕШЕННЫХ АФФИКСНЫХ КОНТЕКСТНО-СВОБОДНЫХ ГРАММАТИК

Актуальность. Рассматривается задача повышения эффективности аффиксных грамматик над конечной решеткой (AGFL). AGFL – это контекстно-свободная грамматика с гибкой и компактной формой записи продукций для разбора текстов на естественных языках.

Цель работы. Целью работы является повышение эффективности разбора предложений с помощью модификации AGFL, которая добавляет семантические атрибуты в продукции грамматики и вводит новую форму продукций под названием «шаблонная продукция». Эта модификация помогает уменьшить количество продукций, необходимых для описания языка, и позволяет уменьшить вычислительную сложность алгоритма синтаксического анализа.

Метод. Разработана математическая модель шаблонной продукции, и доказана теорема о том, что существует нормальная форма шаблонных продукций, и процедура нормализации порождает эквивалентную грамматику. Нормальная форма используется для повышения эффективности разбора украинских предложений. Шаблонные продукции помогают описывать правила на основе онтологии в краткой и вычислительно эффективной форме. Изучается нормальная форма шаблонных продукций, и предлагается эффективный алгоритм для разбора предложений. В наихудшем случае вычислительная сложность предлагаемого алгоритма составляет $O\left(n^3 \cdot m_p^3 \cdot m_r\right)$, где n – длина входной строки терминалов, m_p – максимальное число комбинаций символов и атрибутов, которые могут порождать одну и ту же строку терминалов, и m_r – максимальное число продукций, которые имеют тот же стартовый нетерминальный символ в правой части. Время синтаксического анализа оказалось почти линейной функцией от количества слов в предложении при разборе тестовой базы предложений украинской художественной литературы.

Результаты. Разработанный метод был реализован в программном обеспечении UkrParser, которое доступно с открытым исходным кодом на GitHub.

Выводы. Разработанный алгоритм был протестирован на базе данных украинских предложений и продемонстрировал в десять раз большую скорость разбора, чем анализатор “Stanford Parser”. Будущие исследования могут быть сфокусированы на разработке грамматически дополненных онтологий для более широкого набора предметных областей, что должно улучшить результаты семантического анализа предложений.

Ключевые слова: взвешеная контекстно-свободная аффиксная грамматика, семантический разбор предложений, разбор предложений на основе онтологий, шаблонная продукция.

REFERENCES

1. Najmi E., Hashmi K., Malik Z., Rezgui A., Khanz H. U. ConceptOnto: An upper ontology based on ConceptNet, *Computer Systems and Applications (AICCSA), 2014 IEEE/ACS 11th International Conference on Computer Systems and Applications (AICCSA), November 10–13, Doha*. Qatar, 2014, pp. 366–372. DOI: 10.1109/AICCSA.2014.7073222.
2. Eddy S. R., Durbin R. RNA sequence analysis using covariance models, *Nucleic Acids Research*, 1994, Vol. 22, No. 11, pp. 2079–2088.
3. Koster C.H.A. Affix Grammars for natural languages / C.H.A. Koster, *In: Attribute Grammars, Applications and Systems, International Summer School SAGA, Lecture Notes in Computer Science, Prague, Czechoslovakia*, 1991, Vol. 545, pp. 469–484.
4. Smith N. A., Johnson M. Weighted and Probabilistic Context-Free Grammars Are Equally Expressive, *Computational Linguistics*, 2007, Vol. 33, No. 4, pp. 477–491. DOI:10.1162/coli.2007.
5. Chomsky N. Three models for the description of language, *IRE Transactions on Information Theory,* No. 2 (3), 1956, pp. 113–124. DOI:10.1109/TIT.1956.1056813.
6. Oostdijk N. An Extended Affix Grammar for the English Noun Phrase, In*: Jan Aarts and Wim Meijs (eds), Corpus Linguistics. Recent Developments in the Use of Computer Corpora in English Language Research, Amsterdam*, Rodopi, 1984.
7. Smith T. C., Cleary J. G. Probabilistic Unification Grammars, *In Australasian Natural Language Processing Summer Workshop*, 1997, pp. 25–32.
8. Lozynska O., Davydov M. Information technology for Ukrainian Sign Language translation based on ontologies, *ECONTECHMOD: an international quarterly journal on economics of technology and modelling processes*, 2015, Vol. 04, No. 2, pp. 13–18.