

## ЗАМЕТКА О НЕКОРРЕКТНОЙ РАБОТЕ АЛГОРИТМА СИНТАКСИЧЕСКОГО АНАЛИЗА КОВИНГТОНА

Спирицева П.И.

Тверской государственный университет, г. Тверь

---

*Поступила в редакцию 05.04.2021, после переработки 08.04.2021.*

---

Рассмотрен алгоритм Ковингтона для синтаксического анализа проективных зависимостей LSUP. Показано, что алгоритм анализирует не все проективные структуры (в частности, которые указаны в самой статье), поэтому использование данного алгоритма приводит к некорректным результатам.

**Ключевые слова:** синтаксический анализ, проективная зависимость, корпусная лингвистика, деревья зависимостей.

*Вестник ТвГУ. Серия: Прикладная математика. 2021. № 1. С. 71–75.*  
<https://doi.org/10.26456/vtpmk610>

### Введение

Активно развивающиеся технологии, рост вычислительных мощностей, доступность специальных размеченных коллекций текстов (корпусов) на различных языках способствуют увеличению интереса к корпусной лингвистике. Одним из актуальных направлений исследований в области обработки текстов является синтаксический анализ. Для него существует несколько подходов, в частности, детерминированный инкрементальный анализ на основе деревьев зависимостей.

### 1. Описание исследования

В статье [1] Михаэля Ковингтона предлагается фундаментальный алгоритм разбора естественного языка на основе деревьев зависимостей. Из описания следует, что данный алгоритм работает «по одному слову за раз, прикрепляя каждое слово к дереву, как только оно может быть прикреплено, что соответствует свойствам, заявленным для синтаксического анализатора в человеческом мозге».

В начале статьи автор приводит основные предположения о процессе разбора и результате анализа. Среди них выделим:

- Единство: результатом процесса синтаксического анализа является одно дерево (с уникальным корнем), содержащее все слова входной строки.

- Уникальность: у каждого слова только одна голова; то есть ссылки на зависимости действительно формируют дерево, а не какой-либо другой вид графа.
- Проективность: если слово  $A$  зависит от слова  $B$ , то все слова между  $A$  и  $B$  также подчинены  $B$ . Это эквивалентно отсутствию пересекающихся ветвей в дереве составляющих.

## 2. Описание алгоритма

Рассмотрим итоговый алгоритм синтаксического анализа проективных зависимостей с выполнением предположения об уникальности, приведённый в статье:

---

### Алгоритм LSUP

---

Дан список слов для анализа и два рабочих списка *Headlist* и *Wordlist*.

*Headlist* := []; (Слова, для которых ещё не найдена голова)

*Wordlist* := []; (Все слова, которые встретились на текущий момент)

#### Повторять

(Принять слово и добавить его в *Wordlist*)

$W$  := следующее слово для разбора;

*Wordlist* :=  $W$  + *Wordlist*

(Поиск зависимых от  $W$ ; это могут быть только последовательные элементы *Headlist*, начиная с самого последнего добавленного)

**Цикл**  $D$  := каждый элемент *Headlist*, начиная с первого **выполнять**

**Если**  $D$  может зависеть от  $W$  **то**

Связать  $D$  в качестве зависимого от  $W$

Удалить  $D$  из *Headlist*

**иначе**

Прервать цикл

**Конец условия**

**Конец цикла**

(Поиск головы  $W$ ; она должна содержать слово, предшествующее  $W$ )

$H$  := слово, непосредственно предшествующее  $W$  во входной строке;

**Заикнуть**

**Если**  $W$  может зависеть от  $H$  **то**

Связать  $W$  в качестве зависимого от  $H$

Прервать цикл

**Конец условия**

**Если**  $H$  независим **то**

Прервать цикл

**Конец условия**

$H$  := голова  $H$

**Конец заикливания**

**Если** голова для  $W$  не найдена **то**

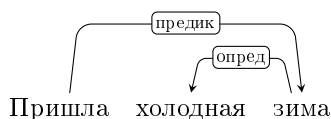
*Headlist* :=  $W$  + *Headlist*

**Конец условия**

**Пока** не будут разобраны все слова

---

В своей работе Михаэль Ковингтон также приводит примеры некоторых проективных (непересекающихся) структур, которые должен обрабатывать любой анализатор зависимостей. Рассмотрим пример одной из таких структур:



Применим алгоритм синтаксического анализа LSUP к данному предложению:

---

$Headlist := []$ ;  $Wordlist := []$ ;

Подробно пройдем по каждой итерации внешнего цикла:

1.  $W := \text{"Пришла"}$ ;  $Wordlist := [\text{"Пришла"}]$ ;  $Headlist := [\text{"Пришла"}]$ ;
2.  $W := \text{"холодная"}$ ;  $Wordlist := [\text{"холодная"}, \text{"Пришла"}]$ ;
  - Поиск зависимых от  $W$ :  
 $D := \text{"Пришла"}$ ;  $D$  зависит от  $W$ ? Нет  $\Rightarrow$  выходим из цикла.
  - Поиск головы  $W$ :  
 $H := \text{"Пришла"}$ ;  $W$  зависит от  $H$ ? Нет. У текущего  $H$  нет головы  $\Rightarrow$  выходим из цикла.  
 Голова для  $W$  не найдена  $\Rightarrow$  добавляем  $W$  в  $Headlist$ :  $Headlist := [\text{"холодная"}, \text{"Пришла"}]$ .
3.  $W := \text{"зима"}$ ;  $Wordlist := [\text{"зима"}, \text{"холодная"}, \text{"Пришла"}]$ ;
  - Поиск зависимых от  $W$ :  
 $D := \text{"холодная"}$ ;  $D$  зависит от  $W$ ? Да. Связываем "холодная" как зависимого от "зима". Удаляем  $D$  из  $Headlist$ :  $Headlist := [\text{"Пришла"}]$ .  
 $D := \text{"Пришла"}$ ;  $D$  зависит от  $W$ ? Нет  $\Rightarrow$  выходим из цикла.
  - Поиск головы  $W$ :  
 $H := \text{"холодная"}$ ;  $W$  зависит от  $H$ ? Нет.  
 $H := \text{"зима"}$  (голова для "холодная");  $W$  зависит от  $H$ ? Нет. У текущего  $H$  нет головы  $\Rightarrow$  выходим из цикла.  
 Голова для  $W$  не найдена  $\Rightarrow$  добавляем  $W$  в  $Headlist$ :  $Headlist := [\text{"зима"}, \text{"Пришла"}]$ .

Конец внешнего цикла (все слова в предложении обработаны).

---

Представленный алгоритм синтаксического анализа работает на данном примере некорректно. В результате работы LSUP получилось связать только «холодная» и «зима». Построить одно дерево, содержащее все слова, не вышло. Из этого можно сделать вывод, что не все из представленных в статье проективных структур могут быть проанализированы рассмотренным алгоритмом анализа. Необходимо провести его модификацию или использовать другие синтаксические анализаторы.

## Заключение

В данной заметке был рассмотрен алгоритм синтаксического анализа проек-

тивных зависимостей LSUP из статьи Михаэля Ковингтона [1]. Данный алгоритм анализирует не все проективные структуры (в частности, которые указаны в самой статье). Использование данного алгоритма приводит к некорректным результатам. Его необходимо модифицировать или применять другие алгоритмы синтаксического анализа. Стоит отметить, что на момент написания заметки было найдено подтверждение данного факта в статье [2]. В ней отмечается, что данный синтаксический анализатор действительно «...не анализирует все структуры проективных зависимостей, потому что при создании левых ссылок он предполагает, что голова узла  $i$  должна быть рефлексивно-транзитивной головой узла  $i - 1$ , что не всегда так».

#### Список литературы

- [1] Covington M.A. A fundamental algorithm for dependency parsing // Proceedings of the 39th annual ACM Southeast Conference. 2001. Pp. 95–102.
- [2] Gomez-Rodriguez C., Carroll J., Weir D.J. Dependency Parsing Schemata and Mildly Non-Projective Dependency Parsing // Computational Linguistics. 2011. Vol. 37, № 3. Pp. 541–586.

#### Образец цитирования

Спирицева П.И. Заметка о некорректной работе алгоритма синтаксического анализа Ковингтона // Вестник ТвГУ. Серия: Прикладная математика. 2021. № 1. С. 71–75. <https://doi.org/10.26456/vtprm610>

#### Сведения об авторах

1. **Спирицева Полина Игоревна**

магистрант факультета прикладной математики и кибернетики Тверского государственного университета.

*Россия, 170100, г. Тверь, ул. Желябова, д. 33, ТвГУ.*

## A NOTE ON THE INCORRECT OPERATION OF THE COVINGTON PARSING ALGORITHM

**Spiritseva Polina Igorevna**

Master student of the Faculty of Applied Mathematics and Cybernetics,  
Tver State University

*Russia, 170100, Tver, 33 Zhelyabova str., TverSU.*

---

*Received 05.04.2021, revised 08.04.2021.*

---

The Covington algorithm LSUP for the syntactic analysis of projective dependencies is considered. It is shown that the algorithm does not analyze all the projective structures (in particular, those specified in the article itself), so the use of this algorithm leads to incorrect results.

**Keywords:** syntactic analysis, projective dependency, corpus linguistics, dependency trees.

### Citation

Spiritseva P.I., “A note on the incorrect operation of the Covington parsing algorithm”, *Vestnik TvGU. Seriya: Prikladnaya Matematika [Herald of Tver State University. Series: Applied Mathematics]*, 2021, № 1, 71–75(in Russian). <https://doi.org/10.26456/vtpmk610>

### References

- [1] Covington M.A., “A fundamental algorithm for dependency parsing”, *Proceedings of the 39th annual ACM Southeast Conference*, 2001, 95–102.
- [2] Gomez-Rodriguez C., Carroll J., Weir D.J., “Dependency Parsing Schemata and Mildly Non-Projective Dependency Parsing”, *Computational Linguistics*, **37:3** (2011), 541–586.