Артем Копецкий, 152

HW-5 «Базовые алгоритмы зависимостного парсинга»

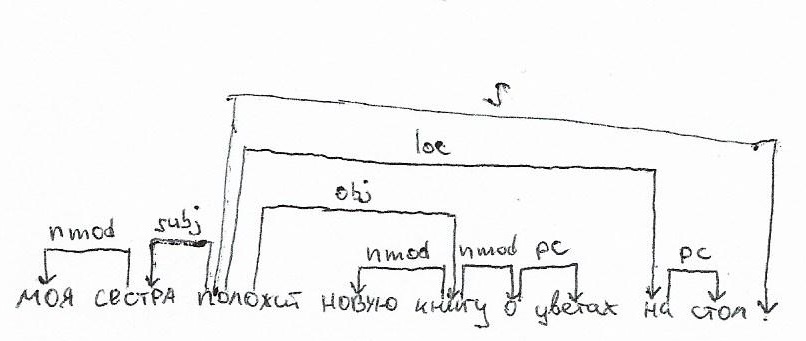
Как было оговорено на консультации, задание заключается в применении 3 базовых алгоритмов для парсинга данного предложения:

*Моя сестра положит новую книгу о цветах на стол.*

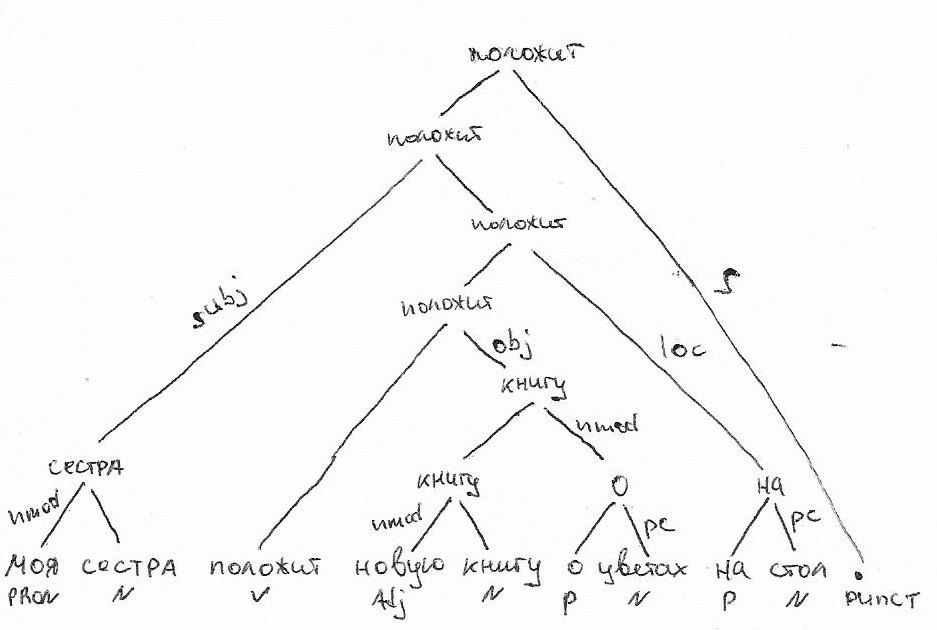
**1. Переход от НС-дерева к дереву зависимостей**

Здесь все просто: построить дерево в терминах НС, затем заменить фразовые категории в узлах вершинами и приписать на соответствующие ребра тип зависимости между зависимым и его вершиной.

Структура зависимостей



Дерево зависимостей (я решил строить его бинарным, т.к. бинарность является требованием для применения алгоритма CYK, например)



**2. Алгоритм, основанный на ограничениях**

В целом эти алгоритмы представляют синтаксическую структуру в виде полносвязного графа, где слова – вершины, зависимости – ребра с метками. Изначально предполагается, что все вершины могут быть связаны (то есть все управляет всем и все подчиняется всему), однако затем на основе составляемых ограничений неверные и лишние (нарушающие ограничения) связи удаляются. Важное правило, насколько я понял, заключается в том, что связи (ребра) не должны пересекаться.

Выпишем ограничения для полного прилагательного, местоимения и существительного (я понимаю задания так, что это нужно сделать на основе имеющегося предложения, а не устройства русского в общем):

1. Прилагательное

word(pos(x)) = ADJ → label(x) = NMOD, word(mod(x)) = N, pos(x) < mod(x)

2. Местоимение

word(pos(x)) = PRON → label(x) = NMOD, word(mod(x)) = N, pos(x) < mod(x)

3. Существительное

3.1 word(pos(x)) = N → label(x) = SUBJ, word(mod(x)) = V, pos(x) < mod(x)

3.2 word(pos(x)) = N → label(x) = OBJ, word(mod(x)) = V, mod(x) < pos(x)

3.3 word(pos(x)) = N → label(x) = PC, word(mod(x)) = P, mod(x) < pos(x)

Теперь посмотрим на допустимые разборы для связи местоимения *моя* и существительного *сестра*, заполнив соответствующую таблицу.

Фрагмент соответствующего графа:

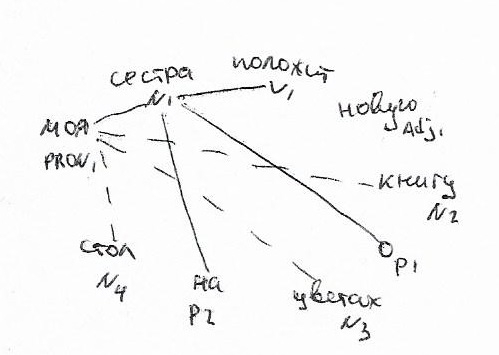


Таблица разборов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PRON1 / N1 | <SUBJ, V1> | <OBJ, V1> | <PC, P1> | <PC, P2> |
| <NMOD, N1> | 1 | 1 | 1 | 1 |
| <NMOD, N2> | 1 | 1 | 0 | 0 |
| <NMOD, N3> | 1 | 1 | 1 | 0 |
| <NMOD, N4> | 1 | 1 | 1 | 1 |

Разборы <NMOD, N2> / <PC, P1>, <NMOD, N2> / <PC, P2> и <NMOD, N3> / <PC, P2> недопустимы, потому в таком случае ребра зависимостей пересекаются. Все остальные разборы при таких ограничениях допустимы. Как видно, требуются еще ограничения, чтобы разрешить только верный разбор (например, ввести ограничения на грамм. категории).

«Нарушением каких ограничений можно объяснить отсутствие связи между прилагательным *новую*и существительным *книгу.*» – честно говоря, этот вопрос непонятен, потому как связь между *новую* и *книгу* явно должна существовать.

Наконец, опишем ограничения для верного привязывания предложных групп к вершинам, среди которых следующие:

1. word(pos(x)) = P → label(x) = NMOD, word(mod(x)) = N, mod(x) < pos(x)

2. word(pos(x)) = P → label(x) =LOC, word(mod(x)) = V, mod(x) < pos(x)

3. Отношение *word(pos(x)) = P → label(x) = NMOD, word(mod(x)) = N, mod(x) < pos(x)* запрещено, если уже есть отношение *word(pos(x)) = PRON → label(x) = NMOD, word(mod(x)) = N, pos(x) < mod(x)*

4. Не может быть двух отношений *word(pos(x)) = P → label(x) = NMOD, word(mod(x)) = N, mod(x) < pos(x)* для одного и того же имени

Если аналогичным образом по графу и этим ограничениям составить таблицу для двух имеющихся предлогов, будет видно, что эти 4 ограничения обеспечивают только верное приписывание обеих предложных групп к своим вершинам, то есть <NMOD, *книгу*> для предлога *о* и <LOC, *положит*> для предлога *на*.

**3. Shift-reduce алгоритм Nivre**

Здесь я просто пошагово применю данный алгоритм к имеющемуся предложению, следуя описанию процедуры и доступных операций из презентации Nivre.

*Моя сестра положит новую книгу о цветах на стол.*

НАЧАЛО

1. [ROOT]s q [моя]: shift

2. [ROOT моя]s q [сестра]: left-arc (nmod)

3. [ROOT]s q [сестра]: shift

4. [ROOT сестра]s q [положит]: left-arc (subj)

5. [ROOT]s q [положит]: right-arc (pred)

6. [ROOT положит]s q [новую]: shift

7. [ROOT положит новую]s q [книгу]: left-arc (nmod)

8. [ROOT положит]s q [книгу]: right-arc (obj)

9. [ROOT положит книгу]s q [о]: right-arc (nmod)

10. [ROOT положит книгу о]s q [цветах]: left-arc (pc)

11. [ROOT положит книгу о]s q [на]: shift

12. [ROOT положит книгу о на]s q [стол]: left-arc (pc)

13. [ROOT положит книгу о на]s q [.]: reduce (loc)

14. [ROOT положит книгу о]s q [.]: reduce (nmod)

15. [ROOT положит книгу]s q [.]: reduce (loc)

16. [ROOT положит]s q [.]: reduce (obj)

17. [ROOT]s q [.]: reduce (pred)

18. [ROOT .]s q []: right-arc (S)

19. [ROOT]s q []: reduce (S)

КОНЕЦ

Извлеченные зависимости:

моя ← сестра (nmod)

сестра ← положит (subj)

положит ← ROOT (pred)

новую ← книгу (nmod)

книгу ← положит (obj)

о ← книгу (nmod)

цветах ← о (pc)

стол ← на (pc)

на ← положит (loc)

. ← ROOT (S)

Итого, видно, что с помощью этого алгоритма успешно найдены и описаны все зависимости в этом предложении.