



НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ  
УНИВЕРСИТЕТ

# Dependency grammar and dependency parsing

Francis M. Tyers

[ftyers@hse.ru](mailto:ftyers@hse.ru)

<https://www.hse.ru/org/persons/209454856>

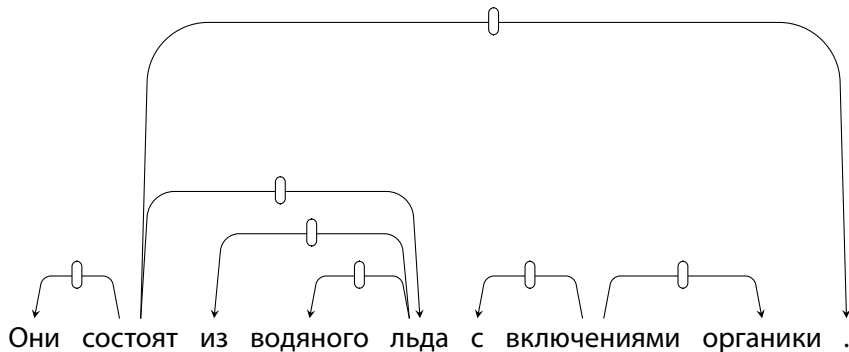
Национальный исследовательский университет  
«Высшая школа экономики» (Москва)

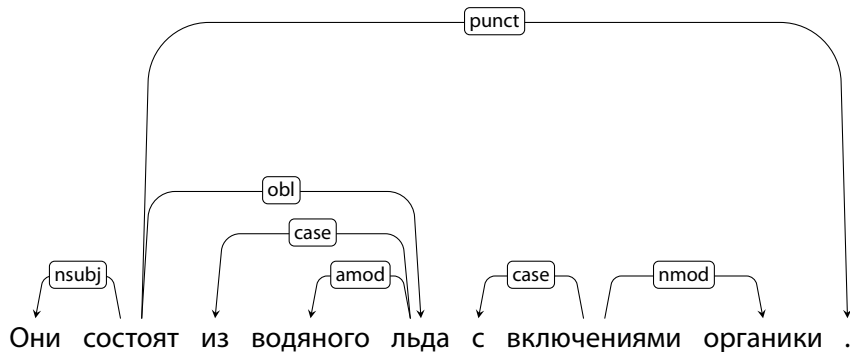
15 марта 2018 г.



- Word based
- No non-terminals
- Words are linked by one-way binary relations
- Relations may be typed or untyped

Они состоят из водяного льда с включениями органики .





<b>Superior</b>	<b>Inferior</b>	
Head	Dependent	
Governor	Modifier	
Regent	Subordinate	...
Mother	Daughter	
Parent	Child	

x



x

x



Dependency structures explicitly represent:

- head-dependent relations (directed arcs)
- functional categories (arc labels)

Phrase structures explicitly represent:

- phrases (non-terminal nodes)
- structural categories (non-terminal labels)

- Criteria for a syntactic relation between a head  $H$  and a dependent  $D$  in a construction  $C$  (Zwicky, 1985)<sup>1</sup>
  1.  $H$  determines the syntactic category of  $C$ ;  $H$  can replace  $C$
  2.  $H$  determines the semantic category of  $C$ ;  $D$  specifies  $H$
  3.  $H$  is obligatory,  $D$  may be optional
  4.  $H$  selects  $D$  and determines optionality of  $D$
  5. The form of  $D$  depends on  $H$  (agreement or government)
  6. Linear position of  $D$  is specified with reference to  $H$
- An issue:
  - Syntactic (and morphological) versus semantic criteria

---

<sup>1</sup>Zwicky, A. (1985) "Heads" *Journal of Linguistics*, 21:1–29

- Complex verb groups (auxiliary–main verb)
- Subordinate clauses (complementiser–verb)
- Coordination (coordinator–conjuncts)
- Adpositional phrases (adposition–nominal)
- Punctuation

A dependency graph,  $G$

- a set of  $V$  nodes,
- a set of  $A$  arcs,
- a linear precedence order  $<$  on  $V$

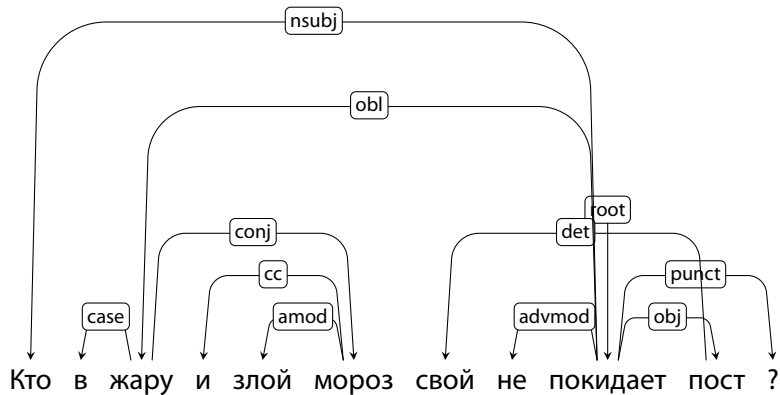
Labelled graphs:

...

- Nodes in  $V$  are labelled with word forms (and annotation)
- Arcs in  $A$  are labelled with dependency types









# Transition-based



## Data structures:

- Stack:
  - Starts as containing only the ROOT
- Buffer
  - Starts as containing the full sentence
- Arcs
  - Starts as empty

## Operations:

- SHIFT: Take the word on top of the buffer and put it on the stack
- LEFT-ARC: Make the word at the top of the stack the head of the word below it
  - Then remove the word at the top
- RIGHT-ARC: Make the word second from top the head of the word above it
  - Then remove the second from top word

ROOT Мы пошли домой

**Stack**

**Buffer**

ROOT Мы пошли домой

SHIFT

ROOT Мы пошли домой

**Stack**

**Buffer**

ROOT Мы пошли домой

SHIFT

ROOT Мы пошли домой

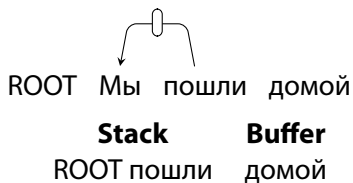
**Stack**

**Buffer**

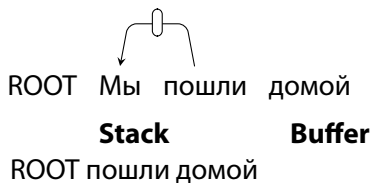
ROOT Мы пошли домой



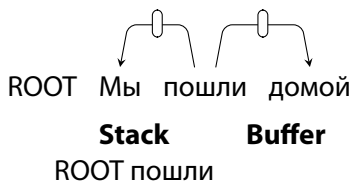
## LEFT-ARC



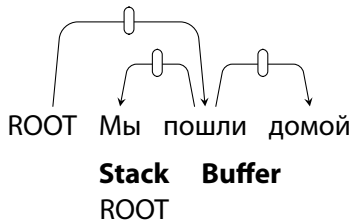
SHIFT



## RIGHT-ARC



## RIGHT-ARC



A **configuration** is a snapshot of the state of the parser at a given time.

- A stack: Representing the word(s) currently being processed
- A buffer: Representing the remaining words
- A set of arcs representing a (partial) tree

We can conceive parsing as transitioning from one configuration to another via an operation.

How do we get the sequence of operations?

**Deterministic algorithm:**

- LEFT-ARC: Configuration has arc from the top of stack to the word below
- RIGHT-ARC: Configuration has arc from the of the stack to the first word in the input buffer
  - In addition: The dependent must have no dependents of its own
- SHIFT: All other cases

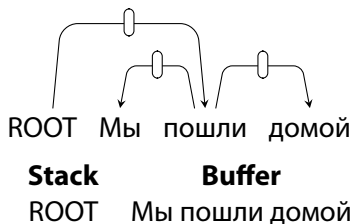


How do we get the sequence of operations?

### Deterministic algorithm:

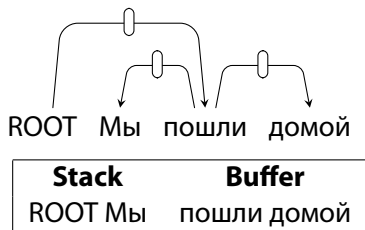
- LEFT-ARC: Configuration has arc from the top of stack to the word below
- RIGHT-ARC: Configuration has arc from the of the stack to the **first word in the input buffer**
  - In addition: The **dependent** must have no dependents of its own
- SHIFT: All other cases



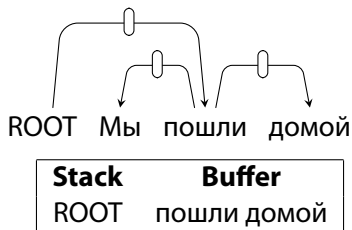


- Is there an arc from the first word in the buffer to the top of the stack?
  - (Мы, ROOT)
- Is there an arc from the top of the stack to the first word in the buffer?
  - (ROOT, Мы)
- → then SHIFT

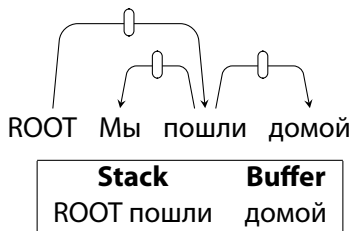




- Is there an arc from the first word in the buffer to the top of the stack?
  - (пошли, Мы) — YES, LEFT-ARC
- Is there an arc from the top of the stack to the first word in the buffer?
  - (Мы, пошли)



- Is there an arc from the first word in the buffer to the top of the stack?
  - (пошли, ROOT)
- Is there an arc from the top of the stack to the first word in the buffer?
  - (ROOT, пошли) – YES, but *пошли* still has dependents
- → then SHIFT

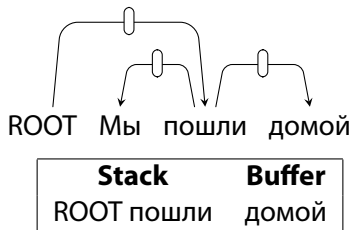


- Is there an arc from the first word in the buffer to the top of the stack?
  - (домой, пошли)
- Is there an arc from the top of the stack to the first word in the buffer?
  - (пошли, домой) – YES, and *домой* has no dependents
  - → RIGHT-ARC

The “only” training data required is a treebank.

- Collection of sentences annotated for dependency structure
- Universal dependencies: 67 languages, 100s of treebanks

Data trains a classifier to predict a transition from a configuration.



Features indexed by address (in stack or buffer) and attribute name.

## Traditional:

- (Stack[0], Form) = пошли
- (Buffer[0], Form) = домой
- (Stack[0], UPOS) = VERB
- (Stack[1], Form) = ROOT

## Indicator:

- Combinations of such features, e.g.
  - (Stack[0], Form) = пошли & (Buffer[0], UPOS) = ADP

Can return NULL.



# Graph-based

## Simple evaluation:

- Unlabelled attachment score, **UAS**: correct heads/total heads
- Labelled attachment score, **LAS**: (correct heads+labels)/total heads