

LING/C SC 581:

Advanced Computational Linguistics

Lecture 14

Today's Topics

Crash Blossoms

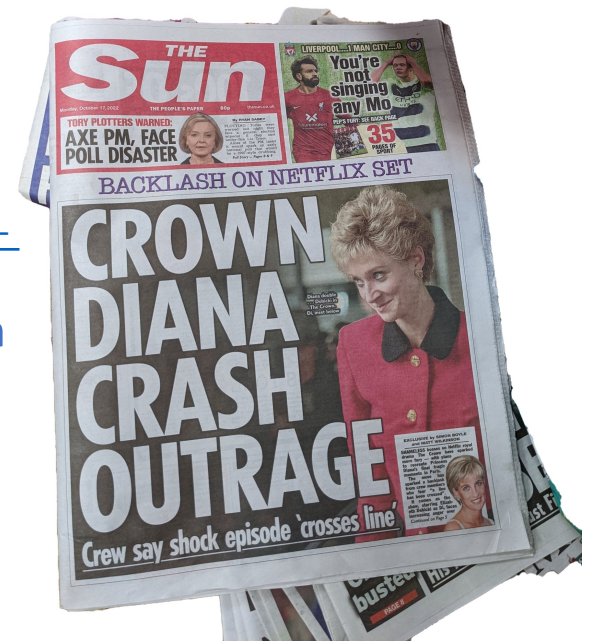
- Other Popular Parsers:
 - CoreNLP, Stanza
- Homework 7:
 - install Stanza into nltk
 - find Crash Blossoms
 - run them on parsers

Crash Blossoms

- **Headlines:**

- headlines are compressed (*due to a lack of space*)
- Leaving out words, e.g. copulas etc., may lead to unfortunate structural ambiguities
- <https://www.nytimes.com/2010/01/31/magazine/31FOB-onlanguage-t.html>
- Legendary headlines [...] include “Giant Waves Down Queen Mary’s Funnel,” “MacArthur Flies Back to Front” and “Eighth Army Push Bottles Up Germans.”
- The *Columbia Journalism Review* published two anthologies with the classic titles “Squad Helps Dog Bite Victim” and “Red Tape Holds Up New Bridge.”

a tabloid



Are these Crash Blossoms?

 **Rich Neville**
@RichNeville Follow


I don't understand how a snake even begins to organise a trip like that.




Entertainment

Lindsay Lohan bitten by snake on holiday in Thailand

< **The Independent's post** ... Q

 **The Independent** ✓
4 h · 🌐

Complete list ↓



independent.co.uk
All the easyJet flights cancelled today from UK airports

 **(((Sigourney Beaver)))**
@sigourneybeaver

What a terrible gift.



John Cena surprises 7-year-old boy with cancer on his birthday
winnipegssun.com

- Perhaps not. Funny, yes, but are they the most likely parse/reading?

Crash Blossoms

Last August, however, one emerged in the Testy Copy Editors online discussion forum. Mike O’Connell, an American editor based in Sapporo, Japan, spotted the headline “Violinist Linked to JAL Crash Blossoms” and wondered, “What’s a crash blossom?” (The article, from the newspaper Japan Today, described the successful musical career of Diana Yukawa, whose father died in a 1985 Japan Airlines plane crash.)

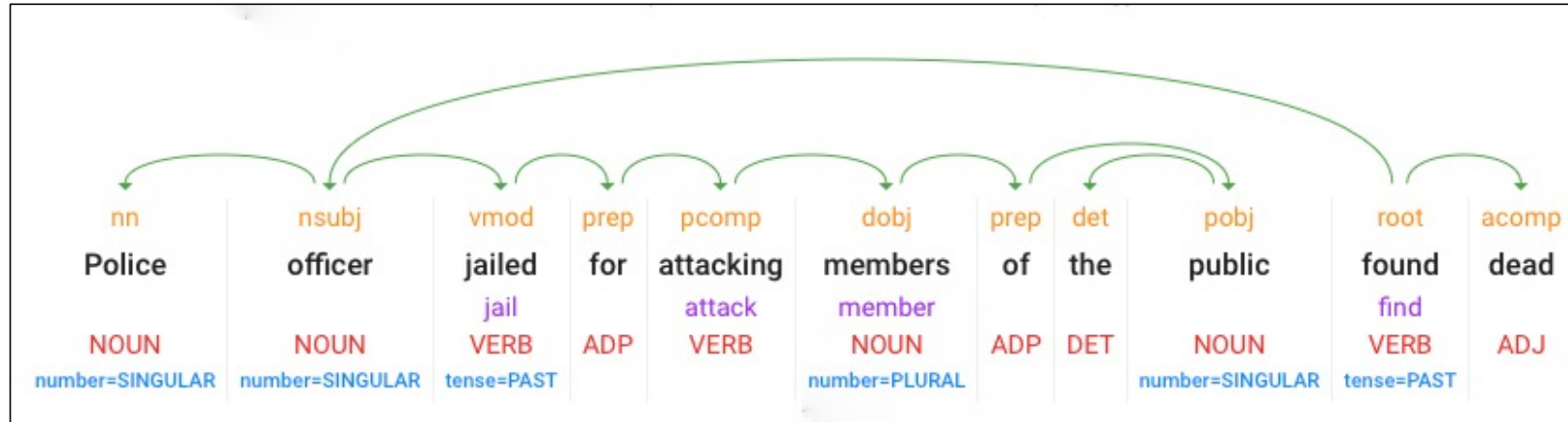
Crash Blossom Archive

- <https://languagelog.ldc.upenn.edu/nll/?cat=118>

Police officer jailed for attacking members of the public found dead

- *vmod*: reduced non-finite verbal modifier
 - *jailed* modifies *police officer*
- *pcomp*: prepositional complement,
 - *for* clausal complement
- *acomp*: adjectival complement

<https://www.theguardian.com/uk-news/2021/dec/29/pc-jailed-for-attacking-members-of-the-public-found-dead-at-home>



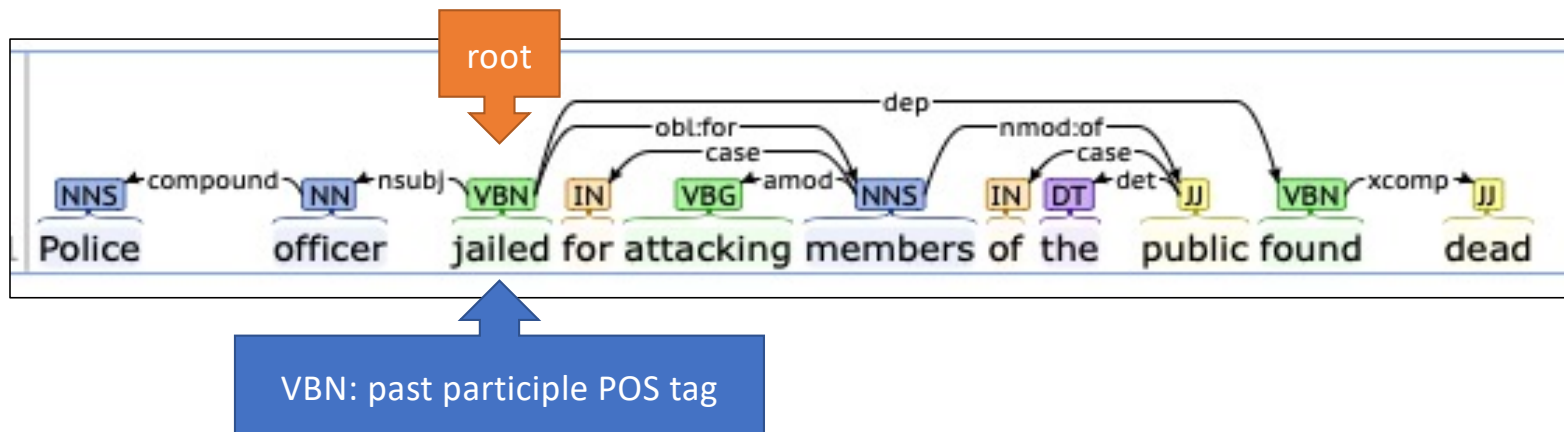
Crash Blossom Archive

- <https://languagelog.idc.upenn.edu/nll/?cat=118>

Police officer jailed for attacking members of the public found dead

Stanford CoreNLP: <https://corenlp.run>

CoreNLP is your one stop shop for natural language processing in Java!
CoreNLP currently supports 8 languages: Arabic, Chinese, English, French, German, Hungarian, Italian, and Spanish.



Crash Blossom Archive

- <https://languagelog.idc.upenn.edu/nll/?cat=118>

Police officer jailed for attacking members of the public found dead

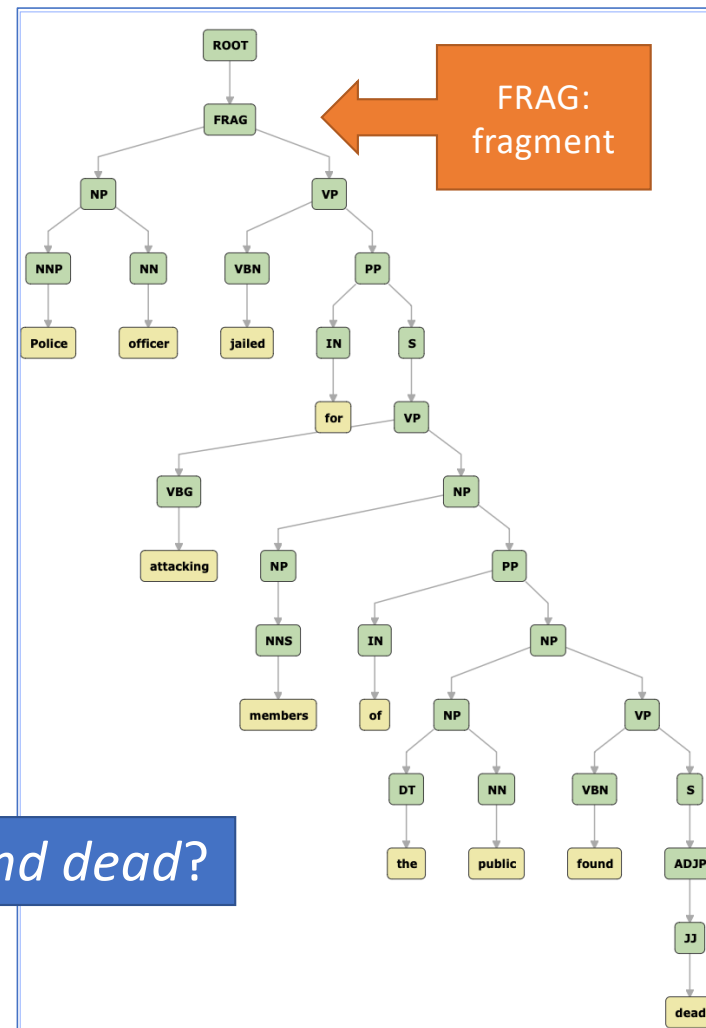
Stanford CoreNLP: <https://corenlp.run>

Police officer jailed for attacking members of the public found dead

— Annotations —

constituency parse ✕

Who is found dead?



Crash Blossom Archive

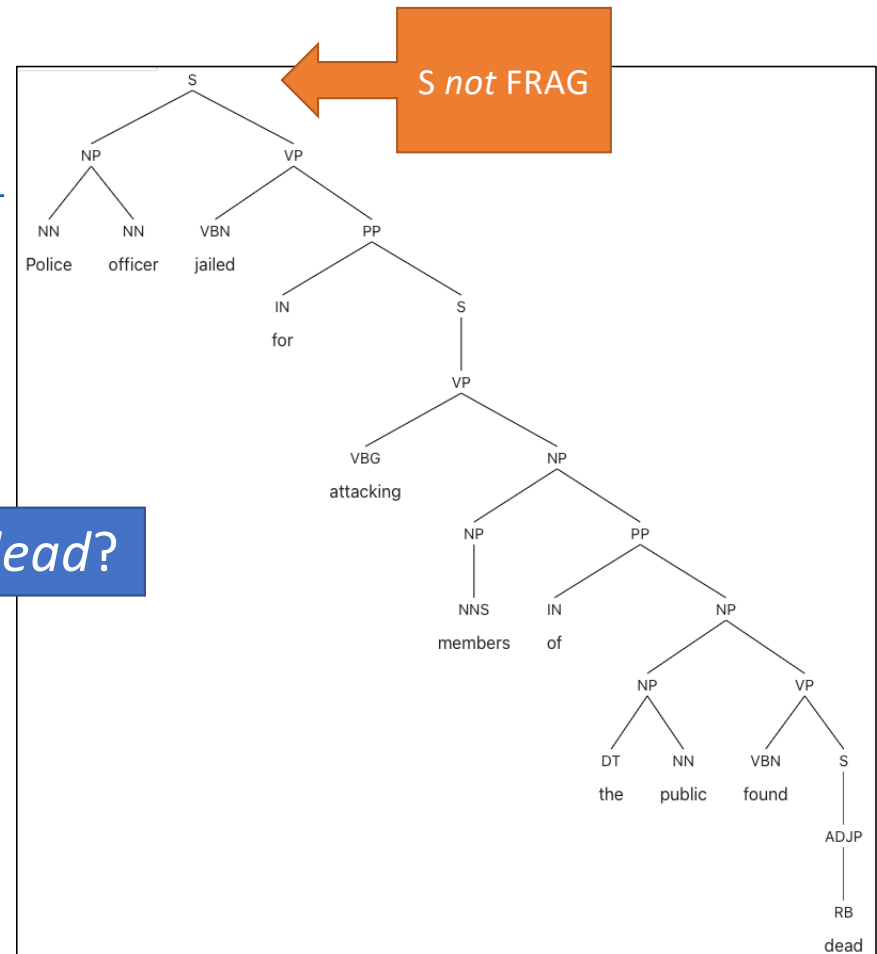
- <https://languagelog.ldc.upenn.edu/nll/?cat=118>

Police officer jailed for attacking members of the public found dead

Berkeley Neural Parser: <https://parser.kitaev.io>

see the VP attachment

Who is found dead?



Stanza

- Stanza – A Python NLP Package for Many Human Languages

```
conda install -c conda-forge stanza
```

Or

```
$ pip install stanza
```

```
Collecting stanza
```

```
  Downloading stanza-1.5.0-py3-none-any.whl (802 kB)
```

```
    |████████████████████████████████████████| 802 kB 2.6 MB/s
```

```
...
```

```
Successfully built emoji
```

```
Installing collected packages: torch, protobuf, emoji, stanza
```

```
Successfully installed emoji-2.2.0 protobuf-4.22.1 stanza-1.5.0 torch-2.0.0
```

- Run:

```
$ python
```

```
Python 3.9.12 (main, Jun 1 2022, 06:34:44)
```

```
>>> import stanza
```

```
>>> stanza.download('en')
```

<http://stanza.run>

demo

502 Bad Gateway

nginx/1.10.3 (Ubuntu)

https://github.com/stanfordnlp/stanza/blob/main/demo/Stanza_Beginners_Guide.ipynb

Stanza

- Initialize English neural pipeline:

```
>>> nlp = stanza.Pipeline('en')
```

```
'''
```

```
2023-03-29 17:50:15 INFO: Loading these models for language: en (English):
```

```
=====
```

```
| Processor      | Package      |
```

```
-----
```

```
| tokenize      | combined     |
```

```
| pos           | combined     |
```

```
| lemma        | combined     |
```

```
| constituency  | wsj          |
```

```
| depparse     | combined     |
```

```
| sentiment    | sstplus     |
```

```
| ner          | ontonotes   |
```

```
=====
```

```
2023-03-29 17:50:15 INFO: Using device: cpu
2023-03-29 17:50:15 INFO: Loading: tokenize
2023-03-29 17:50:15 INFO: Loading: pos
2023-03-29 17:50:15 INFO: Loading: lemma
2023-03-29 17:50:15 INFO: Loading:
constituency
2023-03-29 17:50:15 INFO: Loading: depparse
2023-03-29 17:50:15 INFO: Loading: sentiment
2023-03-29 17:50:16 INFO: Loading: ner
2023-03-29 17:50:16 INFO: Done loading
processors!
```

Crash Blossom Archive

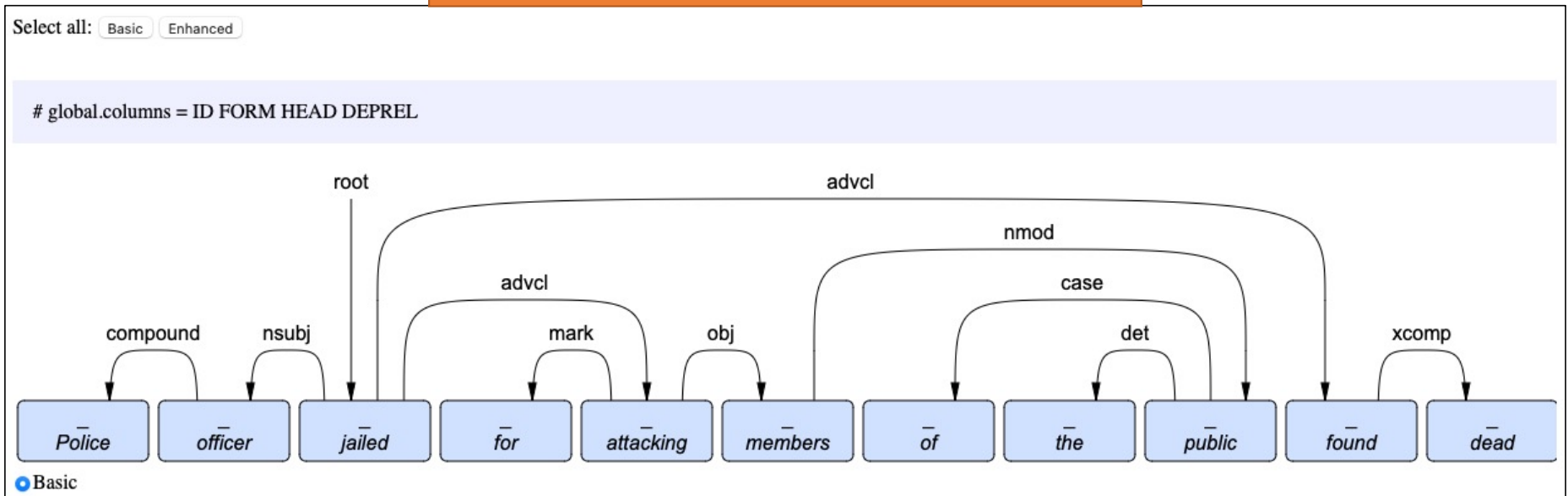
```
>>> doc = nlp("Police officer jailed for attacking members of the public found dead")
>>> s = ''
>>> words = doc.sentences[0].words
>>> for i,w in enumerate(words):
...     s += '{:<3d}\t{:12s}\t{:6s}\t{:<3d}\t{:15s}\n'.format(i+1,w.text,w.pos,w.head,w.deprel)
...
>>> print(s)
1 Police      NOUN  2  compound
2 officer    NOUN  3  nsubj
3 jailed     VERB  0  root
4 for        SCONJ 5  mark
5 attacking  VERB  3  advcl
6 members    NOUN  5  obj
7 of         ADP   9  case
8 the        DET   9  det
9 public     NOUN  6  nmod
10 found     VERB  3  advcl
11 dead      ADJ   10 xcomp
```

Crash Blossom Archive

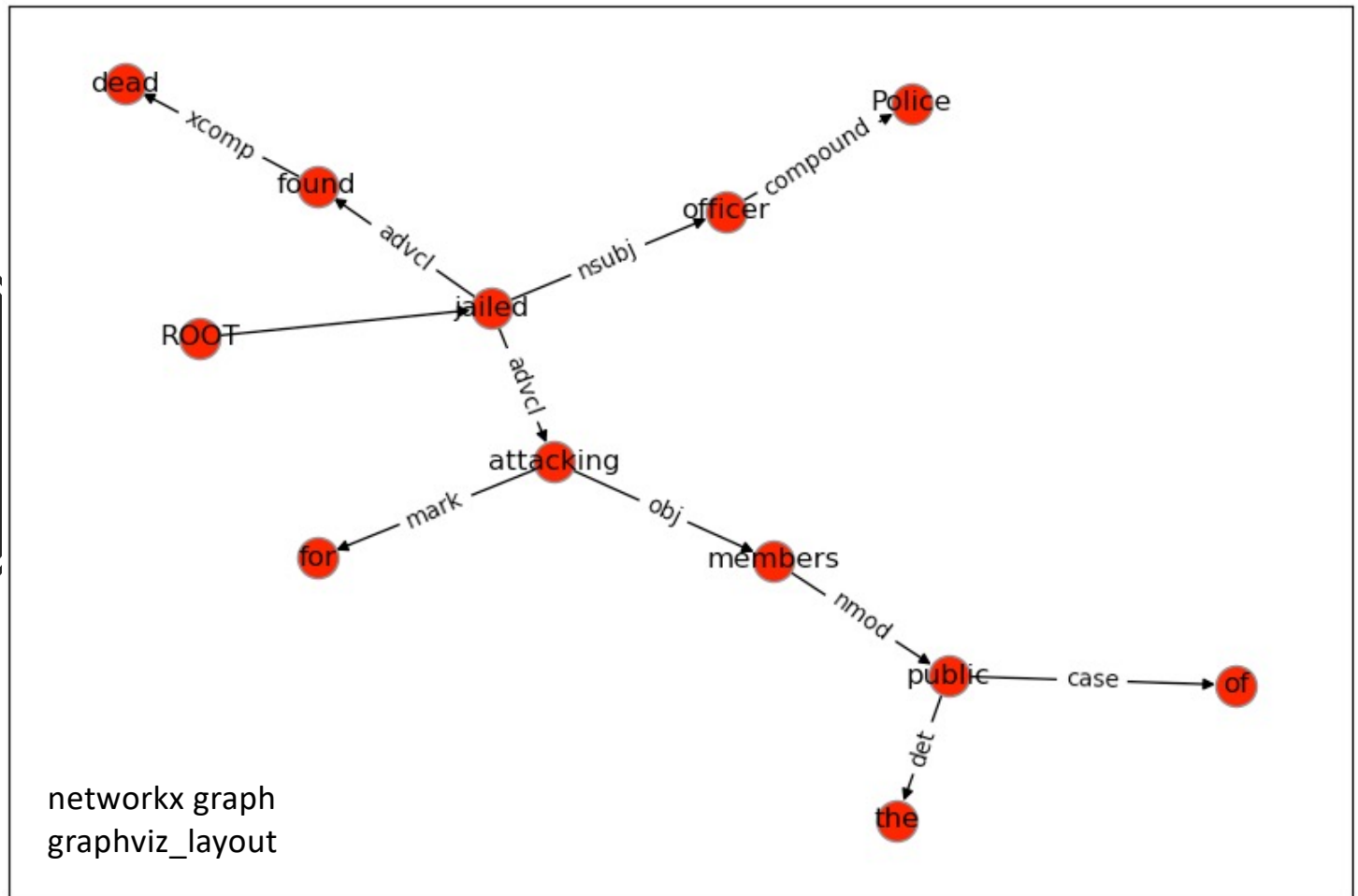
```
>>> print(doc)
[
  [
    {
      "id": 1,
      "text": "Police",
      "lemma": "police",
      "upos": "NOUN",
      "xpos": "NNS",
      "feats": "Number=Plur",
      "head": 2,
      "deprel": "compound",
      "start_char": 0,
      "end_char": 6,
      "ner": "0",
      "multi_ner": [
        "0"
      ]
    },
    {
      "id": 2,
      "text": "officer",
      "lemma": "officer",
      "upos": "NOUN",
      "xpos": "NN",
      "feats": "Number=Sing",
      "head": 3,
      "deprel": "nsubj",
      "start_char": 7,
      "end_char": 14,
      "ner": "0",
      "multi_ner": [
        "0"
      ]
    },
    {
      "id": 3,
      "text": "jailed",
      "lemma": "jail",
      "upos": "VERB",
      "xpos": "VBD",
      "feats": "Tense=Past|VerbForm=Part",
      "head": 0,
      "deprel": "root",
      "start_char": 15,
      "end_char": 21,
      "ner": "0",
      "multi_ner": [
        "0"
      ]
    },
    {
      "id": 4,
      "text": "for",
      "lemma": "for",
      "upos": "SCONJ",
      "xpos": "IN",
      "head": 5,
      "deprel": "mark",
      "start_char": 22,
      "end_char": 25,
      "ner": "0",
      "multi_ner": [
        "0"
      ]
    },
    {
      "id": 5,
      "text": "attacking",
      "lemma": "attack",
      "upos": "VERB",
      "xpos": "VBG",
      "feats": "VerbForm=Ger",
      "head": 3,
      "deprel": "advcl",
      "start_char": 26,
      "end_char": 35,
      "ner": "0",
      "multi_ner": [
        "0"
      ]
    },
    {
      "id": 6,
      "text": "members",
      "lemma": "member",
      "upos": "NOUN",
      "xpos": "NNS",
      "feats": "Number=Plur",
      "head": 5,
      "deprel": "obj",
      "start_char": 36,
      "end_char": 43,
      "ner": "0",
      "multi_ner": [
        "0"
      ]
    },
    {
      "id": 7,
      "text": "of",
      "lemma": "of",
      "upos": "ADP",
      "xpos": "IN",
      "head": 9,
      "deprel": "case",
      "start_char": 44,
      "end_char": 46,
      "ner": "0",
      "multi_ner": [
        "0"
      ]
    },
    {
      "id": 8,
      "text": "the",
      "lemma": "the",
      "upos": "DET",
      "xpos": "DT",
      "feats": "Definite=Def|PronType=Art",
      "head": 9,
      "deprel": "det",
      "start_char": 47,
      "end_char": 50,
      "ner": "0",
      "multi_ner": [
        "0"
      ]
    },
    {
      "id": 9,
      "text": "public",
      "lemma": "public",
      "upos": "NOUN",
      "xpos": "NN",
      "feats": "Number=Sing",
      "head": 6,
      "deprel": "nmod",
      "start_char": 51,
      "end_char": 57,
      "ner": "0",
      "multi_ner": [
        "0"
      ]
    },
    {
      "id": 10,
      "text": "found",
      "lemma": "find",
      "upos": "VERB",
      "xpos": "VBD",
      "feats": "Mood=Ind|Person=3|Tense=Past|VerbForm=Fin",
      "head": 3,
      "deprel": "advcl",
      "start_char": 58,
      "end_char": 63,
      "ner": "0",
      "multi_ner": [
        "0"
      ]
    },
    {
      "id": 11,
      "text": "dead",
      "lemma": "dead",
      "upos": "ADJ",
      "xpos": "JJ",
      "feats": "Degree=Pos",
      "head": 10,
      "deprel": "xcomp",
      "start_char": 64,
      "end_char": 68,
      "ner": "0",
      "multi_ner": [
        "0"
      ]
    }
  ]
]
```

Graphing: CoNLL-U Plus Format

How to print this? See next slides



Graphing: rolling your own



Graphing: CoNLL-U Plus Format

```
>>> s = ''
>>> for i,w in enumerate(words):
...     s += '{:<3d}\t{:12s}\t{:<3d}\t{:15s}\n'.format(i+1,w.text,w.head,w.deprel)
...
>>> print(s)
1  Police          2  compound
2  officer         3  nsubj
3  jailed          0  root
4  for             5  mark
5  attacking       3  advcl
6  members         5  obj
7  of              9  case
8  the             9  det
9  public          6  nmod
10 found           3  advcl
11 dead           10 xcomp
```


Graphing: CoNLL-U Plus Format

<https://urd2.let.rug.nl/~kleiweg/conllu/>

Upload a file with one or more sentences annotated in [CoNLL-U](#) format:

Choose File no file selected

Submit

Here is an [example](#)

-- *OR* --

Enter something in CoNLL-U format here:

```
# global.columns = ID FORM HEAD DEP REL
1 Police 2 compound
2 officer 3 nsubj
3 jailed 0 root
4 for 5 mark
5 attacking 3 advcl
6 members 5 obj
7 of 9 case
8 the 9 det
9 public 6 nmod
10 found 3 advcl
11 dead 10 xcomp
```

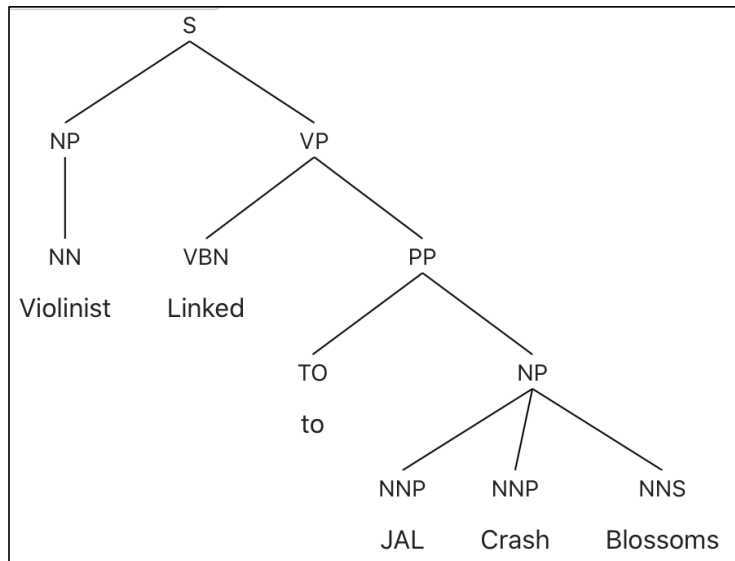
global.columns = ID FORM HEAD DEP REL

Graphing: CoNLL-U Plus Format

- <https://universaldependencies.org/ext-format.html>
- While a CoNLL-U file has always precisely ten columns separated by TAB characters, a CoNLL-U Plus file can have any non-zero number of columns.
- In addition, the first line must be a sentence-level comment (i.e., starting with a # character) that lists the names of the columns used in this file.
- Example:

```
# global.columns = ID FORM LEMMA UPOS XPOS FEATS HEAD DEPREL DEPS MISC
```

Violinist Linked to JAL Crash Blossoms



nsubj	root	prep	nn	nn	pobj
Violinist	Linked	to	JAL	Crash	Blossoms
	link				blossom
NOUN	VERB	ADP	NOUN	NOUN	NOUN
number=SINGULAR	tense=PAST		number=SINGULAR	number=SINGULAR	number=SINGULAR
			proper=PROPER	proper=PROPER	

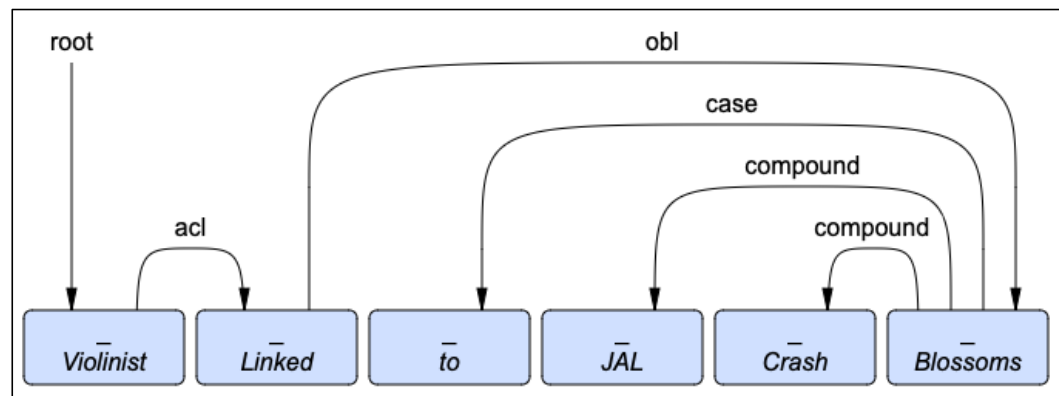
pobj : nmod

The **pobj** relation in SD has been renamed to **nmod** in UD.

Note that prepositional phrases are treated differently in UD as [Prepositional Phrases](#) for more details on the new treatment.

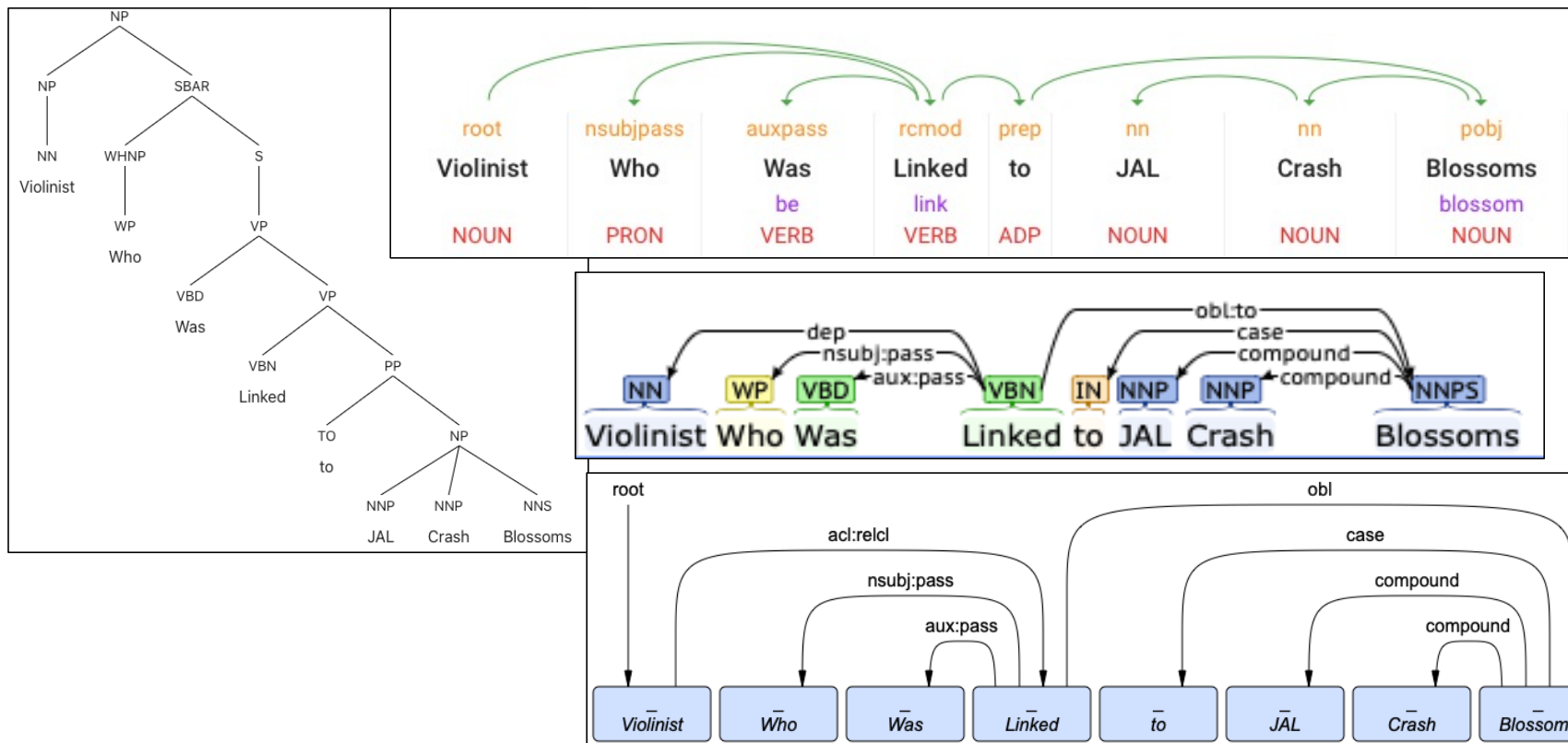
Violinist Linked to JAL Crash Blossoms

```
>>> doc = nlp("Violinist Linked to JAL Crash Blossoms")
>>> words = doc.sentences[0].words
>>> s = ''
>>> for i,w in enumerate(words):
    s += '{:<3d}\t{:<12s}\t{:<3d}\t{:<15s}\n'.format(i+1,w.text,w.head,w.deprel)
...
>>> print(s)
1 Violinist 0 root
2 Linked 1 acl
3 to 6 case
4 JAL 6 compound
5 Crash 6 compound
6 Blossoms 2 obl
```

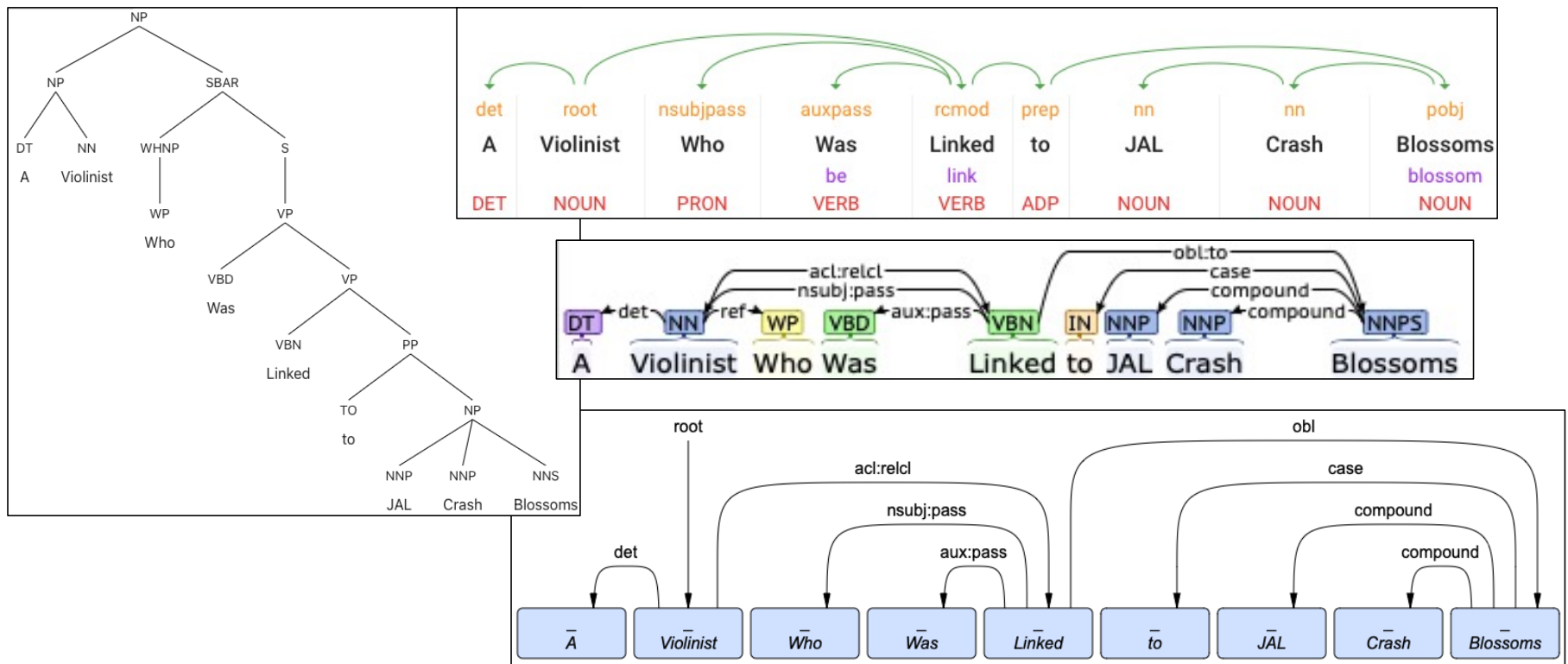


acl: clausal modifier of noun

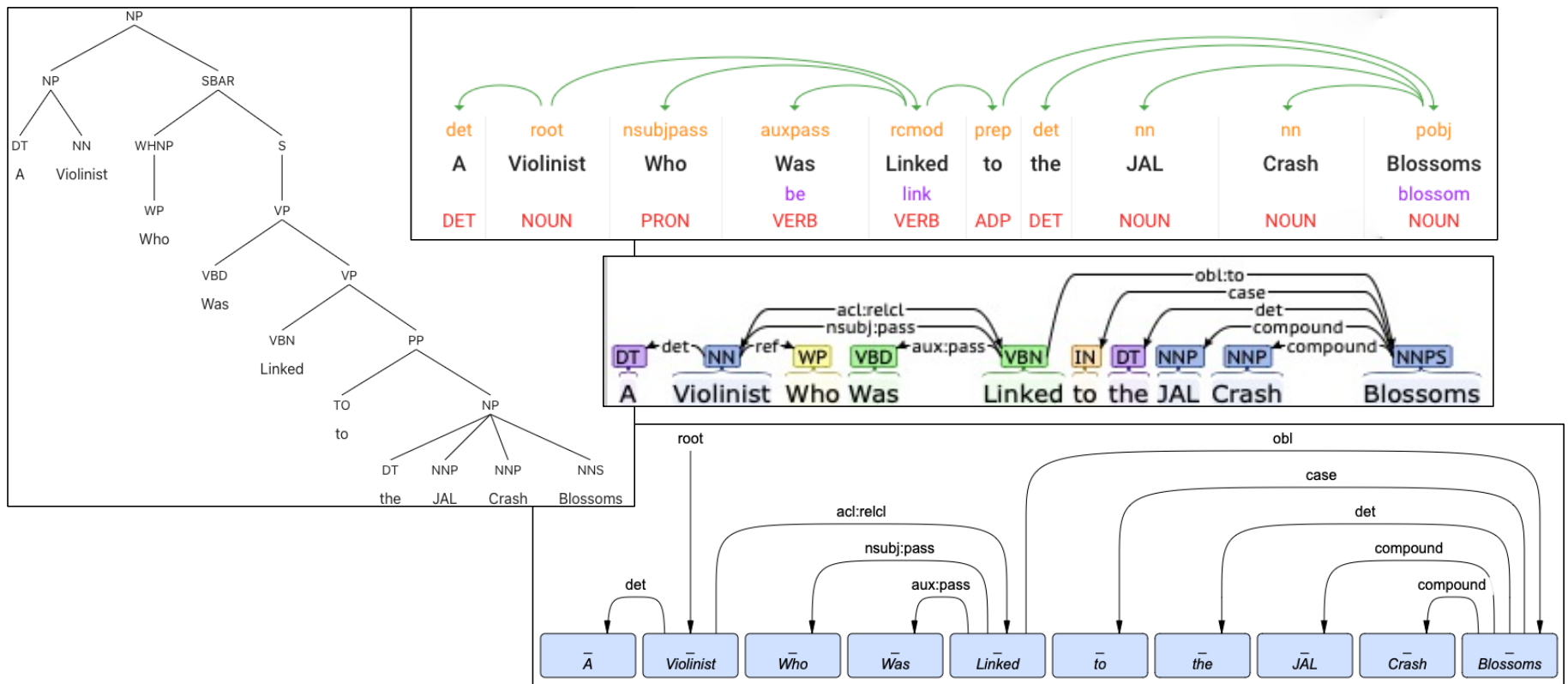
Violinist *Who Was Linked to JAL Crash Blossoms*



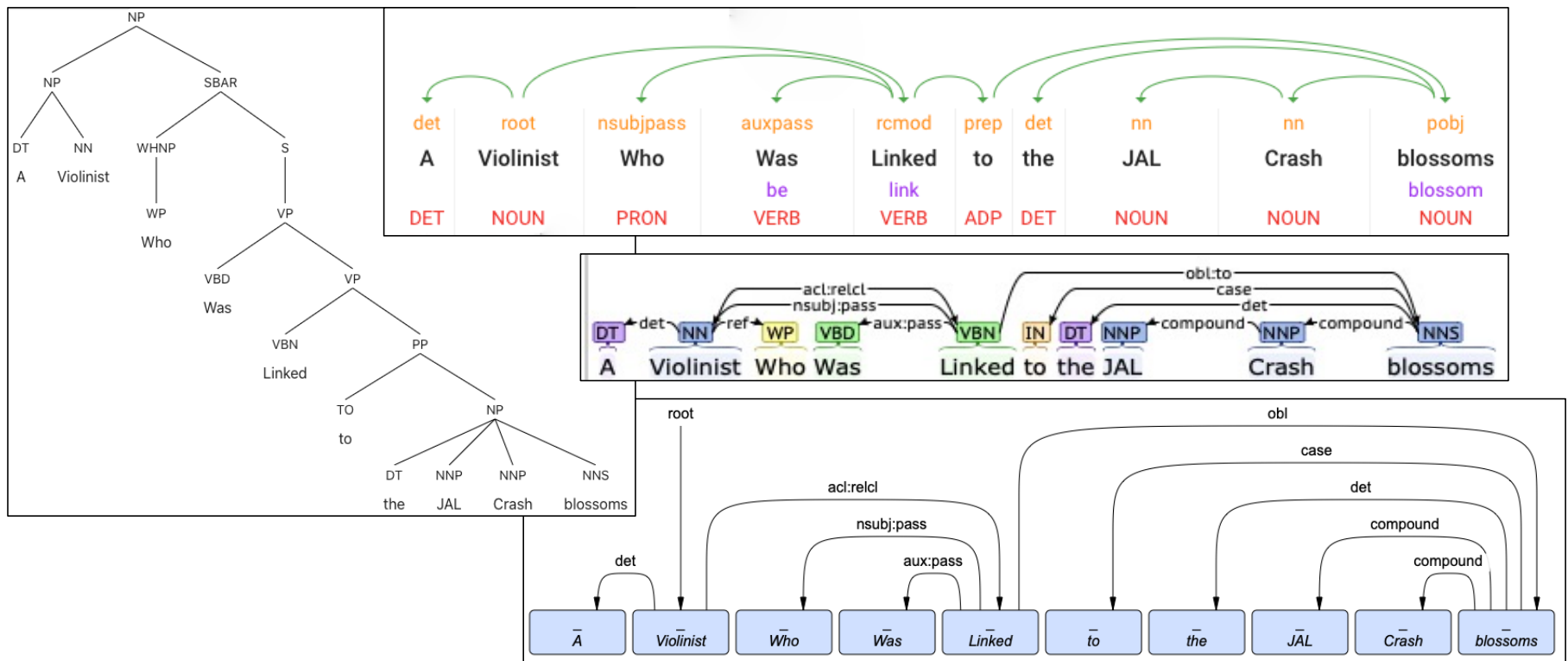
A Violinist Who Was Linked to JAL Crash Blossoms



A Violinist Who Was Linked to the JAL Crash Blossoms



A Violinist Who Was Linked to the JAL Crash blossoms



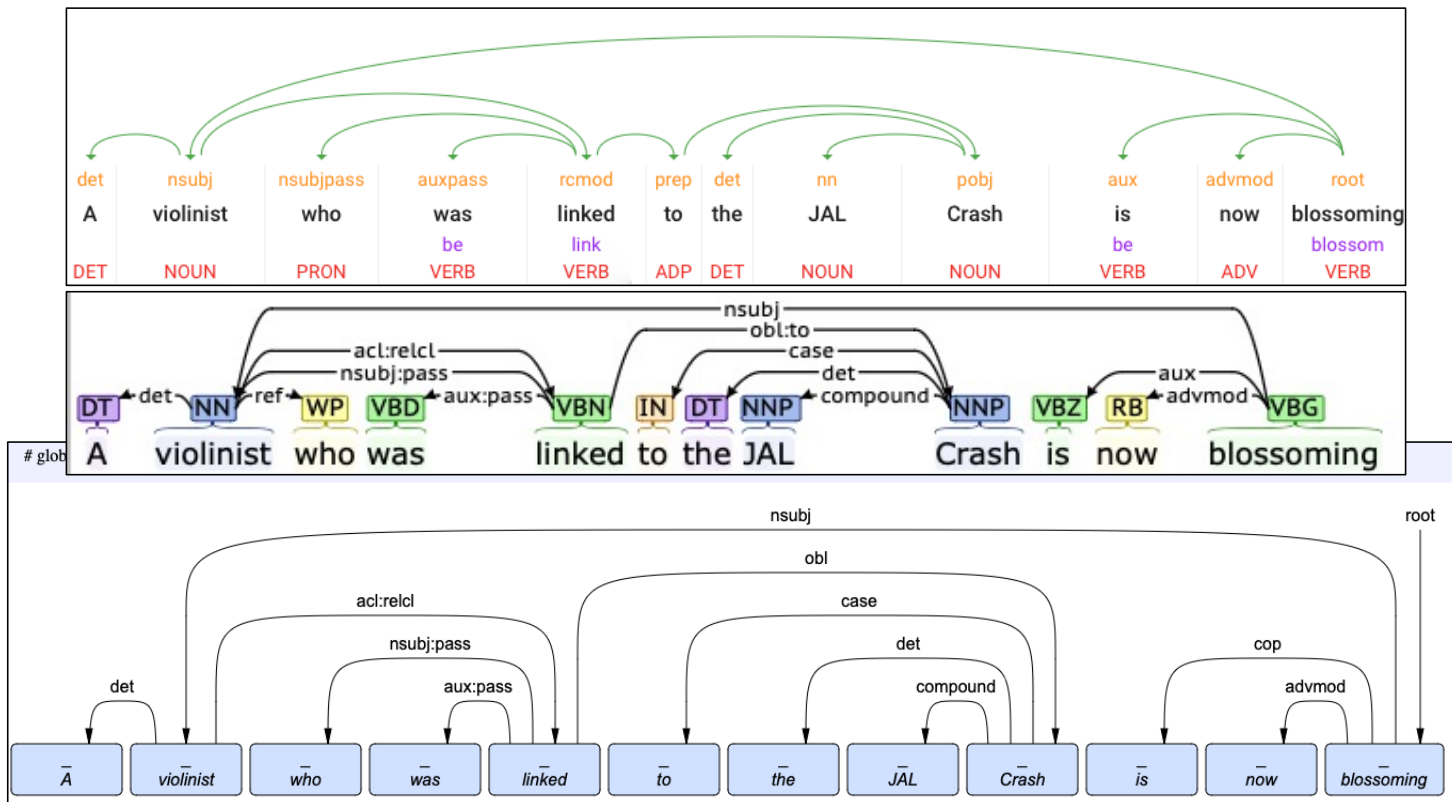
Crash Blossoms

- *Violinist Linked to JAL Crash Blossoms*
- *Violinist **Who Was** Linked to JAL Crash Blossoms*
- *A Violinist **Who Was** Linked to JAL Crash Blossoms*
- *A Violinist **Who Was** Linked to **the** JAL Crash Blossoms*
- *A Violinist **Who Was** Linked to **the** JAL Crash **blossoms***
- *A **violinist who was linked** to **the** JAL Crash **blossoms***
- *A **violinist who was linked** to **the** JAL Crash **is now blossoming***

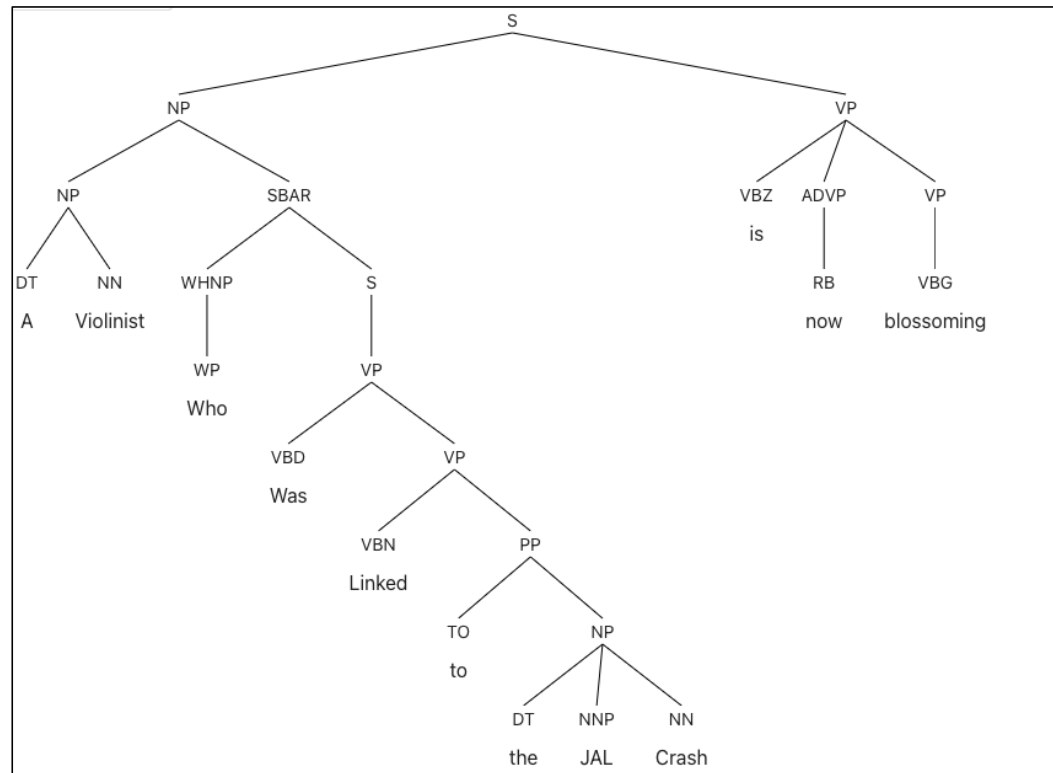


This one!

Crash Blossoms



Crash Blossoms



Homework 7

- Step 1: Read the New York Times article on *Crash Blossoms* (CB)
- Step 2: Install Stanza into nltk on your machine
- Step 3: Find two headlines that you consider to be CBs
 - either from the internet
 - or look at the Crash Blossom Archive
 - <https://languagelog.idc.upenn.edu/nll/?cat=118>
 - Explain why they are a CB,
 - e.g. *what is the misread, why they may be misinterpreted.*

Homework 7

- Step 4: Now run the headlines you chose on **all** of the following parsers:
 1. CoreNLP <https://corenlp.run>
 2. Stanza, on your computer, and
 3. Berkeley Neural Parser <https://parser.kitaev.io>
- Explain each parse, i.e. whether it got the intended reading, your CB reading or some other reading.

Homework 7

- Submit to sandiway@arizona.edu
- **SUBJECT**: 581 Homework 7 **YOUR NAME**
- One PDF file (for grading)
 - include your screenshots in your answer
 - put a web link to each headline (*if you found it online*)
- Extended Deadline:
 - **Spring Break**
 - midnight Monday March 11th
 - we will review the homework on Tuesday March 12th