Funded by
the European Union

11. Context-Free Grammars

This series of tutorials is based upon work from COST Action
Multi3Generation CA18231, supported by COST
(European Cooperation in Science and Technology).

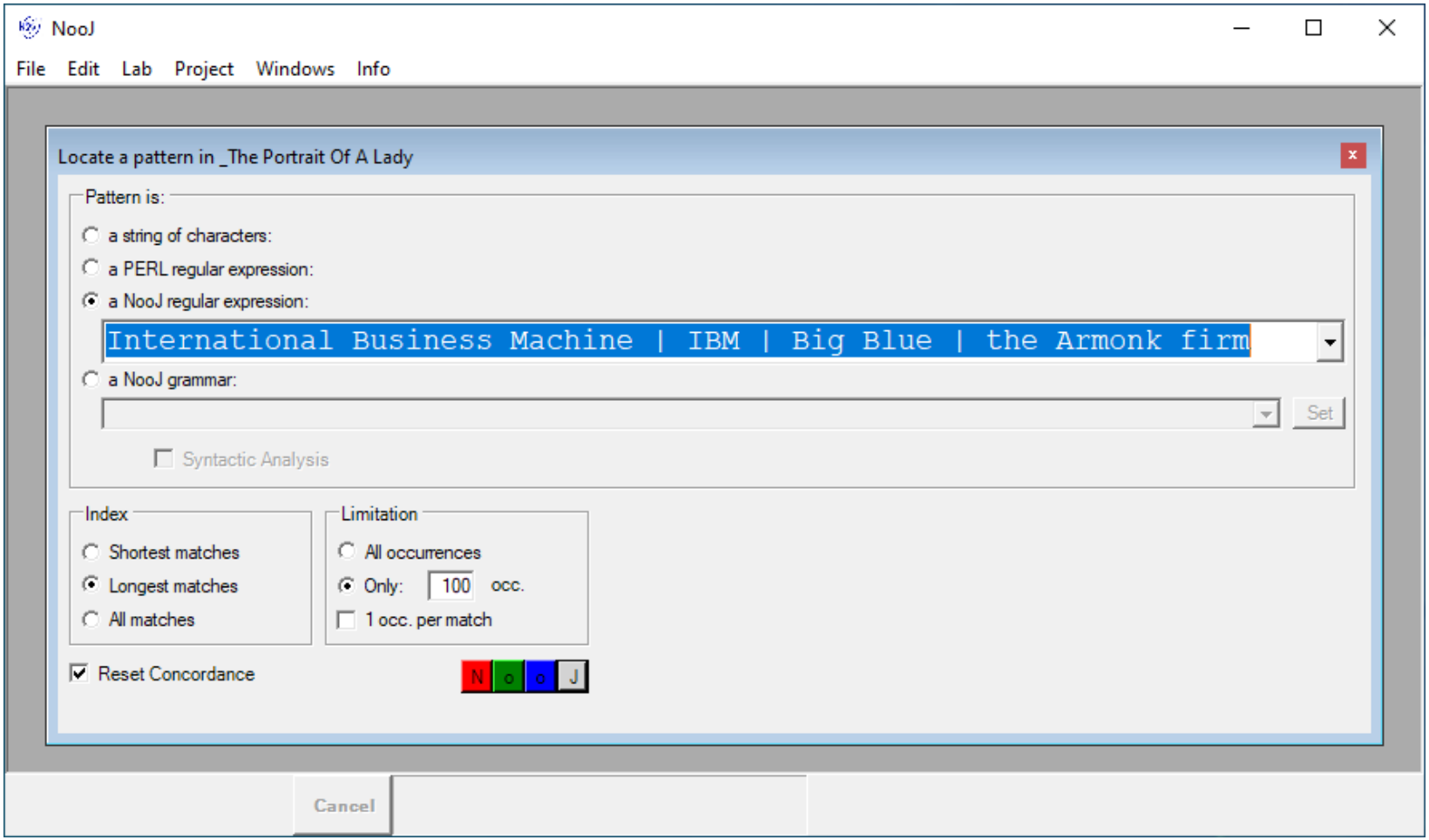
COST (European Cooperation in Science and Technology) is a funding agency for research and innovation networks. Our Actions help connect research initiatives across Europe and enable scientists to grow their ideas by sharing them with their peers. This boosts their research, career and innovation, cf. www.cost.eu

Context-Free Grammars

- Context-free grammars are equivalent to Type 2 Generative Grammars in Chomsky-Schützenberger's hierarchy
- Context-free grammars can be more powerful than regular grammars.
- They can recognize any context-free language, are very well-adapted to describe structured and embedded linguistic units, and can be processed by push-down automata.

From regular expressions to context-free grammars

- TEXT > Locate
- Queries are regular expressions



From regular expressions to context-free grammars

- Queries can be named

IBM = International Business Machine |
IBM | Big Blue | the Armonk firm

From regular expressions to context-free grammars

- Queries can be named

<buy> = buy | buys | buying | bought | buyer | buyers

From regular expressions to context-free grammars

- Queries can be named and then re-used

<buy> = buy | buys | buying | bought | buyer | buyers

<trade> = <buy> | <sale> | <acquire> | <investment> | <purchase> | <deal>

From regular expressions to context-free grammars

- Queries can be named and then re-used

<buy> = buy | buys | buying | bought | buyer | buyers

<trade> = <buy> | <sale> | <acquire> | <investment> | <purchase> | <deal>

<op> = <company> <trade> <company> | <company> <hire> <person>

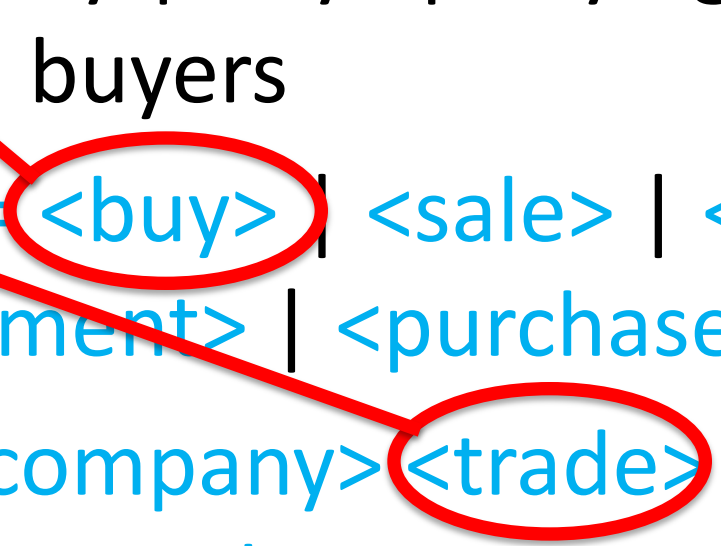
From regular expressions to context-free grammars

- Queries can be named and then re-used

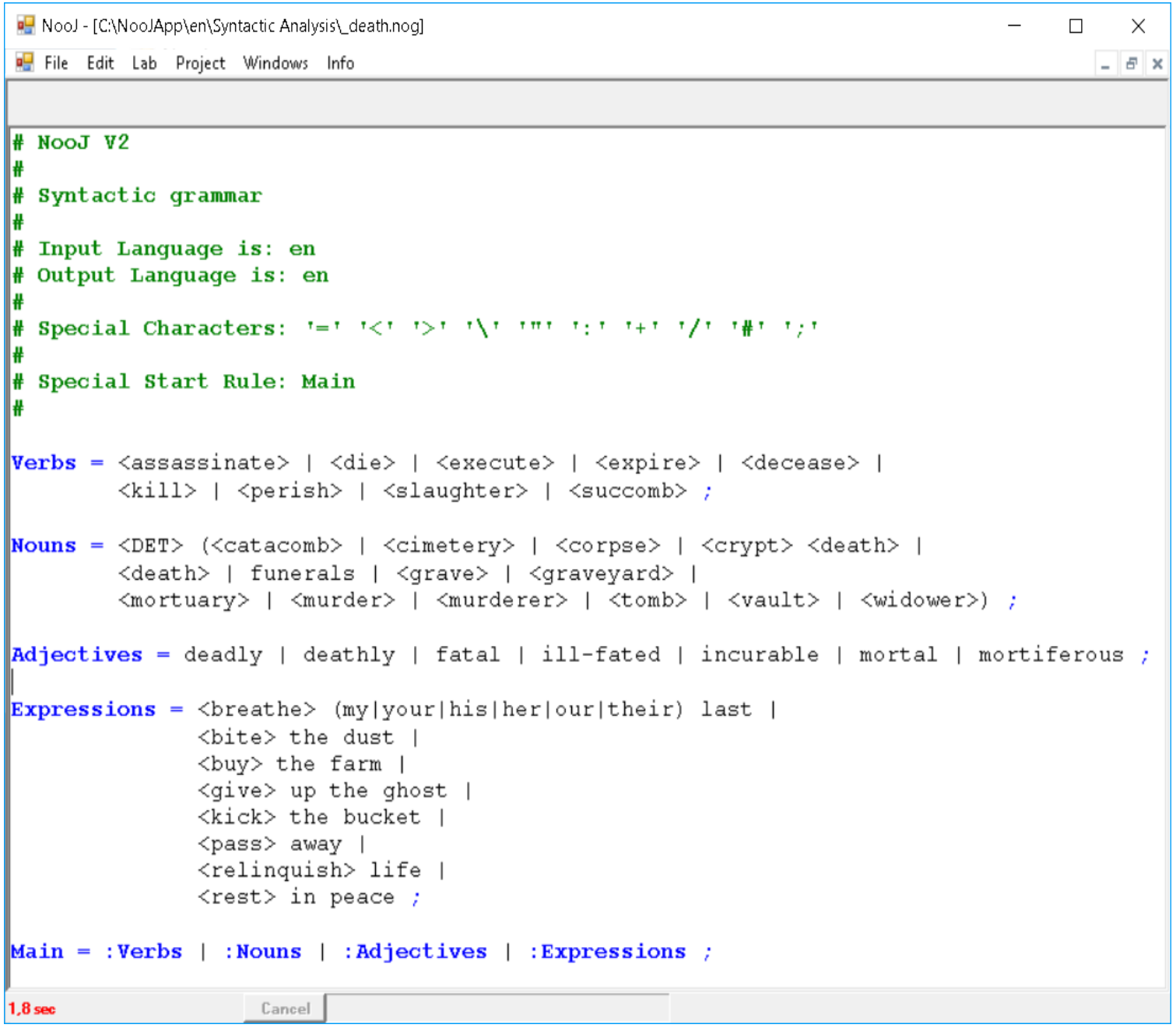
<buy> = buy | buys | buying | bought | buyer | buyers

<trade> = <buy> | <sale> | <acquire> | <investment> | <purchase> | <deal>

<op> = <company> <trade> <company> | <company> <hire> <person>



NLP Application: A thematic analysis



NLP Application: A thematic analysis

NooJ - [C:\NooJApp\en\Syntactic Analysis_death.nog]

File Edit Lab Project Windows Info

```
# NooJ V2
#
# Syntactic grammar
#
# Input Language is: en
# Output Language is: en
#
# Special Characters: '=' '<' '>' '\' '"' ':' '+' '/' '#' ';'
#
# Special Start Rule: Main
#

Verbs = <assassinate> | <die> | <execute> | <expire> | <decease> |
        <kill> | <perish> | <slaughter> | <succomb> ;

Nouns = <DET> (<catacomb> | <cimetery> | <corpse> | <crypt> <death> |
        <death> | funerals | <grave> | <graveyard> |
        <mortuary> | <murder> | <murderer> | <tomb> | <vault> | <widower>)

Adjectives = deadly | deathly | fatal | ill-fated | incurable | mortal | m

Expressions = <breathe> (my|your|his|her|our|their) last |
              <bite> the dust |
              <buy> the farm |
              <give> up the ghost |
              <kick> the bucket |
              <pass> away |
              <relinquish> life |
              <rest> in peace ;

Main = :Verbs | :Nouns | :Adjectives | :Expressions ;
```

1,8 sec Cancel

webnooj.univ-fcomte.fr

McGrawDictionary of Scientific and Technical Terms site:fnac.com - Google Search NooJ

NooJ

français A Corpus processor used in the Digital Humanities - A Linguistic Development Environment - A linguistic Engine for developing Natural Language Processing Software Applications.

Log In For any question or remark, contact max.silberztein@univ-fcomte.fr

The Portrait of a Lady (Henry James, 1881)

CHAPTER I

Under certain circumstances there are few hours in life more agreeable than the hour dedicated to the ceremony known as afternoon tea. There are circumstances in which, whether you partake of the tea or not--some

Select a query: Death

Enter a query:

Apply Query

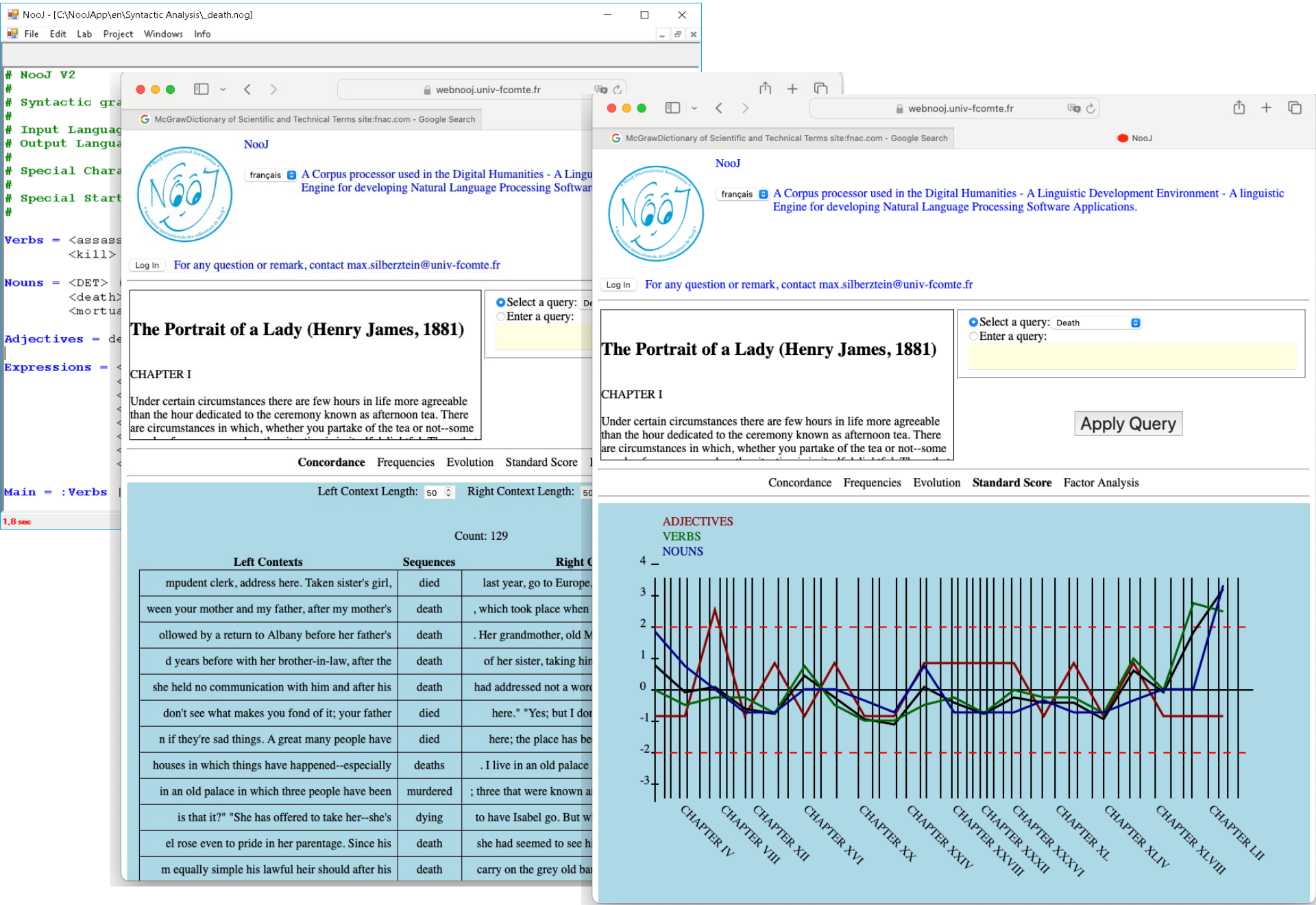
Concordance Frequencies Evolution Standard Score Factor Analysis

Left Context Length: 50 Right Context Length: 50 Set

Count: 129

Left Contexts	Sequences	Right Contexts	Outputs
mpudent clerk, address here. Taken sister's girl,	died	last year, go to Europe, two sisters, quite indep	<VERBS>
ween your mother and my father, after my mother's	death	, which took place when I was a child. In conseque	<NOUNS>
ollowed by a return to Albany before her father's	death	. Her grandmother, old Mrs. Archer, had exercised,	<NOUNS>
d years before with her brother-in-law, after the	death	of her sister, taking him to task for the manner	<NOUNS>
she held no communication with him and after his	death	had addressed not a word to his daughters, who ha	<NOUNS>
don't see what makes you fond of it; your father	died	here." "Yes; but I don't dislike it for that," t	<VERBS>
n if they're sad things. A great many people have	died	here; the place has been full of life." "Is that	<VERBS>
houses in which things have happened--especially	deaths	. I live in an old palace in which three people ha	<NOUNS>
in an old palace in which three people have been	murdered	; three that were known and I don't know how many	<NOUNS>
is that it?" "She has offered to take her--she's	dying	to have Isabel go. But what I want her to do when	<VERBS>
el rose even to pride in her parentage. Since his	death	she had seemed to see him as turning his braver s	<NOUNS>
m equally simple his lawful heir should after his	death	carry on the grey old bank in the white American	<NOUNS>

NLP Application: A thematic analysis



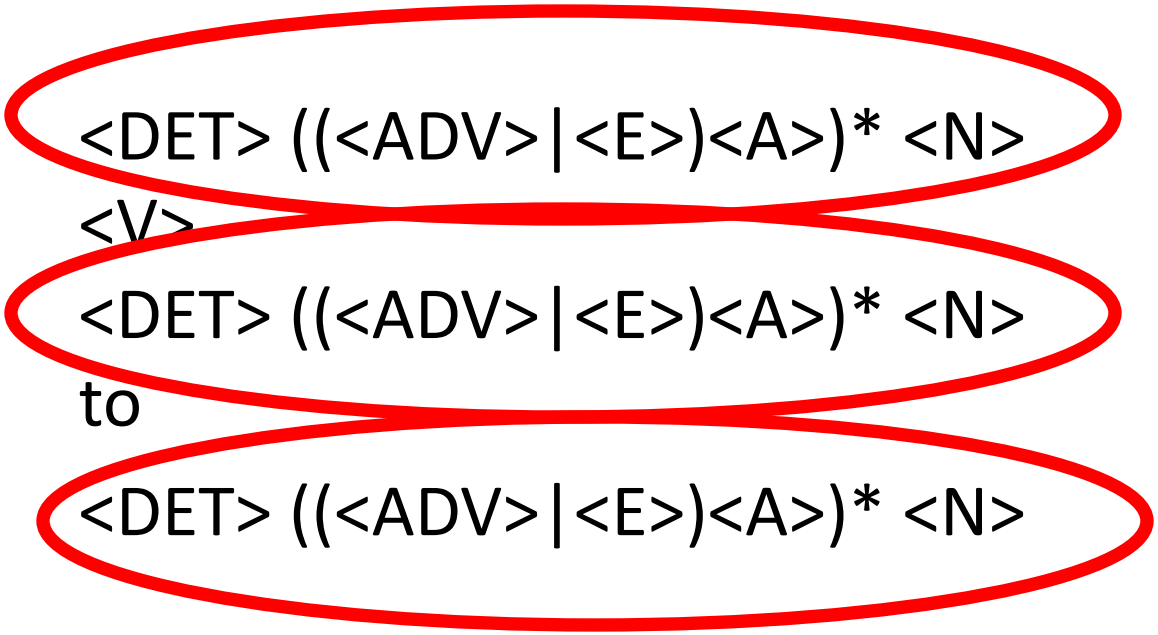
Use Context-Free Grammars to describe regular languages

- Describe all sentences that have two complements, e.g. *The very mean neighbor gave the red pen to my extremely rich cousin*

<DET> ((<ADV>|<E>)<A>)* <N>
<V>
<DET> ((<ADV>|<E>)<A>)* <N>
to
<DET> ((<ADV>|<E>)<A>)* <N>

Use Context-Free Grammars to describe regular languages

- Describe all sentences that have two complements, e.g. *The very mean neighbor gave the red pen to my extremely rich cousin*



Use Context-Free Grammars to describe regular languages

- Re-use rules in an “engineering” approach:

NP = <DET> ((<ADV>|<E>)<A>)* <N>

Main = NP <V> NP to NP

Use Context-Free Grammars to describe regular languages

- A context-free grammar that recognizes Roman Numerals:

Units = I | II | III | IV | V | VI | VII | VIII | IX

Tens = X | XX | XXX | XL | L | LX | LXX | LXXX | XC

RomanNumerals = Tens (Units | <E>) | Units

Use Context-Free Grammars to describe regular languages

- Exercise: Complement the grammar so that it recognizes Roman Numerals up to 3,999

  Units = I | II | III | IV | V | VI | VII | VIII | IX

  Tens = X | XX | XXX | XL | L | LX | LXX | LXXX | XC

  Hundreds = ...

  Thousands = ...

  RomanNumerals = ...

Use Context-Free Grammars to describe regular languages

- Exercise: Complement the grammar so that it recognizes Roman Numerals up to 3,999

Units = I | II | III | IV | V | VI | VII | VIII | IX

Tens = X | XX | XXX | XL | L | LX | LXX | LXXX | XC

Hundreds = C | CC | CCC | CD | D | DC | DCC | DCCC | CM

Thousands = M | MM | MMM

RomanNumerals = (<E>|Thousands) (<E>|Hundreds) (<E>|Tens) (<E>|Units)

First attempt...

Use Context-Free Grammars to describe regular languages

- Exercise: Complement the grammar so that it recognizes Roman Numerals up to 3,999

Units = I | II | III | IV | V | VI | VII | VIII | IX

Tens = X | XX | XXX | XL | L | LX | LXX | LXXX | XC

Hundreds = C | CC | CCC | CD | D | DC | DCC | DCCC | CM

Thousands = M | MM | MMM

RomanNumerals = (<E>|Thousands) (<E>|Hundreds) (<E>|Tens) (<E>|Units)

First attempt incorrect because the grammar recognizes the empty string

Use Context-Free Grammars to describe regular languages

- Exercise: Complement the grammar so that it recognizes Roman Numerals up to 3,999

Units = I | II | III | IV | V | VI | VII | VIII | IX

Tens = X | XX | XXX | XL | L | LX | LXX | LXXX | XC

Hundreds = C | CC | CCC | CD | D | DC | DCC | DCCC | CM

Thousands = M | MM | MMM

RomanNumerals =

(<E>|Thousands) (<E>|Hundreds) (<E>|Tens) Units |

(<E>|Thousands) (<E>|Hundreds) Tens |

(<E>|Thousands) Hundreds |

Thousands

Enhanced Context-Free Grammars

- Enhancing the grammar so that it produces the number in Arabic notation:

Units = I/1 | II/2| III/3 | IV/4 | V/5 | VI/6 | VII/7 | VIII/8 | IX/9

Tens = X/1 | XX/2 | XXX/3 | XL/4 | L/5 | LX/6 | LXX/7 | LXXX/8 | XC/9

Hundreds = C/1 | CC/2 | CCC/3 | CD/4 | D/5 | DC/6 | DCC/7 | DCCC/8 | CM/9

Thousands = M/1 | MM/2 | MMM/3

RomanNumerals =

Thousands (<E>/0|Hundreds) (<E>/0|Tens) (<E>/0|Units) |

Hundreds (<E>/0|Tens) (<E>/0|Units) |

Tens (<E>/0|Units) |

Units

Recursion

- In a Context-Free Grammar, the expression (to the right) can contain auxiliary symbols
- What happens if a rule is defined recursively from itself?

Sentence = **Subject Verb** (<E> | *that* **Sentence**)

Subject = *Jean* | *Mary* | *Eva* | *Joe*

Verb = *said* | *hoped* | *thought* | *slept*

Jean said that Mary hoped that Eva thought that Joe slept

Recursion

- In a Context-Free Grammar, the expression (to the right) can contain auxiliary symbols
- What happens if a rule is defined recursively from itself?

Sentence = **Subject Verb** (<E> | *that* **Sentence**)

Subject = *Jean* | *Mary* | *Eva* | *Joe*

Verb = *said* | *hoped* | *thought* | *slept*

Jean said that Mary hoped that Eva thought that Joe slept

Recursion

There are three types of recursion:

- Left Recursion

  NP = **NP** (*from* | *in* | *around*) *Paris* | *the house*

- Right Recursion

  Sentence = **NP** (*thought* | *said* | *hopes* | *saw*) **NP**

  NP = *Joe* | *Mary* | *that* **Sentence**

- General Recursion

  Sentence = **DET NOUN** (*that* | <E>) **Sentence VERB**

Recursion

Remove Left Recursion

- Left Recursion:

  $\mathbf{NP} = \mathbf{NP}\ (from \mid in \mid around)\ Paris \mid the\ house$

- Left recursions can be removed automatically:

  $\mathbf{NP} = the\ house\ (\ (from \mid in \mid around)\ Paris\ )^*$

Recursion

Remove Right Recursion

- Right Recursion:

  Sentence = **NP** (*thought* | *said* | *hopes* | *saw*) **NP**

  NP = *Joe* | *Mary* | *that* **Sentence**

- Right recursions can be removed automatically:

  Sentence = ((*Joe* | *Mary*) (*thought* | *said* | *hopes* | *saw*) that)*
  (*Joe* | *Mary*) (*thought* | *said* | *hopes* | *saw*)

Recursion

Cannot remove General Recursion

Example of a general recursion

Sentence = **DET NOUN** ((*that* | <E>) **Sentence VERB** | **VERB**)

The cat is sleeping

The cat that my cousin bought is sleeping

? The cat that the neighbour my cousin knows bought is sleeping

** The cat that the neighbour the cousin her friend saw bought is sleeping*

…

Recursion

Conclusions

- Linguists should not hesitate to use recursions to simplify their grammar
- NooJ can remove left and right recursions automatically. If the resulting grammar does not contain any more recursion, it is equivalent to a regular grammar. NooJ can then apply it to texts very efficiently.
- NooJ cannot remove all recursions. However, NooJ can impose a limitation on the depth of the recursion (*e.g.*, maximum 5 embedded relative clauses). The resulting grammar can then be transformed into a regular grammar and thus can be applied to texts efficiently.



CONGRATULATIONS


You know how to construct Context-Free Grammars and apply them to texts to implement various NLP applications