HUMBOLDT-UNIVERSITÄT ZU BERLIN

LaTeX for Linguists

L4L 07: Math mode 2 & trees

Antonio Machicao y Priemer
https://www.linguistik.hu-berlin.de/staff/amyp

MGK Workshop – SFB 1412, Berlin

Contents

1. Math mode 2
   - Non-exhaustive lists of symbols
   - Example: Set theory
   - Example: Propositional Logic
   - Example: Quantifiers
   - Meaning brackets
   - Writing formulae
2. Trees
   - Loading forest
   - forest syntax
   - Trees in example environments
   - Abbreviating nodes
   - Glossing or translating
   - Sub- and superscript
   - Arrows
   - Marking nodes
   - Syllabic structures
   - Trees for Typology
   - Further examples
   - Further features

Reader

LaTeX Reader (Freitag & Machicao y Priemer 2019):
https://doi.org/10.13140/RG.2.2.29299.27682

Exercises and **Handouts**:
https://www.linguistik.hu-berlin.de/de/staff/amyp/latex

1. Math mode 2

2. Trees

Non-exhaustive lists of symbols

Symbols you could need (the following lists are by no means exhaustive):

$=$	=	$\sim$	\sim	∞	\infty
$\pm$	\pm	$\approx$	\approx	$\emptyset$	\emptyset
$\cdot$	\cdot	$\subset$	\subset	$\Box$	\Box
$\times$	\times	$\supset$	\supset	%	\%
$\circ$	\circ	$\subseteq$	\subseteq	$	\$
$\in$	\in	$\cap$	\cap	&	\&
$\ni$	\ni	$\cup$	\cup	#	\#
$\neq$	\neq	$\forall$	\forall	\	\backslash
$\leq$	\leq	$\exists$	\exists	$\dots$	\dots
$\geq$	\geq	$\land$	\land	<	<
$\ll$	\ll	$\lor$	\lor	>	>
$\gg$	\gg	$\lnot$	\lnot		

Table 1: Some non-specific symbols

$\rightarrow$	\rightarrow	$\Downarrow$	\Downarrow	$\{\}$	\{\}
$\leftarrow$	\leftarrow	$\mapsto$	\mapsto	$\mathcal{A}$	\mathcal{A}
$\leftrightarrow$	\leftrightarrow	$\leadsto$	\leadsto	$\mathfrak{A}$	\mathfrak{A}
$\Rightarrow$	\Rightarrow	$\xrightarrow[abc]{xyz}$	\xrightarrow[abc]{xyz}	$\mathbb{R}$	\mathbb{R}
$\Leftarrow$	\Leftarrow	()	()	$\aleph$	\aleph
$\Leftrightarrow$	\Leftrightarrow	[]	[]		

Table 2: Some arrows, brackets, fonts

α	\alpha	θ	\theta	ε	\varepsilon
γ	\gamma	ϕ	\phi	ϑ	\vartheta
δ	\delta	Γ	\Gamma	Φ	\Phi
ϵ	\epsilon	Δ	\Delta	φ	\varphi

Table 3: Some Greek letters and variants

$\tilde{a}$	\tilde{a}	$\notin$	\notin	$\widetilde{abc}$	\widetilde{abc}
$\bar{a}$	\bar{a}	$\dot{a}$	\dot{a}	$\overline{abc}$	\overline{abc}
$\vec{a}$	\vec{a}	$\ddot{a}$	\ddot{a}	$\overrightarrow{abc}$	\overrightarrow{abc}
$\hat{a}$	\hat{a}	$\dot{=}$	\ddot{=}	$\widehat{abc}$	\widehat{}

Table 4: Some combinations of symbols

Some lists of symbols for LaTeX:

- List of logic symbols (Wikipedia): `https://en.wikipedia.org/wiki/List_of_logic_symbols`
- LaTeX for Logicians: `http://www.logicmatters.net/latex-for-logicians/`
- The Great, Big List of LaTeX Symbols (Carlisle et al. 2001)
- The Comprehensive LaTeX Symbol List – Symbols accessible from LaTeX (Pakin 2017)

Draw the symbol and get the code: `http://detexify.kirelabs.org`

As a reminder you can use `\textrm{}`, `\textnormal{}`, or `\text{}` (with the package `amsmath`) to use **normal text inside math mode** and combine it with other commands.

(1) `$A ß Ü \textsc{a}$`: *A*A

(2) `$\textrm{A ß Ü \textsc{a}}$`: A ß Ü A

(3) `$\text{A ß Ü \textsc{a}}$`: A ß Ü A

(4) `$\textnormal{A ßÜ \textsc{a}}$`: A ß Ü A

Set theory

```
$\{\text{a}\} \subset \{\text{a, e}\}$
```

(5) $\{\text{a}\} \subset \{\text{a, e}\}$

```
$\emptyset \subseteq \{\text{a, b}\}$
```

(6) $\varnothing \subseteq \{\text{a, b}\}$

```
$\# \{\emptyset, \text{a} \} = 2$
```

(7) $\#\{\varnothing, \text{a}\} = 2$

```
$\emptyset \in \{\emptyset, \text{a} \}$
```

(8) $\varnothing \in \{\varnothing, \text{a}\}$

```
$\emptyset \notin \{\text{a}\}$
```

(9) $\varnothing \notin \{\text{a}\}$

```
If $|\text{A}| = n$ then $|\mathfrak{P}(\text{A})|=2^{n}$
```

(10) If $|\text{A}| = n$ then $|\mathfrak{P}(\text{A})| = 2^{n}$

```
$\{\text{a, e}\} \setminus \{\text{e, u}\} = \{\text{a}\}$
```

(11) $\{\text{a, e}\} \setminus \{\text{e, u}\} = \{\text{a}\}$

```
DeMorgan: $ \overline{[ \text{A} \cup \text{B} ]} =
[ \overline{\text{A}} \cap \overline{\text{B}} ] $
```

(12) DeMorgan: $\overline{[\text{A} \cup \text{B}]} = [\overline{\text{A}} \cap \overline{\text{B}}]$

Propositional Logic

```
DeMorgan's law:
$\lnot (P \lor Q ) \Leftrightarrow (\lnot P \wedge \lnot Q)$

Biconditional law:
$(P \leftrightarrow P) \Leftrightarrow ((P \rightarrow Q) \wedge (Q \rightarrow P))$

Logical consequence:
$((p \rightarrow q) \wedge p) \Rightarrow q$
```

(13) DeMorgan's law: $\lnot (P \lor Q) \Leftrightarrow (\lnot P \wedge \lnot Q)$

(14) Biconditional law: $(P \leftrightarrow P) \Leftrightarrow ((P \rightarrow Q) \wedge (Q \rightarrow P))$

(15) Logical consequence: $((p \rightarrow q) \wedge p) \Rightarrow q$

Quantifiers

```
$\exists x [\text{\textsc{woman}} (x) \land \textsc{sleep} (x)]$

$\forall x [\text{\textsc{woman}} (x) \rightarrow \textsc{sleep} (x)]$
```

(16) **Existential quantifier:** *A woman sleeps.*
$\exists x[\textsc{woman}(x) \land \textsc{sleep}(x)]$
↛ There is only one sleeper.

(17) **Universal quantifier:** *Every woman sleeps.*
$\forall x[\textsc{woman}(x) \rightarrow \textsc{sleep}(x)]$
↛ Only women are sleepers.

Meaning brackets

In order to use the meaning brackets $\llbracket\,\rrbracket$ you can

1. (using XeLaTeX) copy the Unicode symbol,
2. make an own command for the symbol to use the Unicode symbol,
3. use the package `MnSymbol`. It provides the meaning brackets a.o. symbols.

```
\usepackage{MnSymbol}
```

Meaning brackets can be used **only in math mode**:

```
$\lsem \alpha \beta \rsem = \lsem \beta \rsem (\lsem \alpha \rsem)$
```

(18) $\llbracket \alpha\beta \rrbracket = \llbracket \beta \rrbracket (\llbracket \alpha \rrbracket)$ [Function application]

Writing formulae

```
$\lsem [_{\text{PP}}\text{\emph{in Amsterdam}}] \rsem (s’)
 = \lambda P \lambda x [P(x) \land [x \text{ is in Amsterdam in } s’]]$
```

(19) $\llbracket [_{\text{PP}} \textit{in Amsterdam}] \rrbracket (s') = \lambda P \lambda x [P(x) \land [x \text{ is in Amsterdam in } s']]$

- *in Amsterdam*: object language
- s', x, P: variables
- is in Amsterdam: invariable predicate
- PP: Index

Exercise

- Write the following expressions:

(20) a. $\varnothing \subseteq \{\text{tea, foil, computer}\}$

b. $\#\{\varnothing,\ 20, \text{kitchen cabinet, e}\} = 4$

c. $\varnothing \notin \{\text{tea, foil, computer}\}$

(21) De Morgan: $\overline{[\mathrm{A} \cup \mathrm{B}]} = [\overline{\mathrm{A}} \cap \overline{\mathrm{B}}]$

(22) $\llbracket \textit{red or round} \rrbracket = \llbracket \textit{red} \rrbracket \cup \llbracket \textit{round} \rrbracket$

Exercise

- Write the following expressions:

(23) $(A \land B) \land C \Leftrightarrow A \land (B \lor C)$

(24) $\neg(A \Leftrightarrow B) \Leftrightarrow (A \Leftrightarrow \neg B)$

(25) $\llbracket \textit{All students read a book} \rrbracket := \forall x \exists y [\textsc{student}(x) \rightarrow \textsc{read}(x)(y)]$

(26)
$$\begin{aligned}
&\llbracket [\textit{Lola runs}] \rrbracket (s_3)\\
&= \llbracket \textit{runs} \rrbracket (s_3)(\llbracket \textit{Lola} \rrbracket (s_3))\\
&= \lambda s \lambda x [x \text{ runs in } s](s_3)(\lambda s[\text{Lola}](s_3))\\
&= \lambda x [x \text{ runs in } s_3](\text{Lola})\\
&= [\text{Lola runs in } s_3]\\
&= 1
\end{aligned}$$

1. Math mode 2
2. Trees

Trees

There are different packages for drawing trees:

- `qtree`
- `pstrees` (complex syntax, but more powerful than `qtree`)
- `tikz-qtree`
- `forest` (simple syntax, more powerful than `pstrees` and `qtree`, based on `tikz`)
- ...

Loading forest

```
\usepackage{forest}
```

`forest` provides many features for trees needed in linguistics.

These features can be loaded specifying the **option** `linguistics`.

```
\usepackage[linguistics]{forest}
```

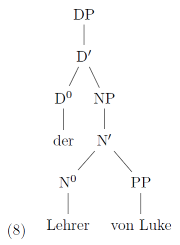


Figure 1: without `linguistics`

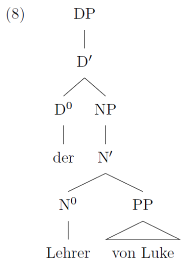


Figure 2: with `linguistics`

`gb4e` re-defines some commands needed for `forest`. If you are using `gb4e`, you must load `forest` first and `gb4e` after.

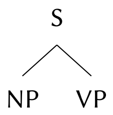
```
\usepackage[linguistics]{forest}

\usepackage{gb4e}
```

forest syntax

1. Use the `forest` **environment**.
2. Inside the `forest` environment, write the **bracket notation** for your tree.
3. Do **not** use **empty lines**!

```
\begin{forest}
[S [NP] [VP]]
\end{forest}
```

(27)



- Practice the bracket notation: `http://ironcreek.net/phpsyntaxtree/`

Exercise

- Load the package `forest` with the option `linguistics`.
- Replicate the following:

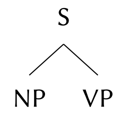
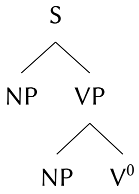


Figure 3: Simple tree

Figure 3 shows a simple tree created with the package `forest`.

For bigger trees, it is useful – for the sake of clarity – not to write the bracket notation linearly.

```
\begin{forest}
[S
  [NP]
  [VP
    [NP]
    [V$^{0}$]
  ]
]
\end{forest}
```



vs.

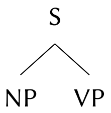
```
\begin{forest}
[S [NP] [VP [NP] [V$^{0}$]]]
\end{forest}
```

Trees in example environments

When using the option `linguistics`, you can embed the tree in an example environment.

```
\ea
\begin{forest}
[S [NP] [VP]]
\end{forest}
\z
```

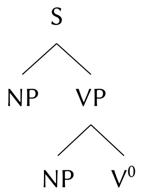
(28)



Exercise

- Copy your tree from Figure 3 and expand it as in (29).
- Replicate the following:

(29)



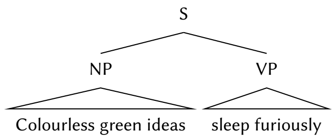
The tree in 29 shows a tree embedded in an example environment.

Abbreviating nodes

With the option `roof`, you can abbreviate nodes.

```
\ea
\begin{forest}
[S
  [NP [Colourless green ideas, roof]]
  [VP [sleep furiously, roof]]
]
\end{forest}
\z
```

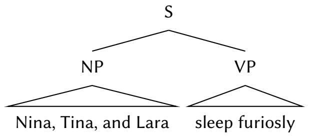
(30)



Take into account that options in `forest` (based on `TikZ`) are given by a **comma**. That means, you can use commas only when you **protect** them.

```
\ea
\begin{forest}
[S
  [NP [Nina{,} Tina{,} and Lara, roof]]
  [VP [sleep furiosly, roof]]
]
\end{forest}
\z
```

(31)



Exercise

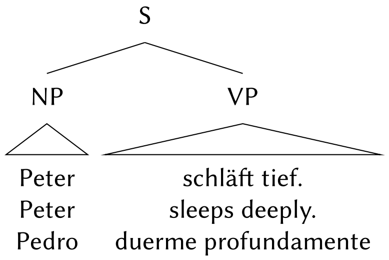
- Copy you tree (29) in an example environment.
- Put some words in the NPs separated by commas and a verb and use `roof` to abbreviate the nodes.

Glossing or translating

With `\\`, you can add **glosses or translations** to your tree.

```
\begin{forest}
[S
  [NP
    [Peter \\ Peter \\ Pedro, roof]
  ]
  [VP
    [schläft tief. \\ sleeps deeply. \\
         duerme profundamente, roof]
  ]
]
\end{forest}
```

(32)



Sub- and superscript

The characters ^ and _ are used in **math mode** for sub- and superscript, respectively.

```
$x^1$

$x_1$
```

(33) $x^1$

(34) $x_1$

The **default scope** of ^ and _ is only one character (35), use { } to **expand** it, see (36).

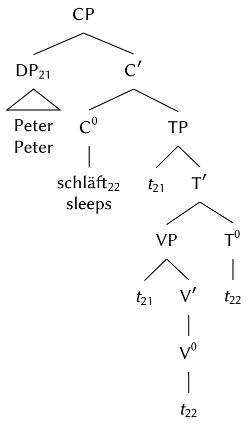
```
X$^1$ Y$^21$ X$_1$ Y$_21$

X$^{1}$ Y$^{21}$ X$_{1}$ Y$_{21}$
```

(35) X^1 $Y^2 1$ X_1 $Y_2 1$

(36) X^1 Y^{21} X_1 Y_{21}

Tree with sub- and superscripts

```
[CP
  [DP$_{21}$ [Peter \\ Peter, roof]]
  [C$^{\prime}$
    [C$^{0}$ [schläft$_{22}$ \\ sleeps]]
    [TP
      [$t_{21}$]
      [T$'$
        [VP
          [$t_{21}$]
          [V$^{\prime}$
            [V$^{0}$ [$t_{22}$]]
          ]
        ]
        [T$^{0}$ [$t_{22}$]]
      ]
    ]
  ]
]
```



See also how primes are generated.

Exercise

- Replicate the following:

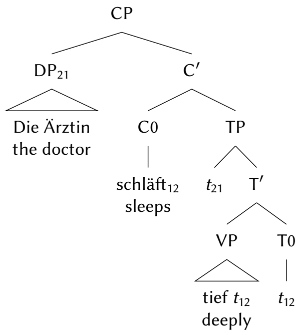


Figure 4: Complex tree

Arrows

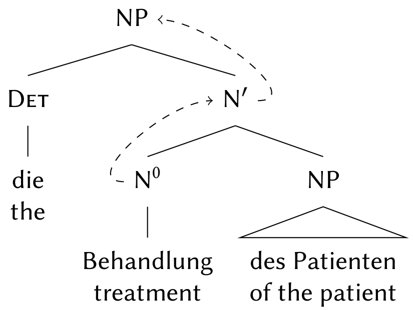
Arrows/lines **from node to node** (e.g. for movement, projection, etc.) can be drawn easily.

Give the nodes a **name** (command: `, name=`) and draw an arrow with the following command:

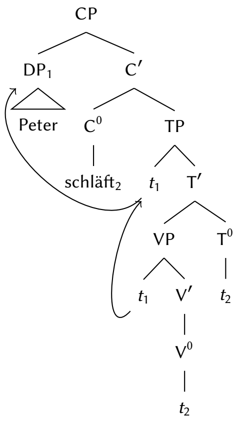
```
\draw[X] (Y) to[out=V, in=W] (Z);

\draw[->] (T10) to[out=south west, in=south west](T11);
```

- X: type of arrow/line (`->` `<-` `<->` `-`)
- Y: name of start node
- Z: name of end node
- V: starting position of the arrow at the start node (`south`/`north` + `east`/`west`)
- W: end position of the arrow at the end node (`south`/`north` + `east`/`west`)
- ;: end of the command

```
[NP, name=N2
  [\textsc{Det} [die \\ the]]
  [N$'$, name=N1
    [N$^0$, name=N0 [Behandlung \\ treatment]]
    [NP [des Patienten \\ of the patient, roof]]
  ]
]
\draw[->,dashed] (N0) to[out=west,in=west] (N1);
\draw[->,dashed] (N1) to[out=east,in=east] (N2);
```



```
[CP
  [DP$_{1}$, name=T12 [Peter, roof]]
  [C$^{\prime}$
    [C$^{0}$ [schläft$_{2}$, name=T22]]
    [TP
      [$t_{1}$, name=T11]
      [T$^{\prime}$
        [VP
          [$t_{1}$, name=T10]
          [V$^{\prime}$
            [V$^{0}$ [$t_{2}$, name=T20]]
          ]
        ]
        [T$^{0}$ [$t_{2}$, name=T21]]
      ]
    ]
  ]
]
\draw[->] (T10)
to[out=south west, in=south west](T11);
\draw[->] (T11)
to[out=south west, in=south west](T12);
```



Exercise

- Copy your tree in Figure 4 and make some changes to replicate the following:

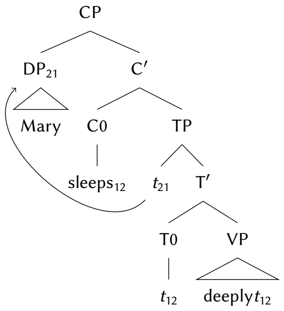


Figure 5: Complex tree with arrow

Exercise

- Create the following tree:

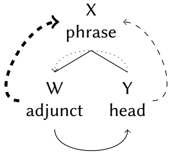


Figure 6: Head-adjunct relation

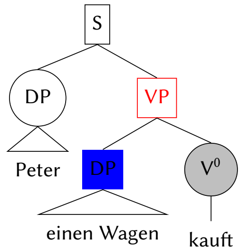
You will need the following specifications for the arrows: `dotted`, `dashed`, `ultra thick`.

Marking nodes

Some options:

- draw: square
- circle, draw: circle
- red: marking node with red
- fill=X: fill background of node with colour *X*
- circle, draw, fill=lightgray: circle around node, background in grey

```
[S, draw
  [DP, circle, draw
    [Peter, roof]]
  [VP, draw, red
    [DP, fill=blue
      [einen Wagen, roof]]
    [V$^{0}$, circle, draw, fill=lightgray
      [kauft]]
  ]
]
```



Hint: with the command \scalebox{.8}{FIGURE} you can rescale your figures.

Exercise

- Create the following tree:

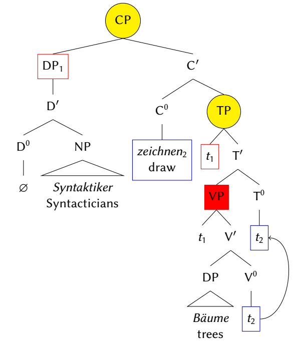


Figure 7: Complex trees

- Rescale your tree to 50% of the original size.

Syllabic structures

The package forest offers the style GP1 for syllabic structures.

```
\begin{forest} GP1,
[
[$\sigma$
  [O
    [[C[\textipa{l}]]]
  ]
  [R
    [N [V[\textipa{a}]]
    ]
  ]
]
[$\sigma$
  [O
    [[C[\textipa{t}]]]
  ]
  [R
    [N [V [\textipa{E}] ] ]
    [K [C [\textipa{\c{c}}] ] ]
  ]
]
]
\end{forest}
```

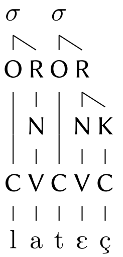


Figure 8: Two syllables with GP1

```
\begin{forest} GP1
[ [$\sigma$
    [O
      [[C[\textipa{S}]]]
      [[C[\textipa{t}]]]
      [[C[\textipa{\textscr}]]]
    ]
    [R
      [N
        [V[\textipa{U}]]
      ]
      [K
        [C[\textipa{m}]]
        [C[\textipa{\t{pf}}]]
        [C[\textipa{s}]]
        [C[\textipa{t}]]
      ]
    ]
  ]
]
\end{forest}
```

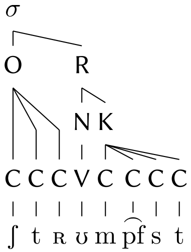


Figure 9: Complex syllable with GP1

Without using GP1, you can draw your syllabic structures with forest. You will need the (TikZ) commands , phantom and , tier=word.

```
\begin{forest}
[,phantom
  [$\sigma$
    [O
      [x, tier=word[\textipa{f}]]
      [x, tier=word
        [\textipa{\textscr }]
      ]
    ]
    [R
      [N
        [x, tier=word
          [\textipa{E}]
        ]
      ]
      [K [x[\textipa{\c{c}}]]]]
      ]
]
\end{forest}
```

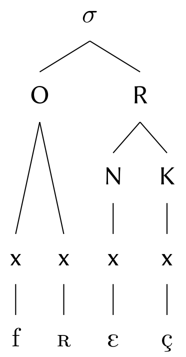


Figure 10: Syllable without GP1

```
\begin{forest}
[,phantom
  [$\sigma$
    [O
      [x, tier=word [\textipa{f}]]
      [x, tier=word[\textipa{K}]]]
    [R
      [N
        [x, tier=word[\textipa{\textopeno}]]]
      [K
        [x[\textipa{s}]]]]
  ]
  [$\sigma$
    [O
      [x, tier=word [\textipa{t}]]]
    [R
      [N [x[\textipa{I}]]]
      [K[x[\c{c}]]]]
  ]
]
\end{forest}
```

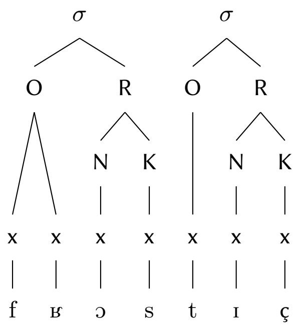


Figure 11: Two syllables

```
\begin{forest}
[,phantom
  [$\sigma$
    [O
      [x, tier=word
        [\textipa{P}]]]
    [R
      [N
        [x, tier=word
          [\textipa{\t{aU}}, name=aU] ]
        [x, name=x]
      ]
      [K
        [x [\textipa{x}] ] ] ]
  ]
]
{\draw[black] (aU.north)--(x.south);}
\end{forest}
```

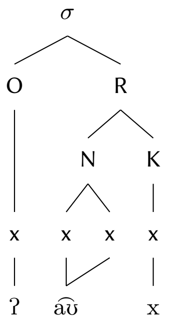


Figure 12: Diphthongs and long vowels

```
\begin{forest}
[,phantom
  [$\sigma$
    [O [x, tier=word [\textipa{t}] ]
    ]
    [R
      [N [x, tier=word [\textipa{I}] ] ]
      [K [x, name=x [\textipa{k}] ] ]
    ]
  ]
  [$\sigma$
    [O, name=onset]
    [R
      [N [x [\textipa{@}] ] ]
      [K [x [\textipa{n}] ] ] ]
    ]
  ]
\draw[black] (x.north)--(onset.south);
\end{forest}
```

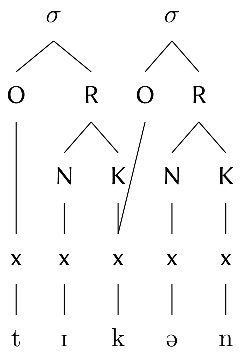


Figure 13: Ambisyllabic consonant

Exercise

- Create the following tree with the `forest` package:

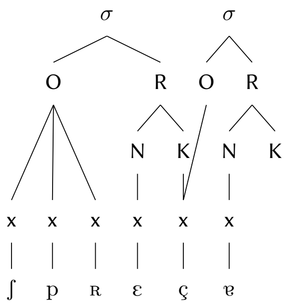


Figure 14: Syllabic structure with `forest`

Exercise

- Create the following tree with the `forest` package and the `GP1` option:

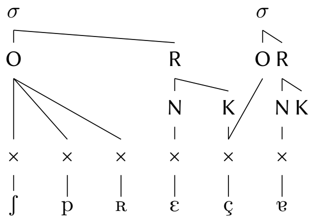


Figure 15: Syllabic structure with `forest` and `GP1`

Trees for Typology

Trees for typological purposes with the package `tikz-qtree`:

```
\usepackage{tikz-qtree}
\usetikzlibrary{positioning}
```

The setting specifies the differences in the branching.

```
\begin{tikzpicture}
%%% TikZ set
\tikzset{edge from parent/.style={draw,edge from parent path={(\tikzparentnode.south)--
      +(0,-8pt)-| (\tikzchildnode)}}}

\Tree
[.A
  [.B
    [.S  [.I P R T V U ] [.J ] ]
    [.I  [.K ] [.L ] ] ]
  [.M
    [.L  [.M ] [.N ] ]
    [.A  [.O ] [.P ] ]
  ]
]
\end{tikzpicture}
```

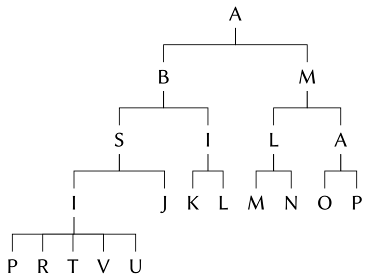


Figure 16: TikZ-qtree

Exercise

- Load the package `tikz-qtree` with the library `\usetikzlibrary{positioning}`
- Create the following tree with the TikZ-qtree package.
- Do not forget to put the TikZ-setting:

```
\tikzset{edge from parent/.style={draw,edge from parent path={(\tikzparentnode.south)--
 +(0,-8pt)-| (\tikzchildnode)}}}
```

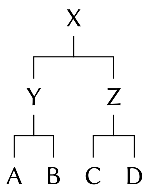


Figure 17: Tree with TikZ-qtree

- Comment out the TiKZ setting, compile and see what happens.

Further examples

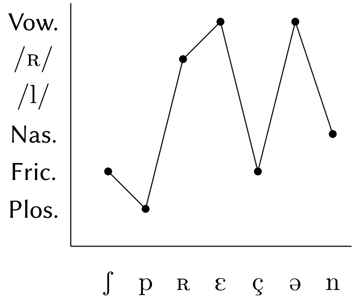


Figure 18: Sonority profile with TikZ

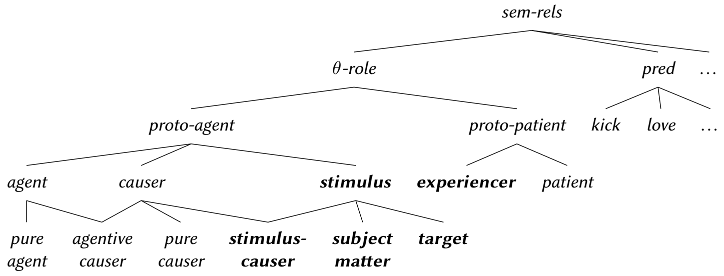


Figure 19: Type hierarchy for *semantic-relations*

Further features

- `forest` and `TikZ` are a very powerful packages. Check the package documentations (Tantau 2013; Živanović 2017) to see all benefits.
- Also, check the `forest` *Quick start guide* for linguists (Vanden Wyngaerd 2016).

Internet sources I

- Link: Akzente und Sonderzeichen in LaTeX.
  `https://de.wikibooks.org/wiki/LaTeX/_Akzente_und_Sonderzeichen` [Access: 10/10/2017]
- Link: Detexify
  `http://detexify.kirelabs.org` [Access: 08/12/2017]
- Link: LaTeX/Special Characters.
  `https://en.wikibooks.org/wiki/LaTeX/Special_Characters` [Access: 02/01/2019]
- Link: List of logic symbols – Wikipedia
  `https://en.wikipedia.org/wiki/List_of_logic_symbols` [Access: 08/12/2017]
- Link: LaTeX for Logicians:
  `http://www.logicmatters.net/latex-for-logicians/` [Access: 08/12/2017]
- Link: The Comprehensive LaTeX Symbol List – Symbols accessible from LaTeX (Pakin 2017):
  `https://ctan.org/tex-archive/info/symbols/comprehensive/` [Access: 08/12/2017]
- Link: The Great, Big List of LaTeX Symbols (Carlisle et al. 2001):
  `https://www.rpi.edu/dept/arc/training/latex/LaTeX_symbols.pdf` [Access: 08/12/2017]
- Package: Coffee Stains
  `http://hanno-rein.de/archives/349` [Zugriff: 06/01/2020]
- Package: forest (Živanović 2017)
  `https://www.ctan.org/pkg/forest` [Zugriff: 06/08/2019]
- Package: TikZ (Tantau 2013)
  `https://www.ctan.org/pkg/pgf` [Zugriff: 06/08/2019]

Literature I

Carlisle, David, Scott Pakin & Alexander Holt. 2001. The great, big list of LaTeX symbols. Handbuch. `https://www.rpi.edu/dept/arc/training/latex/LaTeX_symbols.pdf`.

Freitag, Constantin & Antonio Machicao y Priemer. 2019. LaTeX-Einführung für Linguisten. Manuscript. `https://doi.org/10.13140/RG.2.2.29299.27682`.

Knuth, Donald E. 1986. *The TeX book*. Boston, MA: Addison-Wesley.

Kopka, Helmut. 1994. *LaTeX: Einführung*, vol. 1. Bonn: Addison-Wesley.

Machicao y Priemer, Antonio. 2018. Kopf. In Stefan Schierholz & Pál Uzonyi (eds.), *Grammatik: Syntax* (Wörterbücher zur Sprach- und Kommunikationswissenschaft (Online) 1.2), Berlin: De Gruyter. `https://www.researchgate.net/publication/325046855_Kopf_Pre-Print`.

Machicao y Priemer, Antonio. 2019. Materialien zum gk linguistik. Script.

Machicao y Priemer, Antonio & Elisabeth Eberle. 2019. LaTeX for Linguists – Slides. Workshop for PhD candidates given at the PhD day of the Department of German Studies and Linguistics – Humboldt-Universität zu Berlin – 09. Oktober 2019.

Machicao y Priemer, Antonio & Paola Fritz-Huechante. 2018. Korean and Spanish psych-verbs: Interaction of case, theta-roles, linearization, and event structure in HPSG. In Stefan Müller & Frank Richter (eds.), *The 25th International Conference on Head-Driven Phrase Structure Grammar*, 155–175. University of Tokyo: CSLI Publications. `http://csli-publications.stanford.edu/HPSG/2018`.

Machicao y Priemer, Antonio & Robyn Kerkhof. 2016. LaTeX-Einführung für Linguisten – Slides. Präsentation beim 7. linguistischen Methodenworkshop an der Humboldt-Universität zu Berlin – 22.–24. Februar 2016. `https://www.researchgate.net/publication/295667182_LaTeX-Einfuhrung_fur_Linguisten`.

Literature II

Machicao y Priemer, Antonio & Sebastian Nordhoff. 2019. LaTeX for Linguists – Slides. Class for PhD candidates given at the LOT-Summer school – Universiteit van Amsterdam – 07.–18. Januar 2019.

Pakin, Scott. 2017. The comprehensive LaTeX symbol list – symbols accessible from LaTeX. Handbuch. `https://ctan.org/tex-archive/info/symbols/comprehensive/`.

Tantau, Till. 2013. Tikz & pgf: Manual for version 3.0.0. *CTAN: Comprehensive TeX Archive Network* `http://www.ctan.org/pkg/pgf`.

Vanden Wyngaerd, Guido. 2016. Forest quickstart guide for linguists. Manual. `https://ling.auf.net/lingbuzz/003391`.

Živanović, Sašo. 2017. Forest: a pgf/tikz-based package for drawing linguistic trees v2.1.5. *CTAN: Comprehensive TeX Archive Network* `https://ctan.org/pkg/forest`.