

Gramatické formalismy pro ZPJ II

Aleš Horák

E-mail: hales@fi.muni.cz
http://nlp.fi.muni.cz/poc_lingv/

Obsah:

- ▶ HPSG – Head-driven Phrase Structure Grammar
- ▶ Metagramatika systému synt

HPSG – Head-driven Phrase Structure Grammar – pokrač.

- ▶ gramatika je v HPSG modelována pomocí **uspořádaných příznakových struktur**, které korespondují s typy výrazů přirozeného jazyka a jejich částmi
- ▶ cílem teorie je detailní specifikace, které příznakové struktury jsou **přípustné**
- ▶ příznakové struktury definují **omezení** hodnoty příznaků mohou být jednoho ze čtyř typů
 - atomy
 - příznakové struktury
 - množiny příznakových struktur ($\{...\}$)
 - nebo seznamy příznakových struktur ($\langle...\rangle$)

HPSG – Head-driven Phrase Structure Grammar

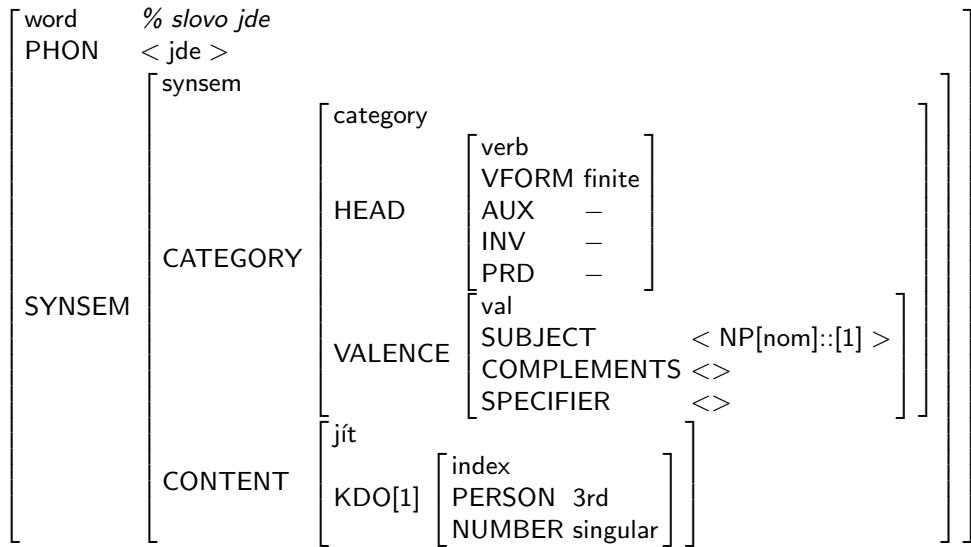
- ▶ HPSG, **Head-driven Phrase Structure Grammar** – Pollard & Sag, 1994
- ▶ navazuje na Gazdar, **Generalized Phrase Structure Grammar**, 1985
- ▶ **lexikalizovaná** teorie generativní gramatiky přirozeného jazyka
- ▶ *neterminály* CFG jsou nahrazeny **příznakovými strukturami**
- ▶ založená na **omezeních** (constraints)
- ▶ modeluje jazyk pomocí **deklarativních omezení** typovaných struktur. Pro vyhodnocení omezení se používá **unifikace** mezi příznakovými strukturami.
- ▶ **příznaky** jsou propojeny pomocí **strukturního sdílení**, tedy předáváním proměnných mezi podstrukturami dané struktury
- ▶ HPSG je **nederivační**, na rozdíl od jiných formalismů, kde jsou různé úrovně syntaktické struktury sekvenčně odvozovány pomocí transformačních operací

HPSG – lexikální hlava

- ▶ **slova** (lexikální položky) obsahují **hodně informací** – podle psycholingvistiky se podobá *zpracování v lidském mozku*
- ▶ **lexikální hlava** – základní prvek frázové struktury HPSG
 lexikální hlava = jedno slovo, jehož položka specifikuje informace, které určují základní gramatické **vlastnosti fráze**, kterou hlava zastupuje
 gramatické vlastnosti zahrnují:
 - morfologické informace (part-of-speech, POS)
 N zastupuje NP, VP zastupuje S, V zastupuje VP
 - relace závislosti (např. valenční rámec slovesa)
- ▶ lexikální hlava obsahuje také klíčové **sémantické informace**, které sdílí se zastupovanou frází

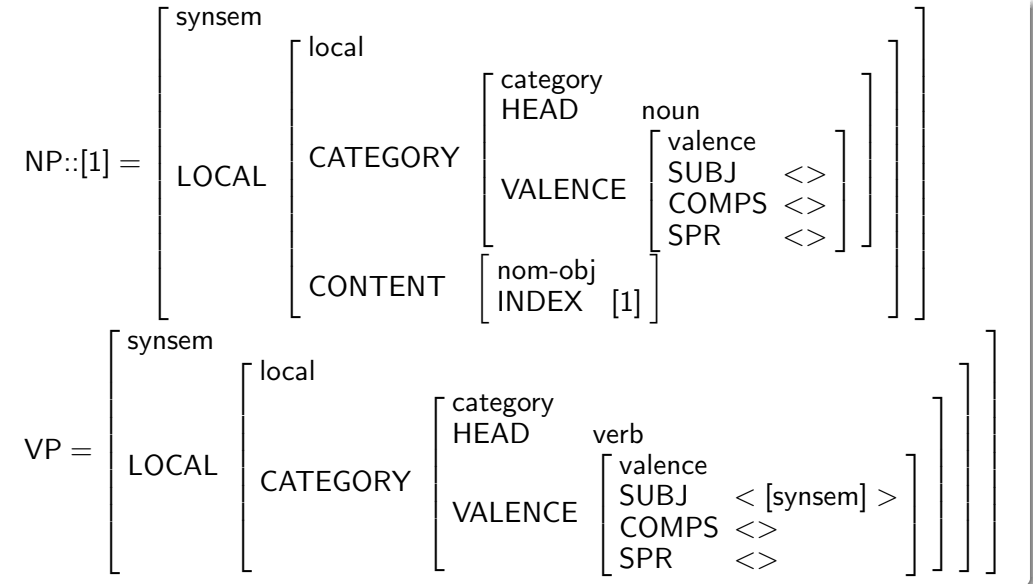
HPSG – struktury

HPSG struktury jsou **typované příznakové struktury**
 zapisují se pomocí AVM – **příznaky** velkými písmeny, **typy** malými



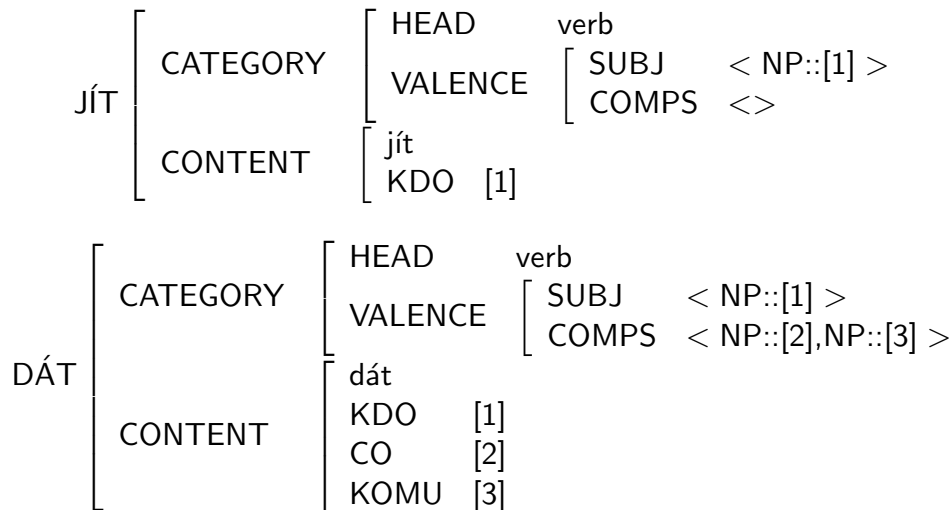
HPSG – syntaktické kategorie

symboly **syntaktických kategorií** – zkratky určitých příznakových popisů:



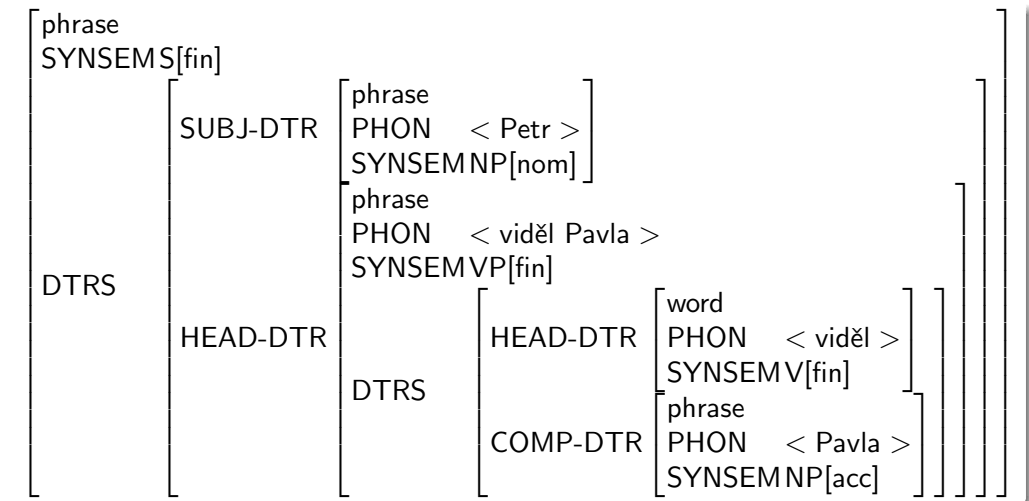
HPSG – lexikální položky

velké množství akcí je v **lexikonu**:



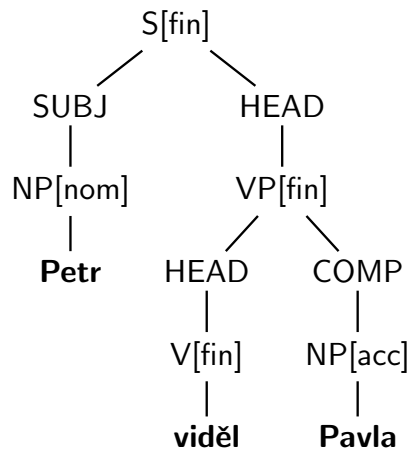
HPSG – fráze

reprezentace **frází** – v HPSG obdoba reprezentace **slov**
 navíc příznak **DAUGHTERS** – struktura členů fráze



HPSG – fráze – pokrač.

pro snazší čtení popisů frází používáme **stromový zápis**:



ve skutečnosti se ovšem jedná o **příznakovou strukturu**, ne strom!

HPSG – deklarace typu

pro popis omezení geometrie příznaku se používají **typové deklarace**:

category: [HEAD: head, VALENCE: valence]

```

head # příznaková struktura složená z příznakových struktur
noun: [CASE: case]
verb: [VFORM: vform, AUX: boolean, INV: boolean]
prep: [PFORM: pform]
...
  
```

```

vform # jednoduchý příznak, forma slovesa – možné hodnoty:
fin # určitý tvar slovesa
inf # neurčitý tvar slovesa – infinitive
...
  
```

```

case # jednoduchý příznak, gramatický pád
nom # 1. pád, nominativ
acc # 4. pád, akuzativ
...
  
```

HPSG – dobře utvořené příznakové struktury

dobře utvořené příznakové struktury musí splňovat **omezení daná gramatikou**

příznaková struktura je **dobře utvořená** ⇔:

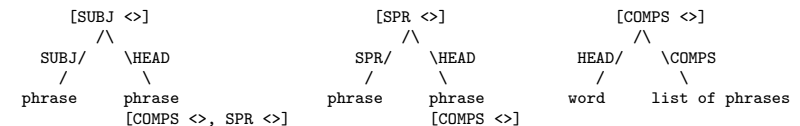
- ▶ každý uzel splňuje **omezení geometrie příznaku**
- ▶ každý uzel vstupního slova splňuje **omezení některé lexikální položky**
- ▶ každý frázový uzel splňuje **frázová omezení** – *omezení přímé dominance* (immediate dominance, viz dále), *omezení hlavových příznaků* (head feature), *valenční omezení*, ...

omezení geometrie příznaku specifikují:

- ▶ s jakými **typy** se pracuje
- ▶ jaká je použitá **typová hierarchie** – který typ je podtypem jiného typu
- ▶ pro každý typ – jaké příznaky přísluší tomuto typu
- ▶ pro každý typ a každý příznak – jakých typů mohou být hodnoty tohoto příznaku

HPSG – dobře utvořená slova a fráze

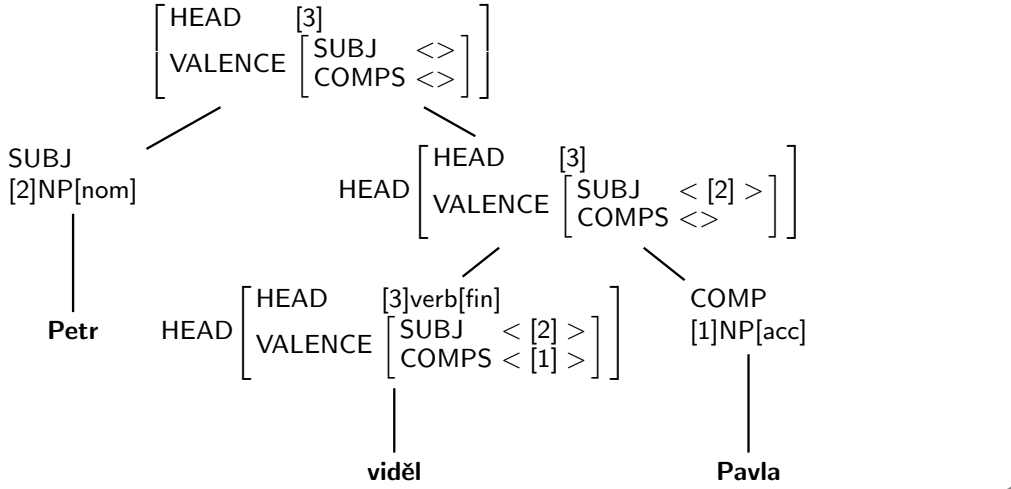
- ▶ každé vstupní **slovo** musí splňovat některou **lexikální položku**
- ▶ **fráze** musí splňovat **frázová omezení** (constraints):
 - **omezení přímé dominance** – každá fráze musí odpovídat jednomu ze schémat – schéma *head-subject*, schéma *head-specifier*, schéma *head-complement*, ...



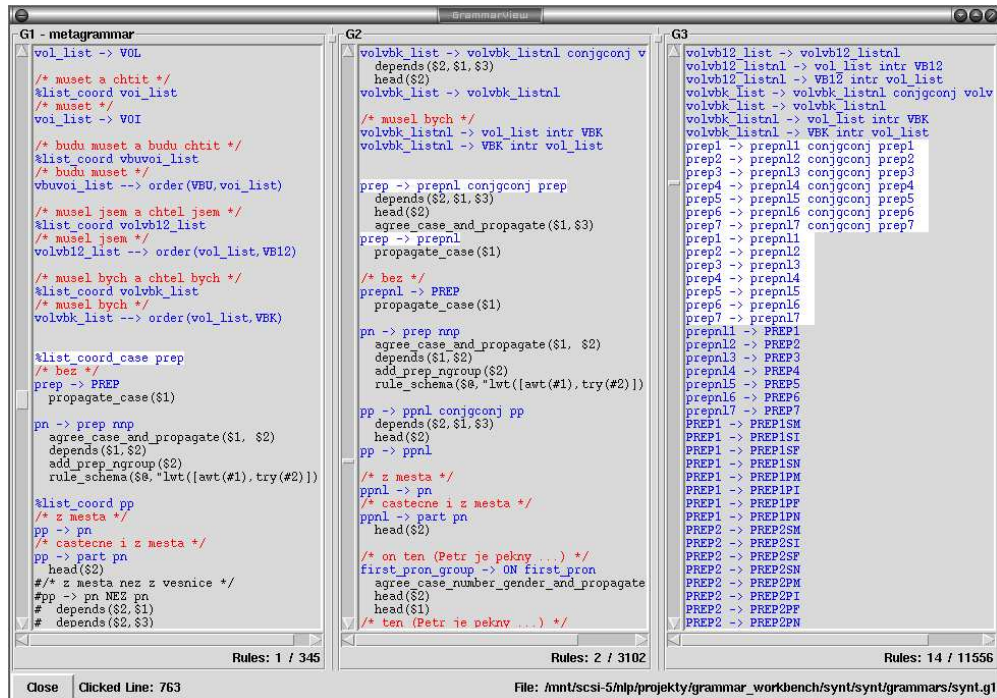
- **omezení hlavových příznaků** – pro každou frázi, která má hlavu, musí být hlavové příznaky fráze shodné s hlavovými příznaky potomka, který je hlavou
- **valenční omezení** – pro každý z valenčních příznaků (SUBJECT, COMPLEMENTS, ...) – hodnota příznaku na hlavové frázi musí odpovídat hodnotě na potomku, který je hlavou, minus ty příznaky, které jsou splněny některým z nehlavových potomků

HPSG – dobře utvořené příznakové struktury

omezení ve větě 'Petr viděl Pavla.':



DEMO: **GG** – HPSG pro němčinu, DFKI Language Technology Lab, Saarbrücken
<http://hpsg.fu-berlin.de/~stefan/Babel/Interaktiv/beispiel.html>



Metagramatika systému synt

3 formy (meta)gramatiky: [ukázka](#)

- ▶ metagramatika (G1)
 - ▶ pravidla s kombinatorickými konstrukty + globální omezení pořadí
 - ▶ akce (= gramatické testy + kontextové akce)
 - ▶ česká lingvistická tradice – závislostní struktury, kontrola shody, pravidla pro pořadí slov, ...
- ▶ generovaná gramatika (G2)
 - ▶ bezkontextová pravidla
 - ▶ akce
- ▶ expandovaná gramatika (G3)
 - ▶ jen bezkontextová pravidla

Metagramatika – kombinatorické konstrukty

kombinatorické konstrukty se používají pro generování variant pořadí daným terminálů a neterminálů

hlavní kombinatorické konstrukty:

- ▶ order() generuje všechny možné permutace zadaných komponent
- ▶ first() argument musí být na prvním místě
- ▶ rhs() doplní všechny pravé strany svého argumentu

/* budu se ptát */
 clause ==> order(VBU,R,VRI)

/* který ... */
 relclause ==> first(relprongr) rhs(clause)

Metagramatika – typy pravidel

- ▶ -> normální CF pravidlo
- ▶ --> vložit intersegment mezi každé dva prvky
- ▶ ==> + kontrola správného pořadí příklonek
- ▶ ===> intersegmenty na začátku a konci RHS, spojky, ...

```
ss -> conj clause
/* budu muset číst */
futmod --> VBU VOI VI
/* byl bych býval */
cpredcondgr ==> VBL VBK VBL
/* musím se ptát */
clause ===> VO R VRI
```

clause pravidla se zadávají pomocí **pravidlových vzorů**

Metagramatika – generativní konstrukty

skupina výrazů %list_* – produkují nová pravidla pro seznamy (s oddělovači/bez oddělovačů, s různými testy na shody, ...)

```
/* (nesmím) zapomenout udelat - to forget to do */
%list_nocoord vi_list
vi_list -> VI
```

```
%list_coord_case np
%list_coord_case_number_gender left_modif
/* krasny velky pes a mala kocka - beautiful dog and small cat */
np -> left_modif np
```

koncovky *_case, *_number_gender and *_case_number_gender určují typ shody

Metagramatika – globální omezení pořadí

globální omezení pořadí zakazuje některé kombinace pořadí preterminálů

%enclitic – které preterminály jsou brány jako **příklonky**

%order – zajišťuje dodržení precedence zadaných preterminálů

```
/* jsem, bych, se */
%enclitic = (VB12, VBK, R)
```

```
/* byl — četl, ptal, musel */
%order VBL = {VL, VRL, VOL}
```

Metagramatika – pravidlové vzory

pravidla pro slovesné skupiny – cca 40% všech pravidel metagramatiky
pravidlové vzory %group – definují časté skupiny konstrukcí v pravidlech

```
%group verbP={
  V: verb_rule_schema($@,"(#1)")
    groupflag($1,"head"),
  VR R: verb_rule_schema($@,"(#1 #2)")
    groupflag($1,"head"),
}
```

```
%template clause ===> order(RHS)
```

```
/* ctu/ptam se - I am reading/I am asking */
clause %> group(verbP) vi_list
  verb_rule_schema($@,"#2")
  depends(getgroupflag($1,"head"), $2)
```

Metagramatika – pravidlové vzory – pokrač.

- ▶ předchozí příklad – skupina verbP = dvě skupiny preterminálů (V a VR R) s příslušnými akcemi
- ▶ při použití v clause vytvoří postupně dvě různé pravé strany
- ▶ (get)groupflag – odkaz na prvek uvnitř %group
- ▶ **vzor celého pravidla** – speciální pravidlová šipka %> %template definuje vzor každého pravidla s %>

Gramatika G2 – kontextové akce

- ▶ gramatické **testy na shody** – pád, rod, číslo
- ▶ **testy na zanoření vedlejších vět** – test_comma
- ▶ akce pro specifikaci **závislostních hran**
- ▶ akce **typové kontroly** logických konstrukcí

```
np -> adj_group np
rule_schema($@, "lwtx(awtx(#1) and awtx(#2))")
rule_schema($@, "lwtx([[awtx(#1),#2],x])")
```

rule_schema – schéma pro tvorbu logické konstrukce ze subkonstrukcí

pojdu jenom kombinace, které **typově vyhovují** danému schématu

Metagramatika – úrovně pravidel

- ▶ používá se pro **ohodnocení** výstupních stromů pro jejich **třídění**
- ▶ doplněk trénování na **stromových korpusech** (6.000 vět)
- ▶ zadané **lingvistou** – specialistou na vývoj gramatiky
- ▶ **základní úroveň** – 0, **vyšší úrovně** – méně frekventované fenomény
- ▶ pravidla vyšších úrovní mohou být v průběhu analýzy **zapnuté/vypnuté**

```
3:np -> adj_group
propagate_case_number_gender($1)
```

Expandovaná gramatika G3

- ▶ překlad testů na shody do CF pravidel
- ▶ v češtině – 7 gramatických pádů, dvě čísla a 4 rody → 56 možných variant pro plnou shodu mezi dvěma prvky

počty pravidel

metagramatika G1	253
gramatika G2	3091
expandovaná gramatika G3	11530

Výstupy syntaktické analýzy

synt nabízí více možností zpracování výsledných struktur:

- ▶ **syntaktické stromy** (varianty: technická/lingvistická, uspořádané/neuspořádané) [▶ ukázka](#)
- ▶ struktura **chart** – komprimovaný les všech stromů [▶ ukázka](#)
- ▶ **závislostní graf** – graf všech závislostí vytvořených akcemi [▶ ukázka](#)
- ▶ seznamy **frází** v dané větě, získané přímo ze struktury *chart* [▶ ukázka](#)
- ▶ částečné **zjednoznačnění morfologických značek** na vstupu [▶ ukázka](#)

manuál ke **GDW** – Grammar Development Workbench

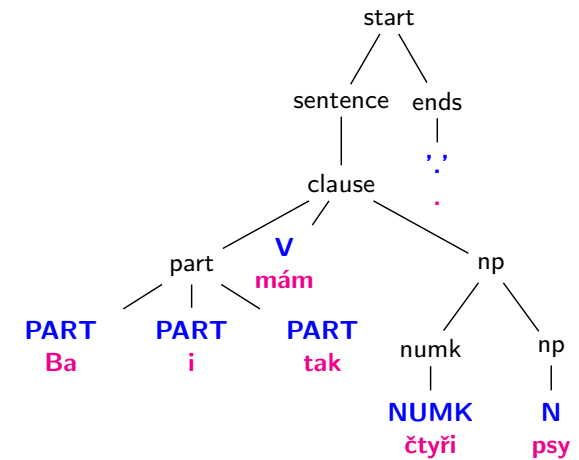
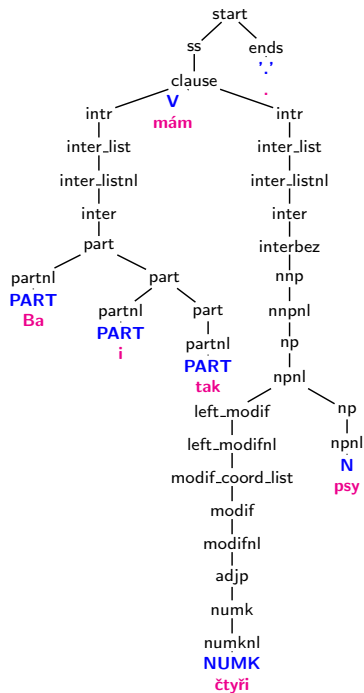
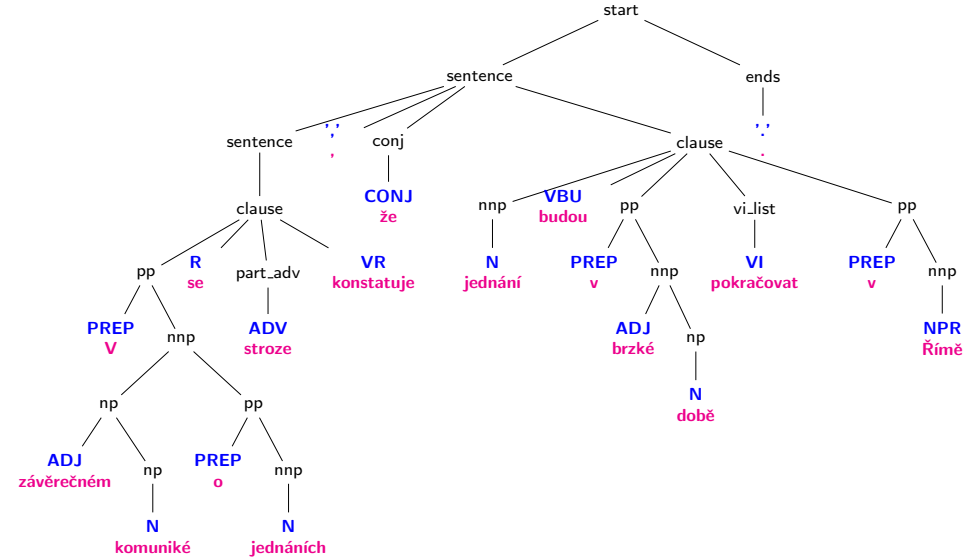
http://nlp.fi.muni.cz/projekty/grammar_workbench/manual/

DEMO: **wwwsynt** – webové rozhraní k syntu

<http://nlp.fi.muni.cz/projekty/wwwsynt/>

[▶ přeskočit příklady](#)

V závěrečném komuniké o jednáních se stroze konstatuje, že jednání budou v brzké době pokračovat v Římě.



[◀ Zpět](#)

System synt – příklad logické analýzy

vyhodnocení rule_schema pro np 'pečené kuře'

```
4, 6, -npl -> . left_modif np .: k1gNnSc145
agree_case_number_gender_and_propagate OK
rule_schema: 2 nterms, 'lwtx(awtx(#1) and awtx(#2))'
And constrs, Abstr and Exi vars are just gathered
1 (1x1) constructions:
    λw2λt3λx4([pečenýw2t3, x4] ∧ [kuřew2t3, x4])... (ol)τω
And constrs: none added
Exi vars: none added
```

System synt – příklad logické analýzy – pokrač.

vyhodnocení verb_rule_schema pro celou clause

```
verb_rule_schema: 3 groups
no acceptable subject found: supplying an inexplicit one
inexplicit subject: k3xPgMnSc1,k3xPgInSc1: On...l
Clause valency list:      jíst <v>#1:(1)hA-#2:(2)hPTc1,      ...
Verb valency list:       jíst <v>#2:hH-#1:hPTc4ti
Matched valency list:    jíst <v>#2:(1)hH-#1:(2)hPTc4ti
time span: λt12dnest12... (oτ)
frequency: Onc... ((o(oτ))π)ω
verbal object: x15... (o(oπ))(oπ)
present tense clause:
λw17λt18(∃i10)(∃x15)(∃i16)([Doesw17t18, On, [Impw17, x15]] ∧ [večeřew17t18, i10] ∧
[pečenýw17t18, i16] ∧ [kuřew17t18, i16] ∧ x15 =
[jíst, i16]w17 ∧ [[kw17t18, i10]w17, x15])... π
clause:
λw19λt20[Pt20, [Oncw19, λw17λt18(∃i10)(∃x15)(∃i16)([Doesw17t18, On, [Impw17, x15]] ∧
[večeřew17t18, i10] ∧ [pečenýw17t18, i16] ∧ [kuřew17t18, i16] ∧ x15 =
[jíst, i16]w17 ∧ [[kw17t18, i10]w17, x15]], λt12dnest12]]... π
```