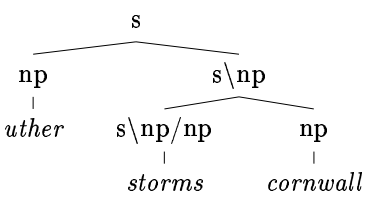
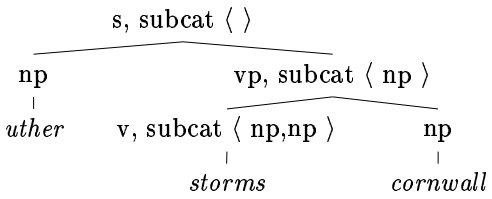
PATR, část 2

Gramatika 2

shoda podmět – přísudek
komplexní subkategorizace (valenční rámce)
S → NP VP
VP → V
VP → VP X
N: *uther, cornwall, knights*
V: *sleeps, sleep, storms, stormed, storm, has, have, persuades, to*

(1) Syntaktický strom věty *Uther storms Cornwall* schématicky podle kategoriální gramatiky a podle "Gramatiky 2"
subkategorizace: postupné krácení seznamu argumentu inspirováno kategoriální gramatikou (transitivní sloveso: S\NP/NP)





(2) transitivní sloveso se seznamem valencí v notaci s lomenými závorkami ⟨ ⟩, pořadí v seznamu valencí je dáno pořadím identifikace slovesných doplnění při analýze zdola nahoru, podmět je tedy na posledním místě

$$\textit{storms} \mapsto \begin{bmatrix} \text{cat} & \text{v} \\ \text{head} & \begin{bmatrix} \text{form} & \text{finite} \end{bmatrix} \\ \text{subcat} & \left\langle \begin{bmatrix} \text{cat} & \text{np} \end{bmatrix}, \begin{bmatrix} \text{cat} & \text{np} \\ \text{head} & \begin{bmatrix} \text{agr} & \begin{bmatrix} \text{num} & \text{sg} \\ \text{per} & 3 \end{bmatrix} \end{bmatrix} \end{bmatrix} \right\rangle \end{bmatrix}$$

(3) totéž se seznamem valencí v notaci first-rest

$$\textit{storms} \mapsto \begin{bmatrix} \text{cat} & \text{v} \\ \text{head} & \begin{bmatrix} \text{form} & \text{finite} \end{bmatrix} \\ \text{subcat} & \begin{bmatrix} \text{first} & \begin{bmatrix} \text{cat} & \text{np} \end{bmatrix} \\ \text{rest} & \begin{bmatrix} \text{first} & \begin{bmatrix} \text{cat} & \text{np} \\ \text{head} & \begin{bmatrix} \text{agr} & \begin{bmatrix} \text{num} & \text{sg} \\ \text{per} & 3 \end{bmatrix} \end{bmatrix} \end{bmatrix} \\ \text{rest} & \text{end} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

(4) pomocné sloveso *have* – ekvivalent *subject raising* v GB: podmět syntakticky závislého (zde významového) slovesa se ztotožní s podmětem slovesa řídícího (zde pomocného), v další verzi gramatiky uvidíme, že tento podmět zůstane sémantickým argumentem jen u slovesa závislého

$$has \mapsto \begin{bmatrix} \text{cat} & \text{v} \\ \text{head} & \begin{bmatrix} \text{form} & \text{finite} \end{bmatrix} \\ \text{subcat} & \left\langle \begin{bmatrix} \text{cat} & \text{vp} \\ \text{head} & \begin{bmatrix} \text{form} & \text{pple} \end{bmatrix} \\ \text{subcat} & \left\langle \boxed{1} \right\rangle \end{bmatrix}, \boxed{1} \begin{bmatrix} \text{cat} & \text{np} \\ \text{head} & \begin{bmatrix} \text{agr} & \begin{bmatrix} \text{num} & \text{sg} \\ \text{per} & 3 \end{bmatrix} \end{bmatrix} \end{bmatrix} \right\rangle \end{bmatrix}$$

(5) pomocné "sloveso" *to*, podobné řešení jako u *has*:

$$to \mapsto \begin{bmatrix} \text{cat} & \text{v} \\ \text{head} & \begin{bmatrix} \text{form} & \text{inf} \end{bmatrix} \\ \text{subcat} & \left\langle \begin{bmatrix} \text{cat} & \text{vp} \\ \text{head} & \begin{bmatrix} \text{form} & \text{nonfin} \end{bmatrix} \\ \text{subcat} & \left\langle \boxed{1} \right\rangle \end{bmatrix}, \boxed{1} \begin{bmatrix} \text{cat} & \text{np} \end{bmatrix} \right\rangle \end{bmatrix}$$

(6) sloveso se 3 valencemi, typu *object control* – ekvivalent *object equi* v GB: podmět závislého slovesa se ztotožní s podmětem slovesa řídícího, v další verzi gramatiky uvidíme, že tento podmět se stane sémantickým argumentem u obou sloves; pořadí prvků v seznamu odpovídá pořadí identifikace slovesných doplnění při analýze zdola nahoru

$$persuades \mapsto \begin{bmatrix} \text{cat} & \text{v} \\ \text{head} & \begin{bmatrix} \text{form} & \text{finite} \end{bmatrix} \\ \text{subcat} & \left\langle \boxed{1} \begin{bmatrix} \text{cat} & \text{np} \end{bmatrix}, \begin{bmatrix} \text{cat} & \text{vp} \\ \text{head} & \begin{bmatrix} \text{form} & \text{inf} \end{bmatrix} \\ \text{subcat} & \left\langle \boxed{1} \right\rangle \end{bmatrix}, \begin{bmatrix} \text{cat} & \text{np} \\ \text{head} & \begin{bmatrix} \text{agr} & \begin{bmatrix} \text{num} & \text{sg} \\ \text{per} & 3 \end{bmatrix} \end{bmatrix} \end{bmatrix} \right\rangle \end{bmatrix}$$

(7) totéž se seznamem ve tvaru first-rest:

$$persuades \mapsto \begin{bmatrix} \text{cat} & \text{v} \\ \text{head} & \begin{bmatrix} \text{form} & \text{finite} \end{bmatrix} \\ & \begin{bmatrix} \text{first} & \boxed{1} \begin{bmatrix} \text{cat} & \text{np} \end{bmatrix} \\ \text{subcat} & \text{rest} & \begin{bmatrix} \text{first} & \begin{bmatrix} \text{cat} & \text{vp} \\ \text{head} & \begin{bmatrix} \text{form} & \text{inf} \end{bmatrix} \\ \text{subcat} & \begin{bmatrix} \text{first} & \boxed{1} \\ \text{rest} & \text{end} \end{bmatrix} \end{bmatrix} \\ & \text{rest} & \begin{bmatrix} \text{first} & \begin{bmatrix} \text{cat} & \text{np} \\ \text{head} & \begin{bmatrix} \text{agr} & \begin{bmatrix} \text{num} & \text{sg} \\ \text{per} & 3 \end{bmatrix} \end{bmatrix} \end{bmatrix} \\ & \text{rest} & \text{end} \end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

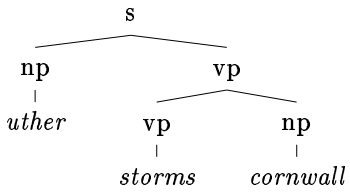
(8) pravidlo $S \rightarrow X \quad VP$:

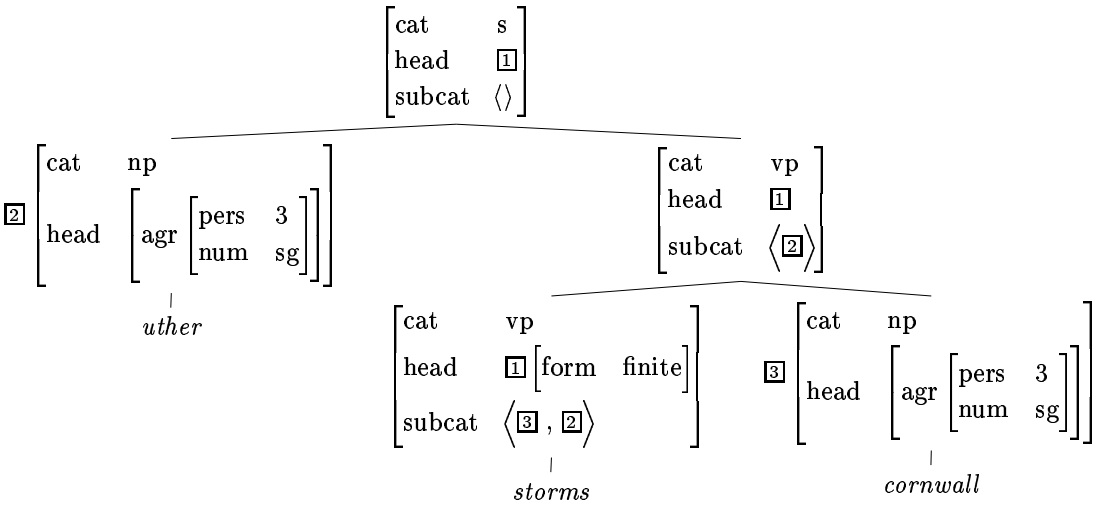
$$S \begin{bmatrix} \text{cat} & \text{s} \\ \text{head} & \boxed{1} \\ \text{subcat} & \langle\rangle \end{bmatrix} \quad \rightarrow \quad X \boxed{2}\begin{bmatrix} \end{bmatrix} \quad VP \begin{bmatrix} \text{cat} & \text{vp} \\ \text{head} & \boxed{1} \\ \text{subcat} & \left\langle \boxed{2} \right\rangle \end{bmatrix}$$

(9) pravidlo $VP_1 \rightarrow VP_2 \quad X$:

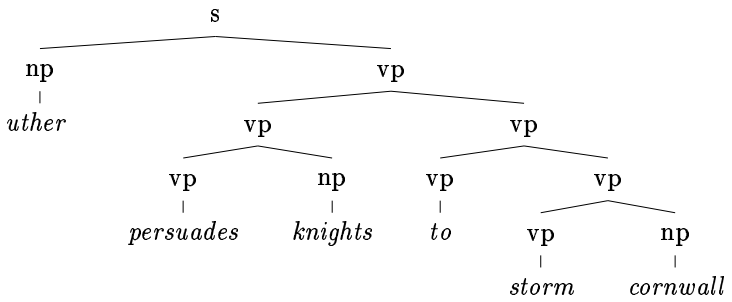
$$VP_1 \begin{bmatrix} \text{cat} & \text{vp} \\ \text{head} & \boxed{1} \\ \text{subcat} & \boxed{2} \end{bmatrix} \quad \rightarrow \quad VP_2 \begin{bmatrix} \text{cat} & \text{vp} \\ \text{head} & \boxed{1} \\ \text{subcat} & \left\langle \boxed{3} \mid \boxed{2} \right\rangle \end{bmatrix} \quad X \boxed{3}\begin{bmatrix} \end{bmatrix}$$

(10) Syntaktický strom věty *Uther storms Cornwall*





(11) Syntaktický strom věty *Uther persuades knights to storm Cornwall*



(syntaktický strom s uzly ohodnocenými sestavami rysů viz příloha)

Úkoly

1. Projděte gramatiku 2 a vyzkoušejte ji na všech uvedených příkladech. Ujistěte se, že pravidlům a slovníkovým heslům rozumíte. Zkuste např. upravit některé slovníkové heslo.

2. Rozšiřte gramatiku 2 o slovníkové heslo pro sloveso *gives*, které vyžaduje nepřímý a přímý předmět. Vyzkoušejte, zda heslo funguje na větách typu *uther gives knights cornwall.* Přidejte další tvary *give* a znovu vyzkoušejte.

3. Rozšiřte gramatiku 2 o slovníkové heslo pro modální sloveso *could.* Zkuste, zda gramatika správně analyzuje věty jako např. *uther could have persuaded knights to storm cornwall.*

4. Rozšiřte gramatiku 2 o slovníkové heslo pro sloveso *thinks*, jehož předmětem je klauze, a zkuste analyzovat věty typu *uther thinks cornwall sleeps.*

5. Vyzkoušejte kapacitu gramatiky a použité implementace PATRu na velmi dlouhých větách. (Zkuste např. věty typu *uther thinks cornwall thinks ... uther thinks knights sleep.*) Jakou nejdelší větu se vám podařilo analyzovat? Narazili jste na potíže? Jakou větu se vám již nepodařilo analyzovat?

6. Zatím jsme pracovali se syntaktickou strukturou, v níž byly každému uzlu přímo podřízeny nejvýše dva uzly jiné. Dejme tomu, že při analýze slovesných frází typu *gives uther cornwall* nebo *persuade uther to sleep* bychom raději viděli, kdyby podřízených uzlů mohlo být více, tedy kdyby hlavní sloveso bylo sesterským uzlem všech svých nesubjektových doplnění. Zkuste upravit gramatiku tak, aby připouštěla i struktury se třemi přímo podřízenými uzly. (Toto je náročnější úkol.)

7. Syntaktická struktura v podobě stromového grafu tak, jak se objevuje na výstupu, podává jasný obraz aplikace gramatických pravidel, tedy syntaktických složek/konstituentů, ale zajímá-li nás více závislostní nebo sémantická struktura, potřebovali bychom něco jiného. Zkuste navrhnout řešení, jak aplikací pravidel bezkontextové gramatiky v PATRu takovou strukturu vytvořit. (Náročný úkol.) Necháte se poddat? Podívejte se na gramatiku 3.

Příloha 1: Gramatika 2 v kódu QPATR

```
/*********************************************************************/
/*  SHIEBER2.GRM                                                     */
/*  demonstration grammar two (pp. 71-76) in QPATR syntax            */
/*     subject-verb agreement                                        */
/*     complex subcategorization                                     */
/*  Stuart M. Shieber, An Introduction to Unification-Based          */
/*  Approaches to Grammar. Stanford, 1986.                           */
/*********************************************************************/

% grammar rules ******************************************************

%   sentence formation
1 # s(S) ---> np(NP), vp(VP) :: S/head *= VP/head,
                                S/head/form *= finite,
                                VP/syncat/first *= NP,
                                VP/syncat/rest *= end.

%   trivial verb phrase
2 # vp(VP) ---> v(V) :: VP/head *= V/head,
                        VP/syncat *= V/syncat.

%   complements
3 # vp(VP_1) ---> vp(VP_2), xp(XP) ::
                     VP_1/head *= VP_2/head,
                     VP_2/syncat/first *= XP,
```

```
                    VP_2/syncat/rest  *= VP_1/syncat.

% lexicon ***********************************************************

uther     lex  np(F) ::
               F/head/agreement/gender *= masculine,
               F/head/agreement/person *= third,
               F/head/agreement/number *= singular.
cornwall  lex  np(F) ::
               F/head/agreement/gender *= masculine,
               F/head/agreement/person *= third,
               F/head/agreement/number *= singular.
knights   lex  np(F) ::
               F/head/agreement/gender *= masculine,
               F/head/agreement/person *= third,
               F/head/agreement/number *= plural.
%-------------------------------------------------------------------
sleeps    lex  v(F) ::
               F/head/form *= finite,
               F/syncat/first/cat *= np,
               F/syncat/first/head/agreement/person *= third,
               F/syncat/first/head/agreement/number *= singular,
               F/syncat/rest *= end.
sleep     lex  v(F) ::
               F/head/form *= finite,
               F/syncat/first/cat *= np,
               F/syncat/first/head/agreement/number *= plural,
               F/syncat/rest *= end.
sleep     lex  v(F) ::
               F/head/form *= nonfinite,
               F/syncat/first/cat *= np,
               F/syncat/rest *= end.
%-------------------------------------------------------------------
storms    lex  v(F) ::
               F/head/form *= finite,
               F/syncat/first/cat *= np,
               F/syncat/rest/first/cat *= np,
               F/syncat/rest/first/head/agreement/person *= third,
               F/syncat/rest/first/head/agreement/number *= singular,
               F/syncat/rest/rest *= end.
stormed   lex  v(F) ::
               F/head/form *= pastparticiple,
               F/syncat/first/cat *= np,
               F/syncat/rest/first/cat *= np,
               F/syncat/rest/rest *= end.
storm     lex  v(F) ::
               F/head/form *= nonfinite,
               F/syncat/first/cat *= np,
               F/syncat/rest/first/cat *= np,
               F/syncat/rest/rest *= end.
%-------------------------------------------------------------------
has       lex  v(F) ::
               F/head/form *= finite,
               F/syncat/first/cat *= vp,
               F/syncat/first/head/form *= pastparticiple,
               F/syncat/first/syncat/rest *= end,
```

```
               F/syncat/first/syncat/first *= F/syncat/rest/first,
               F/syncat/rest/first/cat *= np,
               F/syncat/rest/first/head/agreement/number *= singular,
               F/syncat/rest/first/head/agreement/person *= third,
               F/syncat/rest/rest *= end.
have      lex  v(F) ::
               F/head/form *= finite,
               F/syncat/first/cat *= vp,
               F/syncat/first/head/form *= pastparticiple,
               F/syncat/first/syncat/rest *= end,
               F/syncat/first/syncat/first *= F/syncat/rest/first,
               F/syncat/rest/first/cat *= np,
               F/syncat/rest/first/head/agreement/number *= plural,
               F/syncat/rest/rest *= end.
%-----------------------------------------------------------------------
persuades lex  v(F) ::
               F/head/form *= finite,
               F/syncat/first/cat *= np,
               F/syncat/rest/first/cat *= vp,
               F/syncat/rest/first/head/form *= infinitival,
               F/syncat/rest/first/syncat/rest *= end,
               F/syncat/rest/first/syncat/first *= F/syncat/first,
               F/syncat/rest/rest/first/cat *= np,
               F/syncat/rest/rest/first/head/agreement/number *= singular,
               F/syncat/rest/rest/first/head/agreement/person *= third,
               F/syncat/rest/rest/rest *= end.
%-----------------------------------------------------------------------
to        lex  v(F) ::
               F/head/form *= infinitival,
               F/syncat/first/cat *= vp,
               F/syncat/first/head/form *= nonfinite,
               F/syncat/first/syncat/rest *= end,
               F/syncat/first/syncat/first *= F/syncat/rest/first,
               F/syncat/rest/first/cat *= np,
               F/syncat/rest/rest *= end.

% category labels ************************************************

category_label(F, C) :- F/cat *= C.

s(F)  :- category_label(F, s ).
np(F) :- category_label(F, np).
vp(F) :- category_label(F, vp).
v(F)  :- category_label(F, v ).
xp(F). % dummy label

% example sentences **********************************************

ex1(1, s, [uther, sleeps]).
ex1(2, s, [knights, sleep]).
ex1(3, s, [uther, storms, cornwall]).
ex1(4, s, [uther, has, stormed, cornwall]).
ex1(5, s, [knights, have, stormed, cornwall]).
ex1(6, s, [uther, persuades, knights, to, sleep]).
ex1(7, s, [uther, persuades, knights, to, storm, cornwall]).
```