

Univerzita Karlova v Praze  
Matematicko-fyzikální fakulta

## DIPLOMOVÁ PRÁCE



Josef Toman

### **Automatická anotace angličtiny na tektogramatické rovině**

Ústav formální a aplikované lingvistiky

Vedoucí diplomové práce: prof. RNDr. Jan Hajič, Dr.  
Studijní program: Informatika, Matematická lingvistika



Rád bych touto cestou poděkoval všem, kteří se nějakým způsobem zasloužili o vznik mé diplomové práce. Na prvním místě je zcela určitě prof. RNDr. Jan Hajič, Dr., vedoucí mé práce a ředitel Ústavu formální a aplikované lingvistiky, kterému vděčím za cenné rady a připomínky. I přes velké pracovní vytížení se se mnou ochotně dělil o své bohaté zkušenosti ve dne i v noci, v pracovně i na letišti. Také mi umožnil účastnit se letní školy v USA a několika zahraničních vědeckých konferencí, čehož si velmi vážím.

Hned na dalším místě musím zmínit své nejbližší spolupracovníky, se kterými se podílím na vzniku korpusu PEDT. Mezi nimi patří můj největší dík Mgr. Silvii Cinkové, bez které by tato práce nevznikla. V podobném duchu děkuji všem kolegům z ÚFALu za výjimečné pracovní prostředí, které je přátelské, inspirativní a stěží opakovatelné.

Jsem vděčný svým rodičům a celé rodině za zázemí a podporu, díky kterým se ze mě stal člověk, který právě píše poslední řádky své diplomové práce. Oceňuji pochopení své přítelkyně pro to, že jsem se jí v poslední době věnoval méně než jindy, a děkuji kamarádům za to, že život je s nimi veselejší a obsahuje i jiné věci, než jen přibývající stránky textu.

Závěrem bych chtěl vyjádřit své díky prof. Vojtěchu Jarníkovi za to, že před lety dokázal, že Lenin pochází z opice, což v konečném důsledku vedlo k založení matfyzu<sup>1</sup>, a já tak na této báječné škole mohl studovat.

Prohlašuji, že jsem svou diplomovou práci napsal samostatně a výhradně s použitím citovaných pramenů. Souhlasím se zapůjčováním práce.

V Praze dne 17. dubna 2009

Josef Toman

---

<sup>1</sup><http://www.ucw.cz/~michal/matfyzak.html>



# Obsah

<b>1</b>	<b>Úvod</b>	<b>1</b>
<b>2</b>	<b>PEDT</b>	<b>3</b>
2.1	Funkční generativní popis . . . . .	3
2.2	Pražský závislostní korpus – PDT . . . . .	4
2.3	Rozdíly mezi PEDT a PDT . . . . .	5
2.4	PEDT v širším kontextu . . . . .	6
<b>3</b>	<b>Zdroje dat</b>	<b>7</b>
3.1	Penn Treebank – Wall Street Journal . . . . .	7
3.2	Jmenné fráze . . . . .	9
3.3	BBN . . . . .	10
3.3.1	Pojmenované entity . . . . .	11
3.3.2	Textová koreference . . . . .	12
3.4	Nombank . . . . .	13
<b>4</b>	<b>Metodika evaluace</b>	<b>15</b>
4.1	Příprava dat pro testování . . . . .	15
4.2	Sledování prováděných akcí . . . . .	17
4.3	Párování . . . . .	18
4.4	Výpočet výsledků . . . . .	20
4.5	Popis tabulek . . . . .	21
<b>5</b>	<b>Automatická anotace při přípravě dat</b>	<b>23</b>
5.1	Hlava neterminálu . . . . .	23
5.2	Atribut nodetype . . . . .	24
5.3	Atribut <i>is_generated</i> . . . . .	25
5.4	Negace . . . . .	26
5.5	Stopy . . . . .	26
5.6	Interpunkce na konci vět . . . . .	27
5.7	Předložka u souřadných spojení . . . . .	28
5.8	\$600 a share . . . . .	29
5.9	„\$600 a share” v předložkové vazbě . . . . .	31
5.10	Parenthese jako apozice . . . . .	33
5.11	Apozice s vlastním jménem . . . . .	35
5.12	Vedlejší věta s nevyjádřeným aktantem . . . . .	35
5.13	Postponovaný přívlastek . . . . .	39
5.14	Číslovka ve formě přívlastku . . . . .	39
5.15	Číslovka jako součást složené adjektivní fráze . . . . .	42
5.16	Aktanty PAT a EFF u vybraných sloves . . . . .	43
5.17	Přímá řeč jako téma . . . . .	43

5.18	Identifikace aktantu u vybraných sloves . . . . .	45
5.19	Logický subjekt v pasivu . . . . .	47
5.20	Funktory MAT a APP po předložce „of“ . . . . .	48
5.21	Několik ustálených frází . . . . .	50
5.22	Dvouslovné souřadné spojky . . . . .	51
5.23	Vícenásobná koordinace . . . . .	55
5.24	Cizí názvy . . . . .	57
5.25	Boston, Massachusetts . . . . .	58
5.26	Funktor NE . . . . .	61
5.27	Rozvití pomocí pojmenované entity . . . . .	62
5.28	Čísla . . . . .	63
<b>6</b>	<b>Automatické kontroly při anotaci</b>	<b>69</b>
<b>7</b>	<b>Automatická anotace po odevzdání dat</b>	<b>71</b>
7.1	Doplňování koreference . . . . .	72
<b>8</b>	<b>Souhrnné vyhodnocení</b>	<b>75</b>
<b>9</b>	<b>Závěr</b>	<b>79</b>
	<b>Literatura</b>	<b>81</b>

Název práce: Automatická anotace angličtiny na tektogramatické rovině  
Autor: Josef Toman  
Katedra (ústav): Ústav formální a aplikované lingvistiky  
Vedoucí diplomové práce: prof. RNDr. Jan Hajič, Dr.  
e-mail vedoucího: hajic@ufal.mff.cuni.cz

Abstrakt: Tektogramatická rovina je velmi složitá a její anotace je náročná a nákladná. Na rozdíl od jiných korpusů je *Prague English Dependency Treebank* (PEDT) založen na datech, pro která již existuje syntaktická anotace, byť principiálně odlišná. Cílem práce je navrhnout a implementovat metody automatické anotace využívající dostupná data a vedoucí k minimalizaci úsilí vynaloženého na manuální anotaci. Důležité je kvalitní vyhodnocení, aby bylo možné ověřit přínos použitých metod. Vzniklo několik desítek modulů, které jsou zaměřeny na různé aspekty anotace. Analýza jejich činnosti je komplikovaná a vyžádala si vytvoření složitého systému, s jehož pomocí je možné provést velmi podrobný rozbor. Dosažené výsledky jsou pozitivní a vybízejí k pokračování v započaté práci a jejímu dalšímu rozšiřování.

Klíčová slova: PEDT automatická anotace evaluace

Title: Automatic annotation of English on the tectogrammatical level  
Author: Josef Toman  
Department: Institute of Formal and Applied Linguistics  
Supervisor: prof. RNDr. Jan Hajič, Dr.  
Supervisor's e-mail address: hajic@ufal.mff.cuni.cz

Abstract: Tectogrammatical layer is very complex and its annotation is difficult and expensive. Unlike other corpora, the *Prague English Dependency Treebank* (PEDT) is based on data for which there already exists a syntactic annotation, even though a fundamentally different one. The goal of this work is to propose and implement methods of automatic annotation that are using the available data and (preferably) would lead to minimization of the effort needed for a manual annotation. A high-quality evaluation is important so that the contribution of the used methods can be verified. Tens of modules, which focus on various aspects of annotation, were created. The analysis of their activity is complicated and required a complex system to be created. The analyses created with it are very detailed. The outcome is positive and urges to continue the work and extend it further.

Keywords: PEDT automatic annotation evaluation



# Kapitola 1

## Úvod

Základním stavebním kamenem každého lingvistického nástroje je bohatá a pestrá sbírka jazykových dat – korpus. Zdaleka přitom nemusí jít pouze o produkty založené na statistických metodách, které potřebují tzv. trénovací data. I autoři gramatických formalismů nebo systémů založených na pravidlech potřebují svůj výzkum opřít o perfektní znalost jazyka – tak podrobnou, že se sotva obejdou bez korpusu. I člověk nej-povolanější a nejvzdělanější, který zná každý jazykový jev a chápe všechny jeho funkce a důsledky, nakonec zjistí, že není schopen kvantifikace. Nedokáže posoudit, jak často se daný jev vyskytuje, případně které ze dvou vyjádření téhož je častější apod.<sup>1</sup>

Užitek plynoucí z korpusů je značný a několikanásobný. Stejná data mohou najít uplatnění jak v teoretické lingvistice, tak v aplikované informatice. Není tedy divu, že zejména v posledních letech vzniká stále více korpusů a rozsah jednotlivých exemplářů se zvětšuje. Tvorba korpusů je však složitá a nákladná (zejména těch ručně anotovaných), proto se vždy hledají cesty, jak usnadnit práci nebo snížit náklady.

Jedním z korpusů, které v současné době vznikají, je *Prague English Dependency Treebank* (PEDT), který má jednu zásadní specifickou vlastnost. Jde o novou anotaci dat z jiného, proslaveného korpusu, kterým je *Penn Treebank*. Díky tomu je možné hned od začátku využít předchozí anotaci, jenže kvůli zcela odlišnému pojetí obou korpusů to není snadný úkol.

Tato práce pojednává o využití automatických metod anotace pro usnadnění, urychlení, a tím pádem i zlevnění vývoje PEDT. Principem je přenos informace mezi naprosto odlišnými formalismy (složková vs. závislostní syntax). Hlavními součástmi práce jsou podrobné popisy jednotlivých metod a detailní vyhodnocení dosažených výsledků. Okrajovou součástí je několik kapitol, které popisují další způsoby, jak usnadnit manuální anotaci.

Nejde a ani nemůže jít o práci izolovaného jedince, což mimo jiné ovlivňuje volbu jazykové stylu, který vysvětlím hned v úvodu, aby čtenář nebyl zmaten. V pasážích, ve kterých budu popisovat svoji vlastní práci a rozhodnutí, budu používat první osobu jednotného čísla (*já*). První osoba množného čísla (*my*) bude vyhrazena pro odlišení činnosti celého týmu nebo jeho části a také pro pasáže, ve kterých budu vysvětlovat různé postupy. Vždy se budu snažit jednoznačně uvádět, co je má vlastní práce a co je výsledek úsilí celého týmu, případně úplně jiných lidí.

---

<sup>1</sup>A nebo si myslí, že to dokáže, ale pouze do té doby, než zažije překvapivé setkání s realitou.



# Kapitola 2

## PEDT

Cílem této kapitoly je seznámit čtenáře s projektem PEDT v kontextu příbuzných korpusů a s jeho historickým a teoretickým pozadím. Začnu krátkým popisem *funkčního generativního popisu* (FGD<sup>1</sup>) Petra Sgalla (viz knihy [19] a [20]). Na příkladu *pražského závislostního korpusu* (PDT<sup>2</sup>) (vydán ve verzích 1.0 [7] a 2.0 [8]) ukážu, které myšlenky FGD byly použity v praxi a jak. Dnes jde již o docela známý korpus, proto bude užitečnejší začít jeho stručným popisem a pokračovat výčtem věcí, kterými se PEDT od PDT liší. FGD představuje teoretický základ nejen pro PDT a PEDT, ale i pro ostatní pražské korpusy<sup>3</sup>.

### 2.1 Funkční generativní popis

Podle FGD je větu možné popsat pomocí systému několika hierarchicky propojených rovin. V původním návrhu je jich pět:

1. Tektogramatická rovina (hloubková syntax) – nejvyšší rovina
2. Větněčlenská rovina (povrchová syntax)
3. Morfématická rovina
4. Morfonologická rovina
5. Fonetická rovina – nejnižší rovina

Každá rovina popisuje větu z jiného hlediska. Důraz se postupně posunuje od zcela popisných veličin jako délka a znělost hlásky na fonetické rovině až po význam „skrývající se za slovy“ na rovině tektogramatické. Pro každou rovinu jsou definovány *elementární* a *komplexní* jednotky. Skrze relaci *kombinace* se elementární jednotky spojují do komplexních. Oba typy jednotek mohou být napříč rovinami (vždy pouze o jeden stupeň) propojeny pomocí relace *reprezentace*. Tím se dostávám ke *vztahu formy a funkce*, což je pojem často spojovaný s FGD. Jednotka z vyšší roviny představuje *funkci* jednotky z roviny nižší, ta je naopak *formou* svého výše položeného protějšku. Například funkcí akuzativu bývá předmět a formou předmětu bývá akuzativ (nebo infinitiv).

Zatímco na třech nižších rovinách vystačíme s obyčejnou posloupností jednotek seřazených podle pořadí ve větě, popisem na prvních dvou rovinách je závislostní

---

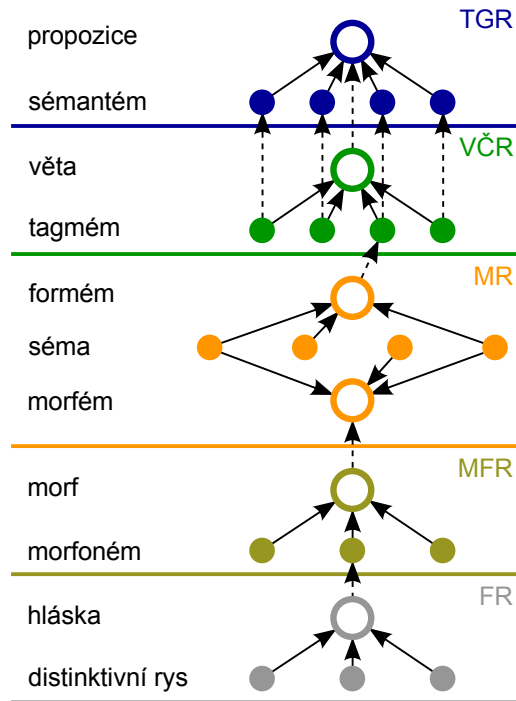
<sup>1</sup>Functional Generative Description

<sup>2</sup>Prague Dependency Treebank

<sup>3</sup>Např. PADT – *Prague Arabic Dependency Treebank*

strom, což vychází z tradice evropské lingvistiky, která preferuje závislostní syntax před složkovou, jež je oblíbená na druhé straně Atlantiku.

Schéma relací mezi rovinami je znázorněno na obrázku 2.1. Popis jednotlivých elementárních a komplexních složek lze nalézt v původních publikacích Petra Sgalla nebo stručně a výstižně v dizertaci Jana Štěpánka [24]. Dále jsem vynechal spoustu dalších důležitých vlastností FGD, např. valenci a aktuální členění.



Obrázek 2.1: Schéma relací mezi elementárními a komplexními jednotkami na pěti rovinách funkčního generativního popisu. Plné šipky znázorňují relaci kombinace, čárkované představují relaci reprezentace. Elementární jednotky jsou reprezentovány menšími plnými kroužky, komplexní jednotky jsou větší a nevyplněné.

## 2.2 Pražský závislostní korpus – PDT

Rané verze korpusu se začaly objevovat ke konci devadesátých let 20. století. Od té doby vývoj pokračoval až ke korpusu PDT 2.0, který byl vydán v roce 2006 [8]. Obsahuje přes milion slov, texty jsou tvořeny články, které vycházely v první polovině devadesátých let v několika českých novinách. Pro specifikaci datového formátu byl stejně jako u všech následujících korpusů použit jazyk PML [16].

PDT sice vychází z teorie funkčního generativního popisu, přináší ale spoustu změn. Věty jsou reprezentovány pouze třemi rovinami – morfologickou, analytickou a tektogramatickou. První z nich obsahuje prostý seznam slov s morfologickými informacemi (poziční morfologická značka). Na analytické a tektogramatické rovině je již věta reprezentována závislostním stromem.

Analytická rovina přibližně odpovídá větnečlenské rovině popsané ve FGD, od které se v několika důležitých věcech liší. Předně stále obsahuje všechna slova věty, přitom podle původního návrhu FGD některá slova (např. předložky) mizí již na morfématické rovině. Přítomnost předložek a jiných neplnovýznamových slov implikuje další rozdíly. Protože předložky (zůstaňme u tohoto příkladu) je potřeba ve stromu někam umístit, vznikají hrany, které nepředstavují skutečné závislosti. Slovo „postel“ ve frázi „pod

postelí“ by v analytickém stromě viselo na předložce „pod“, z hlediska FGD ale nejde o skutečnou závislost. Podobná situace nastává v případě souřadných spojek. Například ve slovním spojení „vidličky a nože nebo židličky a lože“ by řídicím uzlem výrazu byla spojka „nebo“, na ní by visely obě spojky „a“ a teprve na nich všechna substantiva.

Informace o větě zachycené na tektogramatické rovině se dají rozdělit do tří hlavních oblastí: struktura a funktoři [15], aktuální členění [23] a koreference [11]. Významnou vlastností struktury je, že některá slova (např. předložky, podřadící spojky, pomocná slovesa) již nejsou přítomna ve formě samostatných uzlů. Naproti tomu může tektogramatický strom obsahovat uzly, které se nedají promítnout do morfologické roviny, například „vygenerovaný“ uzel pro nevyjádřený podmět.<sup>4</sup> Stát se může i to, že se více tektogramatických uzlů vztahuje k jedinému slovu v povrchovém zápisu věty. Oblíbeným příkladem je výraz „červené a bílé víno“, ve kterém je elidováno jedno ze dvou vín. Pro každé z nich bude existovat samostatný uzel. Jiným příkladem je věta: „Pepa programoval v Pascalu a Petra v Pythonu.“<sup>5</sup> Uzel sám o sobě představuje složitou strukturu skládající se z desítek atributů, jejichž hodnoty mohou být jak atomické, tak různě strukturované.

Mezi další důležité prvky tektogramatické roviny patří rozličné koreferenční vztahy, které ve stromu představují jiný typ hran než závislostní, valenční rámce sloves a některých substantiv (viz valenční slovník PDT–VALLEX [9]) a také odkazy do analytické roviny.

Společným rozdílem analytické a tektogramatické roviny vůči FGD je technický kořen stromu. Kořenem každé věty je místo slovesa speciální uzel, který nepatří do reprezentované věty. Jde o čistě technické řešení některých implementačních problémů.

Kompletní popis tektogramatické roviny PDT stejně jako pokyny pro její anotaci obsahují výše zmíněné manuály [15], [23] a [11].

## 2.3 Rozdíly mezi PEDT a PDT

Největší odlišnost spočívá ve zdroji dat. PEDT je tektogramatická anotace ekonomických článků z novin *Wall Street Journal* (WSJ), konkrétně dat již dříve anotovaných v rámci korpusu *Penn Treebank* (PTB). I PEDT obsahuje tři roviny popisu věty (stejně jako PDT), ale místo morfologické roviny má PEDT frázovou rovinu, která obsahuje kompletní data PTB–WSJ (složkové stromy) převedená do formátu PML. Na frázové rovině jsou veškeré informace potřebné pro morfologickou rovinu, jak je definovaná v PDT, proto je možné vytvořit ji v případě potřeby konverzí.

Další významný rozdíl se týká analytické roviny, která na rozdíl od PDT není ručně anotovaná. Používá se pouze jako mezikrok při automatickém vytváření tektogramatické roviny. Ostatně anotace PTB–WSJ je minimálně srovnatelná s analytickou rovinou v podání PDT, takže analytická rovina v PEDT částečně ztrácí význam.

Přejděme k tektogramatické rovině. V PEDT se zatím vůbec neanotuje koreference ani aktuální členění, obojí je odloženo na později. Od jisté doby je vypuštěn atribut *is\_name\_of\_person*, protože jsme získali úplnou manuální anotaci pojmenovaných entit, která je podrobnější a hodnota atributu se z ní dá odvodit (podrobněji v části 3.3.1). Pro anotaci angličtiny je používán i funktoři DESCR (deskriptivní přívlastek), který v českých datech nenašel uplatnění. Valenční rámce jsou prozatím vyplňovány pouze u sloves. Pro tyto účely byl vytvořen slovník valenčních rámců anglických sloves *EngValLex* [2], [18], který stále podléhá postupnému vývoji.

<sup>4</sup>Tento příklad není úplně přesný. Pro správné vysvětlení bych však musel zabíhat hlouběji do teorie valence a aktantů, což je z hlediska této práce zbytečné.

<sup>5</sup>Ještě o něco zajímavější je znění: „Pepa programoval v Pascalu a Petra v pyžamu.“

Anotace PEDT je v neposlední řadě odlišná i z výlučně jazykových důvodů. V angličtině se například celá řada věcí vyjadřuje pomocí infinitivu („I am too young to marry“, „To get back to what I was saying, . . .“), což v češtině není možné. Pro tyto případy bylo nutné vytvořit nová anotační pravidla. Více viz manuál [3].

Odlišnosti způsobené přímo jazykem jsou však spíš rozšířením teorie o další jevy, které se dosud nevyskytly. Ostatně jedním z cílů PEDT je prokázat obecnou použitelnost FGD a způsobu anotace použitého v PDT. Zejména tektogramatická rovina byla od počátku navrhována s ohledem na pozdější aplikaci na jiné jazyky.

## 2.4 PEDT v širším kontextu

PEDT je „polovinou“ připravované nové verze PCEDT (*Prague Czech-English Dependency Corpus*), který byl poprvé publikován v roce 2004 [6]. Souběžně s PEDT se pracuje i na anotaci českého překladu vět z PTB–WSJ. Cílem je vytvořit paralelní česko-anglický korpus anotovaný na tektogramatické rovině a čítající více než milion slov, který by našel využití zejména při vývoji aplikací pro automatický překlad.

# Kapitola 3

## Zdroje dat

Každý anotovaný korpus musí začít s nějakými daty. Většinou jde o obyčejné texty, například novinové články. Jejich anotace začíná „na zelené louce“, v lepším případě ve stavu, kam je možné dostat se s dnešními parsery. V případě PEDT je výchozí situace odlišná. Výchozími daty jsou novinové články z *Wall Street Journalu* (WSJ), které již byly anotovány a tvoří základ korpusu *Penn Treebank* [12], [13]. Již na začátku tedy máme nejen texty, ale i kvalitní manuální anotaci. Problém je v tom, že Penn Treebank (PTB) obsahuje složkové stromy, zatímco PEDT je závislostní korpus, proto hotovou anotaci nelze využít úplně snadno – o tom je ostatně celá tato práce.

V právě začínající kapitole se ovšem soustředím pouze na základní technickou stránku věci. Vysvětlím, jakým způsobem vzniká první verze dat, která není ničím jiným, než pouhou kopií PTB–WSJ ve formátu PML,<sup>1</sup> a navážu popisem dalších částí přípravy dat, během kterých vznikají první verze analytické a tektogramatické roviny. To vše pouze přibližně, neboť je to spíše okrajové téma a z velké části nejde o moji práci.

PTB–WSJ je natolik proslavený a používaný,<sup>2</sup> že ve světě vznikají jeho stand-off anotace, které k původnímu korpusu dodávají další užitečně informace. Několik takových dodatečných zdrojů jsme využili a informace v nich obsažené připojili k datům získaných z PTB–WSJ. Jejich význam je různý. Šetří práci anotátorům tím, že něco již není třeba dělat vůbec, nebo tím, že anotátoři mají při práci více informací. A také rozšiřují možnosti automatické anotace.

Součástí této kapitoly proto budou kromě popisu PTB i informace o vedlejších datových zdrojích. Představím jednotlivé korpusy, popíši jejich formát a postup, jakým jsem data připojil k PEDT. Budu zacházet do větších podrobností, jelikož jde výhradně o moji práci a problematika již více souvisí s hlavním tématem.

### 3.1 Penn Treebank – Wall Street Journal

Korpus je distribuován v textovém formátu (viz příklad 5.1). K vyznačení syntaktických struktur slouží závorková notace. Stromy obsahují dva typy uzlů – *neterminály* a *terminály* (listy). Neterminálům se také dá říkat „fráze“, jejich hlavním atributem je právě značka pro frázi neterminálem reprezentovanou (kdo by neznal NP a VP). Dále v textu budu velmi často používat spojení „neterminál <fráze>“ jako zkratku za dlouhé a neobratné „neterminál, jehož atribut *phrase* je <fráze>“. Terminály reprezentují jednotlivá slova (přesněji tokeny – slova, interpunkce aj.) věty a jejich nejdůležitějším

<sup>1</sup>Výraz „formát PML“ není úplně správný, protože PML je jazyk pro popis formátu dat. Přesto ho budu pro jednoduchost používat.

<sup>2</sup>Ostatně NLP se občas žertovně přezdívá „Wall Street Journal Science“

atributem (když vynecháme samotnou povrchovou formu) je *tag* nebo česky *značka*. Podobně jako u neterminálů budu dále v textu používat spojení „terminál <tag>“ místo špatně použitelného přesného znění „terminál, jehož atributem *tag* je <tag>“.

Jeden z atributů neterminálů je *functions*. Slouží k upřesnění funkce neterminálu. Jde o atribut typu seznam, takže jeden neterminál je možné označit více funkcemi. Příkladem může být SBJ (subjekt) nebo LOC (příslovečné určení místa). Dále v textu budu používat slovní spojení „neterminál s *funkcí* FCE“ nebo jinou podobnou variantu místo přesnějšího „neterminál, jehož atribut *functions* obsahuje hodnotu FCE“. V případě tagů jako NN, NNS, NNP, NNPS nebo JJ, JJR, JJS budu občas pro označení celé skupiny používat pseudotag NN\*, respektive JJ\*. Znak hvězdičky tady zastupuje libovolný (i prázdný) řetězec.

Převod dat do formátu PML a výroba dalších dvou rovin probíhá v systému *TectoMT* [25], jehož „duchovním otcem“ je Zdeněk Žabokrtský a v současné době se na vývoji podílí ještě spousta dalších lidí. Jde o prostředí, které slouží primárně k překladu, ale možnosti jeho použití jsou zcela otevřené. Nejde o uživatelský software, ale o nástroj pro lingvistického výzkumníka–programátora. Systém používá svůj vlastní datový XML formát, který je definován pomocí PML a který je velmi blízký formátu dat v PEDT.<sup>3</sup> Zjednodušeně řečeno jde o bohatou kolekci různých drobných nástrojů nazývaných *bloky*, které je možné skládat do větších celků, a ty do ještě větších celků. Každý si tedy může sestavit vlastní nástroj přesně podle svých potřeb a chybějící věci vytvořit, existující přizpůsobit.

Ale zpět k tématu. Nejdříve ze všeho je zapotřebí vytvořit p–rovinu („p“ podle „phrasal layer“). Jde o přímou konverzi dvou formátů. Žádné informace se neztrácejí ani nejsou doplňovány. Nejsou potřeba žádné složité technologie. Postačují standardní nástroje jazyka *Perl*, zejména regulární výrazy. Příklad 3.1 ukazuje, jak takový převod vypadá.

---

**Příklad 3.1** Příklad převodu z PTB–WSJ do p–roviny PEDT. Je vidět, že původní zápis (první řádek příkladu) je mnohem úspornější.

---

(NP (NNP Pierre) (NNP Vinken) )

```
<nonterminal id="EnglishP-wsj_0001-s1-n3">
  <phrase>NP</phrase>
  <children>
    <terminal id="EnglishP-wsj_0001-s1-t1">
      <form>Pierre</form>
      <tag>NNP</tag>
    </terminal>
    <terminal id="EnglishP-wsj_0001-s1-t2">
      <form>Vinken</form>
      <tag>NNP</tag>
    </terminal>
  </children>
</nonterminal>
```

---

Po dokončení převodu je nad p–rovinou pomocí necelé desítky bloků vytvořena analytická rovina. Hlavní důraz je kladen na lemmatizaci slovních forem a strukturu závislostního stromu. Analytické funkce nejsou až na výjimky určovány. Příprava tek-

---

<sup>3</sup>Dokonce by se s trochou nadsázky dalo říci, že odlišnost formátů PEDT a *TectoMT* je způsobena „historickým nedorozuměním“. Datový formát PEDT byl založen na starším korpusu PDT. Teprve potom se začalo pořádně pracovat na *TectoMT*. Pokud by se tyto dvě věci udály v opačném pořadí, je možné, že by formát dat byl stejný.

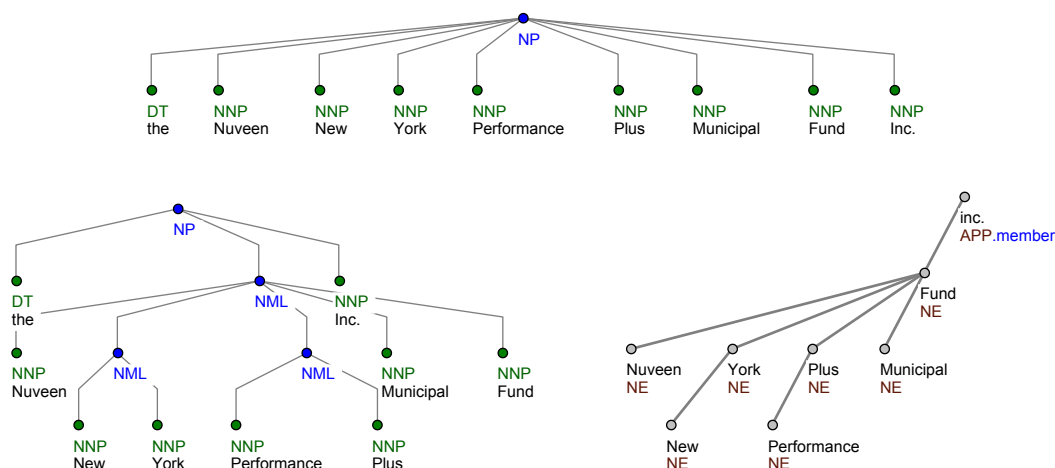
togramatické roviny je mnohem náročnější. Výmluvný je už jen počet bloků, který se v současné době pohybuje kolem třicítiky. Náplň jejich práce je různorodá. Je třeba rozhodnout, ze kterých analytických uzlů vzniknou t–uzly a ze kterých se stanou pouhé reference, a v návaznosti na tyto změny upravit strukturu vznikajícího tektogramatického stromu. Je třeba vyplnit základní atributy a spojit uzly na t–rovině i mezi rovinami různými typy relací. Více se o povaze jednotlivých bloků a vůbec celém procesu vytváření tektogramatických stromů systémem *TectoMT* píše v článcích [25] a [4].

Výstupem této fáze přípravy dat jsou první verze všech tří rovin: frázové, analytické i tektogramatické. Zatímco první dvě se už nikdy nebudou měnit, tektogramatickou čeká ještě spousta změn. V tuto chvíli ještě zdaleka není správně, ale má mnohem blíže k zamýšlené konečné podobě než k prosté sekvenci uzlů, se kterou by anotátoři museli začínat, kdyby neexistovalo žádné předzpracování.<sup>4</sup>

## 3.2 Jmenné fráze

Jedním z hlavních problémů PTB–WSJ je nedostatečná anotace složitých jmených frází. Například výraz „the Nuveen New York Performance Plus Municipal Fund Inc.“ je anotován jako prostá posloupnost slov bez další vnitřní struktury. To ovšem neznamená, že žádnou nemá. Úkolem našich anotátorů je odhalit jednotlivé závislosti a zachytit je na tektogramatické rovině.

Naším cílem bylo vyřešit anotaci „plochých“ jmených frází automaticky, jenže se ukázalo, že to vůbec není lehký úkol. Jisté pokusy proběhly, výsledek se začal rýsovat a po nějaké době jsem skutečně dokončil modul, ve kterém jsem implementoval přibližně desítku různých pravidel. Výsledky nebyly dokonalé, ale přesto přínosné.



Obrázek 3.1: „Plochá“ jmená fráze: původní PTB–WSJ (nahore), stav po přidání anotace jmených frází (dole vlevo) a tektogramatická struktura vygenerovaná z vylepšených dat (dole vpravo). Text fráze zní: „the Nuveen New York Performance Plus Municipal Fund Inc.“

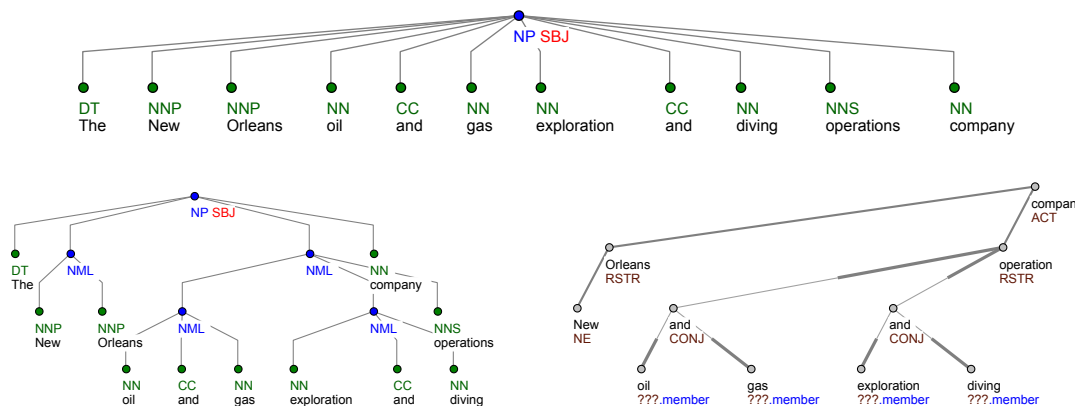
O několik dnů později jsme zjistili, že nedostatečná anotace jmených frází, kterou jsme se snažili řešit automaticky pomocí sady pravidel, je již vyřešena kompletní

<sup>4</sup>Mám na mysli předzpracování ve smyslu anotace, protože i pouhé vytvoření uzlů ze slov lze považovat za předzpracování. Začínat úplně od nuly by pro anotátora znamenalo, že by znal pouze text věty a celý strom (včetně uzlů) by musel vytvořit ručně.

manuální anotací [22]. David Vadas zvolil vcelku sympatický přístup. Jeho anotace do již existujícího stromu přidává další neterminály<sup>5</sup> dvojího druhu: NML pro substantivní fráze a JJP pro adjektivní fráze. Veškeré informace obsažené v původním PTB–WSJ zůstávají nezměněny. Jde o rozšíření, které pouze přidává nové informace. Jmenné fráze tak získávají svoji strukturu (více v [21]). Rozdíl v anotacích výrazu uvedeného výše je dobře patrný na obrázku 3.1.

Spojení anotace „plochých“ jmenných frází s PTB–WSJ je výjimečné tím, že je velmi snadné. Data samotná jsou téměř nečitelná, ale autor společně s nimi distribuuje i skript,<sup>6</sup> který všechnu práci udělá sám. Samozřejmě je nutné mít k dispozici originální Penn Treebank. Výsledkem jsou data obohacená o fráze NML a JJP, která se syntaxí nijak neliší od původních. Proto jsem mohl použít existující nástroje pro konverzi PTB do p–roviny zcela beze změny.

Přínos byl ovšem ohromný. Dá se říci, že úplně zadarmo jsme získali mnohem lepší automatickou anotaci složitých jmenných frází, a to včetně koordinací (viz obrázek 3.2), které také představovaly značný problém. Sebelepší automatická anotace založená na pravidlech se té manuální nemůže vyrovnat.



Obrázek 3.2: „Plochá“ jmenná fráze s vnořenými souřadnými strukturami: původní PTB–WSJ (nahore), stav po přidání anotace jmenných frází (dole vlevo) a tektogrammatická struktura vygenerovaná z vylepšených dat (dole vpravo). Text fráze zní: „The New Orleans oil and gas exploration and diving operations company“

### 3.3 BBN

Velmi cenným objevem byl korpus BBN [5]. Ve skutečnosti jde spíše o dva korpusy, neboť BBN obsahuje anotaci koreference a pojmenovaných entit. Obě součásti jsou na sobě zcela nezávislé a není mezi nimi žádná vazba. Jediným společným znakem je to, že jde o „stand-off“ anotace PTB–WSJ.

Tyto anotace jsou přínosné hned z několika důvodů. Dodatečné informace mohou pomoci anotátorům při práci. Ve velmi složitých větách, jejichž plné pochopení vyžaduje znalosti místních zvyklostí, kultury a reálií, docela pomůže, když člověk ví, že za nějakým neznámým názvem se skrývá například most. Stejně přínosná může být znalost jednotlivých antecedentů ve větách, ve kterých se vyskytuje příliš mnoho zájmen.

<sup>5</sup>Také by se dalo říct, že přidává další závorky do textové reprezentace dat

<sup>6</sup>Pro jeho spuštění je nutné mít interpret jazyka Python

	ENAMEX	NUMEX	TIMEX	Celkem
Počet výskytů	121 003	29 120	21 739	171 862
Počet typů	87	12	6	105

Tabulka 3.1: Počet výskytů a typů tagů pro pojmenované entity

Typ	Počet	Typ	Počet
PER_DESC	26 352	MONEY	11 097
ORGANIZATION:CORPORATION	23 441	CARDINAL	10 321
DATE:DATE	16 123	PERCENT	5 976
ORG_DESC:CORPORATION	15 186	GPE:CITY	5 602
PERSON	13 751	GPE:COUNTRY	5 079

Tabulka 3.2: Deset nejčastějších typů tagů pro pojmenované entity

Na anotaci pojmenovaných entit je založeno několik pravidel pro automatickou anotaci (části 5.24, 5.25, 5.26 a 5.27). A anotace koreference se hodí mimo jiné i proto, že ji teď nemusí dělat naši anotátoři.

### 3.3.1 Pojmenované entity

První částí BBN korpusu je anotace pojmenovaných entit. Data jsou zapsána v SGML formátu a rozdělena do textových souborů. Formát je již na první pohled srozumitelný a snadno čitelný (viz příklad 3.2). Každý element DOC obsahuje vazbu na PTB soubor (element DOCNO) a jednotlivé věty ve formě textu, do kterých jsou vloženy tagy označující pojmenované entity.

---

#### Příklad 3.2 Příklad z BBN korpusu - pojmenované entity

---

```
<DOC>
<DOCNO> WSJ0001 </DOCNO>
  <ENAMEX TYPE="PERSON">Pierre Vinken</ENAMEX> , <TIMEX TYPE="DATE:AGE">61 years...
  Mr. <ENAMEX TYPE="PERSON">Vinken</ENAMEX> is...
</DOC>
```

---

Vypadá to, že na každém řádku je jedna věta, ale autoři zřejmě nebyli dostatečně důslední. Občas se stává, že na jednom řádku je více vět nebo že jedna věta je zalomena a pokračuje na dalším řádku. Chyby tohoto typu naštěstí pouze zkomplikovaly práci s daty, na výsledek neměly vliv.

Pojmenované entity jsou označeny třemi typy tagů: ENAMEX pro názvy, NUMEX pro číselné hodnoty a TIMEX pro označení času. Každý takový tag má navíc atribut TYPE, který blíže určuje druh entity. V PEDT je zaznamenána pouze hodnota atributu TYPE, protože pomocí ní lze jednoznačně určit i tag, pokud by to někdy bylo třeba. Sestavíme-li pro každý ze tří tagů množinu hodnot, kterých může nabývat atribut TYPE, dostaneme tři disjunktní množiny. Tabulka 3.1 ukazuje základní počty jednotlivých tagů, v tabulce 3.2 je seřazeno deset nejčastějších typů.

Poslední nevyřešenou otázkou je, zdali mohou být jednotlivé tagy zanořeny do sebe. Analýzou dat jsem zjistil, že ano, ale v celém korpusu k tomu došlo pouze dvakrát. Rozhodli jsme se, že vícenásobné značkování nebudeme uchovávat, neboť dané množství je zcela nedostatečné vzhledem k technickým problémům, které by jinak bylo nutné vyřešit. Pokud se nějaké slovo nachází v hlubší než první úrovni, je označeno nejbližším

tagem, vyšší úrovně se nezohledňují.<sup>7</sup>

Vlastní začlenění anotace pojmenovaných entit do PEDT je v principu snadné. Je třeba spárovat slova BBN korpusu<sup>8</sup> s terminály složkových stromů a zkopírovat tagy. Párování je v tomto případě triviální. Stačí, když čteme věty BBN korpusu slovo za slovem a odpovídajícím způsobem procházíme terminály (listy) složkového stromu zleva doprava. Skript samotný je o něco složitější, bylo třeba řešit různé technické detaily. V přepisování vět v BBN například nejsou zaznamenány stopy (traces). Hlavní myšlenka je ovšem jednoduchá.

Dosud jsem popsal přenos dat do p-roviny, která je přesnou kopií PTB-WSJ. Anotaci pojmenovaných entit však potřebujeme i na t-rovině, takže ještě dochází ke kopírování do odpovídajících tektogramatických uzlů, existují-li.

Tímto způsobem jsme získali ruční anotaci pojmenovaných entit 269 504 terminálů ve složkových stromech (21,51 %) a 260 310 uzlů tektogramatické roviny (28,25 %<sup>9</sup>).

### 3.3.2 Textová koreference

Druhou částí BBN korpusu je ruční anotace koreference. Data se skládají ze dvou souborů. V jednom je prosté znění vět, ve druhém je úsporným a srozumitelným způsobem zachycena koreference (viz příklad 3.3). Pro naše potřeby byl užitečný jen druhý z nich, takže až do konce této části budu výrazem „data“ označovat pouze tento soubor.

---

#### Příklad 3.3 Příklad z BBN korpusu - koreference

---

(WSJ0176

```
(
  Antecedent -> S1:1-2 -> Xerox Corp.
  Pronoun -> S1:7-7 -> its
  Pronoun -> S1:15-15 -> it
)
(
  Antecedent -> S2:1-6 -> A~spokeswoman for Crum & Forster
  Pronoun -> S3:1-1 -> She
)
)
```

---

Základní jednotka dat je vždy tvořena antecedentem a koreferujícím zájmenem (příp. zájmeny). Tyto jednotky jsou seskupeny podle souborů, ve kterých se nalézají v PTB.

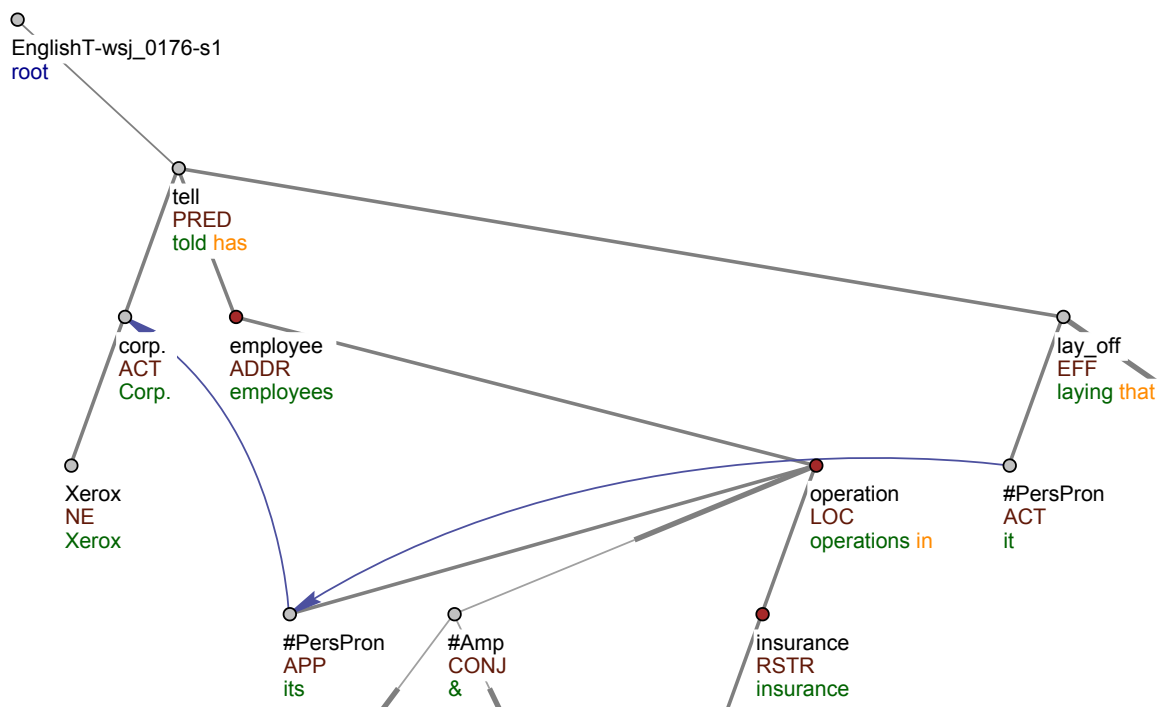
Pokud odhlédneme od obvyklých problémů, jako jsou stopy a nekompatibilní indexování, je zabudování koreference do PEDT stejně snadné jako v případě pojmenovaných entit. Zcela postačuje, když data procházíme lineárně. Když narazíme na antecedent, identifikujeme příslušný uzel na t-rovině. To není úplně samozřejmá záležitost, protože antecedentem je často rozsáhlý výraz, ale na druhou stranu je to stále vcelku mechanická záležitost. V případě koreferujícího zájmena opět vyhledáme odpovídající uzel a zaznameneáme u něj referenci na předchozí uzel (v rámci uzlů uvedených v BBN anotaci). Pokud je jeden antecedent koreferován více zájmeny, tak všechny reference nemíří k antecedentu, ale místo toho tvoří „řetízek“ (viz Obrázek 3.3).

<sup>7</sup>Jinak řečeno – slovu je přiřazen tag z vrcholu zásobníku

<sup>8</sup>Slovy zde rozumím spíše tokeny, takže i interpunkci, číselné výrazy apod.

<sup>9</sup>Procentuální podíl jsem pro větší přesnost určil pouze na základě již anotovaných dat. Počet t-uzlů se během anotace mění, takže výsledek výpočtu procentuálního podílu ve všech (tedy i neanotovaných) datech by byl zavádějící.

Výsledkem je 15 768 antecedentů (1,55 % všech tektogramatických uzlů<sup>10</sup>) a 22 243 koreferujících zájmen.



Obrázek 3.3: Textová koreference získaná z BBN korpusu (modré šipky)

### 3.4 Nombank

Posledním korpusem, který jsem zabudoval do PEDT, je Nombank [14]. Jednoduše řečeno jde o anotaci valence substantiv. Nombank vzniká v koordinaci s projektem Propbank [17], na kterém je založen EngValLex [2], [18], což je anglický protějšek valenčního slovníku PDT-VALLEX [9] používaného v PDT. Všechny tyto datové zdroje jsou založeny na myšlence valenčních rámců. V rámci PEDT probíhá anotace valenčních rámců sloves a substantiva zatím zůstávala stranou. Jejich anotace získaná z Nombanku, která silně připomíná již používané valenční rámce sloves, by proto mohla přinést velký užitek.

Data, se kterými jsem pracoval, byla koncipována jako slovník a uložena v jediném ohromném souboru. Stejně jako v obvyklém slovníku jde o seznam slov obohacený o další informace, kterými mám v tomto případě na mysli seznam anotovaných výskytů daného výrazu v PTB-WSJ. U každého výskytu je uvedena příslušná věta, strojově zpracovatelná adresa slova v korpusu a seznam argumentů („valenční rámec“) (viz příklad 3.4).

Adresa uvádí PTB soubor, ve kterém se slovo nachází, dále číslo věty a pořadí odpovídajícího terminálu ve složkovém stromě. K datům jsem bohužel neměl žádnou technickou dokumentaci, takže jsem musel spoléhat na vlastní analýzu použitého formátu. Došel jsem k následujícím závěrům. Každý argument je uzavřen v kulatých závorkách a uvozen jedním z tagů ARG a REL. První označuje argumenty, druhý se používá pro slovo, o jehož výskyt se právě jedná. Za uvozující tagem následují dvojice „atribut–hodnota“. Jména atributů jsou N, STRINGS, H a F, vždy začínají dvojtečkou. Hodnoty

<sup>10</sup>Procentuální podíl jsem pro větší přesnost opět určil pouze na základě již anotovaných dat.

---

**Příklad 3.4** Zachycení jednoho výskytu slova „shower“ v nombanku a výsledná podoba v PEDT
 

---

Example: At his first school in the U.S. he was thought a little strange for shutting off open water taps and admonishing his schoolmates to take only brief ---> SHOWERS <--- .

Address: wsj/14/wsj\_1455.mrg 34 29

Example Sense: 2

Parts of the Proposition:

(ARG :N "0" :STRINGS ("his schoolmates"))

(ARG :N "SUPPORT" :STRINGS ("take"))

(REL :STRINGS ("showers"))

...

N = 0; STRINGS = "his schoolmates":::N = SUPPORT; STRINGS = "take"

---

jsou většinou v uvozovkách, pouze atribut **STRINGS** má svoji hodnotu navíc uzavřenu i v kulatých závorkách, nebo naopak nemá žádnou, což je reprezentováno slůvkem **NIL**. Pro naše potřeby uchovávám pouze informace o argumentech, a to v o něco lépe zpracovatelné podobě (viz příklad 3.4).

Datový soubor je skutečně obrovský, obsahuje přes milión a půl řádků textu. I s velkou výpočetní silou je třeba zvolit rozumný způsob, jak anotaci zpracovat a přenést do PEDT tak, aby časové nároky zůstaly v normě. Z různých důvodů je nevhodné zpracovávat data sekvenčně výskyt po výskytu. Metoda, kterou jsem nakonec implementoval, představuje kompromis mezi rychlostí, jednoduchostí a dostupnými technickými prostředky. Umožňuje pracovat s každým souborem korpusu samostatně a nezávisle na ostatních. V první fázi shromažďuji všechny záznamy v nombanku, které se týkají daného souboru, a již zpracované je ukládám pro pozdější použití. Ve druhé fázi procházím všechny uzly všech stromů v souboru a postupně do nich ukládám informace získané v první fázi.

Tento postup samozřejmě není optimální, protože celý nombank se čte tolikrát, kolik je v PEDT souborů<sup>11</sup>, ale je výhodný z hlediska paralelního zpracování. Ne vždy je potřeba pracovat s celým korpusem, častěji to bývá pouze malá podmnožina nebo jediný soubor.

Data z nombanku jsem úspěšně přenesl do 105 926 tektogramatických uzlů, což je přibližně 12,86 % celého PEDT. Prozatím nedošlo k žádnému pokusu využít získána data pro anotaci, jelikož věc není dostatečně prozkoumaná.

---

<sup>11</sup>Přesně 2312

# Kapitola 4

## Metodika evaluace

V hlavní části této práce (kapitola 5) uvádím mimo jiné i detailní vyhodnocení činnosti jednotlivých modulů. Ještě před tím ale musím podrobně popsat, jak jsem k výsledkům dospěl. To bude náplní této kapitoly. Rozdělil jsem ji do pěti částí. V první řeknu, jaká data jsem při testování použil (typ, původ, velikost). Další bude věnována způsobu, jakým jsem zaznamenával práci anotačních modulů. Třetí část bude o párování uzlů mezi dvěma stromy. Dále vysvětlím metodu výpočtu výsledků a kapitola uzavřu popisem tabulek, které se objevují v kapitole 5.

Měření, která jsem provedl, jsou velmi podrobná. Zaznamenal jsem každou jednotlivou akci, kterou mé moduly s daty provedly. Chtělo by se mi říci, že jsem dosáhl maximální možné přesnosti, ale to přece jen není pravda, jelikož vždy existuje prostor k dalšímu vylepšení. Na druhou stranu toto odvážné tvrzení v jistých ohledech není daleko od pravdy.

Výsledkem mého snažení je robustní modulární nástroj skládající se ze sady menších programů, pomocí kterého mohu při vynaložení minimálního úsilí provést velmi podrobnou analýzu činnosti libovolného modulu. Považuji ho za druhý nejvýznamnější výsledek mé práce, který bude dále používán pro potřeby PEDT.

### 4.1 Příprava dat pro testování

Při vývoji nějakého nástroje se data standardně dělí na tři skupiny. Největší z nich tvoří trénovací data, další dvě skupiny tvoří řádově menší testovací data. První slouží k testování výsledků při vývoji a pomocí druhé skupiny testovacích dat se změří konečný výsledek po skončení vývoje před vypuštěním nástroje do světa. Práce na PEDT má ale svá specifika, která standardní schéma nabourávají.

Navržená pravidla a skripty podle nich implementované pravděpodobně nikdy nepotkají jiná data než PTB–WSJ a PEDT. Jelikož se nejedná o statistické metody, ale pravidlové, tak pojem trénovacích dat ztrácí svůj smysl. Můžu za ně považovat buď celý korpus nebo nic. Pokud bych se přece jen chtěl držet standardní terminologie, tak asi nejpřesnější je zařadit celý korpus do první skupiny testovacích dat, používaných pro testy při vývoji.<sup>1</sup>

V souvislosti s nimi se často mluví o *přetrénování*. To je jev, kdy je nástroj při vývoji natolik uzpůsoben k tomu, aby podával co nejlepší výsledky na vývojových testovacích datech, že jeho výsledky na jakýchkoliv jiných datech jsou výrazně horší. V případě PEDT a modulech popisovaných v této části by tedy mělo být cílem co největší přetrénování ve smyslu maximalizace výsledků na známých datech, jelikož

---

<sup>1</sup>Tzv. *dtest–data* (*development test data*)

výsledky pro jiná data jsou zcela nevýznamné.

Pro ověření správnosti navržených pravidel jsou v každém případě potřeba nějaká správná data. To je bohužel trochu problém. Korpus v současné chvíli zdaleka není hotový. Jistá část dat již prošla manuální anotací a kontrolami<sup>2</sup>, ale stále nejde o finální produkt – v datech jsou pořád chyby, jejichž intenzita se u jednotlivých atributů liší. Přesto je to to nejlepší, co mohu mít. **Referenčními** daty pro vyhodnocování práce anotačních modulů tedy bude část korpusu, která již prošla manuální anotací. Čítá 14 194 vět (28,84 %) a 362 361 slov (28,92 %).

Kromě toho jsou zapotřebí data, která budou sloužit jako vstup pro anotační moduly. Pro tyto účely jsem tu samou množinu souborů, které jsem zahrnul do referenčních dat, znovu vygeneroval stejným způsobem, jakým se připravují data pro anotaci, ovšem vynechal jsem všechny anotační moduly, které jsem měl v úmyslu vyhodnocovat. Těmto datům budu říkat **testovací**.

Pomocí takto zvolených dat se snažím simulovat podmínky, ve kterých mají anotační moduly běžně pracovat. Jejich vstupem jsou vygenerovaná data, která posléze dostane anotátor. Modul má za úkol vylepšit anotaci, aby měl člověk méně práce. Po odevzdání hotové manuální anotace je pak možné zjistit, do jaké míry se to modulu povedlo. Pokud změny, které provedl, zůstaly zachovány, tak je lze považovat za správné.

Ve skutečnosti to ale tak jednoduché není. Tektogramatická rovina je velmi složitá a na anotaci některých hraničních jevů se neshodnou ani ti nejlepší anotátoři. Mezia-notátorská shoda například na funktorech kolísá okolo 80 %, u struktury se pohybuje kolem 90 %. Anotace je také teprve ve své první fázi a spousta věcí není zcela jasných. Některé jevy mohou být na tektogramatické rovině zachyceny různými způsoby a všechny jsou přibližně stejně správné. Rozhodnout o správnosti jednoho z nich je často možné pouze na základě explicitní definice, takže když se v datech vyskytuje málo zřetelná chyba, anotátor ji ve většině případů neopraví.

Konkrétně pro testování anotačních modulů z toho plyne jeden důsledek. Byli-li soubor referenčních dat ručně anotován dřív, než vznikl anotační modul, tak ho anotátor dostal beze změn, které by modul normálně udělal. Pokud modul opraví chyby, které anotátor nechal být, tak jeho opravy vypadají jako chybné, i když tomu tak ve skutečnosti není. Na druhou stranu, kdyby opravy modulem provedené byly v datech již před anotací, anotátor by je taky nechal být a neměnil je.

Tyto skutečnosti mě vedly k tomu, že jsem referenční a testovací data rozdělil na dvě disjunktivní skupiny. První z nich je tvořena daty, která byla ručně anotována až potom, co na ně byly aplikovány všechny anotační moduly popisované v této práci. Této skupině budu říkat **nová data**. Druhá skupina (**stará data**) je doplňkem první. Vyhodnocení na základě **nových dat** by mělo být mnohem průkaznější. Jejich vážným nedostatkem je, že jich je málo. Jde o pouhých 2 749 vět (5,59 %) a 70 075 slov (5,59 %). **Stará data** tím pádem obsahují 11 445 vět (23,26 %) a 292 286 slov (23,33 %).

Pro každý modul jsem tak získal dvě sady výsledků. Jedna je málo průkazná, protože je založena na datech, která nevznikla úplně korektním způsobem. Druhá je také málo průkazná, tentokrát kvůli nedostatečnému objemu dat. Výsledky jsem proto vždy interpretoval velmi opatrně a často nahlížel přímo do dat. Situace je to nepříjemná, ale nic se s tím nedá dělat.

Abych byl úplně přesný, tak se musím zmínit ještě o třetí sadě dat. Každý modul jsem spustil i na úplně celý korpus. V tomto případě je však možné sledovat pouze absolutní počet výskytů anotovaného jevu, protože správnost provedených operací není jak ověřit.

<sup>2</sup>3. dubna 2009 je ručně anotováno 30,36 % vět

## 4.2 Sledování prováděných akcí

Vyhodnocení může probíhat dvěma způsoby. Mohu určit počáteční relativní (procentuální) správnost testovacích dat v porovnání s referenčními a zjistit, jak se tato hodnota změní po aplikaci modulu. Výsledkem by bylo číslo pohybující se v malém intervalu okolo nuly, kladná hodnota by znamenala zlepšení. Nebo mohu zjistit procentuální správnost pouze těch hodnot, které modul skutečně ovlivnil. V takovém případě by výsledkem byl počet procent na celé své škále. Čím vyšší hodnota, tím lépe. Při použití druhého způsobu by bylo velmi vhodné znát i předchozí správnost.

Rozhodl jsem se pro druhou možnost. Její výhodou je, že zjistím naprosto přesně, co modul udělal, co z toho bylo správně a co špatně. Nevýhodou je, že pro tyto účely je třeba upravit vlastní moduly.

Moje analýza správnosti nepracuje přímo s daty v jejich standardní podobě, nýbrž s několika logy. Nejdůležitějším je záznam všech akcí, které modul při automatické anotaci provádí. Ten získávám úpravou každého analyzovaného modulu. Za každým příkazem, který nějakým způsobem ovlivňuje data, následuje příkaz, který vypisuje informace o této úpravě. Každý zásah do dat je zdokumentován jedním řádkem v logu. Každý řádek se skládá z následující údajů:

- Jednoznačný identifikátor tektogramatického uzlu – *id*
- Typ změny (např. `create_node`), nebo jméno změněného atributu (např. `parent`)
- Původní hodnota, má-li smysl ji uvádět<sup>3</sup>
- Nová hodnota, má-li smysl ji uvádět

Pro každý výskyt sledovaného jevu, jehož zpracování si může vyžádat více různých akcí, se navíc v logu vyskytne řádek s jediným slůvkem HIT. Příklad 4.1 obsahuje výpis zpracování jevu popsáno v části 5.25. Rozhodl jsem se zaznamenávat i „zbytečné“ činy, které nevedly k žádné změně. Příkladem může být snaha změnit funktor na takovou hodnotu, která je již nastavena. Domnívám se, že kromě prosté správnosti je zajímavé vědět i to, jak přínosné nebo naopak zbytečné jsou různé akce, které jsou automatickou anotací prováděny.

---

**Příklad 4.1** Záznam akcí provedených modulem z části 5.25. Následkem prvních čtyř akcí došlo ke skutečným změnám, poslední dvě akce jsou zbytečné, neboť data již byla v zamýšleném stavu.

---

```
EnglishT-wsj_0270-s2-t7 parent EnglishT-wsj_0270-s2-t10 EnglishT-wsj_0270-s2a3
EnglishT-wsj_0270-s2a3 functor ??? LOC
EnglishT-wsj_0270-s2-t7 functor RSTR PAR
EnglishT-wsj_0270-s2-t7 is_parenthesis 0 1
EnglishT-wsj_0270-s2-t7 auxrf_add a#EnglishA-wsj_0270-s2-t6 a#EnglishA-wsj_0270-s2-t6
EnglishT-wsj_0270-s2-t7 auxrf_add a#EnglishA-wsj_0270-s2-t8 a#EnglishA-wsj_0270-s2-t8
HIT
```

---

Druhým typem logu je výpis hodnot atributů z celých referenčních dat. Na rozdíl od záznamu akcí je výpis hodnot atributů společný pro všechny moduly. V souboru je pro každý uzel a každý atribut uveden jeden řádek obsahující následující údaje:

- Jednoznačný identifikátor tektogramatického uzlu – *id*

---

<sup>3</sup>Tento údaj nemá smysl například při vytváření nebo mazání t-uzlu

- Název atributu
- Hodnota atributu. Pokud atributem není atomická hodnota, ale seznam, potom je uvedena délka seznamu a za ní následuje výpis všech prvků.

K analýze chování modulu je nutný ještě třetí soubor obsahující informace o párování uzlů mezi testovacími a referenčními daty, ale o tom pojednává celá následující část.

## 4.3 Párování

Pokud chci porovnávat testovací a referenční data (pokud chci porovnávat jakákoliv data ve formě stromů), základem celé operace je vždy vzájemné porovnání uzlů. To znamená, že musím vědět, které dvojice porovnat. Ke každému uzlu testovacích dat je třeba najít odpovídající uzel v referenčních datech. U některých se to bohužel nemusí podařit, ty potom zůstanou nespárované a budou vyloučeny z následující analýzy.

Stanovil jsem, co je třeba udělat, zbývá vyřešit otázku, jak to udělat. Nejprve jsem si řekl, že v tomto případě přece nejde o žádné obecné párování dvou potenciálně velmi odlišných stromů. Obě verze dat vznikly stejným způsobem, uzly mají svá *id*, která vycházejí z *id* uzlů na *p*-rovině, která se nemění. Proč tedy uzly nepárovat triviálně podle stejných *id*?

První verzi párování jsem skutečně implementoval takto, ovšem záhy jsem rozeznal svůj omyl. Kromě občasných výpadků docházelo k velkým problémům při párování uzlů vytvořených anotacními moduly. Nové uzly vytvořené jindy než při počáteční výstavbě z analytické roviny (tj. později pouštěnými moduly nebo při manuální anotaci), dostávají *id* nahodile, např. podle pořadí, ve kterém vznikají. Vytvářet uzly při automatické anotaci ve stejném pořadí jako anotátor je samozřejmě zcela nemožné, a tak jsem svůj přístup k párování musel přehodnotit.

Stále jsem se držel myšlenky, že vyvíjet úplně obecné párování nebo si nějaké podobné někde opatřit je zbytečné, a navíc pravděpodobně i horší, než využít některé ze speciálních výhod, které jsem měl k dispozici. Tou největší je samozřejmě *p*-rovina neboli *PTB*–*WSJ*. Oba stromy určené ke spárování jsou dvě ne příliš odlišné anotace stejné věty, a navíc oba známým způsobem vznikly ze stejného složkového stromu. Zkusil jsem tedy použít párování využívající okliku přes *p*-rovinu. Tak mohou být spárovány i uzly, které sice nemají stejné *id*, ale jsou tektogramatickým protějškem shodného *p*-uzlu. Jde o účinnější postup než párování podle shodného *id*, vyřeší se tak více než 80 % všech uzlů. Číslo je to sice hezké, ale v tomto kontextu žalostně nízké.

Po této první fázi jsem opakovaně prováděl následující posloupnost kroků tak dlouho, dokud jsem v prvním kroku nacházel nějaké uzly.

1. Najít nespárované uzly, které tvoří pár
2. Odhalit důvod, proč nebyly spárovány
3. Navrhnout metodu, jak takové uzly párovat
4. Implementovat metodu a provést nové párování

Výsledkem jsou čtyři fáze párování, z nichž každá postupně ukrajuje různě velikou část dosud nespárovaných uzlů. Za „nultou“ fázi, kterou ani nepočítám, považuji spárování technických kořenů. To mohu udělat zcela bez obav, protože jedinou funkcí těchto uzlů je být kořenem stromu. Takový uzel je v každém stromě právě jeden, takže omyl je nemožný.

O první fázi jsem se již zmínil, ale rozvedu ji o něco podrobněji. Úplně na začátku procesu párování vyhledáme postupně pro každý  $t$ -uzel obou stromů jeho protějšek ve složkovém stromě, má-li nějaký, a k nalezenému  $p$ -uzlu poznamenejme, s jakým  $t$ -uzlem je spojen. Informace ukládáme pro oba stromy odděleně, abychom dokázali rozlišit, ke kterému stromu referovaný  $t$ -uzel patří. Může se stát, že jeden  $p$ -uzel je spojen s více  $t$ -uzly jednoho stromu, ale jeden  $t$ -uzel je vždy spojen nejvýše s jedním  $p$ -uzlem.

Vybereme pouze takové  $p$ -uzly, které jsou v obou stromech spojeny s právě jedním  $t$ -uzlem a příslušné  $t$ -uzly spárujeme. Tento postup se dá elegantně vyjádřit matematicky. Označme stromy ke spárování jako  $T_1, T_2$ . Pomocí množin přechodů od  $t$ -uzlů stromů  $T_1$  a  $T_2$  k  $p$ -uzlům lze definovat dvě zobrazení  $T_1^p$  a  $T_2^p$ . Zúžíme-li tato zobrazení na bijekce  $B_1^p$  a  $B_2^p$ , tak složené zobrazení  $(B_2^p)^{-1} \circ B_1^p$  je bijekce mezi stromy  $T_1$  a  $T_2$ , jinými slovy párování.

V dalších fázích již budeme využívat i některé atributy  $t$ -uzlů. Budou to *funktor* a *t-lemma*, příznaky *is\_member* a *is\_generated* a rodič. První čtyři atributy budeme triviálně testovat na rovnost. To nejde dělat s rodiči uzlů, proto budeme zjišťovat, zdali byly rodičovské uzly spárovány.

Pro druhou fázi párování zvolíme ty  $p$ -uzly, které v alespoň jednom stromu odpovídají více  $t$ -uzlům. Z každého takového  $p$ -uzlu získáme dvě skupiny  $t$ -uzlů, které je třeba spárovat. Nejprve porovnáme každý uzel z jedné skupiny s každým uzlem z druhé skupiny a pro každou dvojici určíme *podobnost*. Podobnost je počet shodných atributů. Dvojici  $t$ -uzlů s nejvyšší podobností spárujeme. Ze zbývajících kombinací uzlů opět vybereme dvojici s nejvyšší podobností a opět spárujeme. Takto postupujeme tak dlouho, dokud je na obou stranách alespoň jeden nespárovaný uzel a dokud existuje dvojice s nenulovou podobností. Nespárujeme uzly, které se neshodují ani v jednom atributu.

Ve zbývajících dvou fázích již nebudeme používat  $p$ -rovinu. Vždy sestavíme skupiny ke spárování ze všech dosud nespárovaných  $t$ -uzlů a budeme postupovat velmi podobně jako ve druhé fázi. Budeme určovat podobnost dvojic uzlů a ty s nejvyšší podobností spárujeme. Lišit se bude výpočet podobnosti a podmínky pro spárování.

Ve třetí fázi se budeme snažit spárovat uzly pomocí jejich již spárovaných rodičů. Povolíme pouze takové páry, které mají stejného rodiče a shodují se v alespoň jednom dalším atributu. Toho můžeme dosáhnout tak, že za shodné rodiče přidělíme ne jeden, ale např. deset podobnostních bodů, a pro spárování bude nutné dosažení alespoň jedenácti bodů.

Čtvrtá a poslední fáze je párování podle synů. Do výpočtu podobnosti již nebudeme zahrnovat rodiče. Uzly spárujeme, pokud se budou shodovat alespoň ve dvou z vyjmenovaných atributů a budou mít alespoň polovinu (zaokrouhлено nahoru) společných synů. Pokud mají dva uzly různý počet synů, vyžadujeme polovinu z menšího počtu.

V tuto chvíli párování končí, zbylé uzly považujeme za nespárovatelné. Většinou jde o různé generované uzly (listy), které se ve druhém stromě nevyskytují.

Bohužel zde nemohu uvést žádné regulérní vyhodnocení přesnosti popsané metody párování. Důvodem je to, že ji nemám jak otestovat. Nemám k dispozici žádné manuální párování stromů v PEDT. Musel bych ho udělat sám. Jenže v tom případě už bych zase nepotřeboval výše popsaný složitý postup, protože pro testování modulů bych samozřejmě použil ruční párování uzlů. Mohu se spolehnout pouze na to, že jsem prohlédl výsledky párování na několika desítkách stromů a neodhalil jsem chyby.

K dispozici mám jen výsledky jednoduchého experimentu. Zkusil jsem spárovat všechny stromy zahrnuté v *nových datech*. Referenční data jsem použil beze změny, na testovací data jsem aplikoval všechny moduly popsané v kapitole 5. Po těchto úpravách obsahovala testovací data 49 710 uzlů, referenční 49 669. Spárováno bylo 47 243. To

Fáze	0	1	2	3	4
Počet uzlů	2749 (5,82 %)	41 433 (87,70 %)	421 (0,89 %)	2616 (5,54 %)	24 (0,05 %)

Tabulka 4.1: Počet spárovaných uzlů v jednotlivých fázích párování

znamená, že nespárováno zůstalo 2 467 (4,96 %) uzlů testovacích dat a 2 426 (4,88 %) uzlů referenčních dat. V tabulce 4.1 uvádím počty spárovaných uzlů v jednotlivých fázích.

Při návrhu jednotlivých fází párování jsem se inspiroval programem Václava Klimeše (viz část 3.2.1 v [10]), který jsem dosud používal při měření mezianotátorské shody. Můj vlastní program vytvořený podle popisu výše pro potřeby této práce však na datech PEDT dosahuje lepších výsledků a bude dále používán pro potřeby projektu.

## 4.4 Výpočet výsledků

Výsledky samozřejmě počítá opět program, který jako vstup vyžaduje tři soubory popsané v části 4.2. Jeho práce je rozdělena do několika částí. Jako první jsou načteny údaje o párování uzlů ve formě uspořádaných dvojic *id*. Potom je přečten a zpracován log provedených akcí a informace zůstanou uloženy v různých datových strukturách.

V další fázi probíhá čtení výpisu hodnot atributů z referenčních dat. U každé hodnoty program kontroluje, jestli se jí týkala některá z provedených akcí. Pokud ano, tak vyhodnotí správnost akce a správnost hodnoty před akcí. Pro akci provedenou modulem rozlišují dvě různé chyby. Chyba prvního stupně je akce, která nastavila špatnou hodnotu, přičemž původní hodnota byla také mylná. Za chybu druhého stupně považují akci, při které dojde k přepsání správné hodnoty chybno.

Pro atomické hodnoty by tento popis byl dostatečný. Po vyhodnocení správnosti akce dojde k přičtení jedničky k příslušnému čítači. U atributů typu seznam je vyhodnocení složitější. Řekněme, že vyhodnocuji přidávání hodnot do seznamu atributu  $Q$ . K dispozici mám tyto informace:

- Seznam hodnot přidanych do seznamu během automatické anotace ( $S_1$ ).
- Seznam prvků seznamu  $S_1$ , které se v seznamu atributu  $Q$  vyskytovaly již před automatickou anotací ( $S_2, S_2 \subset S_1$ )
- Seznam prvků atributu  $Q$  uvedený v referenčních datech ( $S_r$ )

Pomocí množinových operací sestrojím několik dalších pomocných seznamů:

- $P_1 = S_1 \cap S_r$  – hodnoty, které byly do seznamu zařazeny správně
- $P_2 = S_2 \cap S_r$  – správně přidané hodnoty, které se však v seznamu atributu  $Q$  nacházely již před automatickou anotací
- $D_1 = S_1 \setminus S_2$  – hodnoty, které byly automatickou anotací zařazeny do seznamu atributu  $Q$  a dříve se v něm nenacházely
- $D_2 = D_1 \setminus S_r$  – chybně zařazené hodnoty, které se v seznamu atributu  $Q$  před automatickou anotací nenacházely
- $P_3 = D_1 \cap S_r$  – správně přidané hodnoty, které se nenacházely v původním seznamu před automatickou anotací

U atributů typu seznam nerozlišuji dva typy chyb tak, jako u atomických hodnot. To znamená, že po vyhodnocení akcí pro atribut typu seznam dojde k úpravě čtyř čítačů:

- Správné akce:  $+|P_1|$
- Chybné akce:  $+|S_1| - |P_1|$
- Správné hodnoty před automatickou anotací:  $+|P_2| + |D_2|$
- Chybné hodnoty před automatickou anotací:  $+|S_2| - |P_2| + |P_3|$

V případě odebrání hodnot z atributu typu seznam rozlišuji pouze správně a chybně odebrané. Pravidla automatické anotace totiž při odebrání hodnot ze seznamu nikdy nespécifikují konkrétní uzel. Vždy jde o akce, jejichž úkolem je odebrat danou hodnotu ze seznamů všech uzlů s výjimkou těch, které jsou explicitně vyjmenované. Podobně podrobné vyhodnocování jako u přidávání hodnot do seznamu by proto vedlo k obrovskému počtu správných akcí, které bych však ve skutečnosti neměl počítat. Byly by to akce odebrání takové hodnoty, která se v seznamu nevyskytuje ani před automatickou anotací, ani v referenčních datech, proto by byly vyhodnoceny jako správné. Takto zkreslené výsledky by byly k ničemu. Potom však nemá smysl určovat správnost před provedením akce, jelikož by šlo o pouhé kopie. Správnost před automatickou anotací by se rovnala chybovosti automatické anotace a naopak.

Předposlední část programu je primitivní. Počítá se v ní správnost vytváření a mazání uzlů. Je-li nově vytvořený uzel přítomen v referenčních datech (tj. byl spárován s nějakým uzlem z referenčních dat), jde o správnou akci. V opačném případě byl uzel vytvořen chybně. Při hodnocení správnosti mazání uzlů stačí postupovat zrcadlově.

Problém může nastat pouze ve chvíli, kdy nějaký modul zároveň vytváří i odstraňuje uzly. V takovém případě používám dvě sady párování: jednu pořízenou před spuštěním modulu a druhou vyhotovenou až po provedení změn. Správnost mazání uzlů ověřuji pomocí staršího párování, novější slouží pro ověření správnosti přidávání uzlů.

V poslední fázi své činnosti program vypisuje nashromážděné údaje. Jako první je oznámen počet výskytů anotovaného jevu (počet řádků se slovem HIT). Následuje údaj říkající, jaké množství uzlů bylo ovlivněno provedenými akcemi a kolik z nich bylo úspěšně spárováno. Další hodnoty jsou již vypisovány zvlášť pro každý sledovaný atribut. Jako první je vypsán počet akcí, které se daného atributu týkaly, a údaj říkající, v kolika případech šlo o skutečnou změnu. Dále následuje až pět různých hodnot: správné akce, chyby prvního druhu, chyby druhého druhu a počet hodnot, které byly správně/špatně již před automatickou anotací. Součet prvních tří hodnot se nemusí rovnat celkovému počtu akcí. Pokud nejsou spárovány všechny uzly, tak se může stát, že některé akce byly provedeny na nespárovaných uzlech. Tyto akce nemohou být vyhodnoceny. Součet prvních tří hodnot se ale vždy rovná součtu zbývajících dvou.

## 4.5 Popis tabulek

Údaje vypočtené metodou popsanou v části 4.4 uvádím v této práci formou přehledných tabulek. Jejich rozměry ovšem nedovolují použití zcela jednoznačných popisků. Proto popisu tabulek věnuji samostatnou část textu.

Tabulka má většinou dvě části. Jednu pro *nová data* a druhou pro *stará data*. Typ dat bývá uveden v záhlaví každé části. Na dalších řádcích jsou souhrnné údaje

o počtu výskytů sledovaného jevu, počtu uzlů zasažených automatickou anotací a počtu spárovaných uzlů.

Následují výsledky pro jednotlivé atributy, které jsou vždy uvedeny ve dvou řádcích. První obsahuje absolutní hodnoty, ve druhém je vyjádření v procentech. Je-li název atributu „Uzel“, potom podle písmene uvedeného v závorce jde o vytvoření nového uzlu (N – new) nebo odstranění uzlu (D – delete). Upřesňující údaj v závorce bývá uveden i u atributů typu seznam. Podle použitého písmene jde buď o přidání (A - add) nebo odebrání (D – delete) hodnoty ze seznamu.

Číselné údaje jsou uspořádány do sedmi sloupců. Nemá-li nějaký údaj smysl, je políčko prázdné. Pokud u atributu nerozlišují dva typy chyb, je celkový počet chyb uveden v políčku pro chyby prvního druhu.

1. **Akce** – počet akcí týkajících se daného atributu
2. **Změny** – kolik z uvedených akcí představovalo skutečnou změnu dat
3. **OK** – Počet správných akcí
4. **Err 1** – Počet chyb prvního stupně (nahrazení předchozí chyby chybou)
5. **Err 2** – Počet chyb druhého stupně (nahrazení správné hodnoty chybou)
6. **OK (B)** – Správnost před automatickou anotací (pouze pro hodnoty zasažené provedenými změnami)
7. **Err (B)** – Chybovost před automatickou anotací (pouze pro hodnoty zasažené provedenými změnami)

# Kapitola 5

## Automatická anotace při přípravě dat

V této kapitole popíši stěžejní část práce – moduly, které bezprostředně před manuální anotací zpracovávají data, která jsou výsledkem základního převodu PTB–WSJ → PEDT. Každému modulu nebo skupině podobných modulů bude věnována jedna podkapitola, jejímiž součástmi budou popis jazykového jevu/dat a vyhodnocení úspěšnosti (viz kapitola 4). Vyhledávání automaticky zpracovatelných jevů a prvotní návrhy pravidel jsou v drtivé většině prací Silvie Cinkové. Mým úkolem bylo navržená pravidla analyzovat, implementovat a na základě výsledků dále rozvíjet a upravovat. Výsledná podoba se proto může radikálně lišit od počátečního návrhu.

Oblast působnosti jednotlivých modulů je velmi široká. Některé pracují se složitými lingvistickými jevy, jiné jsou určeny k úpravám spíše technického rázu. Prováděné operace se mohou týkat listů závislostního stromu bez ohledu na kontext stejně jako rozsáhlých struktur na úrovni celých souvětí.

Jednotlivá pravidla, která jsem implementoval ve formě skriptů, se snažím uvádět v pořadí, v jakém byla aplikována na data. Pouze v případech, kdy na pořadí nezáleží, uvádím tématicky příbuzné části pohromadě.

Před dalším čtením důrazně doporučuji prostudování kapitoly 4 nebo alespoň části 4.5.

### 5.1 Hlava neterminálu

Ještě před samotnými moduly se musím krátce zastavit u pojmu „hlava neterminálu“, který velmi často používám v následujících částech, protože jeho význam není zcela intuitivní.

Je-li syntaktická struktura věty zachycena pomocí složkového stromu, je na první pohled viditelné členění věty do jednotlivých celků (frází), ale na rozdíl od závislostního stromu už není zřejmé, které slovo je v dané frázi nejdůležitější – tj. které slovo je *hlavou fráze*. Protože v následujícím textu při popisu složkové struktury mluvím spíše o neterminálech než o frázích (viz také část 3.1), tak místo „hlava fráze“ používám pojem „hlava neterminálu“.

Znalost hlavy neterminálu je potřebná pro převod složkového stromu na závislostní. Základní myšlenku ukážu na příkladu jednoduché věty, jejíž složkový strom je na obrázku 5.1. Věta se na nejvyšší úrovni skládá ze tří částí: jmenné fráze, slovesné fráze a závěrečné tečky. Jako *hlava*<sup>1</sup> je zvýrazněna slovesná fráze (neterminál VP). To

---

<sup>1</sup>V tomto případě jde spíše o označení podstromu, ve kterém se *hlava* nachází.

znamená, že kořenem závislostního stromu věty by mělo být nějaké slovo nacházející se hlouběji v podstromu, jehož kořenem je neterminál VP. V takovém případě je třeba hledat *hlavu* na stále nižších a nižších úrovních, dokud není nalezen příslušný terminál – tedy hned na další úrovni sloveso „rose“.

Na nalezený kořen je zapotřebí zavěsit zbývající části věty, přesněji opět jejich *hlavy*. V tomto konkrétním případě bude na slovese „rose“ viset substantivum „prices“, jelikož jde o hlavu jmenné fráze. Tento postup probíhá rekurzivně od kořene věty až k listům.

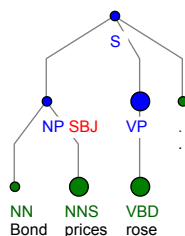
Každý terminál  $T_1$ , který není označen jako *hlava*, musí mít mezi sourozenci nějaký uzel (ať už terminál nebo neterminál), který obsahuje *hlavu*. V závislostním stromě potom  $T_1$  musí viset na nalezené *hlavě*. Například „bond“ na „prices“.

Takto popsany převod ze složkového stromu na závislostní je nutné brát s rezervou. Jde pouze o přibližné vysvětlení hlavních myšlenek. Pro potřeby této práce je důležité pouze to, jakým způsobem najít *hlavu neterminálu* ve složkových stromech PTB–WSJ.

Jedna z prvních fází zpracování systémem *TectoMT* má za úkol nastavit u některých uzlů složkového stromu atribut *is\_head*, jehož interpretace je „obsahuje *hlavu*“. Je-li příznak nastaven u terminálu, příslušný uzel je *hlavou*. U neterminálu tento příznak říká, že *hlava* se nachází někde v jeho podstromu. Pro každý neterminál platí, že právě jeden z jeho přímých potomků je označen příznakem *is\_head*.

Jak už napovídá název této části, *hlavu* hledáme pro zvolený neterminál  $U$ , k čemuž slouží následující rekurzivní algoritmus *NajdiHlavu*.

1. Je-li  $U$  terminál, vrať  $U$  a konec.
2. Najdi mezi přímými potomky  $U$  uzel  $H$ , který má nastaven příznak *is\_head*.
3. Vrať výsledek rekurzivního volání *NajdiHlavu*( $H$ ).



Obrázek 5.1: Složkový strom se zvýrazněnými *hlavami neterminálů* (věta „Bond prices rose.“)

## 5.2 Atribut nodetype

Anotační manuál (kapitola 2 v [15]) stanovuje jasná pravidla, podle nichž se z ostatních atributů uzlu dá odvodit hodnota atributu *nodetype*. Proto je přirozené, že tuto práci udělá počítač a ne člověk. Pravidla znějí takto:

- Kořen stromu je vždy *root*.
- Pokud je *funktorem* uzlu ATT, CM, INTF, MOD, PARTL, PREC nebo RHEM, hodnotou *nodetype* je *atom*.
- Pokud je *funktorem* uzlu CONJ, ADVS, CSQ, DISJ, GRAD, REAS, CONFR, CONTRA, OPER nebo APPS, hodnotou *nodetype* je *coap*.

- Pokud *t-lemma* uzlu je #Idph nebo #Forn, potom jde o list.
- Pokud je *funktořem* uzlu jeden z dvojice FPHR, DPHR, pak je stejná hodnota použita i jako *nodetype*
- Pokud je *t-lemma* uzlu #AsMuch, #Cor, #EmpVerb, #Equal, #Gen, #Oblfm, #QCor, #Rcp, #Some, #Total, #Unsp, #Amp, #Ast, #Percnt, #Bracket, #Comma, #Colon, #Dash, #Period, #Period3 nebo #Slash, hodnotou *nodetype* je qcomplex.
- Ve všech ostatních případech je požadovanou hodnotou complex.

Tato pravidla jsou konzistentní a úplná,<sup>2</sup> proto je snadné vytvořit podle nich modul, který automaticky vyplňuje správnou hodnotu *nodetype*. Jakékoliv vyhodnocování je zbytečné, protože modul hodnotu nastavuje u všech uzlů bez výjimky a vždy správně.

Oprávněnou námitkou může být to, že před manuální anotací nejsou některé atributy, na kterých záleží výběr pravidla, vyplněny správně. Z toho důvodu i stanovená hodnota *nodetype* může být špatná, a proto jsem do anotačního prostředí začlenil kontrolu, která po změně některého z relevantních atributů ihned opraví i *nodetype* stejným způsobem jako při přípravě dat. Tím je zaručeno, že přiřazená hodnota je vždy správná, aniž by byl potřeba zásah anotátora.

### 5.3 Atribut *is\_generated*

Podobně přímočará pravidla je možné stanovit i pro atribut *is\_generated*. Rozdíl je v tom, že tentokrát může výjimečně dojít k chybě a také zdaleka nepokryjí všechny uzly, neboť mě budou zajímat pouze ty, které mají jedno ze zástupných *t-lemmat* (přehled viz část 3.4 v [15]). Jelikož atribut *is\_generated* nabývá pouze dvou hodnot, i pravidla jsou dvě.

- Hodnotou je 0, pokud *t-lemma* uzlu je #Comma, #Colon, #Dash, #Slash, #Bracket, #Amp, #Percnt, #Ast, #Period, #Period3, #Semicolon, #PersPron, #VerbPron, #Rcp nebo #Neg
- Hodnotou je 1, pokud *t-lemma* uzlu je #Separ, #Cor, #QCor, #Gen, #Unsp, #Oblfm, #Some, #Equal, #Total, #Benef, #Forn, #Idph, #EmpNoun, #EmpVerb či #AsMuch

Vyhodnocení na *nových datech*<sup>3</sup> přineslo rozpačité výsledky. Pravidla byla uplatněna na 6 607 uzlů, nedošlo však k žádné skutečné změně. Vždy byla potvrzena předchozí hodnota. Při porovnání s ručně anotovanými daty bylo spárováno 5 006 uzlů (75,77 %) a správnost byla 92,69 %.

Po nějaké době jsem objevil příčinu. Během základního zpracování dat systémem *TectoMT* nevzniká ani jeden generovaný uzel. Ty se objevují až při přidělování valenčních rámců slovesům, když některý obligatorní člen rámce v datech chybí. Takové uzly mohou mít pouze dvě *t-lemmata* – #Gen, nebo #NewNode. Kromě těchto dvou se ve vygenerovaných stromech vyskytují ještě zástupná *t-lemmata* #PersPron, #Neg, #Percnt, #Comma, #Dash, #Semicolon, #Colon a #Amp, u kterých sice nelze úplně vyloučit generovanost, ale je to málo pravděpodobné. Odtud pramení vysoká správnost již před aplikací pravidel uvedených výše. Poměrně malá úspěšnost párování má podobný původ. Velkou část uzlů uměle vygenerovaných kvůli naplnění valenčního

<sup>2</sup>Což znamená, že si navzájem neodporují a popisují všechny možné situace.

<sup>3</sup>Výsledky na *starých datech* byly až malé odchylky totožné.

Nová data							
Výskyty	320						
Spárováno	320 z 320 (100,00 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
Funktor	320	320	301	19	0	0	320
		100,00 %	94,06 %	5,94 %	0,00 %	0,00 %	100,00 %
Stará data							
Výskyty	1 383						
Spárováno	1 253 z 1 383 (90,60 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
Funktor	1 383	1 383	1 161	92	0	0	1 253
		100,00 %	92,66 %	7,34 %	0,00 %	0,00 %	100,00 %

Tabulka 5.1: Negace: výsledky modulu

rámce anotátoři smažou, když identifikují skutečné členy rámce. Záměrně volím spíše přegenerování, protože smazání uzlu je mnohem snazší než tvoření nového. Většina chyb je způsobena špatnou manuální anotací koreferenčních uzlů.

Závěrem je, že tato pravidla sice pracují správně, ale za stávající situace jsou zcela zbytečná. Uplatnění by našla až ve chvíli, kdy by docházelo k odvážnějším pokusům vytvářet uzly se zástupnými *t-lemmaty*. Modul by se také dal přepracovat do podoby automatické kontroly při anotaci (viz kapitola 6).

## 5.4 Negace

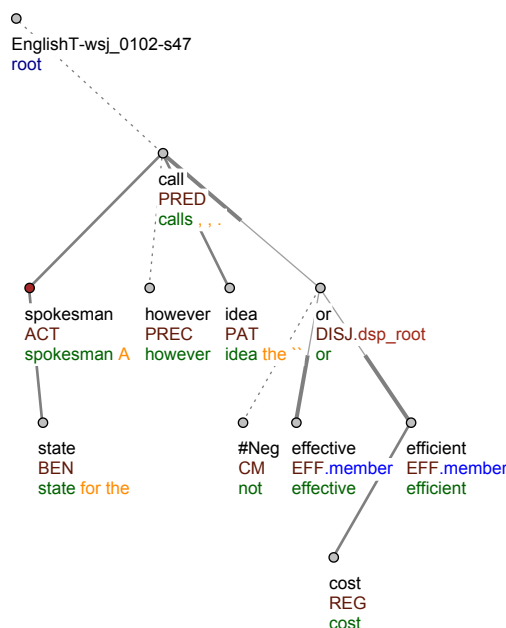
Existuje jedno snadné pravidlo. Syntaktická negace – uzel s *t-lemmatem* #Neg – má funktor RHEM. Jedna anotátorka zjistila, že vygenerovaná data se tímto pravidlem neřídí, proto vznikl triviální modul, který má situaci napravit. Výsledky jsou uvedeny v tabulce 5.1. Za nespárování uzlů a chyby mohou případy, kdy zápor (slovo „not“ a jeho varianty) nevystupuje v obvyklé roli negace, ale například jako přívlástek (viz Obrázek 5.2).

## 5.5 Stopy

Zajímavou vlastností PTB je, že obsahuje *stopy* (*traces*). Jde o terminály, které se objevují v místech, kde by měl být větný člen, ale z nějakého důvodu chybí. Může být nevyjádřený nebo uvedený na jiném místě ve větě (např. při tvoření otázky). Stopy navíc mohou být navázány na relevantní terminály pomocí atributů *index* a *coindex*. Jsou obdobou generovaných uzlů a dají se využít při automatické anotaci. Ukázka stopy je v příkladu 5.1.

Manuální anotace PEDT zatím se stopami nepočítá. Anotátoři si jich nevšímají, neexistují žádná pravidla. Při generování dat se stopy spíše z formálních důvodů přidávají do atributů *aux.rf*.

Po zavedení automatických anotačních kontrol (viz kapitola 6), které pouští sami anotátoři ještě před odevzdáním dat, se začaly množit připomínky, že je se stopami příliš mnoho práce. Často se stávalo, že určitá stopa figurovala v atributu *aux.rf* u více uzlů nebo o ní na tektogramatické rovině nebyl vůbec žádný záznam. Oba případy jsou v rozporu s pravidly anotace. Proto jsme došli k rozhodnutí, že stopy prozatím úplně odstavíme. Jelikož jejich současná anotace byla pouhým vedlejším produktem, a tím



Obrázek 5.2: Příklad negace, která není rematizátorem. Znění věty: *A spokesman for the state, however, calls the idea „not effective or cost efficient“.*

pádem úplně bezcenná, mohli jsme stopy na tektogramatické rovině bez obav zcela zrušit.

---

### Příklad 5.1 Stopa v PTB–WSJ na místě podmětu

---

```
(
(S~(ADVP-TMP (RB Now) )
(NP-SBJ-1 (PRP it) )
(VP (VBZ offers)
(NP (JJR richer) (NNS commissions) )
(S-PRP
(NP-SBJ (-NONE- *-1) )
(VP (TO to)
(VP (VB lure)
(NP
(NP (DT a) (NN broker) )
(NP-ADV (DT a) (NN week) ))))))
(. .) ))
```

---

Technické provedení je přímočaré. Smazal jsem všechny reference (atribut *aux.rf*), které v konečném důsledku směřují k p–uzlu s *tagem -NONE-*. Společně s úpravou dat jsem ještě pozměnil kontrolní moduly, aby si nevšímalý právě p–uzlů s *tagem -NONE-* a nevyžadovaly jejich zachycení na t–rovině. Vyhodnocení přesnosti je v tomto případě bezpředmětné. Uvedu pouze to, že z 1 253 013 terminálů v PTB–WSJ je 79 247 (6,32 %) tvořeno stopami.

## 5.6 Interpunkce na konci vět

Dle pravidla, které je spíše technické než lingvistické, se interpunkce ukončující větu uvádí v atributu *aux.rf* uzlu, jehož rodičem je technický kořen stromu. Jde o typický příklad jevu, který je snadný, jasný a snadno automaticky zpracovatelný, pro anotátora

Nová data							
Výskyty	2 537						
Spárováno	3 019 z 3 041 (99,28 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
aux.rf (D)	512		492	3			
			99,39 %	0,61 %			
aux.rf (A)	2 537	259	1 057	1 475		1 085	1 447
		10,21 %	41,75 %	58,25 %		42,85 %	57,15 %
Stará data							
Výskyty	10 594						
Spárováno	12 545 z 12 696 (98,81 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
aux.rf (D)	2 130		1 910	154			
			92,54 %	7,46 %			
aux.rf (A)	10 594	1 104	7 292	3 216		7 088	3 420
		10,42 %	69,39 %	30,61 %		67,45 %	32,55 %

Tabulka 5.2: Interpunkce na konci vět: výsledky modulu

však pracný až otravný. Příslušný program postupuje od konce věty směrem k začátku, dokud je povrchovou formou jeden ze znaků . ! ? ; : nebo ... (symbol tří teček). Pro každý takový token vyhledá příslušný analytický uzel a jeho referenci uvede v atributu *aux.rf* tektogramatického uzlu, který je přímým potomkem technického kořene věty. Pokud je takových uzlů víc,<sup>4</sup> použije se ten, který je nejvíce vlevo. Referenci je zároveň nutné odstranit ze všech ostatních uzlů, ve kterých se nachází.

Výsledky modulu jsou uvedeny v tabulce 5.2. Zejména výsledky získané na nových datech jsou ale velmi (až podezřele) špatné. Bohužel se projevil lidský faktor. Minimálně jedna z anotátovek vycházela z mylných informací a systematicky odstraňovala přesně ty *aux.rf* reference, které modul vytváří. Výsledky vytváření referencí naměřené na *nových datech* jsou proto úplně k ničemu.

U výsledků získaných ze *starých dat* je nutné volit velmi opatrnou interpretaci. Srovnání s hodnotami před použitím modulu však dokazuje, že i přes špatnou anotaci části dat se modul projevil kladně, jelikož celková správnost vzrostla. Jako přínos lze bez pochyb hodnotit odstraňování nadbytečných referencí.

## 5.7 Předložka u souřadných spojení

Pokud předložková fráze obsahuje více souřadně spojených jmenných či jiných frází, váže se předložka ke každé z nich, nikoliv ke koordinačnímu uzlu. Například ve frázi „se stolem a židlemi“ předložka významově patří ke stolu i židlím. Technicky je potřeba referenci na předložku uvést v atributu *aux.rf* u všech koordinovaných uzlů.

Použitý modul nejprve hledá níže popsanou strukturu ve složkovém stromě a při úspěchu náležitě upravuje tektogramatickou rovinu. Hledanou strukturou je předložková fráze, čili neterminál PP, který má právě dva syny. Vlevo terminál IN (předložka), vpravo fráze NP, která jako přímé potomky může obsahovat pouze terminály s *tagy* , , : , RB, CC a další fráze NP, které navíc musí být alespoň dvě, nesmí mít vyplněný atribut *functions* a jsou na ně kladeny i další podmínky:

- Je-li potomkem fráze (neterminál), musí to být PP

<sup>4</sup>To sice pravidla anotace zakazují, ale výstup TectoMT takto může vypadat.

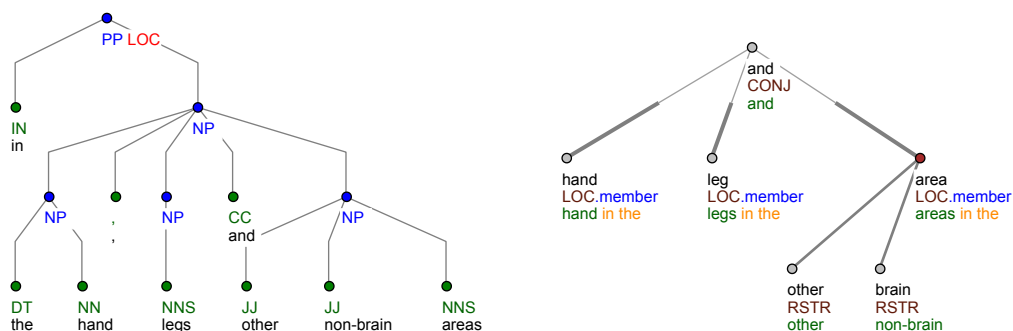
Nová data							
Výskyty	54						
Spárováno	119 z 119 (100,00 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
aux.rf (A)	119	20	94	25		74	45
		16,81 %	78,99 %	21,01 %		62,18 %	37,82 %
Stará data							
Výskyty	182						
Spárováno	386 z 386 (100,00 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
aux.rf (D)	2		2	0			
			100,00 %	0,00 %			
aux.rf (A)	384	91	297	87		252	132
		23,70 %	77,34 %	22,66 %		65,62 %	34,38 %

Tabulka 5.3: Předložka u souřadných spojení: výsledky modulu

- Je-li potomkem terminál, musí mít *tag* CD, RB, JJ\*, DT nebo NN\*
- Mezi potomky je právě jedna předložková fráze (PP) nebo právě jedno podstatné jméno (NN\*)

Při splnění všech podmínek si program zapamatuje všechny terminály NN\* z nejnižší úrovně popsané struktury, nalezne odpovídající tektogramatické uzly a všem do atributu *aux.rf* přidá referenci na předložku.

Příklad výchozí složkové struktury a výsledné podoby na tektogramatické rovině je na obrázku 5.3. Výsledky modulu jsou zachyceny v tabulce 5.3, celkový počet výskytů v celém korpusu je 849.



Obrázek 5.3: Předložka u souřadných spojení: struktura v PTB–WSJ a výsledek na tektogramatické rovině (výraz „in the hand, legs and other non-brain areas“)

## 5.8 \$600 a share

V PTB–WSJ se díky jeho úzké obsahové specializaci<sup>5</sup> velmi často vyskytují fráze typu „\$600 a share“. Použité číslo se samozřejmě případ od případu liší, stejně tak může být použito jiné slovo než akcie a také nemusí jít vždy o počet dolarů, ale například centů.

<sup>5</sup>Ekonomické články

Nová data							
Výskyty	43						
Spárováno	43 z 43 (100,00 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
Funktor	43	43	38	5	0	0	43
		100,00 %	88,37 %	11,63 %	0,00 %	0,00 %	100,00 %
Rodič	43	43	40	3	0	0	43
		100,00 %	93,02 %	6,98 %	0,00 %	0,00 %	100,00 %
Stará data							
Výskyty	195						
Spárováno	195 z 195 (100,00 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
Funktor	195	195	170	24	1	1	194
		100,00 %	87,18 %	12,31 %	0,51 %	0,51 %	99,49 %
Rodič	195	194	181	8	6	7	188
		99,49 %	92,82 %	4,10 %	3,08 %	3,59 %	96,41 %

Tabulka 5.4: \$600 a share: výsledky modulu

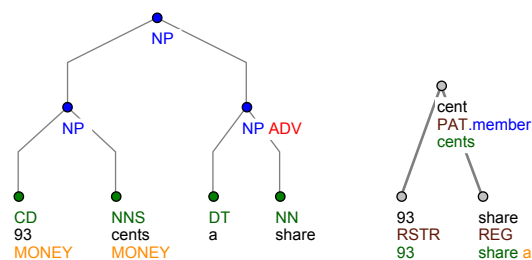
Z hlediska anotace je tento jev jednoduchý, proto se o něj stará modul, který pracuje takto:

Na začátku hledáme neterminál NP s následujícími vlastnostmi:

- jeho *funkce* je ADV
- levým bratrem je neterminál NP nebo QP
- má dva syny, v obou případech terminály
- levý syn je neurčitý člen (*tag* DT a *forma* „a“ nebo „an“)

Po nalezení popsané struktury vyhledáme tektogramatické uzly (hlavy, viz část 5.1) reprezentující hledaný neterminál NP (označme jako  $T_1$ ) a jeho levého bratra ( $T_2$ ). Uzlu  $T_1$  přidělíme funktor REG a zavěsíme ho na uzel  $T_2$ .

Výsledky práce tohoto modulu jsou velmi příznivé (viz tabulka 5.4). Konkrétní příklad je na obrázku 5.4. V celém korpusu existuje 1 702 výskytů tohoto jevu. Modul byl uveden jako příklad a podrobně popsán i v článku [4].



Obrázek 5.4: \$600 a share: struktura v PTB-WSJ a výsledek na tektogramatické rovině (výraz „93 cents a share“)

## 5.9 „\$600 a share” v předložkové vazbě

Jev popisovaný v této části se velmi podobá tomu, který byl popsán v části 5.8. Jde o použití fráze typu „\$600 a share“ v předložkové vazbě. Například „Kimberly-Clark closed at \$66.50 a share“. Na tektogramatické rovině dochází k tomu, že počítaná entita (\$, cent, ...) je rodičem uzlů vyjadřujících množství („600“) a zřetel („a share“). Předložka je uvedena v atributu *aux.rf* u rodiče.

Automatické zpracování však předložku systematicky umísťovalo špatně, někdy dokonce k více uzlům zároveň, proto jsem vytvořil modul, který tyto vazby vyhledává a opravuje. Ve složkovém stromu nás budou zajímat neterminály PP, které mají právě dva potomky. Levým z nich je terminál IN (předložka), pravým je libovolný neterminál, který má alespoň dva potomky, také neterminály. Jeden z nich musí vyjadřovat množství (neterminál  $N_1$ ) a druhý zřetel ( $N_2$ ).

Vlastnosti neterminálu  $N_1$ :

- Právě jeden terminál CD jako přímý potomek (množství)

Vlastnosti neterminálu  $N_2$ :

- Mezi jeho levými sourozenci je neterminál  $N_1$
- Právě dva přímí potomci, žádným z nich není terminál CD
- Levým synem je terminál DT (člen)
- Jeho *funkce* je ADV

Při vyhledávání struktury a ověřování jejích vlastností zároveň identifikujeme celkem pět uzlů – předložka, množství, člen, počítaná entita a zřetel. První tři z nich jsem již zmínil výše. Zřetelem je pravý potomek terminálu  $N_2$ , nezáleží na tom, zdali jde o terminál, či neterminál. Počítanou entitou je buď první potomek neterminálu  $N_1$ , pokud je to terminál zastupující slovo „\$“, nebo potomek úplně vpravo, opět bez rozlišení typu uzlu. Ve speciálních případech se může stát, že počítaná entita a množství splývají – tj. že samo vyjádření množství postačuje k zachycení významu.

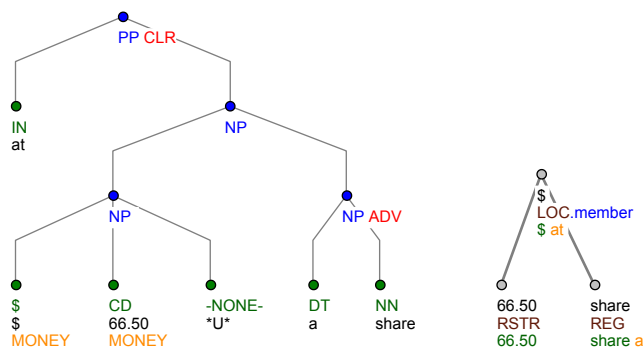
Ve druhé fázi se pokusíme identifikovat příslušnou pětici uzlů na tektogramatické rovině. Najdeme-li samostatné uzly pro předložku a člen, je třeba je smazat. Uzly vyjadřující množství a zřetel pověsíme na uzel počítané entity. Nikde nepřirazuje *funktor*, protože pro množství ho neumíme určit a zřetel má již přiřazen *funktor* REG modulem 5.8. Teď už zbývají pouze předložka a člen. První z nich patří do atributu *aux.rf* počítané entity, druhý ke zřeteli.

Výsledky modulu jsou uvedeny v tabulce 5.5, obrázek 5.5 uvádí příklady popsaných složkových a tektogramatických struktur.

Z výsledků je patrné, že bylo dosaženo významného vylepšení ve struktuře stromů (lepší určování rodiče). Výsledky u atributu *aux.rf* jsou sporné. U *nových dat* nedošlo k vůbec žádné chybě, kdežto u *starých dat* se situace po aplikování modulu ještě zhoršila. Důvodem je lidský faktor – nedůsledná manuální anotace. Po ručním zkontrolování několika chyb jsem zjistil, že se projevilo přesně to, co jsem očekával. Data většinou zůstala v takovém stavu, v jakém je anotátoři dostali, takže jsem zaznamenal vysokou správnost dat ještě před anotací (anotátoři udělali málo změn) a vysokou chybovost (i když modul pracuje správně, jeho akce jsou hodnoceny jako chybné, pokud anotátor nepracuje správně). Skutečná úspěšnost modulu je tak mnohem vyšší, ovšem za stávající situace nejjistitelná. Na závěr ještě dodám, že popsaný jev má v celém korpusu 325 výskytů.

Nová data							
Výskyty	10						
Spárováno	30 z 30 (100,00 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
aux.rf (D)	10		10	0			
			100,00 %	0,00 %			
Rodič	20	10	20	0	0	10	10
		50,00 %	100,00 %	0,00 %	0,00 %	50,00 %	50,00 %
aux.rf (A)	20	10	20	0		10	10
		50,00 %	100,00 %	0,00 %		50,00 %	50,00 %
Stará data							
Výskyty	89						
Spárováno	267 z 267 (100,00 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
aux.rf (D)	89		44	45			
			49,44 %	50,56 %			
Rodič	178	89	173	2	3	92	86
		50,00 %	97,19 %	1,12 %	1,69 %	51,69 %	48,31 %
aux.rf (A)	178	89	131	47		136	42
		50,00 %	73,60 %	26,40 %		76,40 %	23,60 %

Tabulka 5.5: „\$600 a share“ v předložkové vazbě: výsledky modulu



Obrázek 5.5: „\$600 a share“ v předložkové vazbě: struktura v PTB–WSJ a výsledek na tektogramatické rovině (výraz „at \$66.50 a share“)

## 5.10 Parenteze jako apozice

Parenteze, text uvedený zpravidla v závorkách (viz část 4.5 v [3]) má často funkci apozice. Navíc při nepřítomnosti jiné interpunkce nebo spojky funguje levá závorka nebo pomlčka jako koordinační člen. Základní zpracování dat tento jev nerozeznává, proto se mu nyní budu věnovat.

Jako obvykle musíme vycházet z PTB–WSJ. Začneme tím, že budeme hledat neterminál PRN (parenteze) s právě jedním levým sourozencem, kterým musí být neterminál NP (označme  $P_2$ ). Prvním přímým potomkem musí být terminál s *tagem* –LRB– nebo :. Neterminál PRN má kromě terminálů s *tagy* –LRB–, –RRB– a : pouze jednoho dalšího přímého potomka (označme  $P_1$ ). Uzel  $P_1$  nesmí být neterminálem S ani SBAR a jeho prvním přímým potomkem nesmí být žádný z terminálů IN, VBG, VBN, JJ\*. Textem v závorce také nesmí být slovesná fráze, která popisuje, že někdo nějakým způsobem vyjadřuje nějaký názor nebo myšlenku. V modulu je toto omezení implementováno zákazem současného výskytu následujících dvou jevů:

- Uzlem  $P_1$  je neterminál VP
- Mezi potomky uzlu  $P_1$  je terminál s některým z těchto *lemmat*: `say`, `ask`, `claim`, `add`, `tell`, `think`, `believe`

Identifikujeme-li ve složkovém stromě popsanou strukturu, začneme pracovat s tektogramatickou rovinou. Na začátku vyhledáme potřebné t–uzly. Jsou to dva uzly odpovídající hlavám neterminálů  $P_1$  a  $P_2$  (označme  $T_1$  a  $T_2$ ) a uzly pro levou a pravou závorku<sup>6</sup> (případně jiné znaky). Nejprve se musíme postarat o koordinační uzel. Tím je v tomto případě levý symbol ohraničující parentezi, který je výjimečně reprezentován samostatným uzlem (označme  $T_z$ ). Pokud není na t–rovině přítomen, vytvoříme ho a nastavíme tyto atributy: *nodetype* = `coap`, *funktor* = `APPS`. Pokud je *tagem* odpovídajícího p–uzlu –LRB–, nastavíme *t–lemma* na `#Bracket`, jinak na `#Dash`. Zbývá zapojení to stromu, tj. výběr rodiče. Tím je rodič uzlu  $T_2$ . Dále pokračujeme těmito kroky:

- Uzly  $T_1$  i  $T_2$  zavěsíme na  $T_z$  a oběma nastavíme příznak *is\_member*.
- Uzlu  $T_1$  přiřadíme stejný *funktor*, jaký má uzel  $T_2$ .
- Uzlu  $T_1$  i celému jeho podstromu<sup>7</sup> nastavíme příznak *is\_parenthesis*.
- Smažeme t–uzel reprezentující pravý symbol ohraničující parentezi, existuje-li.
- Opravíme informace v attributech *aux.rf*: Reference na pravý ohraničující symbol je uvedena právě u uzlu  $T_z$ . Reference na analytický protějšek uzlu  $T_z$  se pochopitelně nesmí vyskytovat nikde.

Výsledky modulu uvádím v tabulce 5.6. Některé atributy dopadly lépe, jiné hůře. Výsledky na *nových datech* jsou zřetelně lepší, zejména u *aux.rf* je rozdíl velmi vysoký. V celém korpusu jsem zaznamenal 715 výskytů popsané struktury. Příklad parenteze jako apozice ve složkovém a tektogramatickém stromě je na obrázku 5.6.

<sup>6</sup>Ve skutečnosti není moc pravděpodobné, že by tyto uzly na t–rovině byly. Většinou je třeba spokojit se s uzly z a–roviny, na jejichž základě potřebné t–uzly teprve vytvoříme.

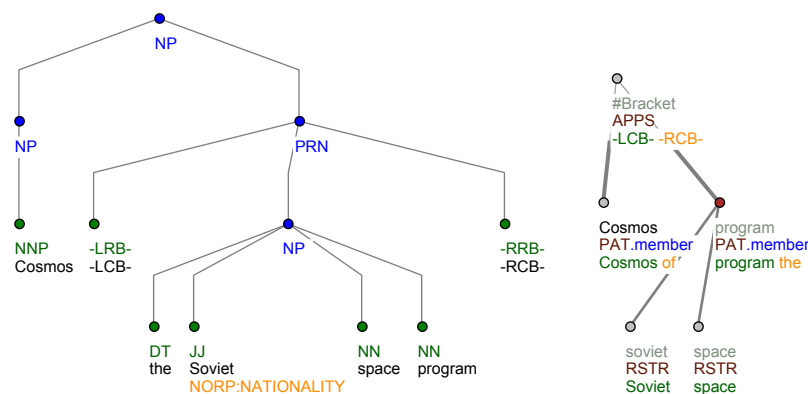
<sup>7</sup>Tím rozumíme všem uzlům vyskytujícím se v podstromu

Nová data							
Výskyty	37						
Spárováno	213 z 224 (95,09 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
aux.rf (D)	46		46	0			
			100,00 %	0,00 %			
Funktor	74	72	33	13	17	17	46
		97,30 %	52,38 %	20,63 %	26,98 %	26,98 %	73,02 %
Uzel (D)	13		13	0			
			100,00 %	0,00 %			
Nodetype	37	37	26	0	0	0	26
		100,00 %	100,00 %	0,00 %	0,00 %	0,00 %	100,00 %
T-lemma	37	23	26	0	0	13	13
		62,16 %	100,00 %	0,00 %	0,00 %	50,00 %	50,00 %
is_member	74	71	40	1	33	35	39
		95,95 %	54,05 %	1,35 %	44,59 %	47,30 %	52,70 %
Uzel (N)	23		13	10			
			56,52 %	43,48 %			
Rodič	97	83	49	22	16	27	60
		85,57 %	56,32 %	25,29 %	18,39 %	31,03 %	68,97 %
is_parenthesis	187	165	151	0	25	47	129
		88,24 %	85,80 %	0,00 %	14,20 %	26,70 %	73,30 %
aux.rf (A)	36	36	18	7		7	18
		100,00 %	72,00 %	28,00 %		28,00 %	72,00 %

Stará data							
Výskyty	143						
Spárováno	727 z 799 (90,99 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
aux.rf (D)	226		198	24			
			89,19 %	10,81 %			
Funktor	286	267	101	83	39	44	179
		93,36 %	45,29 %	37,22 %	17,49 %	19,73 %	80,27 %
Uzel (D)	28		28	0			
			100,00 %	0,00 %			
Nodetype	143	143	76	6	1	1	82
		100,00 %	91,57 %	7,23 %	1,20 %	1,20 %	98,80 %
T-lemma	143	107	48	35	0	28	55
		74,83 %	57,83 %	42,17 %	0,00 %	33,73 %	66,27 %
is_member	286	273	127	2	151	162	118
		95,45 %	45,36 %	0,71 %	53,93 %	57,86 %	42,14 %
Uzel (N)	107		58	49			
			54,21 %	45,79 %			
Rodič	393	357	157	95	83	107	228
		90,84 %	46,87 %	28,36 %	24,78 %	31,94 %	68,06 %
is_parenthesis	656	547	392	12	183	278	309
		83,38 %	66,78 %	2,04 %	31,18 %	47,36 %	52,64 %
aux.rf (A)	135	135	14	62		62	14
		100,00 %	18,42 %	81,58 %		81,58 %	18,42 %

Tabulka 5.6: Parenteze jako apozice: výsledky modulu



Obrázek 5.6: Parenteze jako apozice: struktura v PTB–WSJ a výsledek na tektogramatické rovině (výraz „*Cosmos (the Soviet space program)*“)

## 5.11 Apozice s vlastním jménem

Apozice typu „Mr. Smith, the best IA Manager“ patří k nejjednodušším příkladům tohoto jevu. Začíná vlastním jménem, za kterým následuje čárkami oddělené rozvití. Na tektogramatické rovině je tato apozice reprezentována koordinací. Kořenem souřadné struktury je čárka, členy koordinace jsou vlastní jméno a přístavek. Počet výskytů je bohatý, proto je namísto automatické zpracování.

Struktura této apozice je ve složkovém stromě jednoduchá. Hledáme neterminál NP s právě čtyřmi syny. Druhým a čtvrtým musí být čárka (neterminál s *formou* ,). Prvním synem je neterminál NP, který má mezi svými syny terminál s *tagem* NNP\*. Třetím synem je buď neterminál NP bez dalších požadavků nebo neterminál RRC (restricted relative clause), který mezi svými syny nemá terminál s *tagem* VP\*.

Na tektogramatické rovině nejdříve vyřešíme koordinační uzel, kterým je první ze dvou čárek. Pokud příslušný uzel neexistuje, vytvoříme ho. Jeho *funktořem* je APPS a *t-lemma* je #Comma. Dále najdeme hlavy obou neterminálů zmíněných při popisu složkové struktury. Jejich tektogramatické protějšky se stanou členy koordinace. Oba uzly zavěšíme na kořen koordinace a nastavíme jim příznak *is\_member*. Nakonec zbývají *aux.rf* reference. V kořeni koordinace bude odkaz na druhou čárku, která na *t*-rovině nemá vlastní uzel. Pokud na libovolnou z čárek existují jiné reference, odstraníme je.

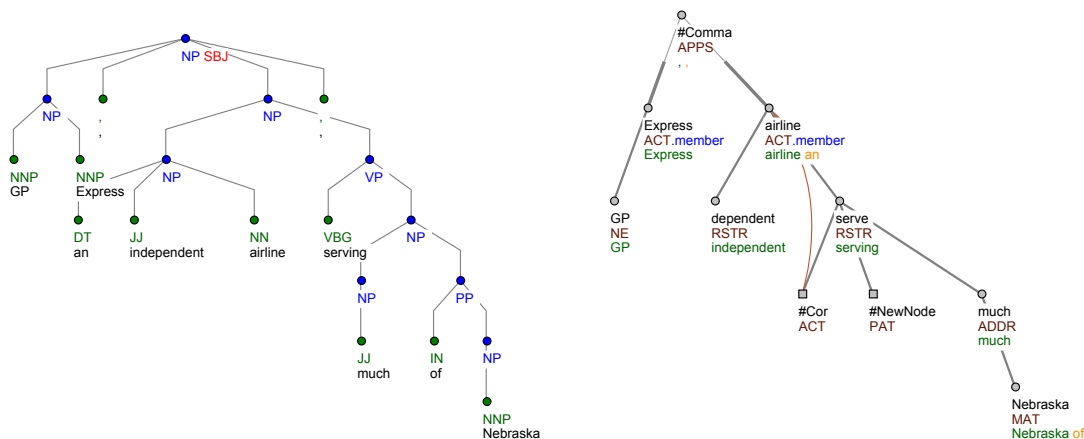
Výsledky modulu nejsou příliš přesvědčivé, po podrobnějším průzkumu bych je ale označil jako dobré. U většiny sledovaných atributů dochází ke zlepšení. Chyby jsou sice časté, ale po srovnání se stavem před použitím modulu je přesto vidět zlepšení. Výjimkou je atribut *aux.rf*. Stejně jako jinde jsou výsledky získané na *starých datech* stěží použitelné. U *nových dat* se opět potvrdilo, že anotátoři se řídili špatnými pokyny a reference spíše mazali. Tomu odpovídá vysoká správnost mazání a mizerný výsledek přidávání referencí. Prohlédl jsem několik výskytů a zjistil jsem, že i v případech, kdy skutečně šlo o popsanou apozici, anotátoři smazali správně určenou referenci. Přesná čísla jsou v tabulce 5.7, počet výskytů v celém korpusu je 2 379. Příklad apozice s vlastním jménem ve složkovém a tektogramatickém stromě je na obrázku 5.7.

## 5.12 Vedlejší věta s nevyjádřeným aktantem

Ve větách typu „Sb. did st., allowing st.“ nebo „Sb did st., driven by st.“ chybí ve vedlejší větě jeden z aktantů. Při použití gerundia je to aktor (ACT), v případě pasiva

Nová data							
<b>Výskyty</b>	94						
<b>Spárováno</b>	292 z 306 (95,42 %)						
<b>Atribut</b>	<b>Akce</b>	<b>Změny</b>	<b>OK</b>	<b>Err 1</b>	<b>Err 2</b>	<b>OK (B)</b>	<b>Err (B)</b>
<b>aux.rf (D)</b>	151		151	0			
			100,00 %	0,00 %			
<b>Funktor</b>	94	31	77	3	0	61	19
		32,98 %	96,25 %	3,75 %	0,00 %	76,25 %	23,75 %
<b>Rodič</b>	62	62	30	31	1	1	61
		100,00 %	48,39 %	50,00 %	1,61 %	1,61 %	98,39 %
<b>T-lemma</b>	94	31	77	3	0	61	19
		32,98 %	96,25 %	3,75 %	0,00 %	76,25 %	23,75 %
<b>is_member</b>	188	185	154	2	32	33	155
		98,40 %	81,91 %	1,06 %	17,02 %	17,55 %	82,45 %
<b>aux.rf (A)</b>	94	94	35	45		45	35
		100,00 %	43,75 %	56,25 %		56,25 %	43,75 %
<b>Uzel (N)</b>	31		19	12			
			61,29 %	38,71 %			
Stará data							
<b>Výskyty</b>	573						
<b>Spárováno</b>	1 659 z 1 815 (91,40 %)						
<b>Atribut</b>	<b>Akce</b>	<b>Změny</b>	<b>OK</b>	<b>Err 1</b>	<b>Err 2</b>	<b>OK (B)</b>	<b>Err (B)</b>
<b>aux.rf (D)</b>	883		471	412			
			53,34 %	46,66 %			
<b>Funktor</b>	575	218	398	21	0	324	95
		37,91 %	94,99 %	5,01 %	0,00 %	77,33 %	22,67 %
<b>Rodič</b>	434	434	143	284	7	7	427
		100,00 %	32,95 %	65,44 %	1,61 %	1,61 %	98,39 %
<b>T-lemma</b>	575	218	400	19	0	326	93
		37,91 %	95,47 %	4,53 %	0,00 %	77,80 %	22,20 %
<b>is_member</b>	1 148	1 125	820	19	309	313	835
		98,00 %	71,43 %	1,66 %	26,92 %	27,26 %	72,74 %
<b>aux.rf (A)</b>	575	575	12	407		407	12
		100,00 %	2,86 %	97,14 %		97,14 %	2,86 %
<b>Uzel (N)</b>	218		85	133			
			38,99 %	61,01 %			

Tabulka 5.7: Apozice s vlastním jménem: výsledky modulu



Obrázek 5.7: Apozice s vlastním jménem: struktura v PTB–WSJ a výsledek na tektogramatické rovině (věta „*GP Express, an independent airline serving much of Nebraska, ...*“)

schází patiens (PAT). Na tektogramatické rovině jsou tyto chybějící aktanty vyjádřeny umělými koreferenčními uzly. Jejich ruční vytváření je poměrně pracné, proto je lepší dělat to automaticky.

V tomto případě poprvé využijeme stopy, o kterých byla řeč již v části 5.5. V PTB–WSJ jsou totiž nevyjádřené členy vyjádřeny stopami, a to i s odkazy na své antecedenty. Díky tomu dokážeme nejen vytvářet uzly na správném místě, ale také přidávat správné koreferenční odkazy.

Ve složkovém stromě hledáme neterminál **S** s *funkcí* **ADV**, který má mezi syny neterminály **NP** a **VP**. Neterminál **NP** musí mít pouze jediného potomka, kterým je terminál s formou ve tvaru *\*-<číslo>*, např.: *\*-3*. Číslo za pomlčkou je *index* koreferovaného větného členu, který vyhledáme. Data PTB–WSJ jsou převedená do formátu PML tak, že uzly mají vyplněný atribut *index*, pokud byl uveden v originálu. Vyhledání antecedentu je proto snadné.

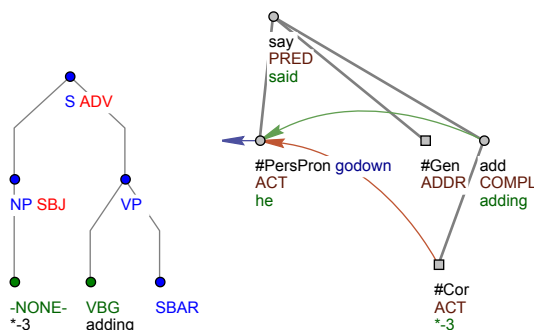
U neterminálu **VP** nás zajímá pouze sloveso, přesněji řečeno jeho značka, která určí *funktor* umělého uzlu na tektogramatické rovině. Bohužel ale nestačí prohledat syny, jelikož občas se vyskytuje více sloves v koordinaci, proto jsem zvolil prohledávání podstromu do šířky a *funktor* je určen prvním nalezeným slovesem s *tagem* **VBG** nebo **VBN**. Pokud takové sloveso není nalezeno, neděláme nic (jde o zanedbatelný počet případů).

Teď již můžeme vytvořit uzel na tektogramatické rovině. Jeho *t-lemma* bude **#Cor** a bude mít nastavený příznak *is\_generated*. Pokud nalezené sloveso mělo *tag* **VBG**, přidělíme uzlu *funktor* **ACT**, v opačném případě *funktor* **PAT**. Také vytvoříme gramatickou koreferenci na nalezený antecedent (tektogramatický protějšek uzlu nalezeného ve složkovém stromě). Poslední neznámou je rodič nového uzlu. Tím by mělo být nalezené sloveso, takže by opět stačilo najít ho na *t*-rovině pomocí relací mezi jednotlivými vrstvami anotace. V tomto případě jsem ale využil analytickou rovinu, na které jsou stopy zachovány ve formě uzlů, a řídil se její strukturou. Na výsledku to nic nemění, liší se pouze technické provedení.

Výsledky uvádím v tabulce 5.8. Hodnoty získané na *nových datech* jsou poměrně přesvědčivé i přesto, že jde o docela malý vzorek dat. Čísla získaná testováním na *starých datech* hovoří úplně jinak, jejich spolehlivost je však minimální. Anotátoři zkrátka umělé uzly až na výjimky nevytvářeli. Popsaný jev se v celém korpusu vyskytuje na 1 697 místech. Jeden příklad je uveden na obrázku 5.8.

Nová data							
Výskyty	77						
Spárováno	58 z 77 (75,32 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
coref_gram (A)	71	71	51	2		2	51
		100,00 %	96,23 %	3,77 %		3,77 %	96,23 %
Funktor	77	77	52	6	0	0	58
		100,00 %	89,66 %	10,34 %	0,00 %	0,00 %	100,00 %
is_generated	77	77	58	0	0	0	58
		100,00 %	100,00 %	0,00 %	0,00 %	0,00 %	100,00 %
T-lemma	77	77	56	2	0	0	58
		100,00 %	96,55 %	3,45 %	0,00 %	0,00 %	100,00 %
Uzel (N)	77		58	19			
			75,32 %	24,68 %			
Stará data							
Výskyty	386						
Spárováno	81 z 386 (20,98 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
coref_gram (A)	363	363	34	41		41	34
		100,00 %	45,33 %	54,67 %		54,67 %	45,33 %
Funktor	386	386	43	38	0	0	81
		100,00 %	53,09 %	46,91 %	0,00 %	0,00 %	100,00 %
is_generated	386	386	69	0	12	12	69
		100,00 %	85,19 %	0,00 %	14,81 %	14,81 %	85,19 %
T-lemma	386	386	38	43	0	0	81
		100,00 %	46,91 %	53,09 %	0,00 %	0,00 %	100,00 %
Uzel (N)	386		81	305			
			20,98 %	79,02 %			

Tabulka 5.8: Vedlejší věta s nevyjádřeným aktantem: výsledky modulu



Obrázek 5.8: Vedlejší věta s nevyjádřeným aktantem: struktura v PTB-WSJ a širší kontext výsledku na tektogramatické rovině (věta „...” he said, adding that ... “)

## 5.13 Postponovaný přívlastek

V této části se zaměřím na postponované přívlastky (uvedené až po rozvíjeném větném členu) vyjádřené slovesnou nebo adjektivní frází. Přívlastky mohou nebo také nemusí být oddělené čárkami. Například „Shaw, based in Dalton, Ga., has . . . “ nebo „. . . a bill boosting the wage floor . . . “.

V PTB–WSJ hledáme neterminál NP, který má právě dva syny (nebo čtyři, pokud je přívlastek oddělen čárkami). Prvním z nich je opět neterminál NP (označme  $P_1$ ). Pokud jsou synové čtyři, musí druhý a čtvrtý z nich být terminál reprezentující čárku – ověříme podle atributu *form*. Zbývající syn (označme  $P_2$ ) je o něco složitější. V každém případě musí jít o neterminál VP nebo ADJP. U přívlastku odděleného čárkami to stačí, ve verzi bez čárek jsou kladeny vyšší nároky. Je-li synem neterminál VP, musí být mezi jeho syny právě jeden terminál s *tagem* VBG, VBN, nebo VBD. Neterminál ADJP zase mezi svými syny nesmí mít žádný terminál –NONE–.

Na tektogramatické rovině vyhledáme uzly odpovídající hlavám neterminálů  $P_1$  a  $P_2$  (označme  $T_1$  a  $T_2$ ). Uzel  $T_2$  pověsíme na  $T_1$  a přidělíme mu *funktor* RSTR, v případě přívlastku odděleného čárkami to bude DESCR. Podobně jako v části 5.12 v některých případech tvoříme umělé koreferenční uzly na místech nevyjádřených aktantů. Je-li uzel  $T_2$  sloveso, které má v PTB–WSJ tag VBG nebo VBN, vytvoříme koreferenční uzel jako jeho syna. Funktor bude ACT nebo PAT podle stejných pravidel jako v části 5.12. I ostatní atributy nastavíme stejně. Koreferovaným uzlem bude  $T_1$ .

Je velmi pravděpodobné, že uzel  $T_2$  již v tuto chvíli má syna s *t-lemmatem* #Gen a se stejným *funktoem* jako nově vytvořený koreferenční uzel (vznikl při přiřazování valenčního rámce). Taková situace je nepřipustná, proto modul nadbytečné uzly maže.

V případě přívlastku odděleného čárkami je ještě potřeba správně vytvořit reference v atributu *aux.rf*. Obě čárky náležejí k uzlu  $T_2$ .

Výsledky modulu (viz tabulka 5.9) jsou výtečné. Je vidět, že struktura je správně již před aplikací modulu (atribut „rodič“), ostatní sledované atributy ovšem ukazují jednoznačný přínos. Pozorného čtenáře může napadnout, jak je možné, že tentokrát i u balíku *starých dat* uvádím velmi vysokou úspěšnost vytváření koreferenčních uzlů, když jsem v části 5.12 psal, že anotátoři tyto uzly nevytvářeli. V tomto případě je situace o něco odlišná, neboť jde o uzly, jejichž antecedentem je vždy rodič jejich rodiče. Všechna data prošla předběžnými manuálními kontrolami (viz kapitola 6), díky kterým bylo možné koreference tohoto typu automaticky doplnit po anotaci (více v části 7.1). Dobrá úspěšnost modulu je navíc podtržena tím, že popsany jev má v korpusu 4336 výskytů. Příklad postponovaného přívlastku je na obrázku 5.9.

## 5.14 Číslovka ve formě přívlastku

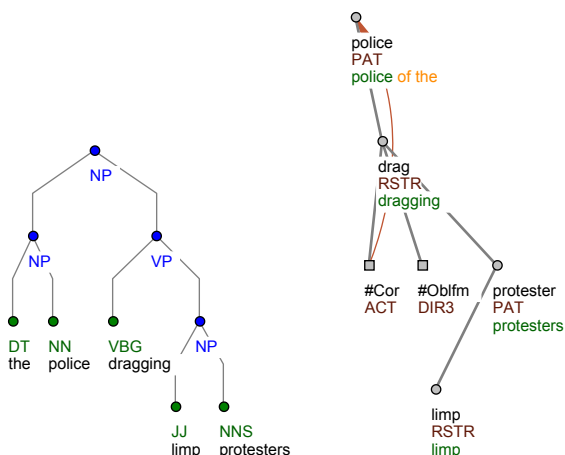
Pokud číslovka rozvíjí substantivum, bere na sebe funkci přívlastku (např. „5 cents“). Tento jev je natolik jednoduchý (a tím pádem i častý), že stojí za to pokusit se zautomatizovat jeho anotaci.

Ve složkovém stromě hledáme neterminál NP, který má právě dva syny, z nichž první je terminál CD. Nalezneme-li tuto strukturu, vyhledáme t–uzel reprezentující terminál CD, přidělíme mu *funktor* RSTR a pověsíme ho na t–uzel, který reprezentuje druhého syna neterminálu NP, ať už to je cokoliv.

Z výsledků, které uvádím v tabulce 5.10, se dají vyčíst dvě významné skutečnosti. Modul je velmi přesný a zcela zbytečný. Grafický příklad považuji v tomto případě za zbytečný. Pro úplnost doplňuji, že v celém PEDT je 8368 výskytů tohoto jevu.

Nová data							
Výskyty	332						
Spárováno	589 z 612 (96,24%)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
aux.rf (D)	54		54	0			
			100,00 %	0,00 %			
Funktor	585	305	533	25	4	278	284
		52,14 %	94,84 %	4,45 %	0,71 %	49,47 %	50,53 %
Uzel (D)	129		126	3			
			97,67 %	2,33 %			
T-lemma	253	253	213	17	0	0	230
		100,00 %	92,61 %	7,39 %	0,00 %	0,00 %	100,00 %
Uzel (N)	253		230	23			
			90,91 %	9,09 %			
coref_gram (A)	253	253	213	17		17	213
		100,00 %	92,61 %	7,39 %		7,39 %	92,61 %
Rodič	332	0	319	13	0	319	13
		0,00 %	96,08 %	3,92 %	0,00 %	96,08 %	3,92 %
is_generated	253	253	229	0	1	1	229
		100,00 %	99,57 %	0,00 %	0,43 %	0,43 %	99,57 %
aux.rf (A)	54	54	22	32		32	22
		100,00 %	40,74 %	59,26 %		59,26 %	40,74 %
Stará data							
Výskyty	1 235						
Spárováno	2 080 z 2 271 (91,59%)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
aux.rf (D)	244		101	143			
			41,39 %	58,61 %			
Funktor	2 149	1 142	1 707	179	72	983	975
		53,14 %	87,18 %	9,14 %	3,68 %	50,20 %	49,80 %
Uzel (D)	390		358	32			
			91,79 %	8,21 %			
T-lemma	914	914	515	228	0	0	743
		100,00 %	69,31 %	30,69 %	0,00 %	0,00 %	100,00 %
Uzel (N)	914		743	171			
			81,29 %	18,71 %			
coref_gram (A)	917	917	379	367		367	379
		100,00 %	50,80 %	49,20 %		49,20 %	50,80 %
Rodič	1 235	0	1 128	87	0	1 128	87
		0,00 %	92,84 %	7,16 %	0,00 %	92,84 %	7,16 %
is_generated	914	914	741	0	2	2	741
		100,00 %	99,73 %	0,00 %	0,27 %	0,27 %	99,73 %
aux.rf (A)	244	244	7	233		233	7
		100,00 %	2,92 %	97,08 %		97,08 %	2,92 %

Tabulka 5.9: Postponovaný přívlástek: výsledky modulu



Obrázek 5.9: Postponovaný přívlastek: struktura v PTB–WSJ a výsledek na tektogramatické rovině (věta „*Hardly a day passes without news photos of the police dragging limp protesters from some building or thoroughfare in one of our cities.*“)

Nová data							
Výskyty	374						
Spárováno	374 z 374 (100,00 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
Funktor	374	0	353	21	0	353	21
		0,00 %	94,39 %	5,61 %	0,00 %	94,39 %	5,61 %
Rodič	374	13	362	11	1	362	12
		3,48 %	96,79 %	2,94 %	0,27 %	96,79 %	3,21 %
Stará data							
Výskyty	1 956						
Spárováno	1 891 z 1 956 (96,68 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
Funktor	1 956	1	1 791	100	0	1 790	101
		0,05 %	94,71 %	5,29 %	0,00 %	94,66 %	5,34 %
Rodič	1 956	73	1 808	44	39	1 839	52
		3,73 %	95,61 %	2,33 %	2,06 %	97,25 %	2,75 %

Tabulka 5.10: Číslovky ve formě přívlastku: výsledky modulu

Nová data							
Výskyty	3						
Spárováno	6 z 6 (100,00 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
Funktor	6	3	5	1	0	2	4
		50,00 %	83,33 %	16,67 %	0,00 %	33,33 %	66,67 %
Rodič	6	0	6	0	0	6	0
		0,00 %	100,00 %	0,00 %	0,00 %	100,00 %	0,00 %
Stará data							
Výskyty	38						
Spárováno	76 z 76 (100,00 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
Funktor	76	38	67	9	0	37	39
		50,00 %	88,16 %	11,84 %	0,00 %	48,68 %	51,32 %
Rodič	76	0	76	0	0	76	0
		0,00 %	100,00 %	0,00 %	0,00 %	100,00 %	0,00 %

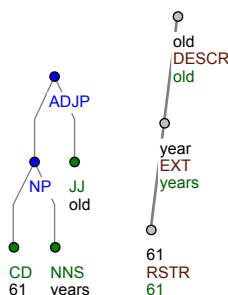
Tabulka 5.11: Číslovka jako součást složené adjektivní fráze: výsledky modulu

## 5.15 Číslovka jako součást složené adjektivní fráze

Podobně jako v části 5.14 se i tady budu krátce zabývat číslovkami s funkcí přívlastku, jenom o něco složitěji. Půjde o adjektivní fráze „56 years old“, „56 % lower“ apod., ve kterých je vnořena substantivní fráze obsahující číslovku.

Ve složkovém stromě hledáme neterminál ADJP, který má právě dva syny. Pravým synem je terminál JJ, JJR nebo JJS (označme  $P_j$ ). Levým synem je neterminál NP, který má také právě dva syny: Levým je terminál CD (označme  $P_c$ ) a pravý je libovolný (označme  $P_n$ ). Na tektogramatické rovině nejprve vyhledáme protějšky uzlů  $P_j$ ,  $P_c$  a  $P_n$  (označme  $T_j$ ,  $T_c$  a  $T_n$ ). Uzlu  $T_c$  přiřadíme *funktor* RSTR a pověsíme ho na uzel  $T_n$ . Ten zase pověsíme na  $T_j$ . Pokud má uzel  $P_j$  tag JJR (komparativ), dostane uzel  $T_n$  *funktor* DIFF. V opačném případě to bude *funktor* EXT.

Výsledky modulu (viz tabulka 5.11) jsou hodně podobné těm, které uvádím v části 5.14. Největší rozdíl je v tom, že výskyty jsou mnohem vzácnější. V celém PEDT je jich pouze 151. Změny struktury jsou zbytečné, jednotlivé uzly jsou již před použitím modulu zavěšeny správně. Změny *funktorů* mají smysl pouze u uzlu  $T_n$ . Na obrázku 5.10 je uveden příklad právě popsaného jevu.



Obrázek 5.10: Adjektivní fráze obsahující substantivní frázi s číslovkou: struktura v PTB–WSJ a výsledek na tektogramatické rovině (výraz „61 years old“)

## 5.16 Aktanty PAT a EFF u vybraných sloves

U sloves, která volně souvisí s vyjadřováním (ať už to je názor, myšlenka, pocit nebo něco jiného) je možné identifikovat aktanty PAT nebo EFF. Začnu zavedením dvou seznamů:  $S_{PAT}$  a  $S_{EFF}$ .

- $S_{PAT}$  = add, insist, suggest, declare, find, reason, hope, believe, conclude, point, assume, presume, promise, fear, assure, reassure, recommend, mention, protest, inform, deny, guarantee, decide
- $S_{EFF}$  = say, tell, announce, think, claim

Pokračovat budu popisem struktury ve složkovém stromě. Hledáme neterminál VP, který má kromě jiných tyto dva syny (v libovolném pořadí a na libovolné pozici):

- Terminál, jehož *lemma* je v některém ze seznamů  $S_{PAT}$  a  $S_{EFF}$  (označme  $P_1$ )
- Neterminál SBAR s prázdným atributem *functions*, který mezi svými syny musí mít neterminál S (označme  $P_2$ ) a může mít terminál IN s *formou* *if*, *whether* nebo *that* (označme  $P_{IN}$ )

Je-li *lemma* uzlu  $P_1$  *point*, potom zkoumaný neterminál VP musí mít mezi syny navíc neterminál PRT, v jehož podstromu se nachází předložka „out“ (označme  $P_{OUT}$ ).

Na tektogramatické rovině nalezneme uzel  $T_1$ , který je protějškem uzlu  $P_1$ , a také uzel  $T_2$ , který reprezentuje hlavu neterminálu  $P_2$ . Pokud se *lemma* uzlu  $P_1$  nachází v seznamu  $S_{PAT}$ , přidělíme uzlu  $T_2$  *funktor* PAT, v opačném případě zvolíme *funktor* EFF.  $T_2$  pověsíme na  $T_1$ . Pokud má  $T_1$  generovaného syna s *funktořem* stejným jako  $T_2$  a *t-lemmatem* #Gen (uzel vytvořený kvůli požadavkům valenčního rámce), je tento umělý uzel nadbytečný a je třeba ho odstranit.

Existuje-li uzel  $P_{IN}$  ( $T_2$  je uvozen podřadící spojkou), musíme přidat příslušnou referenci k  $T_2$  a odstranit všechny chybné. Pokud existuje samostatný t-uzel reprezentující spojku  $P_{IN}$ , smažeme jej. Pokud byl nalezen uzel  $P_{OUT}$ , je třeba přidat referenci k uzlu  $T_1$  a navíc změnit *t-lemma*  $T_1$  na *point\_out*. Případný nadbytečný t-uzel opět smažeme.

Výsledky (tabulka 5.12) ukazují, že modul je velmi přínosný při určování *funktorů* a mazání nadbytečných uzlů. Na strukturu nemá žádný vliv, protože všechno, co by se mělo udělat, už je v datech správně. Přidávání do *aux.rf* je také zbytečné, možná až mírně škodlivé. A o mazání z *aux.rf* se nedá říct vůbec nic, protože výsledky u *nových* a *starých* dat vypadají jako den a noc. Jde o velmi častý jev, počet výskytů v celém korpusu je 7 546. Jako příklad uvádím obrázek 5.11.

## 5.17 Přímá řeč jako téma

PTB-WSJ je korpus složený z novinových článků, proto je v něm velké množství vět se strukturou „...“ sb. said.“. PTB-WSJ na takovou přímou (ale i nepřímou) řeč pohlíží jako na téma věty, což je znázorněno *funkcí* TPC (ToPiC). Tato informace se dá využít při anotaci tektogramatické roviny.

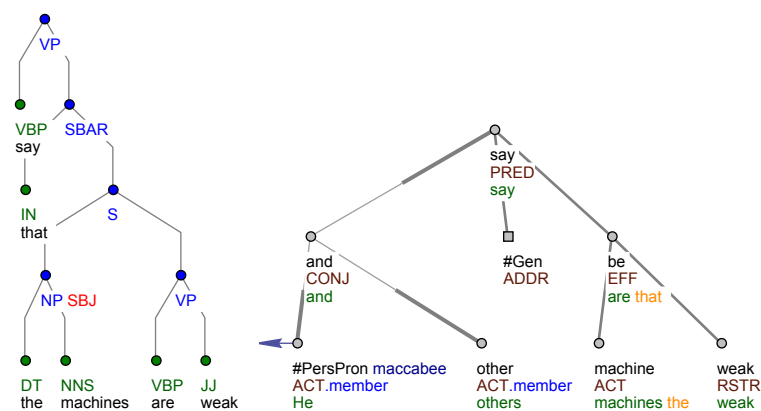
Nejprve musíme najít následující strukturu ve složkovém stromě. Tentokrát nebudeme klást žádné požadavky na konkrétního rodiče, pouze u každého uzlu ověříme, jestli se mezi jeho syny vyskytují dva konkrétní uzly:

Nová data							
Výskyty	412						
Spárováno	426 z 429 (99,30%)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
aux.rf (D)	16		15 93,75 %	1 6,25 %			
Funktor	412	341 82,77 %	380 92,91 %	5 1,22 %	24 5,87 %	94 22,98 %	315 77,02 %
Uzel (D)	307		260 84,69 %	47 15,31 %			
Rodič	412	0 0,00 %	399 97,56 %	10 2,44 %	0 0,00 %	399 97,56 %	10 2,44 %
T-lemma	1	0 0,00 %	1 100,00 %	0 0,00 %	0 0,00 %	1 100,00 %	0 0,00 %
aux.rf (A)	98	6 6,12 %	97 98,98 %	1 1,02 %		93 94,90 %	5 5,10 %

Stará data							
Výskyty	1 816						
Spárováno	1 876 z 1 900 (98,74%)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
aux.rf (D)	74		14 21,21 %	52 78,79 %			
Funktor	1 816	1 570 86,45 %	1 624 90,22 %	14 0,78 %	162 9,00 %	404 22,44 %	1 396 77,56 %
Uzel (D)	1 395		1 216 87,17 %	179 12,83 %			
Rodič	1 816	0 0,00 %	1 761 97,83 %	39 2,17 %	0 0,00 %	1 761 97,83 %	39 2,17 %
T-lemma	10	0 0,00 %	10 100,00 %	0 0,00 %	0 0,00 %	10 100,00 %	0 0,00 %
aux.rf (A)	331	28 8,46 %	297 91,67 %	27 8,33 %		323 99,69 %	1 0,31 %

Tabulka 5.12: Aktanty PAT a EFF u vybraných sloves: výsledky modulu



Obrázek 5.11: Aktanty PAT a EFF u vybraných sloves: struktura v PTB-WSJ a výsledek na tektogramatické rovině rozšířený o aktor (věta „He and others say that the machines are weak enough that they don't jeopardize the memory.“)

Nová data							
Výskyty	186						
Spárováno	231 z 241 (95,85 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
Funktor	216	216	199	7	0	0	206
		100,00 %	96,60 %	3,40 %	0,00 %	0,00 %	100,00 %
Uzel (D)	174		165	9			
			94,83 %	5,17 %			
Rodič	186	0	175	5	0	175	5
		0,00 %	97,22 %	2,78 %	0,00 %	97,22 %	2,78 %
Stará data							
Výskyty	866						
Spárováno	1 073 z 1 096 (97,90 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
Funktor	989	989	937	18	11	11	955
		100,00 %	97,00 %	1,86 %	1,14 %	1,14 %	98,86 %
Uzel (D)	843		788	55			
			93,48 %	6,52 %			
Rodič	866	0	819	32	0	819	32
		0,00 %	96,24 %	3,76 %	0,00 %	96,24 %	3,76 %

Tabulka 5.13: Přímá řeč jako téma: výsledky modulu

- Sourozenec 1: Neterminál VP, mezi jehož syny se nachází terminál, který má *lemma say* (označme  $P_1$ )
- Sourozenec 2: Neterminál S s *funkcí* TPC (označme  $P_2$ )

Na tektogramatické rovině vyhledáme protějšek terminálu  $P_1$  – uzel  $T_1$ . Dále nalezneme  $t$ –uzel odpovídající hlavě neterminálu  $P_2$  (označme  $T_2$ ).  $T_2$  pověsíme na  $T_1$ , a pokud už  $T_1$  měl nějaký generovaný uzel s *funktořem* EFF, můžeme nadbytečný uzel smazat (podobně jako v částech 5.13 a 5.16). Zbývá přidělení správného *funktoru*. Pokud je  $T_2$  kořenem koordinace (typ uzlu *coap*), přidělíme všem koordinovaným uzlům *funktor* EFF. V opačném případě tento *funktor* přiřadíme přímo uzlu  $T_2$ .

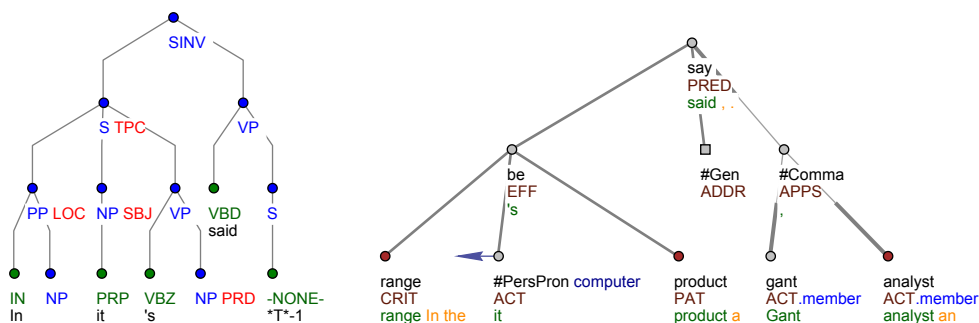
Jako již v několika případech jsem z výsledků zjistil, že uzly jsou na správných místech již před použitím modulu vytvořeného podle popisu výše. Ovšem určování *funktorů* a mazání nadbytečných uzlů je (stejně jako v jiných případech) velmi přínosné (viz tabulka 5.13). Právě popsáný jev má v korpusu 3 528 výskytů. Příklad je na obrázku 5.12.

## 5.18 Identifikace aktantu u vybraných sloves

U sloves uvedených v seznamech  $S_{PAT}$  a  $S_{EFF}$  v části 5.16 je možné snadno identifikovat aktor pomocí PTB–WSJ *funkce* SBJ uváděné u jmenné fráze.

Postup je přímočarý. Ve složkovém stromě hledáme dva sourozence, z nichž jedním je neterminál VP, který má mezi syny jedno z uvedených sloves, a druhým je neterminál NP s *funkcí* SBJ. Pokud najdeme tuto strukturu, tak na tektogramatické rovině identifikujeme uzel, který odpovídá hlavě nalezeného neterminálu NP, a přidělíme mu *funktor* ACT. Opět je třeba zohlednit případnou koordinaci stejně jako v části 5.17.

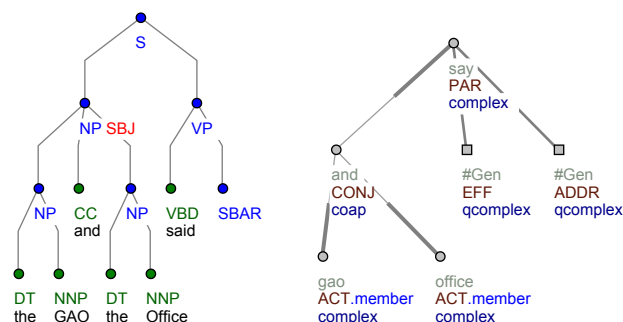
Výsledky (viz tabulka 5.14) ukazují, že ve většině případů je již *funktor* přidělen správně. Popsáný jev je ovšem tak častý (12 549 výskytů v korpusu), že i přesto je modul velmi přínosný. Příklad s vícenásobným aktorem v koordinaci je na obrázku 5.13.



Obrázek 5.12: Přímá řeč jako téma: struktura v PTB–WSJ a výsledek na tektogramatické rovině – pouze hlavní části stromů (věta „*In the price range it’s a tremendously high-performing product ,*” said Sandy Gant, an analyst at the market-research firm InfoCorp.“)

Nová data							
Výskyty	675						
Spárováno	694 z 694 (100,00 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
Funktor	694	162	681	11	2	527	167
		23,34 %	98,13 %	1,59 %	0,29 %	75,94 %	24,06 %
Stará data							
Výskyty	3 031						
Spárováno	3 126 z 3 126 (100,00 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
Funktor	3 126	856	3 028	93	5	2 248	878
		27,38 %	96,87 %	2,98 %	0,16 %	71,91 %	28,09 %

Tabulka 5.14: Identifikace aktantu u vybraných sloves: výsledek modulu



Obrázek 5.13: Identifikace aktantu u vybraných sloves: struktura v PTB–WSJ a výsledek na tektogramatické rovině – pouze části stromů (věta „*Instead, the GAO and the Congressional Budget Office said, the RTC should consider ...*“)

Nová data							
Výskyty	181						
Spárováno	225 z 226 (99,56 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
aux.rf (D)	12		9	3			
			75,00 %	25,00 %			
Funktor	214	214	202	3	8	8	205
		100,00 %	94,84 %	1,41 %	3,76 %	3,76 %	96,24 %
Uzel (D)	82		77	5			
			93,90 %	6,10 %			
aux.rf (A)	214	5	197	16		192	21
		2,34 %	92,49 %	7,51 %		90,14 %	9,86 %

Stará data							
Výskyty	522						
Spárováno	616 z 616 (100,00 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
aux.rf (D)	21		21	0			
			100,00 %	0,00 %			
Funktor	595	595	543	24	28	28	567
		100,00 %	91,26 %	4,03 %	4,71 %	4,71 %	95,29 %
Uzel (D)	235		207	28			
			88,09 %	11,91 %			
aux.rf (A)	595	11	576	19		565	30
		1,85 %	96,81 %	3,19 %		94,96 %	5,04 %

Tabulka 5.15: Logický subjekt v pasivu: výsledky modulu

## 5.19 Logický subjekt v pasivu

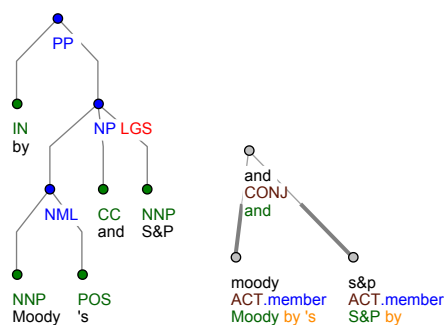
PTB–WSJ používá *funkci* LGS pro označení logického subjektu v pasivních konstrukcích (např. „rated by sb.“). Takto označené subjekty můžeme na tektogramatické rovině označit jako aktory. Postup je následující.

Ve složkovém stromě nás zajímá struktura, jejímž kořenem je neterminál PP, který má právě dva syny: terminál IN a neterminál NP, jehož *funkcí* je LGS.<sup>8</sup>

Na t–rovině nalezneme uzel odpovídající hlavě neterminálu NP, přidělíme mu *funktor* ACT (s přihlédnutím k možné koordinaci) a do atributu *aux.rf* uložíme referenci na slovo reprezentované terminálem IN – tím je vždy předložka „by“. Stejně jako s *funktorem* i s referencí musíme zacházet opatrně vzhledem k možné koordinaci. Nakonec smažeme případné chybné *aux.rf* reference a nadbytečné uzly (nadále nepotřebný generovaný uzel s *funktorem* ACT nebo samostatný uzel pro předložku).

Tento postup přináší velmi dobré výsledky. Správnost práce s *funktory* i s referencemi na pomocná slova je velmi vysoká, ovšem reference jsou v naprosté většině případů správně již před aplikací modulu. O *funktorech* se ale dá říct pravý opak. Dobře se jeví i mazání nadbytečných uzlů. Četnost tohoto jevu je docela vysoká, v korpusu je 3 155 výskytů. Podrobné výsledky jsou v tabulce 5.15, na obrázku 5.14 je příklad toho, jak jev vypadá v kombinaci s koordinací.

<sup>8</sup>Ve skutečnosti je třeba hledat i případy, kdy je *funkce* LGS u neterminálu PP. Dokumentace to sice zakazuje (viz část 2.2.2 v [1]), ovšem realita je jiná.



Obrázek 5.14: Logický subjekt v pasivu: struktura v PTB–WSJ a výsledek na tektogramatické rovině (výraz „by Moody’s and S&P“)

## 5.20 Funktory MAT a APP po předložce „of“

Vyskytuje-li se v PTB–WSJ jmenná fráze bezprostředně následovaná předložkovou frází s předložkou „of“, můžeme v některých případech docela dobře určit *funktor* pro slovo následující po předložce. Některými případy mám v tomto případě na mysli explicitní seznamy slov, které zde kvůli jejich velikosti nebudu uvádět celé. Jejich charakter pouze naznačím několika příklady.

- Seznam  $S_1$ : billions, hundreds, days, decades, supplies, lots, armies, baskets, . . .
- Seznam  $S_2$ : Lemmata většiny slov ze seznamu  $S_1$  a několik nových slov podobného charakteru navíc
- Seznam  $S_3$ : chairman, treasurer, spokesman, father, sister, head, part, front, . . .

A teď ke konkrétnímu popisu. Začneme tím, že ve složkovém stromě najdeme neterminály NP a PP, které spolu bezprostředně sousedí a mají společného rodiče. Hlava neterminálu NP musí splňovat jedno ze tří kritérií:

- *Tag* je NNS a *forma* se nachází v seznamu  $S_1$  – typ 1
- *Tag* začíná na NN a *lemma* se nachází v seznamu  $S_2$  – typ 1
- *Tag* začíná na NN a *lemma* se nachází v seznamu  $S_3$  – typ 2

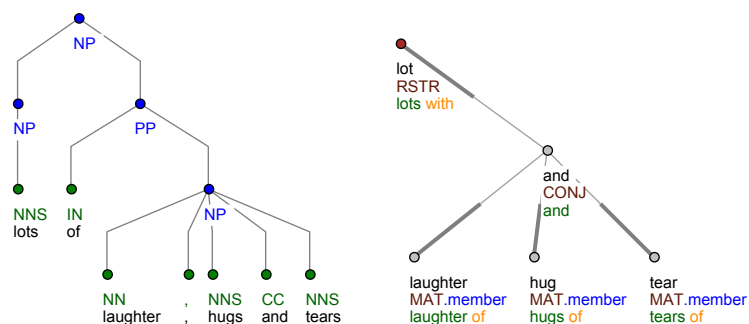
Pro terminál PP platí, že jeho prvním synem musí být předložka „of“ a druhým synem musí být neterminál NP (označme  $P_2$ ).

Na t–rovině je situace jednoduchá. Identifikujeme uzel  $T_2$  odpovídající hlavě neterminálu  $P_2$  a přidělíme mu jeden ze dvou *funktorů*. Pokud bylo splněno kritérium typu 1, použijeme *funktor* MAT, v opačném případě APP. Také zajistíme, aby byla u uzlu  $T_2$  a nikde jinde uvedena reference na předložku „of“. To vše korektně vzhledem k možnému výskytu koordinace.

Výsledky modulu jsou přijatelné. Spolehlivost určování *funktorů* nepřesahuje 90% jako v jiných případech, čísla uvedená v tabulce 5.16 přesto prokazují, že pravidlo (a skript pracující podle něj) je přínosné, a to mimo jiné i proto, že jde o častý jev (3572 výskytů). Zajímavý příklad je na obrázku 5.15.

Nová data							
Výskyty	192						
Spárováno	203 z 203 (100,00 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
aux.rf (D)	5		5	0			
			100,00 %	0,00 %			
Funktor	198	119	158	32	8	67	131
		60,10 %	79,80 %	16,16 %	4,04 %	33,84 %	66,16 %
aux.rf (A)	198	5	193	5		188	10
		2,53 %	97,47 %	2,53 %		94,95 %	5,05 %
Stará data							
Výskyty	732						
Spárováno	766 z 766 (100,00 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
aux.rf (D)	26		26	0			
			100,00 %	0,00 %			
Funktor	740	448	437	196	107	302	438
		60,54 %	59,05 %	26,49 %	14,46 %	40,81 %	59,19 %
aux.rf (A)	740	21	724	16		707	33
		2,84 %	97,84 %	2,16 %		95,54 %	4,46 %

Tabulka 5.16: Funktory MAT a APP po předložce „of“: výsledky modulu



Obrázek 5.15: Funktory MAT a APP po předložce „of“: struktura v PTB–WSJ a výsledek na tektogrammatické rovině (výraz „lots of laughter, hugs and tears“)

## 5.21 Několik ustálených frází

V této části se budu věnovat skupině pravidel, podle kterých jsou na tektogramatické rovině zachyceny některé ustálené fráze. Většina z nich se v korpusu vyskytuje velmi řídko, místy jde i o méně než deset výskytů, ale i tak má automatické zpracování smysl kvůli zachování konzistence. V následujících odstavcích popíši jednotlivé fráze: jak je správně najít<sup>9</sup> a jak je zachytit na tektogramatické rovině.

### no doubt

Ve složkovém stromě hledáme neterminál NP s *funkcí* ADV, který má právě dva syny. Levým je slovo „no“, pravým „doubt“. Na tektogramatické rovině vyhledáme uzel reprezentující slovo „doubt“, přidělíme mu *funktor* ATT, jeho *t-lemma* změním na `no_doubt` a do atributu *aux.rf* zaneseme referenci na slovo „no“. Případné další chybně umístěné reference odstraníme. Má-li slovo „no“ svůj samostatný *t*-uzel, smažeme ho.

### no matter

Nejprve hledáme neterminál ADVP, který má právě tři syny v tomto pořadí: slovo „no“, slovo „matter“, neterminál SBAR. Na *t*-rovině vyhledáme uzel, který reprezentuje hlavu neterminálu SBAR, a nastavíme jeho *funktor* na REG. U stejného uzlu také do *aux.rf* zaznamenejme reference na obě slova „no“ i „matter“ a jako obvykle odstraníme přebytečné uzly a chybné reference.

### regardless of

Hledáme neterminál ADVP, který má právě dva syny. Prvním z nich je slovo „regardless“, druhým je neterminál PP. Ten má také právě dva syny: prvním je předložka „of“, na druhém nezáleží – musíme pouze najít jeho hlavu. K ní nalezneme odpovídající *t*-uzel, přidělíme mu *funktor* REG a do atributu *aux.rf* zaneseme reference na obě slova „regardless“ a „of“. Přebytečné pozůstatky (chybné reference a uzly) mažeme.

### worth

Výjimečně nás zajímá pouhý terminál, jehož *forma* je `worth` a *tag* JJ. Ten má vždy bezprostředního pravého sourozence, jehož podoba nás moc nezajímá. Stačí najít jeho hlavu a identifikovat příslušný uzel na tektogramatické rovině, abychom mu přidělili *funktor* PAT a pověsili ho na uzel, který reprezentuje slovo „worth“.

### all but

Budeme hledat neterminál ADVP, který má právě dva syny – slova „all“ a „but“. Najdeme-li tuto strukturu, je zajištěno, že neterminál ADVP má pravého sourozence, pro kterého určíme hlavu a přejdeme na *t*-rovinu k odpovídajícímu uzlu (označme  $T_1$ ). Dále vyhledáme i *t*-uzel pro slovo „but“, změním jeho *t-lemma* na `all_but`, *funktor* nastavíme na EXT, do *aux.rf* zaznamenejme referenci na slovo „all“ a takto upravený uzel pověsíme na  $T_1$ . Nakonec odstraníme uzel reprezentující slovo „all“ stejně jako další reference na něj, pokud cokoliv z toho existuje.

<sup>9</sup>Hledání na základě prosté textové shody by mohlo selhat. Uvažme např. frázi „at all“ ve smyslu „vůbec“: „not good at all“ vs. „at all of them“

## at best

Ve složkovém stromě hledáme neterminál ADVP, který má právě dva syny: slova „at“ a „best“. Ke slovu „best“ najdeme odpovídající  $t$ -uzel, změním jeho  $t$ -lemma na `at_best` a *funktor* nastavíme na `ATT`. Do atributu `aux.rf` přidáme referenci na slovo „at“. Jako obvykle mažeme přebývající věci.

## at least, at most

Hledáme neterminál ADVP, který má právě dva syny. Prvním je slovo „at“, druhým je jedno ze slov „least“ a „most“. Najdeme-li tuto strukturu, musíme ještě identifikovat uzel, který se stane rodičem na tektogramatické rovině. Počínaje rodičem neterminálu ADVP budeme hledat hlavu příslušného neterminálu do té doby, dokud při hledání neskončíme jinde, než zpátky v podstromu neterminálu ADVP. Při neúspěchu se vždy posuneme o úroveň výš a zahájíme hledání z rodiče toho uzlu, ve kterém jsme začínali při minulém pokusu. Až nakonec uspějeme, určíme odpovídající  $t$ -uzel  $T_1$ .

Na tektogramatické rovině začneme tím, že se pokusíme najít uzel reprezentující slovo „least“ (nebo „most“). Může se stát, že takový uzel na  $t$ -rovině vůbec není, v tom případě ho musíme vytvořit. Podle situace nastavíme jeho  $t$ -lemma a *funktor* na `at_least` a `RHEM` nebo na `at_most` a `EXT`. K uzlu také připojíme referenci na slovo „at“. Výsledek pověsíme na uzel  $T_1$ . Nakonec obvyklým způsobem odstraníme chybné reference a případný přebytečný uzel.

## vice ...

Ve složkovém stromě hledáme dvojici terminálů (bezprostřední sourozence), z nichž levý má *lemma vice* („vice president“, „vice chairman“, apod.). Při úspěchu přejdeme na  $t$ -rovinu a k oběma terminálům najdeme odpovídající  $t$ -uzly. Levý označíme *funktorem* `RSTR` a pověsíme ho na pravý uzel – slovo „vice“ má funkci přívlastku.

Výsledky uvádím pro všechny automaticky anotované fráze dohromady v tabulce 5.17. Nejčastější akce se týkají *funktoru* a rodiče. Ovšem v případě rodiče nelze mluvit o skutečné změně, v naprosté většině případů jsou již data ve stavu, do jakého se je moduly pokouší dostat. V případě *funktorů* dochází ke skutečné změně asi ve čtvrtině případů a lze pozorovat spíše přínos, případně změnu jedné chyby na jinou. Změny *t*-lemmatu nebo mazání uzlu jsou příliš vzácné na to, abych mohl pouze podle těchto čísel dojít k nějakému závěru. Ručně jsem prošel několik výskytů, u kterých byly akce modulů označeny za chybné, a ve všech případech šlo o chyby anotace. Souhrnný počet výskytů v celém korpusu je 714.

Podrobné vyhodnocení ukázalo, že popsané moduly pracují zpravidla správně. Odhaduji, že většina jich ještě bude použita v závěrečných fázích práce na korpusu pro zajištění konzistence.

## 5.22 Dvouslovné souřadné spojky

Některé souřadné spojky se skládají ze dvou slov, která často stojí hned vedle sebe („let alone“), ale v některých případech mohou být i oddělena („either – or“). Ovšem na tektogramatické rovině musí být i tyto spojky reprezentovány jediným uzlem. V této části vysvětlím automatické zpracování pětice takových spojek: „let alone“, „such as“, „both – and“, „either – or“ a „neither – nor“. Jako obvykle vždy začnu popisem struktury ve

Nová data							
<b>Výskyty</b>	36						
<b>Spárováno</b>	34 z 36 (94,44 %)						
<b>Atribut</b>	<b>Akce</b>	<b>Změny</b>	<b>OK</b>	<b>Err 1</b>	<b>Err 2</b>	<b>OK (B)</b>	<b>Err (B)</b>
<b>Funktor</b>	36	11	30	4	0	24	10
		30,56 %	88,24 %	11,76 %	0,00 %	70,59 %	29,41 %
<b>Uzel (D)</b>	1		0	1			
			0,00 %	100,00 %			
<b>Rodič</b>	28	1	26	1	0	26	1
		3,57 %	96,30 %	3,70 %	0,00 %	96,30 %	3,70 %
<b>T-lemma</b>	10	10	7	1	1	1	8
		100,00 %	77,78 %	11,11 %	11,11 %	11,11 %	88,89 %
<b>aux.rf (A)</b>	9	0	8	0		8	0
		0,00 %	100,00 %	0,00 %		100,00 %	0,00 %
Stará data							
<b>Výskyty</b>	164						
<b>Spárováno</b>	163 z 164 (99,39 %)						
<b>Atribut</b>	<b>Akce</b>	<b>Změny</b>	<b>OK</b>	<b>Err 1</b>	<b>Err 2</b>	<b>OK (B)</b>	<b>Err (B)</b>
<b>Funktor</b>	164	40	146	16	1	125	38
		24,39 %	89,57 %	9,82 %	0,61 %	76,69 %	23,31 %
<b>Uzel (D)</b>	10		5	5			
			50,00 %	50,00 %			
<b>Rodič</b>	141	2	135	4	1	136	4
		1,42 %	96,43 %	2,86 %	0,71 %	97,14 %	2,86 %
<b>T-lemma</b>	25	25	13	1	10	10	14
		100,00 %	54,17 %	4,17 %	41,67 %	41,67 %	58,33 %
<b>aux.rf (A)</b>	28	7	24	4		25	3
		25,00 %	85,71 %	14,29 %		89,29 %	10,71 %

Tabulka 5.17: Několik ustálených frází: výsledky modulu pro všechny fráze dohromady

složkovém stromě, na který navážu výčtem toho, co je třeba vykonat na tektogramatické rovině.

### let alone, both – and, either – or, neither – nor

V případě těchto čtyř spojek hledáme neterminál, který má mezi svými syny obě slova, ze kterých se příslušná spojka skládá. Pouze v případě „let alone“ navíc vyžadujeme, aby to byl neterminál ADVP. Za lexikální složku považujeme vždy druhé slovo spojky, jehož uzel se pokusíme vyhledat na  $t$ -rovině. Pokud ho nenajdeme, je třeba ho vytvořit. U tohoto uzlu (ať už nalezeného, nebo nově vytvořeného) nastavíme následující atributy:

- *nodetype* = `coap`
- *t-lemma* = obě slova spojky spojená podtržítkem, např. `both_and`
- *funktor* = `GRAD` pro „let alone“, `CONJ` pro „both – and“, `DISJ` pro „either – or“, `CONJ` pro „neither – nor“
- *aux.rf* – přidáme referenci na první slovo spojky

Na závěr zbývá odstranit případné chybně umístěné reference na první slovo spojky nebo chybně vytvořený samostatný uzel.

### such as

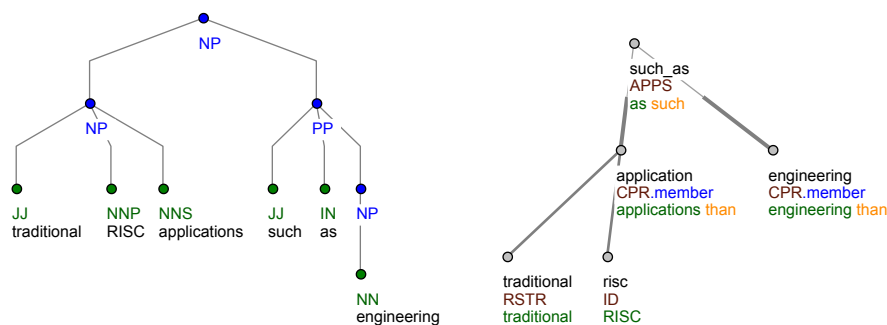
Situace u spojky „such as“ je o něco složitější. Hledáme neterminál PP, který jako syny obsahuje obě slova spojky. Tyto uzly navíc musí být bezprostředními sousedy – nesmí mezi nimi být žádný jiný uzel. Stejně jako ostatní spojky bude i tato na  $t$ -rovině sloužit jako kořen koordinace, tentokrát se však pokusíme najít i oba koordinované členy. Pravým bude neterminál NP nacházející se mezi pravými sourozenci slova „as“. Levého člena koordinace budeme hledat takto: Začneme s terminálem PP (rodič terminálů pro slova „such“ a „as“) a mezi jeho levými sourozenci budeme hledat neterminál NP. Pokud tam žádný není, přesuneme se o úroveň výš k rodiči neterminálu PP a zopakujeme hledání. Takto budeme postupovat pořád výš, dokud nenalezneme neterminál NP nebo dokud se nedostaneme do kořene celého stromu, odkud už není kam stoupat. Pokud mezi předky neterminálu PP narazíme na neterminál PRN, zapamatujeme si, že na  $t$ -rovině máme označit parentezi. Nalezení členů koordinace není podmínkou pro další zpracování.

Se samotnou spojkou provedeme to samé, co je již popsáno u ostatních – *funkto-rem* bude v tomto případě APPS. Pokud se nám navíc podařilo identifikovat některého ze členů koordinace, pověsíme ho (uzel odpovídající hlavě neterminálu NP) na již připravený uzel reprezentující spojkou a nastavíme mu příznak *is\_member*. Je-li třeba označit parentezi a pokud známe pravý člen koordinace, nastavíme tomuto uzlu a všem uzlům v jeho podstromu příznak *is\_parenthesis*.

Výsledky ukazují přínos u všech sledovaných atributů kromě odstraňování uzlů. Podrobné vyhodnocení uvádím v tabulce 5.18. Četnost výskytů nepatří k nejvyšším, přesto jich je nezanedbatelných 781. Na obrázku 5.16 je příklad spojky „such as“. Jsou na něm vidět o oba úspěšně identifikovaní členové koordinace.

Nová data							
Výskyty	71						
Spárováno	177 z 183 (96,72 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
aux.rf (D)	51		48	1			
			97,96 %	2,04 %			
Funktor	71	41	67	2	0	29	40
		57,75 %	97,10 %	2,90 %	0,00 %	42,03 %	57,97 %
Uzel (D)	39		19	20			
			48,72 %	51,28 %			
Nodetype	71	41	62	7	0	23	46
		57,75 %	89,86 %	10,14 %	0,00 %	33,33 %	66,67 %
T-lemma	71	71	44	6	19	19	50
		100,00 %	63,77 %	8,70 %	27,54 %	27,54 %	72,46 %
is_member	62	60	54	0	8	10	52
		96,77 %	87,10 %	0,00 %	12,90 %	16,13 %	83,87 %
Uzel (N)	32		30	2			
			93,75 %	6,25 %			
Rodič	62	62	50	6	6	6	56
		100,00 %	80,65 %	9,68 %	9,68 %	9,68 %	90,32 %
is_parenthesis	23	23	18	0	1	1	18
		100,00 %	94,74 %	0,00 %	5,26 %	5,26 %	94,74 %
aux.rf (A)	71	71	45	24		24	45
		100,00 %	65,22 %	34,78 %		34,78 %	65,22 %
Stará data							
Výskyty	163						
Spárováno	459 z 469 (97,87 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
aux.rf (D)	156		152	0			
			100,00 %	0,00 %			
Funktor	163	107	148	9	0	56	101
		65,64 %	94,27 %	5,73 %	0,00 %	35,67 %	64,33 %
Uzel (D)	71		30	41			
			42,25 %	57,75 %			
Nodetype	163	107	153	4	0	56	101
		65,64 %	97,45 %	2,55 %	0,00 %	35,67 %	64,33 %
T-lemma	163	163	107	13	37	37	120
		100,00 %	68,15 %	8,28 %	23,57 %	23,57 %	76,43 %
is_member	176	169	149	1	26	32	144
		96,02 %	84,66 %	0,57 %	14,77 %	18,18 %	81,82 %
Uzel (N)	92		86	6			
			93,48 %	6,52 %			
Rodič	176	176	137	29	10	10	166
		100,00 %	77,84 %	16,48 %	5,68 %	5,68 %	94,32 %
is_parenthesis	25	22	17	1	7	9	16
		88,00 %	68,00 %	4,00 %	28,00 %	36,00 %	64,00 %
aux.rf (A)	163	163	109	48		48	109
		100,00 %	69,43 %	30,57 %		30,57 %	69,43 %

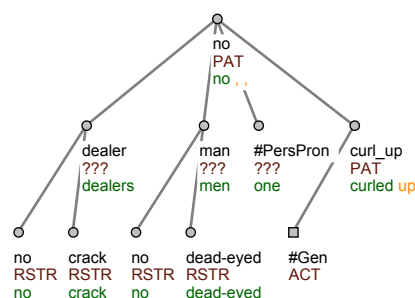
Tabulka 5.18: Dvouslovné souřadné spojky: výsledky modulu



Obrázek 5.16: Dvouslovné souřadné spojky: struktura v PTB–WSJ a výsledek na tektoqramatické rovině pro spojku „such as“ včetně obou členů koordinace (věta „*Mips also wants to wedge into markets other than traditional RISC applications such as engineering.*“)

## 5.23 Vícenásobná koordinace

V několika předchozích částech jsem již zmiňoval pravidla, která mimo jiné řeší správné anotování koordinací, ale téměř vždy šlo o koordinaci pouhých dvou uzlů. V jisté fázi projektu jsme zjistili, že určitý typ koordinací zůstává pravidly nedotčen. Byly to koordinace alespoň tří větných členů, které byly odděleny pouze čárkami bez spojky.<sup>10</sup> Výsledek základního zpracování systémem *TectoMT* dopadal špatně (viz obrázek 5.17), což nejen že znamenalo více práce pro anotátory, ale navíc i narušovalo činnost a zhoršovalo výsledky jiných modulů pro automatickou anotaci. Systém se tvrdohlavě snaží nějakou spojku najít a výsledkem je nesmysl.



Obrázek 5.17: Vícenásobná koordinace: špatná anotace po základním zpracování systémem *TectoMT* (výraz „*no crack dealers, no dead-eyed men ... , no one curled up ...*“)

Proto jsem sestavil modul, který řeší právě vícenásobné koordinace. Samozřejmě je založen na anotaci PTB–WSJ. Hledá neterminál, jehož synové tvoří posloupnost, ve které se pravidelně střídají neterminály a terminály reprezentující symbol čárky. Posloupnost musí začínat i končit neterminálem. Navíc jsou povoleny pouze neterminály VP, NP, PP a ADVP a v jedné konkrétní koordinaci nesmí být různé neterminály. Výjimku tvoří PP a ADVP, ty se mohou vyskytovat pohromadě.

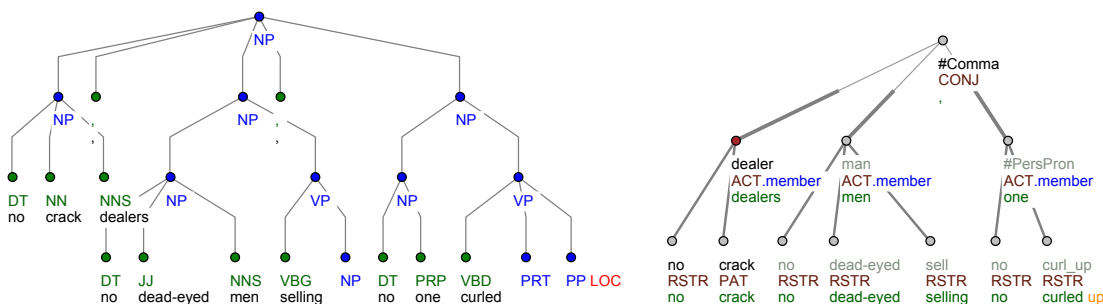
<sup>10</sup>Ve většině koordinací jsou poslední dva členy odděleny spojkou a všechny ostatní čárkou.

Nová data							
Výskyty	3						
Spárováno	16 z 16 (100,00 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
aux.rf (D)	7		7 100,00 %	0 0,00 %			
Rodič	13	13 100,00 %	13 100,00 %	0 0,00 %	0 0,00 %	0 0,00 %	13 100,00 %
Nodetype	3	3 100,00 %	3 100,00 %	0 0,00 %	0 0,00 %	0 0,00 %	3 100,00 %
T-lemma	3	3 100,00 %	3 100,00 %	0 0,00 %	0 0,00 %	0 0,00 %	3 100,00 %
is_member	13	13 100,00 %	13 100,00 %	0 0,00 %	0 0,00 %	0 0,00 %	13 100,00 %
aux.rf (A)	7	7 100,00 %	6 85,71 %	1 14,29 %		1 14,29 %	6 85,71 %
Uzel (N)	3		3 100,00 %	0 0,00 %			
Stará data							
Výskyty	55						
Spárováno	162 z 177 (91,53 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
aux.rf (D)	54		42 77,78 %	12 22,22 %			
Rodič	124	124 100,00 %	66 54,55 %	35 28,93 %	20 16,53 %	20 16,53 %	101 83,47 %
Nodetype	41	41 100,00 %	27 93,10 %	1 3,45 %	1 3,45 %	1 3,45 %	28 96,55 %
T-lemma	41	41 100,00 %	26 89,66 %	3 10,34 %	0 0,00 %	0 0,00 %	29 100,00 %
is_member	124	124 100,00 %	74 61,16 %	0 0,00 %	47 38,84 %	47 38,84 %	74 61,16 %
aux.rf (A)	52	52 100,00 %	3 8,33 %	33 91,67 %		33 91,67 %	3 8,33 %
Uzel (N)	41		29 70,73 %	12 29,27 %			

Tabulka 5.19: Vícenásobná koordinace: výsledky modulu

Na tektogramatické rovině nejprve vytvoříme nový uzel pro poslední čárku v posloupnosti, který se stane kořenem koordinace. Jeho *nodetype* nastavíme na `coap` a *t-lemma* na `#Comma`. Do atributu *aux.rf* přidáme reference na všechny ostatní čárky vyskytující se v koordinaci. Pokud pro ostatní čárky existovaly samostatné *t*-uzly nebo *aux.rf* reference u jiných uzlů, smažeme je. Na nový uzel pověsíme všechny členy koordinace (*t*-uzly odpovídající hlavám neterminálů nalezených ve složkovém stromě) a nastavíme jim příznak *is\_member*.

Podrobné vyhodnocení ukazuje, že nejde o příliš častý jev (187 výskytů v celém korpusu), přesto ale není natolik vzácný, abych hotový modul vyloučil z používání. Sice zdaleka není bezchybný (u *starých dat*), ale je prokazatelně přínosný, jak ukazuje tabulka 5.19. Výsledky na *nových datech* jsou sice téměř dokonalé, ale ze tří správně anotovaných výskytů nelze vyvozovat žádné závěry. Za špatné hodnocení vyplňování atributu *aux.rf* opět může nedokonalá anotace, jelikož zejména s interpunkcí zatím není zacházeno dobře. Reference většinou úplně chybí. Správné zpracování vícenásobné koordinace ukazuje obrázek 5.18.



Obrázek 5.18: Vícenásobná koordinace: struktura v PTB-WSJ a výsledek na tektogramatické rovině (výraz „no crack dealers, no dead-eyed men selling st., no one curled up ...“)

## 5.24 Cizí názvy

V této části popíšeme první ze čtyř modulů, které se věnují pojmenovaným entitám. Všechny čtyři musí proběhnout v určitém pořadí, jinak dochází k problémům. Modul pro zpracování cizích názvů musí být první, protože vytváří struktury, se kterými pracují/počítají další moduly.

Na tektogramatické rovině se cizí názvy (např. „Los Angeles“, „Sao Paulo“) reprezentují specifickým a v jistém smyslu jednoduchým způsobem. Nejprve je nutné vytvořit speciální generovaný uzel označující cizí název, na který se posléze pověsí všechny uzly zastupující jednotlivá slova názvu.

Postup popsany níže nevyhledává všechny cizí názvy. Pracuje se se seznamy „podezřelých“ slov sestavených na základě analýzy dat a zaměřuje se na dvou a tříslavné názvy, které ovšem tvoří většinu všech cizích názvů.

Samotná identifikace je docela jednoduchá. Ve složkovém stromě hledáme dva sousední terminály (sourozence s rodičem  $P_1$ ) takové, že *forma* levého z nich je v seznamu, který obsahuje slova jako „los“, „san“, „del“, „costa“, „rio“ nebo „marina“. Pokud má pravý ze dvou terminálů jako *formu* jedno ze slov „la“, „cru“, „de“, „del“, „bala“, „bel“, můžeme cizí název rozšířit na tři sousední terminály, pokud dva již nalezené terminály mají ještě jednoho bezprostředního pravého bratra.

Na tektogramatické rovině vytvoříme dříve zmíněný generovaný uzel s těmito atributy:

- *t-lemma* = #Forn
- *nodetype* = list
- *is\_generated* = 1 (příznak nastaven)

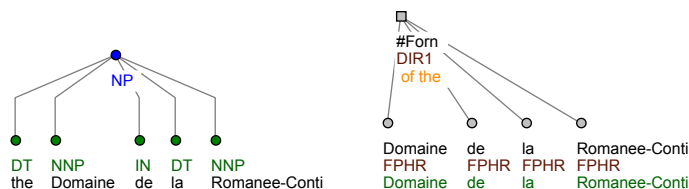
Vyhledáme *t*-uzly odpovídající terminálům nalezeným ve složkovém stromě, všem přidělíme *funktor* FPHR a pověsíme je na nový umělý uzel #Forn, pro který ovšem stále nemáme rodiče. Tím bude původní rodič toho z převěšených *t*-uzlů, který měl před aplikací modulu nejvyšší postavení ve struktuře stromu – vedla z něho nejkratší cesta do kořene. Smysl takové definice<sup>11</sup> může být poněkud nezřetelný, mnohem pochopitelnější by bylo, kdybychom vybrali rodiče toho uzlu, který ve svém podstromu

<sup>11</sup>Slovo „definice“ v tomto odstavci používám pouze pro přibližné popisy. Domnívám se však, že přesné a korektní definice by spíše zhoršily čitelnost textu.

obsahuje všechny ostatní převěšované uzly. Je ovšem zřejmé, že v případě, kdy je možné uplatnit druhou definici, jsou obě definice ekvivalentní. První (méně zřetelná) má ale tu výhodu, že ji můžeme použít i v situaci, ve které druhá definice selhává – tj. v situaci, kdy ani jeden ze zasažených t–uzlů neobsahuje ve svém podstromu všechny ostatní zasažené t–uzly.

Modul dělá ještě jednu věc. Pokud je prvním synem neterminálu  $P_1$  terminál DT, přidáme referenci na něj do atributu *aux.rf* nového uzlu #Forn (např. výraz „a tour de force“).

Výsledky modulu (viz tabulka 5.20) jsou skvělé. Není se čemu divit, protože popsaný jev je jednoduchý a nedá se toho na něm moc zkazit. Modul patří do kategorie těch, které řeší jednoduché a časté věci (1 517 výskytů v korpusu), což usnadňuje práci anotátorům. A vůbec nemusí vadit, když není korektně zpracován celý cizí název, protože to hlavní – vytvoření nového uzlu jako kořene struktury – se provede správně. Převěšení případných zbývajících uzlů a přiřazení *funktorů* zabere minimum času. Příklad cizího názvu je na obrázku 5.19.



Obrázek 5.19: Cizí názvy: struktura v PTB–WSJ a výsledek na tektogramatické rovině (výraz „*The six wines of the Domaine de la Romanee-Conti*“)

## 5.25 Boston, Massachusetts

Druhý ze čtveřice modulů určených ke zpracování pojmenovaných entit se snaží anotovat výrazy typu „Boston, Massachusetts“ ve funkci rozvití, které lokalizuje rozvíjený větný člen. Například ve frázi „a Palo Alto, Calif., computer maker“ je výrobce počítačů blíže určen tím, že je umístěn do Palo Alto v Kalifornii. Na tomto příkladu je vidět i důvod, proč tento modul musí následovat až po tom, který je popsán v části 5.24. Název „Palo Alto“ je z pohledu angličtiny cizí a musí být reprezentován předepsaným způsobem. Pokud by moduly byly spuštěny v opačném pořadí, tak by si spíš škodily.

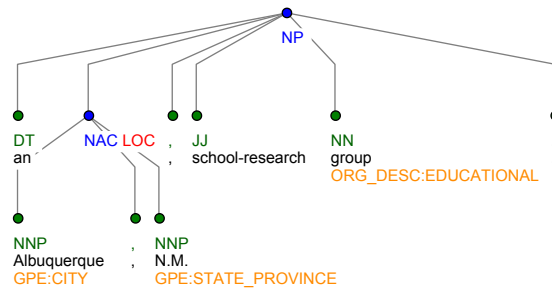
Nejprve popíše strukturu ve složkovém stromu. Hledáme neterminál NP nebo NML, který má mezi svými syny neterminál NAC (označme  $P_1$ ). Uzel  $P_1$  musí mít mezi pravými sourozenci neterminál NP nebo terminál s *tagem* ve tvaru NN\* (označme  $P_2$ ). Uzel  $P_1$  musí mít právě tři nebo čtyři syny. Druhým z nich musí být terminál zastupující symbol čárky nebo levé kulaté závorky (-LRB-). Je-li přítomen čtvrtý syn, musí jít o terminál reprezentující znak čárky nebo pravé kulaté závorky (-RRB-). Pokud  $P_1$  čtvrtého syna nemá a terminál umístěný v podstromu uzlu  $P_2$  nejvíce vlevo<sup>12</sup> splňuje předchozí požadavky, můžeme tento terminál pro potřeby modulu považovat za čtvrtého syna uzlu  $P_1$  (příklad viz obrázek 5.20). Třetí syn  $P_1$  musí splňovat jednu ze tří následujících podmínek:

<sup>12</sup>Takový uzel najdeme tak, že z  $P_2$  sestupujeme do syna umístěného úplně vlevo tak dlouho, dokud nenarazíme na terminál

Nová data							
Výskyty	51						
Spárováno	147 z 153 (96,08 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
aux.rf (D)	1		1	0			
			100,00 %	0,00 %			
Funktor	102	102	97	2	1	1	99
		100,00 %	97,00 %	2,00 %	1,00 %	1,00 %	99,00 %
Nodetype	51	51	47	0	0	0	47
		100,00 %	100,00 %	0,00 %	0,00 %	0,00 %	100,00 %
T-lemma	51	51	47	0	0	0	47
		100,00 %	100,00 %	0,00 %	0,00 %	0,00 %	100,00 %
Uzel (N)	51		47	4			
			92,16 %	7,84 %			
Rodič	102	102	92	6	2	2	98
		100,00 %	92,00 %	6,00 %	2,00 %	2,00 %	98,00 %
is_generated	51	51	47	0	0	0	47
		100,00 %	100,00 %	0,00 %	0,00 %	0,00 %	100,00 %
aux.rf (A)	1	1	1	0		0	1
		100,00 %	100,00 %	0,00 %		0,00 %	100,00 %
Stará data							
Výskyty	269						
Spárováno	789 z 811 (97,29 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
aux.rf (D)	10		8	2			
			80,00 %	20,00 %			
Funktor	539	539	519	11	6	6	530
		100,00 %	96,83 %	2,05 %	1,12 %	1,12 %	98,88 %
Nodetype	269	269	248	2	0	0	250
		100,00 %	99,20 %	0,80 %	0,00 %	0,00 %	100,00 %
T-lemma	269	269	249	1	0	0	250
		100,00 %	99,60 %	0,40 %	0,00 %	0,00 %	100,00 %
Uzel (N)	269		250	19			
			92,94 %	7,06 %			
Rodič	539	539	498	27	11	11	525
		100,00 %	92,91 %	5,04 %	2,05 %	2,05 %	97,95 %
is_generated	269	269	249	0	1	1	249
		100,00 %	99,60 %	0,00 %	0,40 %	0,40 %	99,60 %
aux.rf (A)	9	9	7	2		2	7
		100,00 %	77,78 %	22,22 %		22,22 %	77,78 %

Tabulka 5.20: Cizí názvy: výsledky modulu

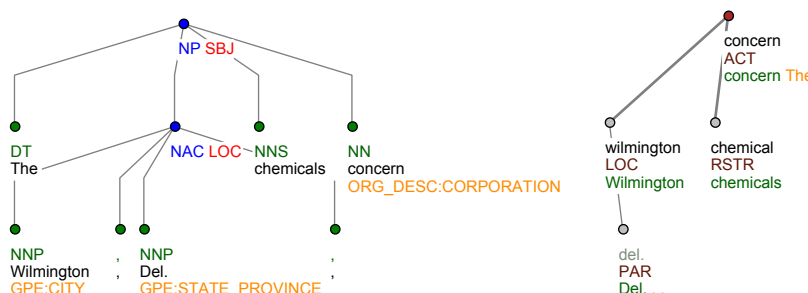
- Jde o neterminál NP nebo NML a všechny terminály, které se nacházejí v jeho podstromu, mají BBN-tag<sup>13</sup> GPE:STATE\_PROVINCE.
- Jde o terminál, jehož BBN-tag je GPE:STATE\_PROVINCE.
- Jde o terminál, který reprezentuje římské číslo (i řadové) a který nemá žádný BBN-tag.



Obrázek 5.20: Boston, Massachusetts: čtvrtý syn neterminálu  $P_1$  převzatý od  $P_2$  (výraz „an Albuquerque, N.M., school-research group“)

Na tektogramatické rovině najdeme uzly, které reprezentují prvního a třetího syna uzlu  $P_1$ , případně hlavy jejich podstromů (označme  $T_1$  a  $T_2$ ). Pokud jako  $T_1$  nalezneme uzel s *funktorem* FPHR, označíme jako  $T_1$  jeho rodiče.<sup>14</sup> Následně uzlům  $T_1$  a  $T_2$  přidělíme *funktory* LOC, respektive PAR a  $T_2$  pověsíme na  $T_1$ . Všem uzlům z podstromu uzlu  $T_2$  i jemu samotnému nastavíme příznak *is\_parenthesis*. Nakonec zbývají druhý a možný čtvrtý syn uzlu  $P_1$ . Ty uvedeme v atributu *aux.rf* uzlu  $T_2$  jako reference.

Výsledky, které uvádím v tabulce 5.21, ukazují, že určování *funktory*, rodiče i příznaku *is\_parenthesis* je velmi dobré. To je ale na druhou stranu pochopitelné, jelikož modul klade vysoké nároky na strukturu ve složkovém stromě, a kvůli tomu je celkový počet výskytů popsaného jevu v korpusu velmi nízký – pouze 239. Výsledky pro atribut *aux.rf* jsou sporné a vyžadovaly by ruční vyhodnocení. Obrázek 5.21 ukazuje příklad úspěšného použití modulu.



Obrázek 5.21: Boston, Massachusetts: struktura v PTB-WSJ a výsledek na tektogramatické rovině (výraz „The Wilmington, Del., chemicals concern“)

<sup>13</sup>Viz. část 3.3.1

<sup>14</sup>Zde využíváme předchozí běh modulu 5.24. Rodičem by měl být generovaný uzel #Forn.

Nová data							
Výskyty	18						
Spárováno	36 z 36 (100,00 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
Funktor	36	36	34	2	0	0	36
		100,00 %	94,44 %	5,56 %	0,00 %	0,00 %	100,00 %
Rodič	18	18	18	0	0	0	18
		100,00 %	100,00 %	0,00 %	0,00 %	0,00 %	100,00 %
is_parenthesis	18	18	18	0	0	0	18
		100,00 %	100,00 %	0,00 %	0,00 %	0,00 %	100,00 %
aux.rf (A)	36	0	14	22		14	22
		0,00 %	38,89 %	61,11 %		38,89 %	61,11 %
Stará data							
Výskyty	69						
Spárováno	139 z 139 (100,00 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
aux.rf (D)	1		1	0			
			100,00 %	0,00 %			
Funktor	138	138	112	18	8	8	130
		100,00 %	81,16 %	13,04 %	5,80 %	5,80 %	94,20 %
Rodič	69	69	64	4	1	1	68
		100,00 %	92,75 %	5,80 %	1,45 %	1,45 %	98,55 %
is_parenthesis	69	67	65	0	4	6	63
		97,10 %	94,20 %	0,00 %	5,80 %	8,70 %	91,30 %
aux.rf (A)	137	1	92	45		91	46
		0,73 %	67,15 %	32,85 %		66,42 %	33,58 %

Tabulka 5.21: Boston, Massachusetts: výsledky modulu

## 5.26 Funktor NE

Třetí z modulů určených ke zpracování pojmenovaných entit zavádí do dat *funktor* NE. Stejně jako v mnoha předchozích případech je motivací mimo jiné i to, aby měli anotátoři méně práce. V průběhu anotace jsme zjistili, že vlastně nevíme, jak na tekto-gramatické rovině anotovat složité pojmenované entity typu „U.S. Assistant Treasury Secretary David Mulford“. Ze všech slov v této frázi mají vzniknout t-uzly. Ale jaká by měla být jejich struktura? Na tuto otázku jsme neznali odpověď a nebyla vhodná situace pro hledání trvalého řešení, proto jsme problém odsunuli na později a zavedli dočasné opatření. Všechny uzly vyjadřující část nějaké pojmenované entity dostanou nový *funktor* NE (named entity), a jakmile bude mít nějaká skupina uzlů *funktor* NE, není třeba měnit strukturu stromu.

Právě tady se skrývá usnadnění práce anotátorům. Pokud není jasné, jak něco dělat, nebudeme to teď dělat vůbec a necháme to na později. Funktory NE samozřejmě přiřazujeme automaticky a jejich výskyt říká anotátorovi, že příslušné uzly může ignorovat.

Modul pracuje jednoduše. Obsahuje definici seznamu BBN-*tagů*, podle kterých *funktor* NE přiřazujeme. Potom stačí procházet jednotlivé uzly a slepě přiřazovat *funktor* NE těm, které jsou označeny jedním z BBN-*tagů* uvedených v seznamu. Jedinou výjimkou jsou struktury pro cizí názvy s uzlem #Forn vytvářené modulem popsáním v části 5.24. Uzly s *funktořem* FPHR zůstávají nezměněny, k nahrazení *funktoru* dochází pouze v kořeni struktury pro cizí název. Z toho důvodu musí být tento modul spuštěn až po modulu 5.24.

Vyhodnocení úspěšnosti v tomto případě postrádá smysl. Pokud modul někde změní

Nová data							
Výskyty	107						
Spárováno	110 z 110 (100,00 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
Funktor	110	110	93	12	5	5	105
		100,00 %	84,55 %	10,91 %	4,55 %	4,55 %	95,45 %
Stará data							
Výskyty	533						
Spárováno	539 z 540 (99,81 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
Funktor	540	540	248	83	208	208	331
		100,00 %	46,01 %	15,40 %	38,59 %	38,59 %	61,41 %

Tabulka 5.22: Rozvití pomocí pojmenované entity: výsledky modulu

*funktor*, tak už to tak v drtivé většině případů i zůstane, protože popravdě řešeno jde spíše o „deanotaci“. Činnost tohoto modulu musím vynechat i z jakýchkoliv souhrnných měření, protože by hrubě ovlivnil a zdeformoval výsledky. Přesnost blížící se 100 % v kombinaci s ohromným počtem výskytů (39 058 v celém korpusu) – to jsou čísla, která se nedají použít společně s žádným jiným výsledkem některého z modulů.

## 5.27 Rozvití pomocí pojmenované entity

Poslední ze čtveřice modulů zaměřených na pojmenované entity poprvé výrazně používá anotaci získanou z BBN korpusu. Pro jeho správné fungování je nutné zachovat správné pořadí zpracování dat – moduly 5.24 a 5.26 musí být spuštěny dříve.

V této části mě bude zajímat situace, kdy je jedna pojmenovaná entita rozvíjena druhou (např. „White House officials“ nebo „John F. Kennedy International Airport“). Podle typu použitých entit ve vztahu *rozvíjející* – *rozvíjený* je možné rozvíjícímu větnému členu přidělit *funktor*. Typem pojmenované entity v tomto případě myslím BBN-tag přidělený příslušnému uzlu. Označím-li množinu všech BBN-tagů jako  $A$ , potom můžu instrukce pro program vyjádřit ve formě relace  $R \subseteq A^2$ , kde  $\forall(a, b) \in R$  znám *funktor*, který je třeba přiřadit rozvíjícímu větnému členu. Nebudu zde uvádět celou relaci  $R$ , ale pouze přibližný popis. Rozvíjeným členem jsou převážně různé organizace nebo lidské stavby jako mosty, silnice nebo budovy (BBN-tagy `ORG_DESC:*` a `FAC_DESC:*`). Typem rozvíjícího členu je téměř vždy některý z pětice `GPE:CITY`, `GPE:COUNTRY`, `GPE:OTHER`, `GPE:STATE_PROVINCE` a `PERSON`. Funktorem je většinou `LOC`.

Pouze BBN-tagy však nestačí, protože i struktura<sup>15</sup> musí splňovat nějaké požadavky. Mezi syny neterminálu NP nebo NML hledáme uzel s příznakem *is\_head*<sup>16</sup>, kterým ovšem nesmí být vlastní jméno<sup>17</sup>. Tím získáme rozvíjený větný člen. Rozvíjícím členem je jeho bezprostřední levý sourozenec, kterým opět nesmí být vlastní jméno.

Když teď máme oba členy, zjistíme jejich BBN-tagy. Mají-li nějaké, podíváme se, jestli se příslušná dvojice nachází v relaci  $R$ . Pokud ano, tak rozvíjícímu  $t$ -uzlu přiřadíme správný *funktor*. Z tohoto přímočarého postupu existují dvě výjimky. Pokud je rozvíjícím členem cizí název (viz část 5.24), přidělujeme *funktor* generovanému uzlu s  $t$ -lemmatem `#Forn`. Nikdy neměníme *funktor* `NE` (viz. část 5.26).

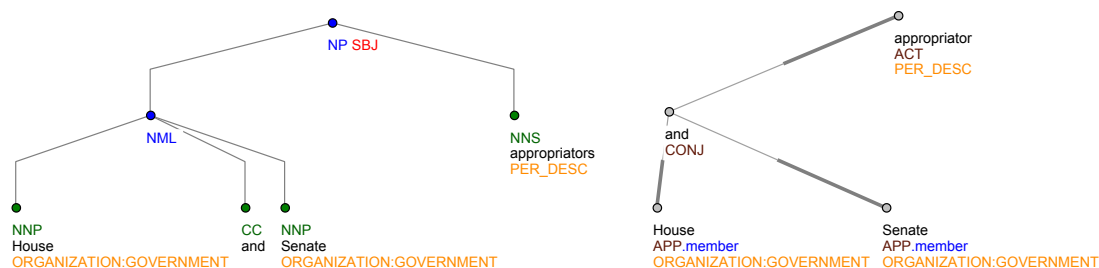
Není snadné interpretovat výsledky modulu, které uvádím v tabulce 5.22. U *nových*

<sup>15</sup>Samozřejmě mám na mysli strukturu ve složkovém stromu.

<sup>16</sup>Pozor: to je něco jiného, než když mluvíme o „hlavě neterminálu“

<sup>17</sup>(proper noun – tag `NNP` nebo `NNPS`)

*dat* je úspěšnost docela dobrá a mnohem vyšší, než u *starých dat*, kde došlo k velkému množství chyb. Další analýzou jsem zjistil, že přibližně tři čtvrtiny všech chyb (216) vznikly tak, že modul přidělil nějaký jiný *funktor* uzlu, u kterého anotátor ponechal RSTR. Všechny chyby druhého stupně (208) vznikly tímto způsobem. Před použitím tohoto modulu mají všechny zasažené t–uzly až na výjimky (méně než 1%) právě *funktor* RSTR nebo ???,<sup>18</sup> proto se domnívám, že anotátoři dříve nerozlišovali dostatečně jemně jednotlivé typy rozvití a spousta akcí modulu je jako chyba označena neprávem. Obrázek 5.22 ukazuje příklad, který nebyl vyhodnocen jako chyba. Jev má v celém korpusu 2 202 výskytů.



Obrázek 5.22: Rozvití pomocí pojmenované entity: struktura v PTB–WSJ a výsledek na tektogramatické rovině (výraz „*House and Senate appropriators*“)

## 5.28 Čísła

Wall Street Journal jsou noviny věnující se ekonomice, takže obsahují spoustu různých číselných výrazů. Vyskytují se v nich běžné číslice, procenta, zlomky, čísla zapisovaná slovy a další číselné výrazy všeho druhu, častokrát složené z více částí. Všechny tyto jevy je třeba nějak smysluplně a hlavně konzistentně reprezentovat na tektogramatické rovině. Automatické zpracování je obzvláště v tomto případě výhodné i z toho důvodu, že práce s číselnými výrazy je myšlenkově velmi snadná, ale často také pracná. Například složené zlomky vyskytující se ve tvaru „3 2/5“ se na tektogramatické rovině zachycují pomocí pěti uzlů, jejichž podoba je přesně daná a není na ní co zkazit. V některých větách se může vyskytovat i pět až deset zlomků. Při jejich anotaci by se z anotátora stával tupý stroj. Anotátoři by se navíc určitě nevyhnuli chybám, a tím by se zvyšovala inkonzistence korpusu.

V následujících odstavcích popíši jednotlivé typy číselných výrazů. Vysvětlím, jak je rozeznat a jak je zachytit na t–rovině. Výsledky uvedu na závěr pro všechny číselné výrazy najednou. Množiny jevů spadajících pod jednotlivé typy číselných výrazů nejsou vždy disjunktní a není to třeba. Jednotlivé typy uvádím v pořadí, v jakém je zpracovává i modul. Začínám nejobecnějšími věcmi a postupně pokračuji ke konkrétnějším. Práce na jednom jevu často začíná tam, kde skončila u jiného. Také se může stát, že jeden číselný výraz je zpracován vícekrát a pokaždé jinak. Zvolené pořadí však zajišťuje, že poslední verze je ta správná. Na pořadí tedy velmi záleží. Jeho změna by mohla značně ovlivnit výsledek.

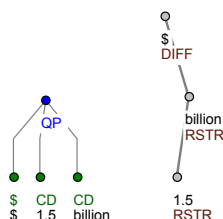
<sup>18</sup>??? je speciální hodnota *funktoru*, která může být použita pouze při generování dat. Označuje zcela neznámý *funktor*. V datech odevzdaných anotátory se již nesmí vyskytovat.

## Posloupnosti číslovek

Je-li v textu za sebou několik číslovek, které společně rozvíjejí nějaké substantivum (např. „one million people“), nesmí všechny číslovky viset na stejné úrovni na substantivu, nýbrž postupně jedna na druhé. V uvedeném příkladu by slovo „million“ viselo na „people“ a slovo „one“ na „million“. Jak tento jev rozeznat ve složkovém stromě? Kandidátem je každý neterminál. Uspořádáme jeho syny do posloupnosti a budeme v ní hledat nepřerušenu podposloupnost délky alespoň dvě s následujícími vlastnostmi:

- Všechny prvky až na první a poslední musí být tvořeny terminály CD
- První a poslední prvek musí být tvořen některým z terminálů CD, NN\*
- V posloupnosti může být nejvýše jeden terminál NN\*

V případě, že odhalíme takovou posloupnost, budeme pokračovat na tektogramatické rovině. Najdeme t–uzly pro všechny terminály obsažené v posloupnosti, případně vytvoříme nové. Nalezneme hlavní, nejvýše položený uzel nově formované struktury: Pokud je prvním prvkem posloupnosti terminál NN\*, stane se jeho tektogramatický protějšek hlavním uzlem struktury, v ostatních případech bude hlavním ten t–uzel, který reprezentuje poslední prvek posloupnosti (i tehdy, když je to číslovka). Ze zbytku posloupnosti vždy vezmeme uzel, který je na jejím konci, pověsíme ho na naposledy zavěšený uzel a přidělíme mu *funktor* RSTR. Při prvním opakování věšíme poslední prvek posloupnosti na zvolený hlavní uzel. Tímto postupem vznikne „řetízek“.<sup>19</sup> Viz. obrázek 5.23.



Obrázek 5.23: Čísla – posloupnosti číslovek: struktura v PTB–WSJ a výsledek na tektogramatické rovině (výraz „\$ 1.5 billion“)

## Koordinace dvou číslovek bez oddělovače

Pokud se v textu hned za sebou vyskytují dvě číslovky, které navíc vyhovují i dalším požadavkům, potom mají být na tektogramatické rovině zachyceny pomocí koordinace a ne pouhé závislosti. Prvním požadavkem je, aby číslovky (terminály CD) byly ve složkovém stromě bezprostředními sourozenci. Za druhé musí číslovky splňovat jedno z následujících pravidel:

- První číslovka je vyjádřena číslem, druhá se skládá ze dvou čísel oddělených lomítkem – složený zlomek

<sup>19</sup>Řetízky bohužel nejsou nijak zvlášť dlouhé, protože v celém korpusu se nevyskytuje ani jeden případ, kdy by takto byly zřetězeny více než dvě číslovky.

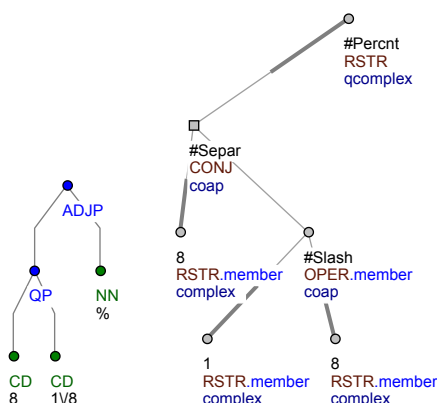
- Obě číslovky jsou vyjádřeny jedním z explicitně povolených slov (např. „three“, „sixteen“, „eighty“, „billion“)

Při splnění všech podmínek se můžeme pustit do úprav *t*-roviny. Nejprve vytvoříme umělý koordinační uzel s těmito atributy:

- *t-lemma* = #Separ
- *nodetype* = coap
- *funktor* = CONJ
- *is\_generated* = 1

Na něj pověsíme oba číselné *t*-uzly, kterým nastavíme příznak *is\_member*. Zbývá nalézt rodiče pro nově vytvořený uzel. Většinou jím bude původní rodič pravé číslovky. Pouze v případě, že pravá číslovka se nachází v podstromu levé číslovky, zvolíme původního rodiče levé číslovky. Příklad koordinace je na obrázku 5.24.

Povšimněme si, že podle uvedených podmínek by všechny výskyty popsané koordinace měly být zpracovány i předchozím pravidlem jako posloupnost číslovek, které ve skutečnosti funguje jako jistá forma „back-off“ přístupu: zachytí všechno a zvláštní případy budou dále zpracovány podrobněji.



Obrázek 5.24: Čísla – složený zlomek vyjadřující počet procent: struktura v PTB-WSJ a výsledek na tektogramatické rovině (výraz „8 1/8 %“). Struktura vzniká postupnými úpravami čtyř výše popsaných typů číselných výrazů: Posloupnost číslovek, Koordinace dvou číslovek bez oddělovače, Koordinace dvou číslovek s oddělovačem, Symbol procenta.

## Symbol procenta

Symbol procenta je na tektogramatické rovině reprezentován uzlem, jehož *nodetype* je *qcomplex* a *t-lemma* je #Percnt. Modul nastavuje právě tyto dva atributy.

Tato část časem ztratila význam, jelikož jsem vytvořil modul výhradně pro opravy atributu *nodetype* (viz část 5.2) a do *TectoMT* jsem zařadil blok, který různé symboly nahrazuje zástupnými *t-lemmaty*.

## Desetinná čísla

V PTB–WSJ se desetinná čísla uvádějí oběma standardními způsoby – s desetinnou tečkou i čárkou. V PEDT jsou desetinná čísla s tečkou ponechána beze změny, desetinná čárka je nahrazena symbolem podtržítka: 1,25 → 1\_25. Změna se týká atributu *t-lemma*.

## Koordinace dvou číslovek s oddělovačem

Na rozdíl od výše popsané koordinace číslovek bez oddělovače se tato část bude týkat výrazů, ve kterých jsou od sebe dvě číslovky nějakým způsobem odděleny. Ve složkovém stromě jde vždy o jediný terminál CD s povrchovou formou jednoho ze čtyř následujících druhů:

1. Čísla „twenty-\*“, „thirty-\*“, . . . , „ninety-\*“
2. Jakýkoliv výraz s pomlčkou kromě těch, které patří k druhu č.1 (např. „Oct. 2-8“)
3. Zlomek – dvě čísla oddělená lomítkem
4. Čísla oddělená dvojtečkou (např. zápis času)

Až na pár detailů je reprezentace všech čtyř druhů výrazů na tektogramatické rovině stejná. Modul je vytvořen tak, aby navázal na práci systému *TectoMT* – očekává existenci jednoho *t*-uzlu, který reprezentuje celý číselný výraz (označme  $T_1$ ). Nejprve vytvoříme nový uzel, který se stane kořenem koordinace a bude reprezentovat samostatný oddělovač. Atributy *nodetype* a *is\_generated* nastavíme podle očekávání: *coap* a 0. Hodnotu atributu *is\_member* zkopírujeme z uzlu  $T_1$ , protože pokud je výraz jako takový členem koordinace, musíme tuto informaci uchovávat v nově vytvořeném uzlu řídicím koordinaci jednotlivých složek výrazu. *Funktor* a *t-lemma* nastavíme v závislosti na druhu výrazu podle tabulky 5.23.

Druh	1	2	3	4
<b>funktor</b>	CONJ	OPER	OPER	OPER
<b>t-lemma</b>	#Dash	#Dash	#Slash	#Colon

Tabulka 5.23: Čísla – koordinace dvou číslovek s oddělovačem: hodnoty atributů *t-lemma* a *funktor*

Dále vytvoříme kopii původního uzlu, čímž zachováme maximum předchozích informací, např. *funktor*. Dostaneme tak dva uzly, kterým musíme změnit pouze *t-lemma*, aby jeden uzel reprezentoval levou část výrazu a druhý pravou. Oba dva uzly pověsíme na koordinační uzel, který jsme vytvořili, a nastavíme u nich příznak *is\_member*. Příklad výrazu typu 3 (zlomek) je na obrázku 5.24.

Souhrnné výsledky (viz tabulka 5.24) ukazují, že modul pracuje dobře a vzhledem k ohromnému počtu výskytů (44 105 v celém korpusu – nejvíce ze všech) je jeho přínos pro anotaci značný. Zároveň se ukazuje značná disproporce mezi výsledky na *nových* a *starých datech*. Rozdíl se projevuje nejvíce u nově vytvořených uzlů – tj. při koordinaci číslovek. Anotátoři tyto uzly většinou sami nevytvářeli, ale jakmile to za ně udělal automat, byli s výsledkem spokojeni a zpravidla ho neměnili.

Nová data							
Výskyty	2 079						
Spárováno	2 151 z 2 172 (99,03 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
Funktor	1 897	163	1 649	227	0	1 508	368
		8,59 %	87,90 %	12,10 %	0,00 %	80,38 %	19,62 %
is_generated	350	65	328	0	1	276	53
		18,57 %	99,70 %	0,00 %	0,30 %	83,89 %	16,11 %
Rodič	1 827	883	1 732	51	23	966	840
		48,33 %	95,90 %	2,82 %	1,27 %	53,49 %	46,51 %
Nodetype	620	160	322	277	0	180	419
		25,81 %	53,76 %	46,24 %	0,00 %	30,05 %	69,95 %
T-lemma	1 090	421	1 061	8	0	669	400
		38,62 %	99,25 %	0,75 %	0,00 %	62,58 %	37,42 %
is_member	354	258	331	5	8	94	250
		72,88 %	96,22 %	1,45 %	2,33 %	27,33 %	72,67 %
Uzel (N)	255		239	16			
			93,73 %	6,27 %			
Stará data							
Výskyty	10 238						
Spárováno	9 293 z 10 270 (90,49 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
Funktor	8 830	726	6 980	928	0	6 795	1 113
		8,22 %	88,27 %	11,73 %	0,00 %	85,93 %	14,07 %
is_generated	1 510	289	669	4	14	583	104
		19,14 %	97,38 %	0,58 %	2,04 %	84,86 %	15,14 %
Rodič	8 554	3 857	6 957	424	251	4 816	2 816
		45,09 %	91,16 %	5,56 %	3,29 %	63,10 %	36,90 %
Nodetype	2 909	698	577	1 452	1	380	1 650
		23,99 %	28,42 %	71,53 %	0,05 %	18,72 %	81,28 %
T-lemma	5 237	1 942	3 600	197	524	3 710	611
		37,08 %	83,31 %	4,56 %	12,13 %	85,86 %	14,14 %
is_member	1 554	1 133	558	121	207	448	438
		72,91 %	62,98 %	13,66 %	23,36 %	50,56 %	49,44 %
Uzel (N)	1 105		419	686			
			37,92 %	62,08 %			

Tabulka 5.24: Čísla: výsledky modulu



# Kapitola 6

## Automatické kontroly při anotaci

Jedním ze způsobů, jak zvýšit kvalitu dat, jsou kontroly prováděné přímo v průběhu anotace a před jejím odevzdáním. Pro potřeby PEDT jsem vyvinul systém maker, které anotátorovi umožňují pohodlně kontrolovat jeho práci přímo v anotačním prostředí programu *TrEd* (viz <http://ufal.mff.cuni.cz/~pajas/tred>). Kontroly na první pohled nesouvisí s tématem této práce, protože samy neprovádějí žádné změny a pro anotátora z krátkodobého hlediska znamenají spíše více práce, jelikož ho nutí opravovat chyby, které by v datech normálně nechal. Alespoň základní pochopení toho, jak kontroly pracují, je však nutné kvůli kapitole 7, ve které je popsána automatická anotace prováděná až po odevzdání dat anotátorem, která využívá právě kontroly.

Základem systému bylo 32 kontrolních modulů, které vytvořil Jan Štěpánek [24]. Jejich základ vznikl při přípravě PDT a nyní se používají ke kontrole odevzdaných dat z paralelně vyvíjeného českého protějšku PEDT v rámci PCEDT. Moje práce se skládala ze dvou kroků. Moduly jsem přizpůsobil pro anotaci angličtiny (původně byly navrženy pro kontrolu anotace českých vět) a vytvořil jsem sadu maker pro *TrEd*, pomocí kterých mohou anotátoři pohodlně používat kontroly při práci.

Celý systém je samozřejmě předmětem neustálého vývoje. Původní kontroly upravuji na základě zpětné vazby od anotátorů a také postupně implementuji některé zcela nové. Momentálně je jich už 43 a jejich počet dál poroste.

Kontroly se věnují různým aspektům dat. V současné chvíli jsou spíše pro formu rozděleny do čtyř skupin: kontroly koordinací, odkazů mezi vrstvami, valence a struktury. Poslední skupina je velmi široká, patří do ní všechno, co se nevešlo do předchozích tří. Většina kontrol pracuje s izolovanými uzly. Některé je však možné spustit pouze na celé stromy nebo dokonce pouze na celý soubor. Například odkazy mezi rovinami mohou překračovat hranice vět, proto je třeba při jejich kontrole pracovat s celým souborem najednou.

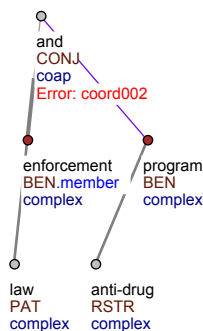
Anotátor, čili uživatel, má k dispozici několik funkcí, které je možné v programu *TrEd* vyvolat klávesovými zkratkami. Mezi ně patří hlavně tři způsoby kontroly, které se liší svým rozsahem: je možné kontrolovat jediný uzel, jeden strom/větu, nebo celý soubor. Pokaždé se spouští všechny dostupné<sup>1</sup> kontroly, rozdíl je pouze v rozsahu kontrolovaných dat (uzel – strom – soubor). V případě nalezení chyby se u příslušného uzlu zobrazí červená správa s kódem chyby a s případnými doplňujícími informacemi. U jednoho uzlu může být i více chyb. V takovém případě se v hlavním okně *TrEdu* zobrazí kvůli přehlednosti pouze první chyba ze seznamu. Kompletní výpis je samozřejmě snadno k dispozici jinde. Anotátor může ještě používat funkci pro přechod k nejbližší

---

<sup>1</sup>Při kontrole uzlu se nespouštějí kontroly určené pro celé stromy a pro celý soubor, při kontrole stromu se nespouštějí kontroly určené pro celý soubor.

následující chybě a funkci, kterou je možné nalezenou chybu zrušit v případě, že o chybu ve skutečnosti nejde.

V žádném případě zde nebudu popisovat všechny používané kontroly, protože to není náplní této práce. Uvedu pouze jeden ilustrativní příklad. Základem kontroly je pravidlo říkající: Kořen souřadné struktury (*nodetype = coap*) má alespoň dva přímé potomky s nastaveným příznakem *is\_member*. Kontrolní modul napsaný podle tohoto pravidla dostane jako vstupní parametr kontrolovaný uzel. Je-li jeho typem *coap*, spočte přímé potomky s příznakem *is\_member*. Pokud je jich méně než dva, ohlásí chybu. Obrázek 6.1 ukazuje, jak taková situace vypadá.



Obrázek 6.1: Výsledek automatické kontroly počtu členů koordinace. U kořene souřadné struktury je hlášení o chybě, neboť mezi jeho přímými potomky je pouze jeden člen koordinace. Uzel reprezentující slovo „program“ má mít nastaven příznak *is\_member*. Podstromem je zachycen výraz „*law enforcement and anti-drug programs*“.

Na závěr několik slov o možném využití kontrol. Hlavním motivem k vývoji systému je samozřejmě kontrola anotace ještě před odevzdáním dat. Anotátoři nesmí odevzdat data, dokud po vyvolání kontroly celého souboru nedostanou odpověď, že nebyly nalezeny žádné chyby. Ovšem nabízejí se i jiné možnosti. Automatické kontroly se dají po domluvě s anotátory využít k automatické anotaci po odevzdání dat – konkrétněji v části 7.1. Údaje získané pomocí kontrolních modulů se také dají použít k přibližnému vyhodnocení práce anotátorů. Součástí pravidelně spouštěného programu vyhodnocujícího množství odvedené práce je funkce, která pro každou dokončenou sekci vypočte poměr počtu chyb odhalených kontrolami před anotací a počtu změn, které anotátor při anotaci provedl. Výsledné číslo sice může být zavádějící, ale zejména při delším sledování poskytuje cenné srovnání práce anotátorů.

## Kapitola 7

# Automatická anotace po odevzdání dat

V kapitolách 5 a 6 jsem popsal, jakým způsobem se v současné době snažím zdokonalovat data před a během ruční anotace pomocí automatických metod. Zbývá už tedy pouze jedna možnost – co se dá dělat tehdy, když manuální anotace skončí a anotátor odevzdá hotová data?

Jako první je třeba si uvědomit, že po skončení manuální anotace lze dělat pouze takové věci, u kterých je zaručeno, že budou správně. Mizí bezpečnostní prvek lidského anotátora, který byl přítomen u modulů pro automatickou anotaci prováděných jako jedna z fází přípravy dat. Tehdy bylo možné dělat věci, které nebyly vždy úplně spolehlivé, protože i kdyby se něco opravdu hodně pokazilo, vždycky byl v záloze člověk, který bude data podrobně anotovat a chyby opravit.

Prvním principem, na kterém se dají založit konkrétní metody, je spoléhání se na správný kontext. Při přípravě dat spousta metod (nebo pravidel, chcete-li) selhává proto, že chybně anotovaný jev se nachází uprostřed komplexnější struktury, která je také chybná. Konkrétní příčiny mohou být různé. Nemusí se podařit nalézt uzlu, který by se měl stát rodičem opraveného/vytvořeného uzlu nebo podstromu, protože ve stromě vůbec není nebo je na úplně špatném místě. Také se může stát, že nějaký jev připouští omezený počet podob svého bezprostředního okolí (rodič, potomci), a na tom je možné založit automatickou anotaci. Příslušné okolí se ale může nacházet v úplně jiném stavu, který není přípustný. Takto bych mohl pokračovat dál, obecně však lze podobné problémy charakterizovat slovy: „není se čeho chytit“.

Po skončení manuální anotace nastává situace, kdy je možné spolehnout se na správnost kontextu. Stačí zvolit problematický jev, například nějaký zdroj inkonzistence, uplatnit všechny myslitelné předpoklady a pohodlně vytvořit program, který uvede data do pořádku. Nápad vypadá slibně, ale tato sympatická teorie má jednu vážnou trhlinu. Vedle sebe stojí dva neslučitelné předpoklady. Podle prvního jsou data zcela správně. Podle druhého je v nich stále něco špatně, protože jinak by přece nemělo smysl zabývat se nějakou automatickou anotací po dokončení té manuální.

Nabízí se otázka, jestli vůbec data po odevzdání anotátorem obsahují nějaké chyby nebo ne. Má vůbec smysl snažit se je ještě nějak vylepšit? Řekl bych, že nějaké chyby se najdou vždycky, další vylepšování už může být sporné. Pojd'me ale zpět k hlavnímu smyslu této práce. Cílem je minimalizovat čas potřebný k manuální anotaci. Představme si situaci, kdy anotátor ví, že musí udělat věc, která je nezbytná, ale také poměrně pracná. Pokud by mohl data zanechat v takovém stavu, ze kterého bude jednoznačně vyplývat, že je potřeba danou věc vykonat, může to za něj posléze udělat stroj.

Tím se dostávám ke druhému principu, podle kterého je možné navrhovat a implementovat metody automatické anotace po ukončení práce anotátora. Velmi důležitá je v tomto případě dohoda s anotátorem, který musí přesně vědět, co za něj posléze udělá stroj a jak toho dosáhnout. Tento princip jsme pro potřeby anotace PEDT také skutečně uvedli do praxe, zatím ale ve formě pouze jednoho pravidelně používaného modulu. V označení místa určeného k dalšímu zpracování hrají roli automatické kontroly popsané v kapitole 6.

Než přikročím k popisu zmíněného modulu, dovolím si ještě několik závěrečných obecných poznámek. V čem se vlastně liší oba popsané principy? Jejich pro i proti jsou svázaná s kvalitou manuální anotace. Udělá-li anotátor chybu, oba principy selžou. Rozdíl je v tom, že v prvním případě jedna chyba způsobí další, kdežto ve druhém anotátorova chyba zůstane chybou bez dalších následků. V podstatě jde o opožděnou anotaci. Tato zjištění oprávněně vedou k otázce, proč by anotátor nemohl provést posloupnost nutných změn (tj. spustit nějaké makro) stejně jednoduše, jako označuje místo určené k následné automatické anotaci? Proč neintegrovat automatickou anotaci prováděnou po odevzdání dat přímo do anotačního nástroje, když příslušné akce jsou ve skutečnosti pořád prováděny na popud anotátora? Na to odpovím, že je to bez výhrad možné a určitě se do toho stavu při anotaci PEDT dostaneme, ale zatím tomu tak není.

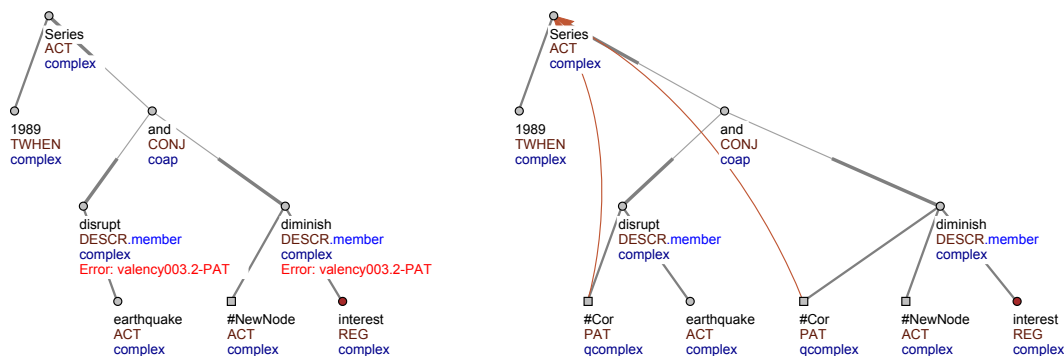
## 7.1 Doplnování koreference

Jedna z automatických kontrol prováděných během anotace ověřuje, jestli mají slovesa všechny potomky vyžadované přiděleným valenčním rámcem. U postponovaných přívlasků (viz část 5.13) se často stává, že chybí aktant ACT nebo PAT. A pokud nechybí, tak je přítomen pouze jako generovaný uzel s *t-lemmatem* #Gen, který vznikl právě při přiřazování rámece. Správně by měl být aktant zastupován koreferenčním uzlem, jak je uvedeno v části 5.13. Při přípravě dat se ale nepodaří odhalit všechny případy, při kterých jde o koreferenci, proto jich část zbývá na anotátory. Vytváření správného uzlu a zaznamenávání koreference je ovšem docela náročné, proto existuje následující dohoda: Anotátoři koreferenční uzel nevytvářejí a zanechávají data ve stavu, ve kterém automatická kontrola vyžaduje doplnění aktantu.

Každá odevzdaná data jsou ošetřena modulem, jež vynechané koreference doplňuje. Modul nejprve spustí automatické kontroly a potom hledá chyby upozorňující na chybějící aktant. Samotný výskyt chyby nestačí, přece jen je lepší ověřit nějaké základní údaje. Chyba se musí vyskytnout u uzlu, který má na p-rovině (ve složkovém stromě) *tag* VBG nebo VBN a jeho efektivním rodičem musí být substantivum nebo zájmeno. Pokud jsou požadavky splněny, můžeme vytvořit nový uzel, který bude generovaný (nastavený příznak *is\_generated*) a jeho *t-lemma* bude #Cor. U chyby je v tomto případě uvedeno i to, který aktant chybí – takto zjistíme *funktor* pro nový uzel. Zbývá vytvořit koreferenční odkaz (atribut *coref\_gram.rf*). Jeho hodnotou bude odkaz na rodiče slovesa, kterému původně chyběl aktant. Při hledání rodiče je nutné počítat s koordinací, aby se nestalo, že by odkaz vedl ke kořeni souřadné struktury (viz obrázek 7.1).

Na závěr ještě ukážu, že tento postup nemůže selhat.<sup>1</sup> Předpokládejme, že automatická kontrola ohlásí patřičnou chybu, ale koreferenční uzel nemá být vytvořen. V tuto chvíli existují následující možnosti:

<sup>1</sup>Samozřejmě za předpokladu, že anotátor neudělá chybu.



Obrázek 7.1: Doplnění koreference po manuální anotaci: stav stromu před použitím modulu s označenými chybami a strom po opravě s doplněnými uzly. Situace je navíc zkomplikována koordinací sloves. Podstrom pochází z věty „*The 1989 Series, disrupted by a devastating earthquake and diminished in national interest because both teams came from the San Francisco Bay area, is likely to ...*“

- Sloveso příslušný aktant vůbec nevyžaduje. V tom případě je špatně přiřazen valenční rámec a anotátor by ho měl opravit.
- Sloveso aktant vyžaduje, ale nejde o koreferenci. Anotátor aktant doplní – buď vytvoří generovaný uzel *#Gen* nebo opraví *funktor* některého ze synů.
- Sloveso aktant vyžaduje a jde o koreferenci, která ale vypadá jinak než ta, kterou by vyrobil modul. V takovém případě anotátor vše udělá ručně.

Žádná z těchto možností nevede k automatickému vytváření umělého koreferenčního uzlu, takže zvolený postup je korektní.



# Kapitola 8

## Souhrnné vyhodnocení

Tato kapitola má jediný a jednoduchý cíl. Chci v ní přinést kompletní vyhodnocení celé mé práce jako celku. Použitý postup se téměř neliší od toho, který je popsán v kapitole 4. První odlišností je to, že jsem na data postupně aplikoval všechny moduly, které data nějakým způsobem mění, a vyhodnotil jsem je najednou, nikoli po jednom. Prostý součet předchozích výsledků by byl velkou chybou.

Také jsem se musel vyrovnat s komplikací, která se neprojeví, když jsou jednotlivé moduly vyhodnocovány zvlášť.<sup>1</sup> V případě tak velkého množství modulů, s jakým pracuji, se může velmi snadno stát, že jeden atribut je editován vícekrát. Kdybych všechny akce započítával zvlášť, tak bych dostal velmi zavádějící výsledky. Mým cílem je ale na celou proceduru, které se účastní desítky modulů, pohlížet jako na celek – na černou skříňku, do které na jedné straně vložím data, na druhé je změněná vytáhnu a zajímá mě pouze celkový rozdíl, protože součet drobných rozdílů je v tomto případě něco docela jiného.

Uvedu příklad. Mějme uzel, jehož *funktor* je `RSTR`. Jeden modul změní *funktor* na `APP`, druhý na `DESCR` a nakonec dojde ještě ke změně na `LOC`. Pozorovatel vně černé skříňky ale vidí pouze změnu z `RSTR` na `LOC`. Podobných situací je více, například po smazání uzlu je třeba zajistit, že všechny operace na něm dosud provedené budou vyloučeny z následující analýzy. Největší komplikace nastávají u atributů typu seznam.

Pro tyto účely jsem implementoval filtr, který je zařazen do procesu evaluace těsně před fází popsanou v části 4.4 (Výpočet výsledků). Vstupem je záznam všech akcí provedených za sebou jdoucími moduly, výstupem je seznam akcí, které vnímá vnější pozorovatel.

S touto výbavou se již mohu pustit do závěrečné analýzy. Součástí rozsáhlého experimentu byly všechny moduly popsané v kapitole 5, ovšem akce provedené moduly z částí 5.2 a 5.26 nebyly použity pro výpočet z důvodů popsaných v části 5.26.

Číselné výsledky jsem rozdělil do tabulek 8.1 a 8.2. Některé atributy, nejvýrazněji pak *aux.rf*, dávají špatné výsledky, zejména u nových dat. Je to způsobeno především důvody popsanými v části 5.6 (chybná anotace referenčních dat zjištěná až po provedení vyhodnocení; tento problém bude třeba řešit zdokonalením anotačních pravidel, nikoli automatických skriptů). Osobně se domnívám, že moduly s atributem *aux.rf* zacházejí poměrně dobře, mnohem lépe, že se zdá při pohledu na čísla uváděná v tabulkách. Svůj názor však nemohu podpořit žádným hodnověrným měřením a nepůjde to do doby, dokud nebude k dispozici kvalitní anotace.

Potvrdil se předpoklad, který jsem vyslovil v části 4.1. Výsledky na *nových datech* jsou až na výjimky lepší než u *starých dat*. Místy i o desítky procent. Druhý největší rozdíl je vidět u vytváření nových uzlů, což je pochopitelné, jelikož jde o pracnou věc,

---

<sup>1</sup>Až na výjimky, např. část 5.28

Výskyty	14 519						
Spárováno	12 995 z 14 216 (91,41 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
aux.rf (D)	856		829	8			
			99,04 %	0,96 %			
Funktor	5 212	2 381	4 682	373	66	2 641	2 480
		45,68 %	91,43 %	7,28 %	1,29 %	51,57 %	48,43 %
Uzel (D)	745		672	73			
			90,20 %	9,80 %			
Nodetype	782	292	460	284	0	203	541
		37,34 %	61,83 %	38,17 %	0,00 %	27,28 %	72,72 %
T-lemma	1 688	941	1 546	41	21	765	843
		55,75 %	96,14 %	2,55 %	1,31 %	47,57 %	52,43 %
is_member	690	586	591	8	81	172	508
		84,93 %	86,91 %	1,18 %	11,91 %	25,29 %	74,71 %
Uzel (N)	726		654	72			
			90,08 %	9,92 %			
coref_gram (A)	325	325	275	23		23	275
		100,00 %	92,28 %	7,72 %		7,72 %	92,28 %
is_generated	6 634	447	5 128	342	2	4 728	744
		6,74 %	93,71 %	6,25 %	0,04 %	86,40 %	13,60 %
Rodič	3 213	1 272	2 988	135	49	1 952	1 220
		39,59 %	94,20 %	4,26 %	1,54 %	61,54 %	38,46 %
is_parenthesis	227	205	186	0	26	48	164
		90,31 %	87,74 %	0,00 %	12,26 %	22,64 %	77,36 %
aux.rf (A)	3 474	559	1 788	1 652		1 764	1 676
		16,09 %	51,98 %	48,02 %		51,28 %	48,72 %

Tabulka 8.1: Souhrnné vyhodnocení všech modulů – *nová data* (vysvětlivky viz část 4.5)

jejíž potřeba navíc není vždy úplně zřejmá (např. v případě zlomků – část 5.28). Úplně největší rozpor mezi výsledky je u atributu *coref\_gram.rf*. To je také pochopitelné, neboť jeho úprava vždy souvisí s vytvořením nového uzlu.

Podstatné však je, že u dvou nejdůležitějších atributů – *rodič* (tj. struktura stromu) a *funktor* – analýza prokázala zlepšení pohybující se mezi 25 až 40 procenty. Navíc jde o první dva nejčastěji měněné atributy, což dále zvyšuje celkový přínos. Kompletní sada modulů má v rámci celého PEDT potenciál určit správně asi 90 000 *funktorů* (přibližně 10 % tektogramatických uzlů),<sup>2</sup> z toho necelou polovinu v případech, kdy *funktor* před automatickým zpracováním správně nebyl.

<sup>2</sup>Čísla jsou nutně pouze orientační, jelikož prozatím není možné určit, kolik uzlů (a tím pádem i funkciorů) bude v hotovém korpusu.

Výskyty	64 146						
Spárováno	54 431 z 61 506 (88,50 %)						
Atribut	Akce	Změny	OK	Err 1	Err 2	OK (B)	Err (B)
aux.rf (D)	3 841		2 952	802			
			78,64 %	21,36 %			
Funktor	23 139	10 652	18 841	1 870	677	11 606	9 782
		46,03 %	88,09 %	8,74 %	3,17 %	54,26 %	45,74 %
Uzel (D)	3 040		2 748	292			
			90,39 %	9,61 %			
Nodetype	3 525	1 258	1 082	1 457	4	444	2 099
		35,69 %	42,55 %	57,29 %	0,16 %	17,46 %	82,54 %
T-lemma	7 766	4 068	5 053	557	572	4 122	2 060
		52,38 %	81,74 %	9,01 %	9,25 %	66,68 %	33,32 %
is_member	3 284	2 820	1 728	144	736	999	1 609
		85,87 %	66,26 %	5,52 %	28,22 %	38,31 %	61,69 %
Uzel (N)	3 135		1 812	1 323			
			57,80 %	42,20 %			
coref_gram (A)	1 282	1 282	448	436		436	448
		100,00 %	50,68 %	49,32 %		49,32 %	50,68 %
is_generated	28 593	1 863	20 720	1 275	23	19 516	2 502
		6,52 %	94,10 %	5,79 %	0,10 %	88,64 %	11,36 %
Rodič	14 751	5 917	12 263	1 050	400	8 882	4 831
		40,11 %	89,43 %	7,66 %	2,92 %	64,77 %	35,23 %
is_parenthesis	745	631	469	12	187	286	382
		84,70 %	70,21 %	1,80 %	27,99 %	42,81 %	57,19 %
aux.rf (A)	14 277	2 451	9 735	4 203		10 115	3 823
		17,17 %	69,85 %	30,15 %		72,57 %	27,43 %

Tabulka 8.2: Souhrnné vyhodnocení všech modulů – *stará data* (vysvětlivky viz část 4.5)



# Kapitola 9

## Závěr

Vše důležité bylo řečeno, touto kapitolou má práce končí. Pokusím se stručně shrnout její obsah a připomenout ty nejdůležitější body. Zhodnotím dosažené výsledky a nastíním možnosti dalšího vývoje.

Analyzoval jsem a korigoval návrhy pravidel automatické anotace připravených Silvií Cinkovou a jejich výslednou podobu jsem implementoval ve formě desítek skriptů. Připravil jsem jednoduché nástroje, pomocí kterých je možné celou sadu modulů nebo její části pouštět na vybraná data, a vše integroval do již existujícího zautomatizovaného procesu přípravy dat. Navíc zodpovídám za kompletní správu dat a udržování a vývoj programů určených speciálně pro projekt PEDT<sup>1</sup>, takže všechny postupy popsané v této práci v kapitolách 3, 5, 6 a 7 a i některé další udržuji aktuální a funkční.

Druhým zásadním bodem je vyhodnocení dosažených výsledků, které je od začátku až do konce výhradně mojí vlastní prací. Výsledkem je modulární systém, který bude bezpochyby používán i v budoucnosti pro potřeby PEDT. Po provedení úprav, které je třeba udělat ve vyhodnocovaném modulu, aby vypisoval záznam svých akcí, stačí k provedení kompletní evaluace zadat jediný příkaz v příkazové řádce. Použité postupy a metody jsou popsány v kapitole 4.

Souhrnné výsledky uvedené v kapitole 8 prokazují, že vykonaná práce je přínosná. Absolutní čísla vypadají „hezky“, i když objem provedených změn ztrácí své kouzlo ve srovnání s velikostí celého korpusu. V tomto případě však kvantita není jediným cílem. Výhoda automatické anotace založené na pravidlech místo na statistických metodách spočívá v tom, že korpus je nyní v případě několika desítek lingvistických jevů velmi konzistentní. Výsledky také ukazují, že má smysl podobným způsobem pokračovat. Nepochybně budou přibývat další pravidla a další moduly, při jejichž vývoji bude možné těžit z dosud vykonané práce.

---

<sup>1</sup>To například není program *TrEd*, ale anotační makra specifická pro PEDT ano



# Literatura

- [1] Ann Bies, Mark Ferguson, Karen Katz a Robert Mac-Intyre. Bracketing guidelines for Treebank II style Penn Treebank project. Technická zpráva, University of Pennsylvania, 1995.
- [2] Silvie Cinková. From PropBank to EngValLex: Adapting the PropBank-Lexicon to the Valency Theory of the Functional Generative Description. V *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006)*, strany 2170–2175, Genoa, Itálie, 2006.
- [3] Silvie Cinková, Jan Hajič, Marie Mikulová, Lucie Mladová, Anja Nedolužko, Petr Pajas, Jarmila Panevová, Jiří Semecký, Jana Šindlerová, Josef Toman, Zdeňka Urešová a Zdeněk Žabokrtský. Annotation of English on the tectogrammatical level: Reference book. Technická zpráva, ÚFAL MFF UK, Praha, 2007.
- [4] Silvie Cinková, Josef Toman, Jan Hajič, Kristýna Čermáková, Václav Klimeš, Lucie Mladová, Jana Šindlerová, Kristýna Tomšů a Zdeněk Žabokrtský. Tectogrammatical Annotation of the Wall Street Journal. Vyjde v *Prague Bulletin of Mathematical Linguistics*, číslo 92, 2009.
- [5] Ralph Weischedel a Ada Brunstein. BBN Pronoun Coreference and Entity Type Corpus. Linguistic Data Consortium, 2005. CAT: LDC2005T33.
- [6] Jan Cuřín, Martin Čmejrek, Jiří Havelka, Jan Hajič, Vladislav Kuboň a Zdeněk Žabokrtský. Prague Czech-English Dependency Treebank 1.0. Linguistic Data Consortium, 2004. CAT: LDC2004T25.
- [7] Jan Hajič, Eva Hajičová, Petr Pajas, Jarmila Panevová, Petr Sgall a Barbora Vidová Hladká. Prague Dependency Treebank 1.0. Linguistic Data Consortium, 2001. CAT: LDC2001T10.
- [8] Jan Hajič, Eva Hajičová, Jarmila Panevová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka a Marie Mikulová. Prague Dependency Treebank 2.0. Linguistic Data Consortium, 2006. CAT: LDC2006T01.
- [9] Jan Hajič, Jarmila Panevová, Zdeňka Urešová, Alevtina Bémová, Veronika Kolářová-Řezníčková a Petr Pajas. PDT-VALLEX: Creating a Large-coverage Valency Lexicon for Treebank Annotation. V *Proceedings of The Second Workshop on Treebanks and Linguistic Theories*, strany 57–68, Vaxjo, Sweden, 2003.
- [10] Václav Klimeš. *Analytical and Tectogrammatical Analysis of a Natural Language*. Dizertace, Univerzita Karlova v Praze, 2006.
- [11] Lucie Kučová, Veronika Kolářová-Řezníčková, Zdeněk Žabokrtský, Petr Pajas a Oliver Čulo. Anotování koreference v Pražském závislostním korpusu. Technická zpráva 19, ÚFAL MFF UK, Praha, 2003.

- [12] Mitchell P. Marcus, Beatrice Santorini a Mary Ann Marcinkiewicz. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1994.
- [13] Mitchell P. Marcus, Beatrice Santorini, Mary Ann Marcinkiewicz a Ann Taylor. Treebank-3. Linguistic Data Consortium, 1999. CAT: LDC99T42.
- [14] Adam Meyers, Ruth Reeves a Catherine Macleod. Nombank v 1.0. Linguistic Data Consortium, 2008. CAT: LDC2008T23.
- [15] Marie Mikulová, Alevtina Bémová, Jan Hajič, Eva Hajičová, Jiří Havelka, Veronika Kolářová-Řezníčková, Lucie Kučová, Markéta Lopatková, Petr Pajas, Jarmila Panevová, Magda Razímová, Petr Sgall, Jan Štěpánek, Zdeňka Urešová, Kateřina Veselá a Zdeněk Žabokrtský. Anotace Pražského závislostního korpusu na tektogramatické rovině: pokyny pro anotátory. Technická zpráva, ÚFAL MFF UK, Praha, 2005.
- [16] Petr Pajas a Jan Štěpánek. A Generic XML-Based Format for Structured Linguistic Annotation and Its Application to Prague Dependency Treebank 2.0. Technická zpráva 29, ÚFAL MFF UK, Praha, 2005.
- [17] Martha Palmer, Paul Kingsbury, Olga Babko-Malaya, Scott Cotton a Benjamin Snyder. Proposition Bank I. Linguistic Data Consortium, 2004. CAT: LDC2004T14.
- [18] Jiří Semecký a Silvie Cinková. Constructing an English Valency Lexicon. V *Proceedings of the Workshop on Frontiers in Linguistically Annotated Corpora 2006*, strany 94–97, Sydney, Austrálie, 2006.
- [19] Petr Sgall. *Generativní popis jazyka a česká deklinace*. Academia, Praha, 1967.
- [20] Petr Sgall, Eva Hajičová a Jarmila Panevová. *The Meaning of the Sentence in Its Semantic and Pragmatic Aspects*. Academia, Praha, 1986.
- [21] David Vadas. Noun Phrase Bracketing Guidelines. Technická zpráva, School of Information Technologies, University of Sydney, 2007.
- [22] David Vadas a James R. Curran. Adding Noun Phrase Structure to the Penn Treebank. V *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL)*, strany 240–247, Praha, 2007.
- [23] Kateřina Veselá a Jiří Havelka. Anotování aktuálního členění, věty v Pražském závislostním korpusu. Technická zpráva 20, ÚFAL MFF UK, Praha, 2003.
- [24] Jan Štěpánek. *Závislostní zachycení větné struktury v anotovaném syntaktickém korpusu (nástroje pro zajištění konzistence dat)*. Dizertace, Univerzita Karlova v Praze, 2006.
- [25] Zdeněk Žabokrtský, Jan Ptáček a Petr Pajas. TectoMT: Highly Modular MT System with Tectogramatics Used as Transfer Layer. V *ACL 2008 WMT: Proceedings of the Third Workshop on Statistical Machine Translation*, strany 167–170, Columbus, Ohio, 2008.