Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra kybernetiky

BAKALÁŘSKÁ PRÁCE

Detekce sémantických entit

Plzeň, 2014 Martin Bulín

Prohlášení

Předkládám tímto k posouzení a obhajobě bakalářskou práci zpracovanou na závěr studia na Fakultě aplikovaných věd Západočeské univerzity v Plzni.

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím odborné literatury a pramenů, jejichž úplný seznam je její součástí.

V Plzni dne Martin Bulín

Poděkování

Děkuji vedoucímu bakalářské práce Ing. Luboši Šmídlovi Ph.D. za poskytnutí zajímavého tématu, cenné rady a metodické vedení práce. Dále bych rád poděkoval Ing. Janu Švecovi za věcné diskuze a zpřístupnění vlastních skriptů pro zpracování a vyhodnocení dat.

Abstrakt

Předkládaná bakalářská práce je věnována automatickému porozumění mluvené řeči na lokální úrovni. Cílem práce je navrhnout bezkontextové gramatiky, které umožňují detekci vybraných sémantických entit z mřížky nejlepších (N-best) hypotéz automatického rozpoznávače řeči.

Na vstupu detekce byly postupně použity kromě ASR mřížek také nejlepší (1-best) hypotézy rozpoznávače a ruční přepisy promluv, což umožnilo paralelně testovat metodu detekce založenou na regulárních výrazech. Testovací data poskytla komunikace mezi operátory řízení letového provozu a piloty. Použité skripty byly vypracovány v programovacím jazyku Python a linuxovém prostředí Bash Shell.

Výstup detekce sémantických entit nabízí základ pro globální strojové porozumění vstupní informaci. Výsledky této práce potvrzují možnost použití metody založené na gramatikách i v dalších úlohách.

Klíčová slova: automatické porozumění mluvené řeči, sémantické entity, bezkontextové gramatiky, konečné automaty

Abstract

The presented thesis is devoted to the local-level spoken language understanding task. The thesis aims to design context-free grammars enabling a detection of chosen semantic entity from the N-best hypotheses lattice of an automatic speech recognizer.

Besides the ASR lattices also the ASR 1-best hypotheses and hand-made speeches transcriptions have been used gradually as a detection input, which allowed a parallel testing of the detection method based on regular expressions. The testing data has been provided by communications between flight traffic operators and pilots. Used scripts have been developed in programming language Python and linux Bash Shell developing environment.

The outcome of the semantic entity detection provides a basis for an automatic global-level understanding task. The results gained by this thesis confirm a possibility of using the method based on grammars in other various tasks.

Keywords: automatic spoken language understanding, semantic entity, context-free grammars, finite state automata

Obsah

Úvod

Základním předpokladem úspěšné komunikace, tedy sdělování informací, myšlenek či pocitů, mezi lidmi je, že komunikující předávané informaci rozumějí. V současné době můžeme za nejdůležitější nástroj dorozumívání označit řeč, ať už v mluvené či psané podobě. Postupný vývoj informační společnosti přináší nové příležitosti, kde by lidé mohli ocenit hlasovou, tedy bezkontaktní, komunikaci také s počítačem, a nezastavitelný pokrok v oblasti kybernetiky postupně umožňuje takové nápady uskutečňovat.

Tato práce čtenáři představuje detekci sémantických entit, jednu z metod strojového porozumění informace. Nutno dodat, že zmíněné porozumění (velice pravděpodobně) nepředstavuje práci lidského mozku při snaze pochopit význam daného sdělení.

If the human brain were so simple that we could understand it, we would be so simple that we couldn't.

Kdyby byl lidský mozek tak prostý, že bychom ho byli schopni pochopit, byli bychom tak prostoduší, že bychom to nedokázali.

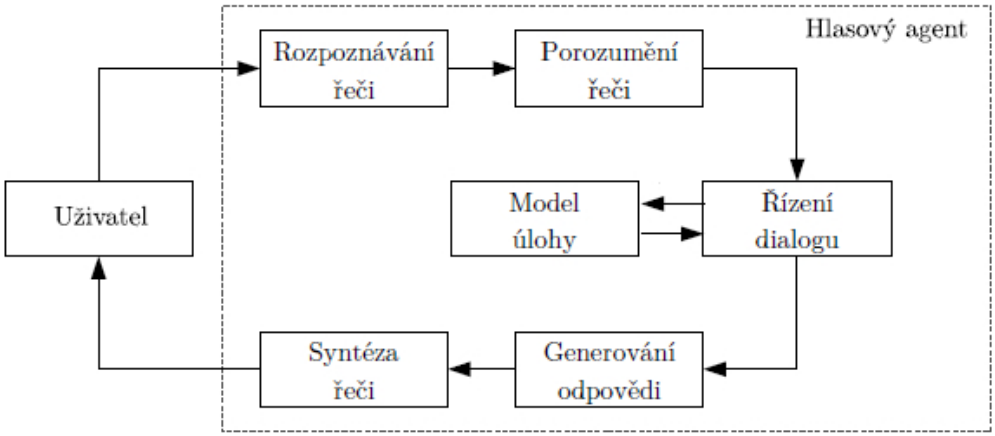
Protože umělé napodobení všech funkcí mozku je výrazně nad rámec této práce, Dr Emerson Pugh a jeho tvrzení by nemuselo být překážkou při pokusech programově aproximovat proces lidského pochopení informace alespoň natolik, aby byl výstup tohoto systému takový, který by při stejném zadání vyprodukoval člověk. Výsledek detekce sémantických entit se tedy porovnává s referenčními daty, která byla ručně připravena - jedině tak je možné ověřit funkčnost algoritmů.

Program je testován na datech obsahujících anglickou komunikaci mezi operátory řízení letového provozu a piloty. Celá práce je součástí projektu „Inteligentní technologie pro zvýšení bezpečnosti letového provozu“ řešeném na katedře kybernetiky Západočeské univerzity v Plzni.

Hlavním úkolem této práce je na základě studie letecké frazeologie navrhnout gramatiky, které v promluvách dokáží detekovat vybraných 17 sémantických entit. Dále na stejných datech vyhodnotíme detekci založenou na regulárních výrazech a výsledky obou metod porovnáme. Významnou výhodou gramatik oproti regulárním výrazům bude jejich snadná modifikace a především možnost detekce z ASR mřížek.

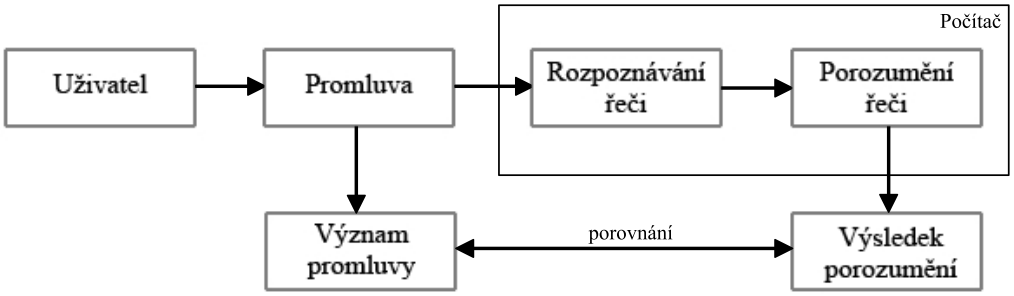
1. Strojové zpracování mluvené řeči

Problematika řešená v této práci je součástí velkého konceptu zvaného **Hlasový dialogový systém**. Jednou z komunikujících stran takového systému je tzv. hlasový agent (můžeme si pod ním představit počítač), který dokáže nejen vnímat mluvenou řeč, ale také na ni patřičně reagovat. Obsahuje tedy rozhodovací algoritmus, který se dokáže "zamyslet"nad vstupem od uživatele, vygenerovat rozumnou odpověď a předat ji ve formě syntetizované řeči.



Obrázek 1.1: Model hlasového dialogového systému [1]

V dalším textu nejprve naznačíme princip metod rozpoznávání řeči a poté se zaměříme především na zmíněnou metodu automatického porozumění, jejíž výsledky porovnáme s lidským porozuměním stejného vstupu. Budeme tedy zkoumat část hlasového dialogového systému znázorněného na obrázku 1.2.

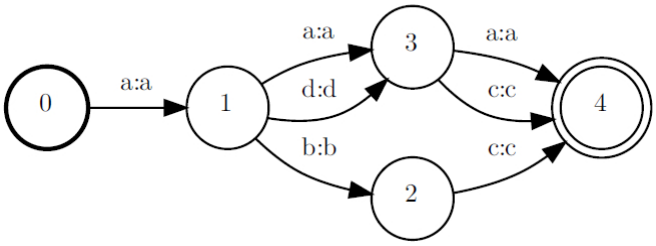


Obrázek 1.2: Část hlasového dialogového systému vyvíjena v této práci

1.1 Konečné automaty a jejich reprezentace

Jedním ze základních stavebních kamenů této práce je teorie konečných automatů (angl. Finite-State Machine - FSM). Využíváme jich při modelování stavového prostoru v úloze rozpoznávání řeči a také pro reprezentaci navržených gramatik při vyhodnocení porozumění řeči.

Konečný automat je model systému přecházející mezi jednotlivými svými stavy na základě vstupu. V každém kroku svého algoritmu přečte jeden symbol ze vstupu a na základě hodnoty tohoto symbolu vybere svůj příští stav. Ohodnotíme-li tyto přechody mezi stavy váhou, vytvoříme tzv. ohodnocený konečný automat.



Obrázek 1.3: Příklad grafického znázornění konečného automatu (počáteční stav: 0, koncový stav: 4, přijímá řetězce: 'aaa', 'ada', 'aac', 'adc', 'abc') [1]

Polookruh

Pro způsob ohodnocení přechodů mezi stavy konečného automatu se využívá tzv. polookruh (angl. semiring), okruh neobsahující negaci, definovaný jako množina K vybavená asociativními operacemi $\oplus$, popř. $\otimes$ s neutrálními prvky $\bar{0}$, popř. $\bar{1}$, přičemž operace $\otimes$ je dvojně distributivní vzhledem k $\oplus$ a $\bar{0}$ je nulový prvek. Polokruh značíme jako uspořádanou pětici $(K, \oplus, \otimes, \bar{0}, \bar{1})$ [2].

V praxi se nejčastěji používá ohodnocení podle pravděpodobností přechodů, resp. podle záporných logaritmů těchto pravděpodobností. Pracuje se tedy s polookruhem $(R \cup \{\inf\}, \oplus, +, \inf, 0)$, kde pro operátor $\oplus$ platí $a \oplus b = -log(e^{-a} + e^{-b})$ pro logaritmický polookruh nebo $a \oplus b = min\{a, b\}$ pro tzv. tropický polookruh.

Logaritmus pravděpodobnosti se využívá z důvodů vyplývajících z reprezentace čísel v počítači. Použití jeho záporné hodnoty umožňuje získat ohodnocení v kladných číslech a pro průchod automatem tak dále využít standardních metod prohledávání v grafu [3].

Konečné transducery

Automaty, které mají pouze binární výstup, se nazývají **akceptory**. Podávají informaci pouze o tom, zda daný vstup přijmou či nepřijmou. Vstupní řetězec je přijat, pokud akceptor obsahuje takovou posloupnost přechodů, která ho dovede do jednoho z definovaných koncových stavů.

Konečné akceptory jsou obecně speciálním případem tzv. **konečných transducerů (angl. Finite-State Transducer - FST)**, konečných automatů, které kromě informace o přijetí vstupního řetězce generují výstup závisející na průchodu mezi stavy.

Transducery lze zapsat jako uspořádanou sedmici $(S, s, F, \Sigma, \delta, \Gamma, \omega)$, kde

- S ... množina stavů automatu
- s ... počáteční stav $s \in S$
- F ... množina koncových bodů $F \subset S$
- Σ ... množina vstupních symbolů
- δ ... přechodová funkce $\delta : S \times \Sigma \rightarrow S$
- Γ ... množina vstupních symbolů
- ω ... výstupní funkce

Pro ohodnocený transducer už poté stačí jen dodefinovat počáteční váhu $\lambda \in R$ a váhovou funkci ρ zobrazující $F \rightarrow R$. Podle způsobu generování výstupu (volby výstupní funkce) rozdělujeme transducery na dva typy. Výstupní funkce **Mooreova automatu** je definována jako $\omega : S \rightarrow \Gamma$, závisí tedy na stavu, do kterého automat přijetím vstupu přejde. Výstup **Mealyho automatu** je k dispozici v okamžiku přijetí vstupu, neboť jeho výstupní funkce je $\omega : S \times \Omega \rightarrow \Gamma$. Oba dva typy automatů jsou vzájemně převoditelné [2].

Optimalizační operace

Konečné automaty zpravidla představují výpočetní modely enormních rozsahů. Uvedeme několik postupů, které se snaží změnou vnitřní struktury modelu dosáhnout zvýšení operační rychlosti a snížení paměťových nároků při práci s automatem. Poznamenejme, že žádný z uvedených algoritmů nemění relaci mezi vstupními a výstupními řetězci automatu [1].

- **Odstranění ϵ-přechodů**

  Znak ϵ u konečných automatů představuje přechod, který se realizuje bez ohledu na vstupní symbol. Tuto neurčitost redukujeme odstraněním všech přechodů e, pro které platí $i[e] = o[e] = \epsilon$.

- **Determinizace**

  Konečné automaty označíme jako nedeterministické, pokud mohou mít pro jeden vstup z jednoho stavu více přechodů (může existovat několik různých cest k přijetí vstupního řetězce). Takové automaty lze procesem **determinizace** převést na deterministické, které mají pro daný vstup v daném stavu vždy jen jediný přechod.

- **Minimalizace**

  Tento algoritmus můžeme použít na deterministických automatech. Po jeho provedení dosáhneme minimálního počtu stavů pro daný automat.

Konkrétní algoritmy ve formě pseudokódů těchto operací najdeme například v [2].

Reprezentace konečných automatů

Jednoduché automaty s malým počtem stavů a hran často vídáme v podobě orientovaných grafů, mluvíme tedy o **grafické reprezentaci** (např. obrázek 1.3). Obdobně se využívá zápis automatu ve formě **tabulky**, ovšem opět pouze pro automaty s malou abecedou.

Obecně existují dva základní formáty reprezentace konečných automatů v počítači. První z nich, udržovaný společností *AT&T*, se zapisuje do textového souboru a je ho možné použít v rámci knihovny **openfst** [4]. Tento formát s příponou **.fst** je používán také v této práci, přičemž soubory vstupující do vyhodnocovacích skriptů jsou ještě optimalizovány odstraněním ϵ-přechodů. Výsledný používaný formát je tedy označován **.rmeps.fst**.

Druhou používanou reprezentací jsou automaty ve formátu **ELF**, které reprezentují Mooreův typ ohodnoceného konečného transduceru. Konkrétní specifikace podoby tohoto formátu v [2].

1.2 Automatické rozpoznávání řeči

Rozpoznáváním řeči (v angličtině Automatic Speech Recognition - dále jen ASR) rozumíme proces převedení mluveného slova na text. Myšlenkou ASR je napodobit skutečné lidské rozpoznávání zvukového signálu, který je ve své fyzikální podstatě spojitá funkce změny akustického tlaku. Ve vnitřním uchu je tato časová funkce převedena do frekvenční oblasti, čímž vzniká frekvenční spektrum neboli tón, který je dále přijímán a vyhodnocován mozkem. [5]

První pokusy o realizaci se objevily již s příchodem prvních počítačů před více než 50. lety. Jednalo se o značně primitivní systémy, které většinou fungovaly na principu porovnávání nahrávek slov s nahrávkami v referenční paměti.

Z tohoto nápadu se postupně vyvinul klasifikátor řeči pracující na principu **porovnávání se vzory** (template matching). Pracuje se zde s celými slovy a s metodou dynamického programování[1], která pomocí nelineární transformace časové osy hledá ve slovníku vzorový obraz takového slova, k němuž má obraz rozpoznávaného slova nejblíže. [3] Tento klasifikátor se dá použít pro rozpoznávání izolovaných slov, naopak výrazné problémy má se spontánní řečí, čímž do jisté míry ztrácí svou aktuálnost.

Protože složitost rozpoznávání jednotlivých diktovaných slov a každodenní souvislé řeči plné změn intonace, přeřeků či neočekávaných váhání se výrazně liší, mohli bychom kvalitu rozpoznávání spontánní řeči označit za jedno z kritérií metod ASR. Mezi další požadavky na klasifikátor zařadíme schopnost vypořádat se s šumy na pozadí, tedy nezávislost na prostředí, a velmi žádaná je také nezávislost na řečníkovi. V průběhu let se systémy ASR vyvinuly natolik, že dokáží rozpoznat milióny slov, jsou nezávislé na mluvčím a dokonce se na něj umí během rozpoznávání adaptovat.

Za tento pokrok vděčíme především **statistickému přístupu k modelování** promluvy založenému na tzv. skrytých Markovových modelech (Hidden Markov Models - dále jen HMM). Na rozdíl od první metody se zde používají modely tzv. subslovních (jazykových) jednotek (např. slabik, fonémů, trifonů) a o klasifikaci rozhoduje největší aposteriorní pravděpodobnost. Pojďme si tento přístup stručně přiblížit.

[1] Dynamické programování je matematický pojem užívaný pro analýzu sekvenčních rozhodovacích procesů. Užívá se zde časově nelineární "bortivé"funkce s přesně specifikovanými vlastnostmi (dynamic time warping - DTW) - více v [3]

1.2.1 Statistický přístup k modelování řeči

Pod pojmem akustický procesor si představme zařízení, které transformuje řečové kmity na posloupnosti vektorů příznaků. Použitím metody extrakce příznaků, která obsahuje Fourierovu transformaci (převedení signálu z časové do frekvenční oblasti), nahradíme výše zmíněnou činnost vnitřního ucha u přírodní paralely.

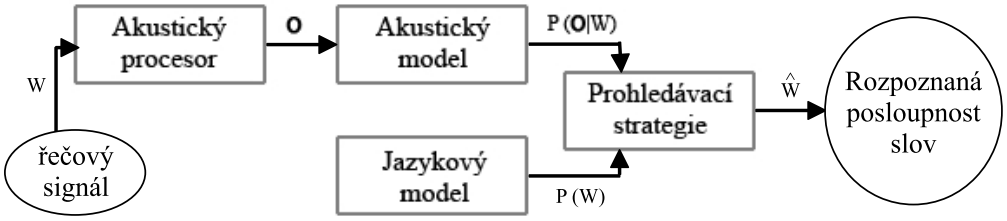
[3] Předpokládejme nyní, že máme řečový signál ve formě posloupnosti jazykových jednotek[2] $W = \{w_1 w_2 ... w_N\}$ a posloupnost vektorů příznaků $\mathbf{O} = \{\mathbf{o}_1, \mathbf{o}_2 ... \mathbf{o}_T\}$ získanou z akustického procesoru. Hledáme nejpravděpodobnější posloupnost jazykových jednotek $\hat{W}$ pro danou akustickou informaci $\mathbf{O}$, tedy takovou, aby podmíněná pravděpodobnost $P(W|\mathbf{O})$ byla maximální. Užitím Bayesova vztahu dostaneme:

$$\hat{W} = \arg\max_W P(W|\mathbf{O}) = \arg\max_W \frac{P(W)P(\mathbf{O}|W)}{P(\mathbf{O})}$$

Protože apriorní pravděpodobnost posloupnosti výstupních vektorů $P(\mathbf{O})$ není funkcí W, lze ji při hledání maxima ignorovat a dostáváme tedy:

$$\hat{W} = \arg\max_W P(W|\mathbf{O}) = \arg\max_W P(W)P(\mathbf{O}|W)$$

Z této rovnice vyplývá, že problém přechází na nalezení maximálního součinu dvou nezávislých pravděpodobností. Podmíněné rozdělení pravděpodobnosti $P(\mathbf{O}|W)$ zastupuje model řečníka neboli tzv. **akustický model** a apriorní pravděpodobnost $P(W)$ nese informaci o tzv. **jazykovém modelu**.



Obrázek 1.4: ASR systém - statistický přístup [3]

[2] V [1] se proměnná W používá pro slova, zde je použita obecně pro jazykovou jednotku.

Akustický model a HMM

Flexibilita, přesnost a účinnost jsou vlastnosti, kterými by měly akustické modely disponovat. Jejich úkolem je poskytnout co nejpřesnější a nejrychlejší odhad podmíněné pravděpodobnosti $P(\mathbf{O}|W)$ pro libovolnou pozorovanou posloupnost vektorů příznaků $\mathbf{O}$ a každou uvažovanou posloupnost jazykových jednotek.

Efektivním způsobem řešení se ukázaly být výše zmíněné skryté Markovovy modely (HMM). Při jejich používání se opět vychází z přírodní představy o generování řeči a z úvahy o konečném počtu stavů artikulačních konfigurací (např. vyslovení určité jazykové jednotky). Posloupnosti konečného počtu stavů dosáhneme vygenerováním tzv. podpůrného Markovova řetězce. Spektrální charakter krátkých úseků řečového signálu pak reprezentuje řetězec vektorů příznaků. V diskrétních časových okamžicích je proces vždy v jediném stavu a lze jej tedy pozorovat prostřednictvím náhodné funkce.

V úlohách ASR se většinou užívá třístavového HMM, kde si každý stav můžeme představit jako směs gaussových křivek a výsledný model poté jako posloupnost těchto stavů.

Původní úloha akustického modelu nalezení nejlepšího odhadu rozdělení pravděpodobnosti $P(\mathbf{O}|W)$ přechází na úlohu určení struktury HMM a nastavení jeho parametrů - tvaru gaussových křivek. Toho dosáhneme využitím expertní znalosti (pomůže zvolit strukturu HMM - zmíněné tři stavy) a trénováním na anotovaných datech (nastavení parametrů). [3]

Jazykový model

Jak již bylo naznačeno, záměrem jazykového modelu je určit, pokud možno v reálném čase, apriorní pravděpodobnost $P(W)$ pro libovolnou posloupnost jazykových jednotek W. Každý jazyk má svůj slovník a pravidla, která určují možné posloupnosti jednotlivých slov, popřípadě jiných jazykových jednotek. Je prakticky nemožné, zvláště ve spontánní řeči, abychom mohli předem s jistotou vyloučit výskyt některé z kombinací. Proto nám nebude stačit deterministické omezení vypovídající pouze o výskytu či nevýskytu (0 nebo 1) dané posloupnosti, ale budeme potřebovat tzv. **stochastický jazykový model** generující skutečnou pravděpodobnost výskytu.

S výhodou pro další úlohu dekódování určujeme pravděpodobnost posloupnosti W obsahující K slov jako:

$$P(W) = P(w_1^K) = P(w_1 w_2 ... w_k) = \prod_{i=1}^{k} P\left(w_i | w_1^{i-1}\right)$$

Protože získání všech takovýchto posloupností jazykových jednotek libovolné délky je v současné době nad možnosti výpočetní techniky, využívají se tzv. n-gramové modely. Takto dosáhneme toho, že všechny posloupnosti, které se shodují v posledních $n-1$ jazykových jednotkách, zařazujeme do jedné skupiny. Jejich historie v průběhu určování pravděpodobnosti je tedy pouze $n-1$ posledních jazykových jednotek. Nejčastěji se používají bigramy (n=2). [2] Podrobnosti viz [3].

Shrnutí úlohy ASR

Pro získání názorného přehledu celé úlohy ASR využijeme její dekompozici na následující subúlohy [3]:

1. Volba **základní jazykové jednotky** rozpoznávání a vytvoření slovníku. Pro představu, anglické rozpoznávače obsahují kolem padesáti tisíc slov, český slovník se kvůli morfologii dostane snadno přes jeden milion slov.

2. Provedení **akustické analýzy** řečového signálu s cílem určit posloupnost vektorů příznaků **O**.

3. Vytvoření **akustického modelu** pro ocenění podmíněné pravděpodobnosti $P(\mathbf{O}|W)$.

4. Vytvoření **jazykového modelu** pro získání pravděpodobnosti $P(W)$.

5. Nalezení nejpravděpodobnější posloupnosti slov aplikací **účinných prohledávacích strategií**.

Úloha se dá převést na problém prohledávání stavového prostoru, který ovšem bývá enormně veliký. Cílem je tedy nalézt přirozeně takovou metodu prohledávání, která úspěšně redukuje výpočty a přitom neztrácí na obecnosti.

ASR mřížky

Tzv. slovní mřížka (angl. lattice) představuje jeden ze způsobu vyjádření **N nejlepších hypotéz**, resp. N posloupností slov, jejichž modely nejpravděpodobněji generují posloupnost pozorování **O**. Tento způsob získání výsledku oceníme při víceprůchodovém prohledávání stavového prostoru, neboť každý nový průchod prohledává detailněji, a tím i přesněji, menší oblast stavového prostoru.

Výsledek obdržíme ve formě hranově ohodnoceného orientovaného grafu, který bude časově uspořádán. Seznam N nejpravděpodobnějších posloupností slov je určen množinou všech cest vedoucích z počátečního uzlu a končících v koncovém uzlu grafu. [3]

Můžeme říci, že z ASR mřížky dostaneme za větší úsilí lepší výsledek než při použití jediné nejlepší hypotézy. Mřížku ukládáme ve formě acyklického ohodnoceného konečného automatu a jedná se tedy o formát výstupu automatického rozpoznávače řeči. Následná detekce sémantických entit přímo z ASR mřížek je výhodou použití metody bezkontextových gramatik oproti regulárním výrazům (bude blíže popsáno níže).

1.3 Automatické porozumění řeči

Tato podkapitola se zabývá dalším krokem strojového zpracování vstupní informace. Zatímco předchozí tématika automatického rozpoznávání řeči byla jasně definována (vstupem byla hlasová promluva, výstupem sdělení ve formě textu), úloha automatického porozumění nabývá výrazně vyšší neurčitosti.

Co ve skutečnosti znamená ono porozumění? Po hlubokém zamyšlení bychom pravděpodobně podlehli nejrůznějším filosofickým úvahám, to ovšem není předmětem této práce. Pokusíme se tedy využít volného přepisu webové definice porozumění: *„Porozumění znamená konstruování významu sdělení včetně orálních, psaných i grafických komunikací“* .

Právě o přiřazení globálního významu vstupní promluvě se úloha automatického porozumění řeči snaží. V případě úspěchu dosáhneme tzv. **vyššího** porozumění, které samozřejmě čerpá z porozumění na **nižší** úrovni (označováno také jako lokální). Jednou z metod dílčího (lokálního) porozumění je **detekce sémantických entit**.

1.3.1 Sémantické entity

Sémantickou entitou myslíme konkrétní objekt zmíněný v dané promluvě a významný z pohledu sémantické analýzy. Může se jednat o slovo, spojení více slov, ale také třeba jen písmeno, číslo či jakýkoliv jiný znak. Sémantické entity mohou být různých typů, například časové údaje, datum, položky rozsáhlých databází (poznávací značky, jména osob).

Úkolem detekce sémantických entit je označit určitou část vstupní informace jako příslušející sémantické entitě daného typu. Pozorného čtenáře později napadne podobnost sémantických entit s tzv. pojmenovanými entitami zmíněnými ve druhé kapitole. Rozdíl mezi těmito metodami by měl být v reprezentaci vstupní informace. Pojmenované entity jsou běžně detekované z textu, sémantické entity ze slovní mřížky. V této práci používáme pojem sémantická entita pro detekci z obou zmíněných vstupů.

K modelování sémantických entit se využívá dvou základních přístupů. **Statistický přístup** používá pravidla závislá na trénovacích datech, od jejichž výběru se později odvíjí i kvalita detekce.

V této práci je použit **znalostní přístup** k modelování entit. U této varianty se používají znalosti získané nastudováním dané problematiky a konzultacemi s oborovým expertem. Danou entitu zpravidla představuje výčet možných hodnot nebo jejich popis vhodnou gramatikou. Znalostní přístup může často vhodně doplňovat statistický přístup a výrazně tak zvýšit robust-

nost výsledného systému.

Dané expertní znalosti mohou být generovány také automaticky z vhodné databáze a pro některé entity mohou být dokonce přenositelné mezi jednotlivými doménami (např. gramatiku entity typu čas lze bez úprav použít v různých úlohách) [1].

V dalším textu se pokusíme popsat způsoby popisu jednotlivých sémantických entit, upozornit na výhody a nevýhody jednotlivých metod a nakonec vyhodnotit detekci těmito způsoby na testovacích datech. Obecně je třeba vytvořit pro každou z entit nějaký regulární jazyk, který vymezuje všechny možné zápisy dané entity.

1.3.2 Regulární výrazy

V polovině dvacátého století přišel americký matematik Stephen Cole Kleene s prvním vzorcem regulárního výrazu (angl. regular expression - **RE**), někdy označovaného též jako **regexp** či **regex**. Jedná se o speciální řetězec znaků, který představuje určitý vzor (masku) pro textové řetězce. Popisuje tedy celou množinu řetězců, která se často označuje jako regulární jazyk [6].

K popisu se využívá tzv. **metaznaků**, mezi které patří například tečka ('.') reprezentující právě jeden libovolný znak, dále celá skupina kvantifikátorů udávajících, kolikrát se smí opakovat předchozí znak, a další. Přehledný popis regulárních výrazů najdeme v [6], na tomto místě bude stačit pro představu příklad regulárního výrazu popisujícího množinu všech možných zápisů e-mailové adresy.

```
[a-zA-Z0-9._-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,4}
```

Regulární výrazy v současné době podporuje většina skriptovacích jazyků a dokonce i některé propracovanější textové editory či souborové manažery. Nejčastěji jsou využívány při vyhledávání dané posloupnosti znaků v textu a při manipulaci s textem. Při detekci sémantických entit jsou využity pro tvorbu regulárního jazyka pro každou z entit. Při jejich aplikování na testovací data ve formě textu poté získáme detekované entity na odpovídajících místech.

Regulární výrazy jsou obecně velice silným nástrojem při práci s textovými řetězci, čemuž odpovídá i kvalita získaných výsledků. Na druhou stranu, právě jejich omezení pouze na vstup ve formě textu je jejich částečnou nevýhodou. Druhý nedostatek čtenář pravděpodobně odhalil již v průběhu představování regulárních výrazů sám - totiž jejich nečitelnost a náročnou modifikovatelnost.

1.3.3 Gramatiky

Další možností popisu sémantických entit je využití gramatik, které slouží k formálnímu popisu jazyka. Zapisují se jako uspořádaná čtveřice $G = (U, V, S, R)$.

- U ... množina neterminálních symbolů

  Symboly, které jsou dále rozšiřovány pomocí pravidel, běžně značené velkými písmeny.

- V ... množina terminálních symbolů, kde $U \cap V = \emptyset$

  Dále nerozšiřitelné symboly značené malými písmeny.

- $S \in U$ ... startovací symbol

- R ... množina pravidel

Pokud máme definovanou gramatiku G, můžeme z ní užitím pravidel z R generovat jazyk $L(G)$. Následně můžeme např. pomocí Turingova stroje [7] či jednodušeji výše zmíněnými konečnými automaty testovat, zda vstupní řetězec w patří do jazyka $L(G)$. Pokud si nyní místo jazyka představíme naše sémantické entity, dostaneme přesně takový nástroj, který pro detekci potřebujeme.

K získání řetězce, který daná gramatika přijme, se využívá tzv. derivačních stromů. Ty dostaneme postupným aplikováním pravidel gramatiky na získané symboly, přičemž začínáme u startovacího symbolu. Výsledný strom má v uzlech neterminály a v listech terminály [2].

V roce 1956 vytvořil americký lingvista Noam Chomsky hierarchii, která gramatiky rozděluje do čtyř tříd - frázové, kontextové, bezkontextové a regulární, přičemž jsou seřazeny od té nejobecnější.

Bezkontextové gramatiky

V této práci se zaměříme na třídu bezkontextových gramatik, které získáme omezením odvozovacích pravidel na tvar $A \rightarrow \gamma$, kde A je neterminál a γ řetězec terminálů i neterminálů.

Jednou z bezkontextových gramatik je gramatika podle specifikace **SRGS** (Speech Recognition Grammar Specification) vytvořená konsorciem W3C. Tato specifikace umožňuje vytváření gramatik ve formě ABNF nebo XML a stanovuje způsob zpracování těchto forem [2]. V této práci využijeme formát **.abnf**, který je uživatelsky přístupný i pro rozsáhlé gramatiky. Příklad bude uveden níže v textu.

2. Současný stav problematiky

Metody strojového porozumění významu informace, kam můžeme zařadit také detekci sémantických entit, jsou cesty, jakými se člověk snaží o vytvoření umělé inteligence.

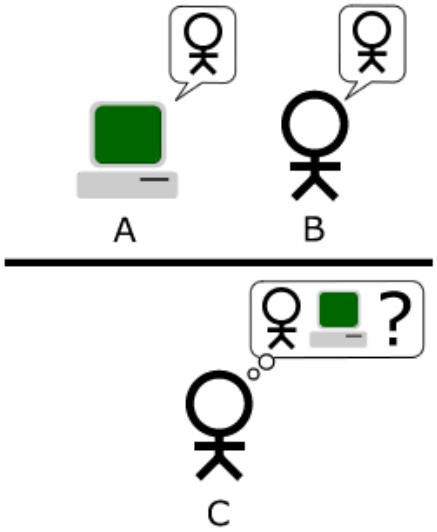
2.1 Související výzkum

Inteligence je pojem, který lze jen těžko definovat a tím hůře testovat. Jednou z možností je test založený na srovnání s člověkem, který prezentoval již v roce 1950 Alan Turing.

Turingův test

Test je ilustrován na obrázku 2.1. Do jedné místnosti umístíme testujícího, do druhé testované zařízení (počítač s umělou inteligencí) a člověka. Testující položí otázku v přirozené řeči a předává ji do druhé místnosti. Náhodně se zvolí, zda mu na ni odpoví člověk nebo stroj a tato odpověď je v neutrální podobě (např. ve formě textu) poslána zpět do první místnosti. Pokud testující po určitém počtu otázek nedokáže rozpoznat, s kým komunikuje, testovaná umělá inteligence splňuje Turingův test [7].

Loebnerova cena (100 000 dolarů) pro počítač, jehož odpovědi budou od člověka zcela nerozlišitelné, dosud nebyla udělena. Mírný pokrok přišel s vývojem tzv. chatterbotů - počítačových programů, kteří simulují inteligentní komunikaci s uživatelem.



Obrázek 2.1: Turingův test [7]

Chatterboti

Snahou autorů chatterbotů je vyvinout program, který dokáže generovat plynulé a smysluplné reakce na repliky člověka a vytvoří tak dojem, že tématu konverzace skutečně rozumí. Prvním průlomem se stal program ELIZA (Joseph Weizenbaum, 1966) chovající se jako psychoterapeut. Mezi další úspěšné pokusy můžeme zařadit například programy A.L.I.C.E., Jabberwacky či Cleverbot. Převážně fungují na principu vyhledávání podle klíčových slov, popřípadě hledají v rozsáhlé databázi takovou odpověď, kterou dříve použil u stejné otázky nějaký uživatel [8]. Všechny zmíněné i další může čtenář zdarma otestovat na internetu.

Argument čínského pokoje

Zamyslíme-li se nad dosavadním obsahem této kapitoly, napadne nás, že tito chatterboti jsou navrženi právě tak, aby splňovali Turingův test. Otázkou je, zda Turingův test skutečně představuje vhodné kritérium inteligence. V roce 1980 ho zpochybnil filosof John Searle tzv. argumentem čínského pokoje, jehož podrobné vysvětlení viz [7]. V principu jde o to, že je možné vytvořit iluzi porozumění použitím deterministicky definovaných mechanických úkonů.

Named entity recognition

Metoda detekce sémantických entit není nová, nicméně její počátky by se daly dohledat v nedávné historii. Velmi podobná je metoda "Named entity recognition", jejíž implementace od různých autorů vyhodnocovala ke konci minulého století americká vládní agentura DARPA pořádáním konferencí známých jako MUC (Message Understanding Conference). Pro představu, nejlepší program z poslední konference v roce 1997 dokázal detekovat zhruba 93% požadovaných informací ze vstupního textu [7].

Než začneme toto vysoké procento úspěšnosti porovnávat s našimi výsledky, musíme mít na paměti, že kvalita výsledků se vždy odvíjí od zadání úlohy. Zatímco ve zmíněné "soutěži"se jednalo o detekci čtyř či pěti entit z náhodných textů (šlo o zjištění stěžejních informací z článků), úloha v této práci se zajímá o detekci hned sedmnácti navzájem velice podobně vypadajících entit.

Autor práce [1] se zabýval detekcí sémantických entit, kde na vstupu je slovní mřížka ze systému automatického rozpoznávání řeči reprezentována pomocí váženého konečného automatu. V této práci se pokusíme detekovat sémantické entity také ze vstupu ve formě textu.

2.2 Motivace pro práci na tématu

Z uvedených výpisek se může zdát, že vyvíjení systémů typu chatterbot či metod porozumění se line špatným směrem a nemůže tak přinést nic užitečného pro společnost, ale opak je pravdou. Zdaleka se ještě nejedná o tzv. silnou umělou inteligenci, ovšem výsledky se dají uplatnit v dílčích oblastech.

V případě chatterbotů můžeme mluvit o automatických "offline"poradnách v oblastech, kde se často opakují otázky zákazníků, dále potom například o systémech automatického získávání informací (většinou se jedná o obsáhlá data a hodiny lidské práce).

Co se týče detekce sémantických entit, také zatím neexistuje systém, který by dokázal pojmout celou množinu možných lidských promluv, neboť k tomu nejsou prostředky ani expertní znalosti. Ovšem pokud se jedná o jednu specifickou oblast, jako například v této práci o leteckou angličtinu, výsledky mohou být až překvapivě dobré. V tomto případě jde o vytvoření pseudopilota podílejícího se na tréninku operátora letového provozu.

Jako další oblast využití bychom mohli uvést situaci při řízení auta. Podle statistik z let 2001-2007 zemřelo jen v USA za volantem vinou řidiče, který za jízdy telefonoval, psal sms zprávu, ovládal navigaci či jen rádio, přes 16000 lidí. Jedním návrhem vládních pracovníků je prý začít do dopravních prostředků montovat rušičky mobilních telefonů, dalším návrhem by mohlo být bezdotykově komunikující auto. Pokud množina autem rozpoznávaných příkazů bude daná a tím i omezená, zdaleka se již nejedná o sci-fi, a výsledky mohou být skutečně prospěšné.

3. Cíle bakalářské práce

Cíle práce, částečně zmíněné již v úvodu, převážně vycházejí z potřeb řešeného projektu. Jejich výklad můžeme založit na teoretickém popisu použitých metod z první kapitoly.

Primární cíl práce

Na základě získaných expertních znalostí **navrhnout bezkontextové gramatiky** pro vybrané sémantické entity a vytvořit prostředí, které po jakékoliv úpravě těchto vstupních gramatik jednoduše vygeneruje nové výsledky detekce. Právě možnost snadné modifikace gramatik je vedle možnosti detekce z ASR mřížek hlavní motivací k jejich implementaci.

Pro vyhodnocení je dále třeba:

1. **Připravit testovací data** z poskytnutých komunikací mezi piloty a operátory řízení letového provozu. To znamená především nastudovat leteckou frazeologii a získané znalosti zkombinovat s poznatky získanými přímo od pracovníků řízení letového provozu a vytvořit tak expertní znalosti k přípravě referenčních dat. S těmito daty se později budou porovnávat výsledky jednotlivých metod detekce.

2. Otestovat na připravených referenčních datech již implementovanou metodu založenou na **regulárních výrazech**. Pro tuto metodu provedeme dva testy. Při prvním z nich bude na vstupu detekce ruční přepis promluv, u druhého testu použijeme nejlepší (1-best) hypotézu automatického rozpoznávače řeči.

3. Zkompilovat **navržené gramatiky** do formátu konečných automatů. Dále pomocí nich detekovat sémantické entity postupně pro:

    - vstup získaný přepisem promluvy
    - vstup získaný z automatického rozpoznávače řeči jako nejlepší (1-best) hypotéza
    - vstup ve formě ASR mřížky, která představuje N nejlepších hypotéz automatického rozpoznávače řeči

4. Detekce sémantických entit v letecké angličtině

Metoda detekce sémantických entit, jak již vyplývá z jejího teoretického popisu v první kapitole, se dá aplikovat téměř kdekoliv, pokud zde dochází k dorozumívání mezi dvěma stranami. Testovací data v této práci představuje komunikace, převážně v anglickém jazyce, mezi operátory řízení letového provozu a piloty letadel.

4.1 Seznámení s oborem

Řídící letového provozu, vybaven mapou, obvykle radarovým zobrazením letadel, radiovou stanicí pro komunikaci a potřebnými znalostmi má za úkol ze země zajistit, aby se letadlo s ničím nesrazilo, ať již v průběhu cesty či v oblasti letiště. Za let po naplánované trati je zodpovědný pilot letadla.

Letecké mapy a předávání řízení

V leteckých mapách můžeme mimo jiné najít tečkovaná kolečka či šestiúhelníky přestavující letecké majáky, zařízení, která ze země vysílají navigační radiový signál. Bývají označovaná třemi písmeny (např. OKG, RAK). Další značka v mapách, která nás může zajímat, je trojúhelníček představující tzv. FIX - myšlený bod na zemi definovaný svou polohou. Tyto FIXy jsou většinou pojmenované pěti písmeny (KOLAD, VARIK, ...). Mezi leteckými majáky a FIXy je síť vzdušných koridorů - leteckých cest určených především pro tzv. IFR lety (lety „podle přístrojů“ podávající letový plán). VFR lety, většinou sportovní a turistická letadla, využívají služeb řídících převážně v oblasti letišť.

Standardní chování IFR letadel po celou dobu jejich aktivity (nastartování motorů, vytlačení od gatu, přejezd k dráze, vzlet, stoupání, let, klesání, stočení se do osy dráhy, dosednutí, příjezd ke gatu, vypnutí motorů) umožňuje vytvoření podobné struktury řídících služeb letového provozu po celém světě. Tyto služby si letadlo předávají a sledují ho po celou dobu jeho činnosti.

Každá služba má svůj volací znak (např. služba GROUND starající se o pohyb letadla v oblasti letiště v Praze-Ruzyni má radiový volací znak „Ruzyně Ground“). Obecně lze služby rozdělit na řízení pohybu na zemi, řízení odletů a příletů na letiště a řízení letadla na trati. Pokud daná služba používá pro

řízení radarovou informaci, název služby se v jejím volacím znaku nahrazuje slovem "Radar". V uvedeném případě by to tedy bylo "Ruzyně Radar"[9].

Z tabulky 4.1 si můžeme udělat představu o struktuře jednotlivých řídících služeb.

Mezin. název	Zkratka	CZ název	Výška [ft ALT]	Prostor působnosti	Radar
DELIVERY	DEL	Delivery	0	jen dává povolení	NE
GROUND	GND	Ground	0	stojánky a TWY	NE
TOWER	TWR	Věž	0 - 5000	RWY a CTR	NE
APPROACH	APP	Přiblížení	3000 - FL125	TMA	ANO
DEPARTURE	DEP	Odlety	3000 - FL125	TMA	ANO
DIRECTOR	DIR	Direktor	FL70 - 3000	TMA	ANO
CONTROL	ACC	Oblast	FL95 - FL660	mimo CTR a TMA	ANO
INFO	INFO	Info	1000 - FL95	vně CTR a TMA	NE

Tabulka 4.1: Struktura služeb řídících letového provozu [9]

Informaci o typu řídící služby jsme v práci nevyužili a v komunikaci jsme označili jednotně všechny promluvy od řídících letového provozu jako „*_ground_*“ (země). Promluvy pilotů byly označeny jako „*_air_*“ (vzduch).

Ukázka komunikace

Přestože se většinou jedná o předávání informace od řídících k pilotům, komunikace není jednostranná. Úkolem pilotů je kromě okamžitého plnění příkazů také zopakování instrukce tak, jak jí porozuměli - tomu se říká „Readback“. K identifikaci letadla se užívá volacích znaků (anglicky call sign - bude zmíněno později), které jsou vždy první informací zprávy. Pilot svou volací značku zopakuje vždy na konci své promluvy. Častou součástí komunikace jsou také zdvořilostní fráze. Pro představu si uvedeme ukázky možných komunikací. Pro zkrácení nyní vynecháme zmíněné „readbacky“ pilotů letadel [8].

- Služba „Ruzyně Delivery“ vydává povolení k provedení letu:

```
_ground_ CSA020, cleared to BRNO, Vozice three mike
departure, squawk 1421.
```

- Následně dá služba „Ruzyně Ground“ povolení ke spuštění motorů a postupnému přesunutí na tzv. holding point - vyčkávací místo před dráhou:

```
_ground_ CSA020, start up and powerback approved.
_ground_ CSA020, taxi to holding position RWY24 via G, B.
```

- Nyní piloti dostanou instrukci přeladit frekvenci na službu „Ruzyně Věž“, a ta následně povolí vzlet:

```
_ground_ CSA020, contact Tower 118.10, naslyš.
_ground_ CSA020, line up RWY24, ... wind calm, RWY24,
cleared for take-off, naslyš.
```

Ještě ukážeme krátkou instrukci od řídícího směrem k pilotovi letadla s volacím znakem Easy 5498, aby kontaktoval řídícího z Prahy na frekvenci 120.275, tentokrát i s „readbackem“.

```
_ground_ Easy 5 4 9 8 contact Praha radar 1 2 0 . 2 7 5 bye bye
_air_ 1 2 0 2 7 5 Easy 5 4 9 8 good night
```

Protože součástí spousty volacích znaků či názvů leteckých společností jsou samostatná písmena, používá se pro snazší dorozumění hláskovací abeceda:

A :: Alpha	**H** :: Hotel	**O** :: Oscar	**V** :: Victor
B :: Bravo	**I** :: India	**P** :: Papa	**W** :: Whiskey
C :: Charlie	**J** :: Juliett	**Q** :: Quebec	**X** :: X-Ray
D :: Delta	**K** :: Kilo	**R** :: Romeo	**Y** :: Yankee
E :: Echo	**L** :: Lima	**S** :: Sierra	**Z** :: Zulu
F :: Foxtrot	**M** :: Mike	**T** :: Tango	
G :: Golf	**N** :: November	**U** :: Uniform	

Tabulka 4.2: Abeceda letecké frazeologie

4.2 Reálný přínos práce pro daný obor

Jak je zmíněno již v úvodu, práce je součástí projektu „Inteligentní technologie pro zvýšení bezpečnosti letového provozu“. Cílem je vyvinout a v experimentálním režimu otestovat komplexní systém inteligentní komunikace mezi operátory řízení letového provozu a automatickým „počítačovým“ pseudopilotem.

Detekce sémantických entit poskytne částečnou informaci o významu dané promluvy a přispěje tak k nalezení celkového („vyššího“) významu promluvy. Pokud uměle vytvořený automatický pseudopilot „porozumí“ situaci, bude schopen vygenerovat smysluplné odpovědi či otázky pro testovaného operátora a simulovat tak pro něj skutečnou leteckou komunikaci.

4.3 Postup při vypracování

V této části popíšeme experimentální část práce podloženou získanými výsledky jednotlivých metod. Vývoj je založen na programovacím jazyce Python a podpořen „shell“ skripty v linuxovém prostředí.

4.3.1 Příprava dat

Pro testování projektu byly získány nahrávky skutečné komunikace z leteckého provozu. Data byla rozdělena na trénovací sadu a dvě testovací sady, označené „test-e3“ (obsahuje celkem 1434 nahrávek) a „test-e7“ (1877 nahrávek). Obě testovací sady budeme dále vyhodnocovat nezávisle.

Rozpoznání promluvy

Nahrávky jednotlivých promluv byly získány v audio formátu **.wav**, tedy v podobě mluvené řeči. Prvním úkolem tedy bylo převést je na text. Využili jsme systému automatického rozpoznávání řeči (**ASR**) zmíněného v první kapitole. Tím byla získána pro každou z promluv ASR mřížka reprezentující N (zde N=60) nejlepších hypotéz a také nejlepší (1-best) hypotéza ASR. Kromě toho byla vytvořena ještě data získaná **ručním přepsáním** nahrávek do textové podoby.

Tyto „čisté“ rozpoznané promluvy jsme uložili do souborů s názvy *_v04_teste3_cleaned.txt* a *_v04_teste7_cleaned.txt* a můžeme je považovat za zdrojová data pro detekci sémantických entit. Dále byly vytvořeny soubory s názvy promluv (*_v04_teste3_scp.txt* a *_v04_teste7_scp.txt*), přičemž vše je synchronizováno po řádcích.

Automaticky rozpoznané promluvy ve formě ASR mřížek (formát .htk.slf) jsme připravili na pozdější zpracování metodami detekce sémantických entit. Zvolené parametry rozpoznávače a detailnější postup při rozpoznávání promluv najdeme níže.

Výběr sémantických entit k detekci

Rozhovory mezi operátory letového provozu a piloty bývají velmi jednotvárné a množina významů možných sdělovaných informací je tedy velmi omezená. Vhodným výběrem jednotlivých sémantických entit můžeme pokrýt celou tuto množinu a umožnit tak porozumění všech možných (v oboru standardních) promluv.

Po analýze poskytnutých dat byly pro detekci zvoleny tyto sémantické entity:

⟨**ALT/**⟩	Altitude	Výška nad zemí
⟨**ATIS/**⟩	ATIS	Automatická informační služba v koncové řízené oblasti
⟨**CO/**⟩	Courtesy	Zdvořilostní fráze
⟨**CS/**⟩	Call Sign	Volací označení (”volačka”) letadla
⟨**FL/**⟩	Flight Level	Letová hladina
⟨**FR/**⟩	Frequency	Radiová frekvence řídící služby
⟨**HE/**⟩	Heading	Směr letu letadla
⟨**PO/**⟩	Point	Myšlený bod („FIX“) nebo maják
⟨**QNH/**⟩	QNH	Q-kód pro tlak vzduchu přepočtený na hladinu moře[1]
⟨**RA/**⟩	Rate	Změna rychlosti
⟨**RC/**⟩	Radar Contact	Volání řídící služby
⟨**RWY/**⟩	Runway	Číslo ranveje či pojezdové dráhy
⟨**SP/**⟩	Speed	Rychlost letadla
⟨**SQ/**⟩	Squawk	Nastavení odpovídače letadla
⟨**TI/**⟩	Time	Čas
⟨**TU/**⟩	Turn	Relativní změna směru letadla
⟨**WI/**⟩	Wind	Rychlost a směr větru

Tabulka 4.3: Vybrané sémantické entity k detekci

Vytvoření referenčních dat

Cílem úlohy je automaticky poskytnout takové výsledky, které by vyprodukoval člověk. Pro porovnávání při vyhodnocení jednotlivých metod tedy potřebujeme referenční data, která můžeme označit za správné řešení dané úlohy. Soubory „cleaned“ byly tedy po řádcích manuálně převedeny na referenční soubory obsahující detekci - „*_v04_teste3_referenceSED.txt*“ a „*_v04_teste7_referenceSED.txt*“.

Příklad správné (ruční) detekce:

- Řádka v souboru „cleaned“ (ruční přepis promluvy):

```
_ground_ Skytravel 1054 Praha dobrý_den radar_contact
climb to FL 280
```

- Odpovídající řádka s detekovanými entitami v referenčním souboru:

```
_ground_ <CS/> <RC/> climb to <FL/>
```

[1] barometrický výškoměr ukazuje nadmořskou výšku letiště, když je letadlo na zemi

Všimněme si, že některá slova či slovní spojení (v tomto příkladu „climb to“) jen přepíšeme, neboť nejsou součástí žádné z entit. Využívají se později k „vyššímu“ porozumění promluvy, což je ovšem nad rámec této úlohy.

Vytvoření „slovníkového“ souboru

Abychom později mohli využít již vytvořené skripty [2], které dokáží vyhodnotit kvalitu detekce sémantických entit, potřebujeme mít porovnávaná data v určitém formátu. Soubory v tomto požadovaném formátu jsme označili jako **„slovníkové“**, neboť jejich struktura skutečně odpovídá zápisu kontejnerového datového typu slovník (dict) v programovacím jazyce Python. Jako příklad uvedeme obsah „slovníkového“ souboru o dvou nahrávkách, kde promluva ”e3_ACCL05O0BQ_00002”obsahuje entity ”⟨CS/⟩”a ”⟨RC/⟩”a promluva ”*e*3_*ACCL*05*O*0*BQ*_00003”entitu ”⟨CO/⟩”. Všechny tyto entity se v promluvách vyskytují s váhou 1, tzn. byly tam obsaženy jedenkrát.

```
{'e3_ACCL-0500BQ_00002': {'CS': 1.0, 'RC': 1.0},
  'e3_ACCL-0500BQ_00003': {'CO': 1.0}}
```

Vytvoření „slovníkových“ souborů z textových souborů s detekovanými entitami nám zajistí skript **detResult2dict.py**. Po této transformaci zavoláme na výsledné soubory ještě skript **entityFilter.py**, který vytvoří ”slovníkový”soubor také pro každou entitu zvlášť. Tímto by všechna referenční data měla být připravena.

4.3.2 Popis způsobu vyhodnocení

K vyhodnocení přesnosti detekce se používají hotové skripty vytvořené v rámci práce [1]. Konkrétně se o to stará program **rocspot.py**, na jehož vstupu je referenční soubor a porovnávaný soubor, kde oba mají „slovníkovou“ strukturu. Výstupem je graf ve formátu .pdf obsahující tzv. **ROC křivku** (Receiver Operating Characteristic).

ROC křivka

V použitém skriptu se jedná o modifikaci klasické ROC křivky, kterou využívá binární klasifikátor. Daná modifikace je nutná z důvodu klasifikace do více než dvou tříd - jedné vstupní promluvě je přiřazeno více různých hypotetických sémantických entit s odpovídajícími aposteriorními pravděpodobnostmi a rovněž referenční sémantické entity jsou tvořeny nikoli jedinou cílovou třídou, ale celou posloupností sémantických entit. [1]

		reference
predikce	1	0
1	TP	FP
0	FN	TN

- **TP** (True Positives) ... počet správně detekovaných entit
- **TN** (True Negatives) ... počet správně nedetekovaných entit
- **FP** (False Positives) ... počet tzv. "falešných poplachů"- entita byla detekována, ale neměla být
- **FN** (False Negatives) ... počet nezachycených entit - entita měla být detekována, ale nebyla

Dále můžeme definovat poměr správně detekovaných příkladů TPR (True Positives Rate) a relativní četnost falešných poplachů na jednu promluvu FPR_{norm} (False Positives Rate):

$$TPR = \frac{TP}{TP + FN}, FPR_{norm} = \frac{FP}{n}$$

kde n je v našem případě počet promluv v testovací množině.

ROC křivka vždy začíná v bodě [0;0], správně by měla končit v bodě [1;1] a je definována jako křivka vyjadřující závislost TPR na FPR_{norm} pro spojitě se měnící hodnotu prahu θ. Tento práh je využíván rozhodovacím pravidlem při přiřazování skóre detekovaným entitám. Abychom pro každou ROC křivku měli nějakou číselnou hodnotu a mohli tak jednotlivé výsledky navzájem porovnávat, je možné použít velikost plochy pod modifikovanou ROC křivkou - tzv. hodnotu AUC (Area Under the Curve). Toto číslo spočítáme z hodnot TPR pro FPR_{norm} z intervalu $\langle 0, 1 \rangle$:

$$AUC = \int_0^1 TPR(FPR_{norm}) dFPR_{norm}$$

ROC křivky najdeme níže u vyhodnocení všech použitých metod detekce sémantických entit.

Druhý skript (**rocpoint.py**) nám pomáhá odhalit chyby detekce ve specifikovaném bodu ROC křivky. Výstupem je „slovníkový" soubor, který pro každou promluvu označí falešné poplachy a entity, které algoritmus nedokázal detekovat.

4.3.3 Detekce regulárními výrazy

Na úvod je třeba říci, že navržení regulárních výrazů pro sémantické entity letecké angličtiny není předmětem této práce. Metoda byla implementována již dříve a k naší úloze byly poskytnuty její výsledky pro porovnání. Pro představu si ovšem můžeme uvést ukázku regulárního výrazu detekujícího jeden z možných zápisů frekvence:

```
(?P<#r__0-9.txt>) (?P<#r__0-9.txt>) (?P<#r__0-9.txt>) &.
(?P<#r__0-9.txt>) (?P<#r__0-9.txt>) (?P<#r__0-9.txt>)
```

Výraz $r\_\_0-9.txt$ představuje název souboru, ve kterém jsou uloženy všechny možnosti zápisu jakékoli číslice. Takový regulární výraz by zachytil například frekvenci ve tvaru: „1 1 2 . 1 8 0“.

Metodu detekce sémantických entit pomocí regulárních výrazů aplikujeme na dva různé typy vstupních dat.

1. vstupní data získaná **ručním přepisem** promluv (později označováno RE-text)

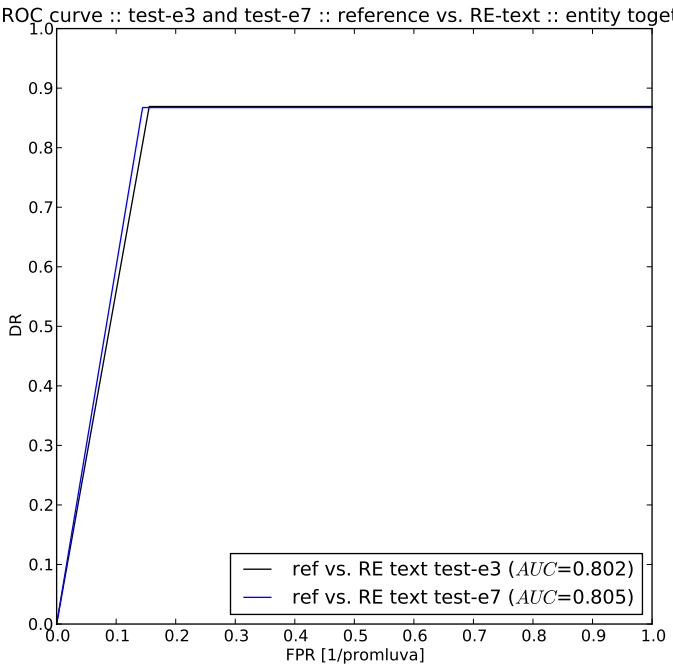
2. vstupní data získaná z automatického rozpoznávače řeči jako nejlepší (**1-best**) hypotéza (RE-Hyp1best)

Výsledky detekce pomocí regulárních výrazů byly pro oba dva typy vstupních dat této práci poskytnuty ve formátu souborů s detekovanými entitami (stejný princip jako při přípravě referenčních dat). K získání „slovníkových“ souborů s výsledky detekce pomocí regulárních výrazů jsme tedy opět použili skript **detResult2dict.py** a posléze jsme pomocí programu **entityFilter.py** vytvořili individuální „slovníkový“ soubor pro každou z entit.

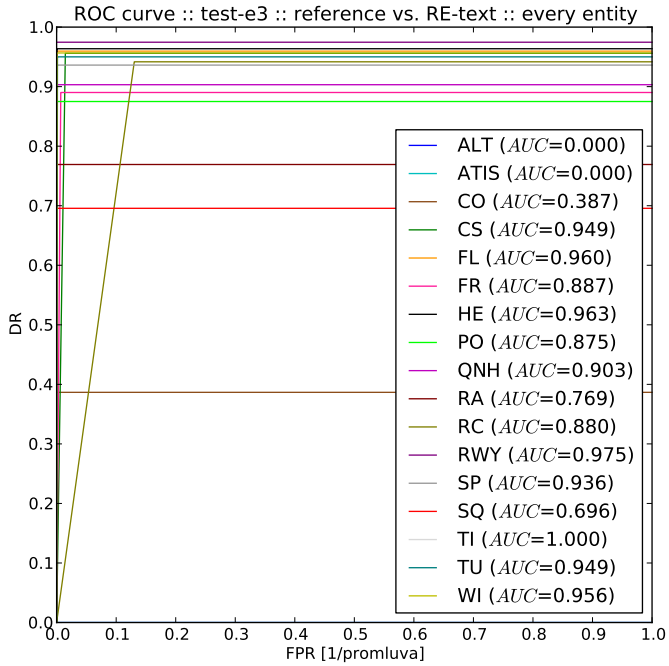
Po získání „slovníkových“ souborů máme již vše potřebné k vyhodnocení metody. Zavoláme tedy skript **rocspot.py**, kde použijeme připravená referenční data a jako hypotetická data pošleme do vyhodnocení výsledky detekce pomocí regulárních výrazů.

Pro oba typy vstupních dat vždy zobrazíme nejprve celkové výsledky, kdy se při vyhodnocení používají slovníky obsahující detekci všech entit. Poté zobrazíme graf znázorňující kvalitu jednotlivých entit samostatně (zde se používají slovníky získané skriptem entityFilter.py).

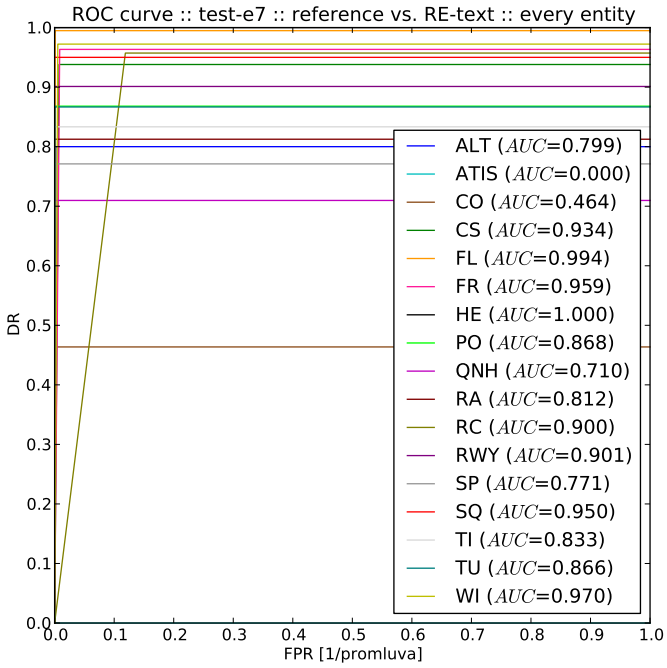
1. metoda RE aplikovaná na vstupní data získaná ručním přepisem



Obrázek 4.1: Regulární výrazy aplikované na ruční přepis promluv pro všechny entity dohromady - sada e3 a sada e7

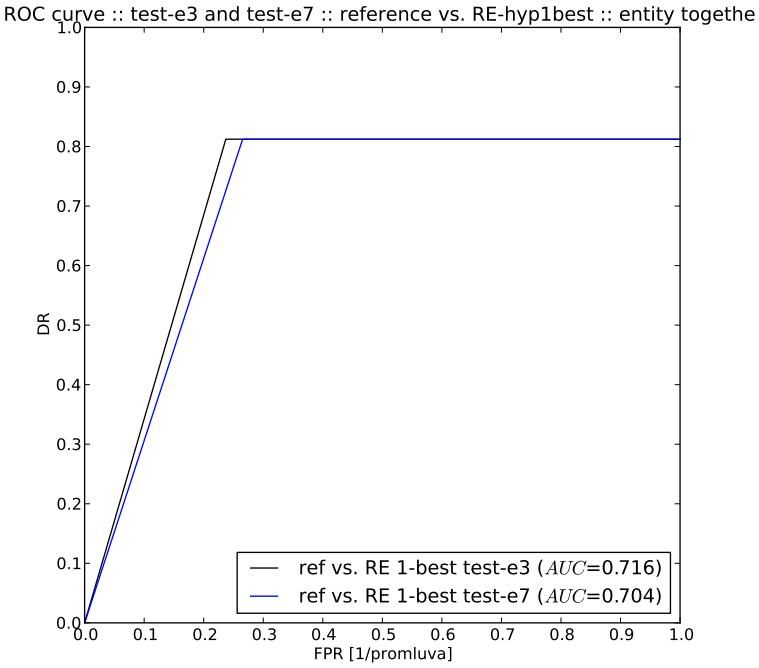


Obrázek 4.2: Regulární výrazy aplikované na ruční přepis promluv pro každou z entit zvlášť - sada e3

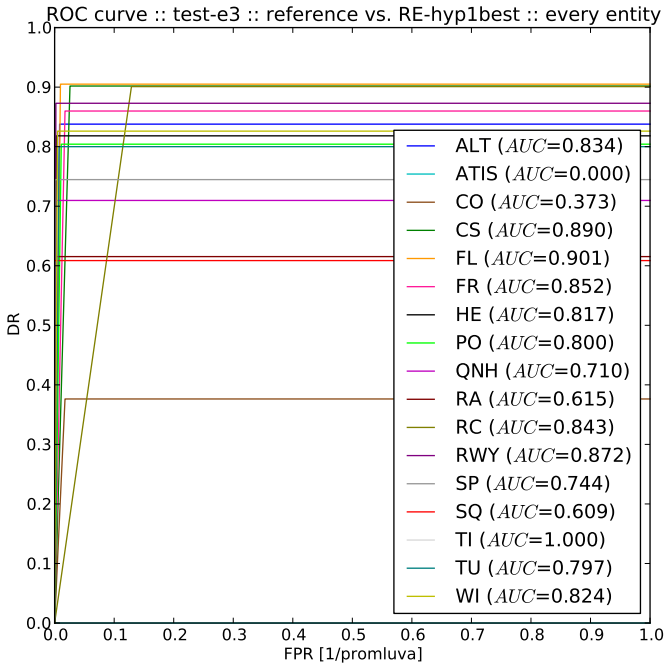


Obrázek 4.3: Regulární výrazy aplikované na ruční přepis promluv pro každou z entit zvlášť - sada e7

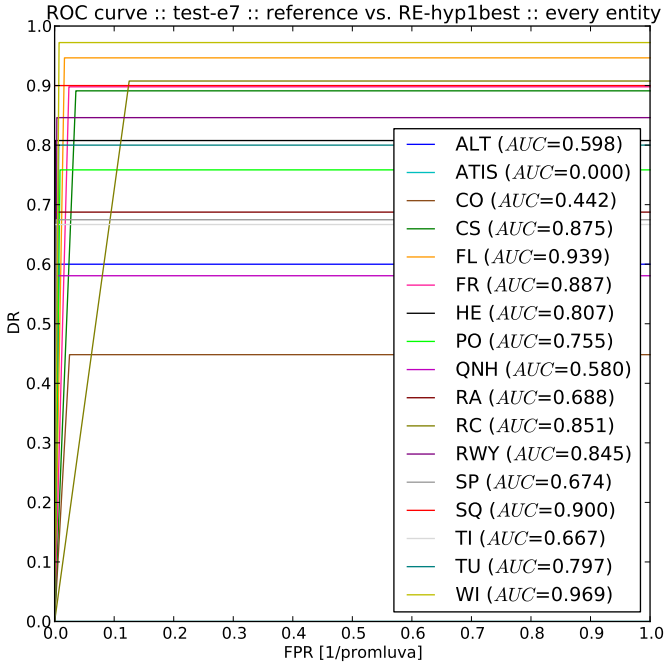
1. metoda RE aplikovaná na ASR 1-best hypotézu



Obrázek 4.4: Regulární výrazy aplikované na nejlepší hypotézu ASR promluv pro všechny entity dohromady - sada e3 a sada e7



Obrázek 4.5: Regulární výrazy aplikované na nejlepší hypotézu ASR promluv pro každou z entit zvlášť - sada e3



Obrázek 4.6: Regulární výrazy aplikované na nejlepší hypotézu ASR promluv pro každou z entit zvlášť - sada e7

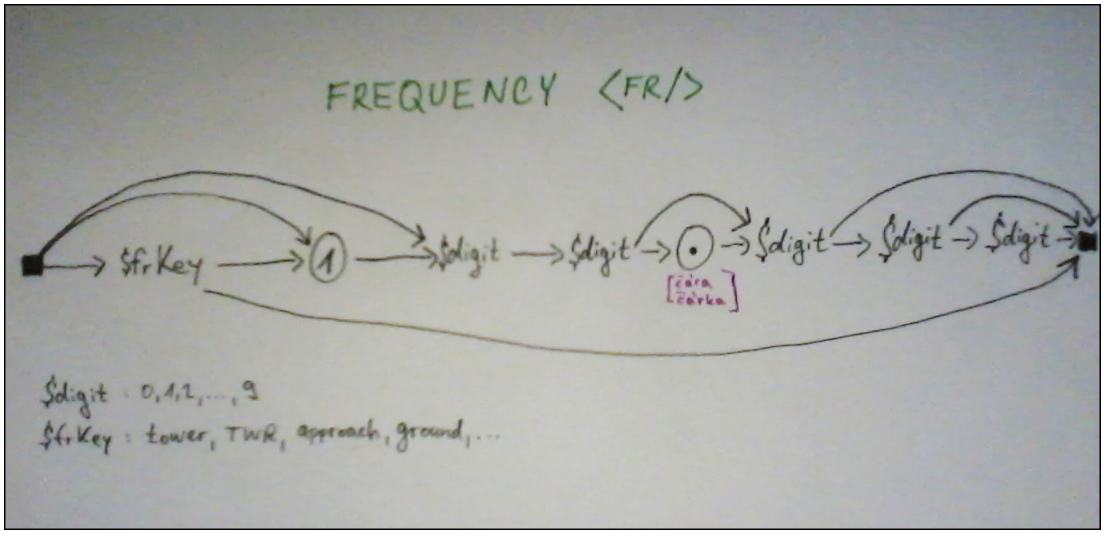
K uvedeným grafům bychom měli doplnit, že metoda regulárních výrazů byla implementována dříve, než byl zkompletován seznam detekovaných entit. Například o detekci entity **ATIS** se RE vůbec nesnaží, a proto je u ní hodnota AUC = 0.0. Návrh popisu entity **CO** by byl adeptem na další úpravy. Naopak si můžeme všimnout, že pro sadu e3 se RE daří dokonale detekovat entitu **TI**. Výsledky detekce ostatních entit jsou vyrovnané a můžeme je označit za kvalitní.

4.3.4 Detekce bezkontextovými gramatikami

Navržení bezkontextových gramatik, které budou s určitou přesností detekovat sémantické entity, je hlavním cílem této práce. Základním předpokladem pro jejich realizaci je spolupráce s oborovým expertem, od kterého získáme informace o všech možných podobách zvolených sémantických entit. Za tímto účelem proběhlo několik konzultací se spoluřešiteli projektu.

Návrh gramatik „na papíře“

Po spojení informací získaných od oborového experta, zkušeností získaných procházením testovaných dat a lidské intuice vznikly představy o možných podobách pro jednotlivé entity. Prvním krokem bylo přenést tyto představy na papír a nějak je graficky znázornit. Pro každou ze 17 vybraných entit byl vytvořen orientovaný graf, kde každá cesta od počátečního bodu ke koncovému značí jednu možnost pronesení dané entity. Pro ukázku přiložíme graf pro entitu frekvence (obr. 4.7). Všechny navržené gramatiky jsou takto zobrazeny v příloze A.



Obrázek 4.7: Návrh gramatiky pro frekvenci - ručně na papír

Gramatiky ve formátu .abnf

Dalším krokem je převedení těchto návrhů gramatik do počítače. K implementaci s výhodou použijeme formát **.abnf**. Ten je totiž uzpůsoben tak, aby si dokázal poradit s vyjádřením dané gramatiky, které je prakticky stejné, jako vidíme na obrázku 4.7. Zápis gramatiky pro entitu frekvence ve formátu .abnf bude vypadat takto:

```
#ABNF 1.0 UTF-8 cs;
grammar FR;

root $FR;
public $FR  = ([$frKey]([&1] {1} $digit09 $digit09
                [(&.|čára|čárka)] {.}
                $digit09 [$digit09 [$digit09]]) ) |
                $frKey;

$digit09 = ( &0 {0} | &1 {1} | &2 {2} | &3 {3} | &4 {4} |
                &5 {5} | &6 {6} | &7 {7} | &8 {8} | &9 {9} );
$frKey = (tower {tower} | TWR {tower} | approach {approach} |
                ground {ground} | information {info} | ...)
$
```

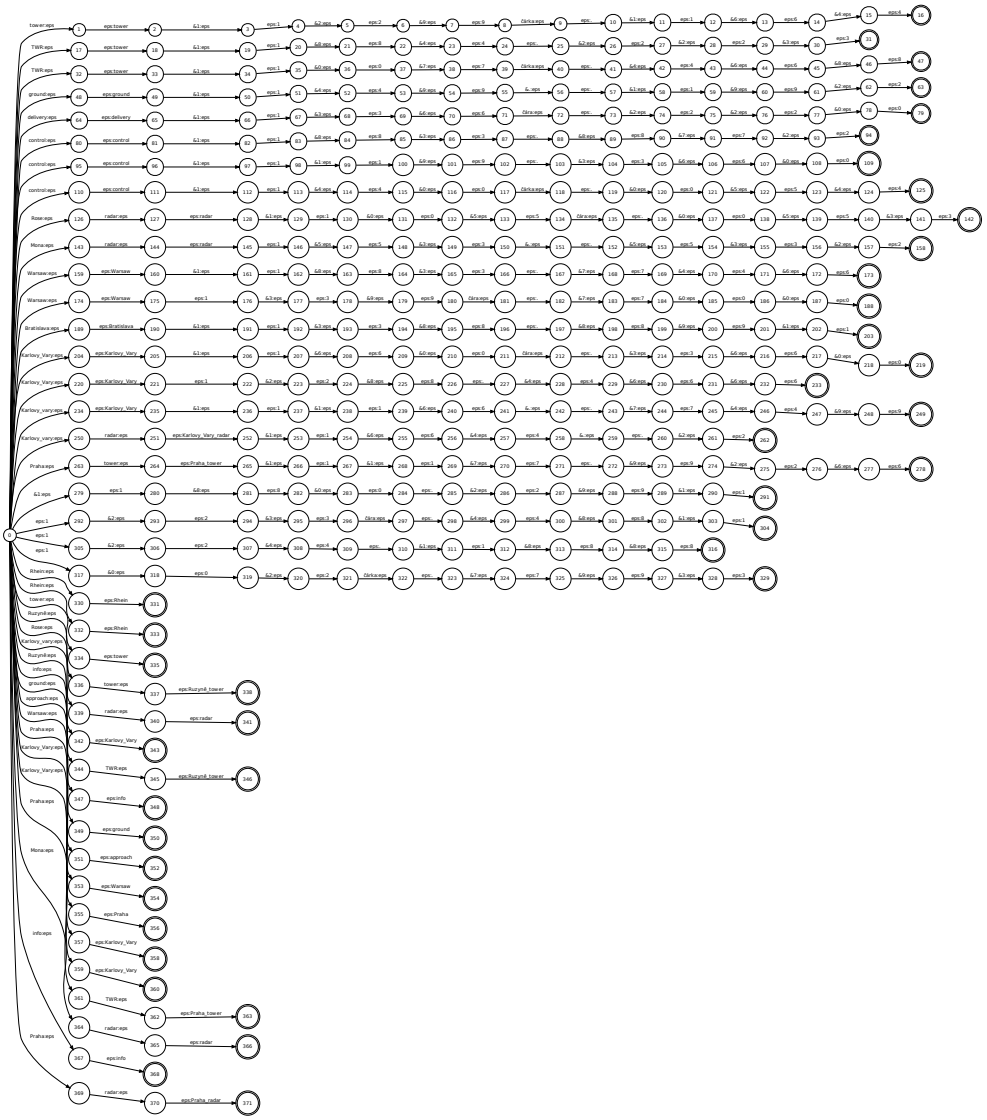
kde **frKey** označuje jakási klíčová slova (pro zkrácení nejsou vypsána všechna) a **digit09** může být zastoupeno jakoukoli číslicí. Do hranatých závorek „[ ]“ dáváme část promluvy, která může být v dané cestě vynechána. Ve složených závorkách „{ }“ udáváme význam předcházejícího prvku.

Převedení gramatik do formátu .fst

Formát .abnf je sice přehledný, příjemný na implementaci a vhodný pro pozdější úpravy, nicméně abychom mohli gramatiky použít, potřebujeme je transformovat do formátu konečného automatu - **.fst**. O to se stará skript *abnf2fst_transducer.sh*, který gramatiky „překlopí“ hned do několika dalších formátů (mimo jiné i do známého .xml), ovšem nás nejvíce zajímá již zmíněný konečný automat .fst a především jeho upravená verze s redukcí prázdných hran **.rmeps.fst**. Toto je již konečný formát gramatiky připravený pro detekci dané entity v promluvě.

Vykreslení automatů ve formátu .pdf

Pro kontrolu správnosti převedení „papírové“ gramatiky do počítače můžeme použít skript *fst2pdf_allSE.sh*, který pro každý konečný automat ve formátu .rmeps.fst vytvoří jeho vizuální podobu ve formátu .pdf a navíc do textových souborů vypíše všechny možné průchody danými konečnými automaty. O vypsání nejprve všech a poté také 40 náhodných cest jednotlivými automaty se stará skript *traverse_path.py* [3]. Vizuální kontrolu gramatik provedeme postupně pro všechny entity. Je zřejmé, že rozsáhlé konečné automaty nemají v malém rozlišení velkou vypovídající hodnotu, přesto pro představu uvedeme 40 náhodných průchodů konečným automatem pro entitu **FR** (frekvence).



Obrázek 4.8: Čtyřicet náhodných průchodů konečným automatem reprezentujícím gramatiku navrženou pro entitu FR

Nyní si pojďme shrnout, na jaká data chceme vytvořené gramatiky ve formě konečných automatů aplikovat. Postupně budeme detekovat sémantické entity ze vstupů:

1. vstup získaný **ručním přepisem** promluvy (označení GRM-text)
2. vstup získaný z ASR jako nejlepší (**1-best**) hypotéza (GRM-Hyp1best)
3. vstup ve formě ASR mřížky představující **N** (v našem případě N=60) nejlepších hypotéz (GRM-HypNbest)

Získání ASR mřížek

Z první kapitoly již víme, co pojem ASR mřížka znamená. Nyní si pojďme popsat, jak jsme mřížky pro jednotlivé promluvy získali pro tuto práci.

Použili jsme standardní rozpoznávač řeči založený na skrytých Markovových modelech (viz 1. kapitola). Trifónový akustický model má čtyři tisíce stavů se směsí 16 Gaussových křivek na stav a třístavovou strukturu. Akustický signál byl vzorkován na 8 kHz a 16 bitech na vzorek. Jazykový model byl trénován jako standardní tri-gramový model z anotovaných trénovacích dat. Použitý slovník obsahuje 10 300 slov. Abychom z rozpoznaných mřížek mohli detekovat sémantické entity, mřížky byly převedeny z formátu **.htk.slf** na formát konečného automatu **.htk.fst**.

Vyhodnocení detekce sémantických entit pomocí gramatik

Pro vyhodnocení procesu detekce potřebujeme získat „slovníkový“ soubor. Skript **gramspot.py** z práce [1] dokáže vytvořit slovníkové soubory pro všechny tři typy vstupu. Protože gramspot.py zapisuje také pro naši úlohu nadbytečnou informaci o konkrétních hodnotách entit, byl napsán skript **dictSimplifier.py**, který hodnoty entit z výsledku skriptu gramspot.py odstraní tak, jak je ukázáno na následujícím příkladu:

- řádka z výstupu gramspot.py:

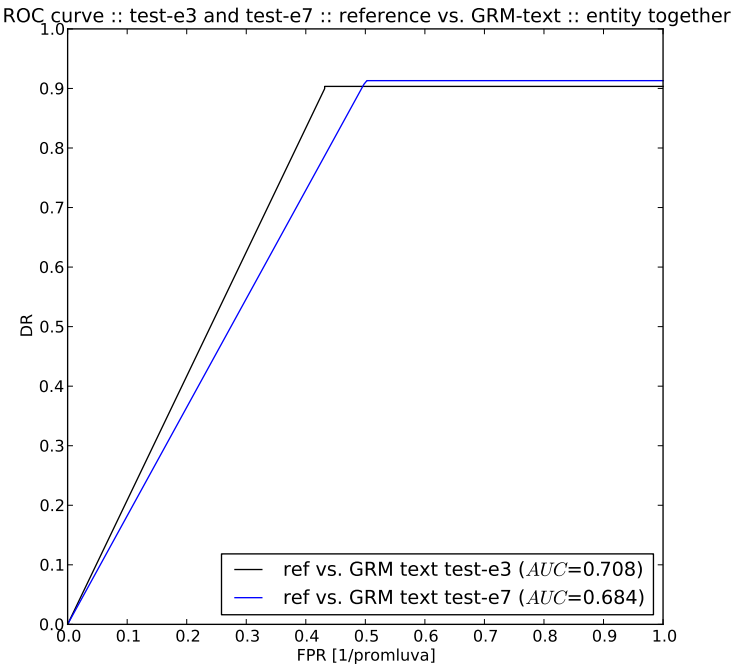
```
u'e3_ACCL-0Ap7RG_00004':
{u'CS:Air_Berlin:5:3:8:M': 1.0, u'FL:1:3:0': 1.0}
```

- tatáž položka slovníku (řádka souboru) po zásahu dictSimplifier.py:

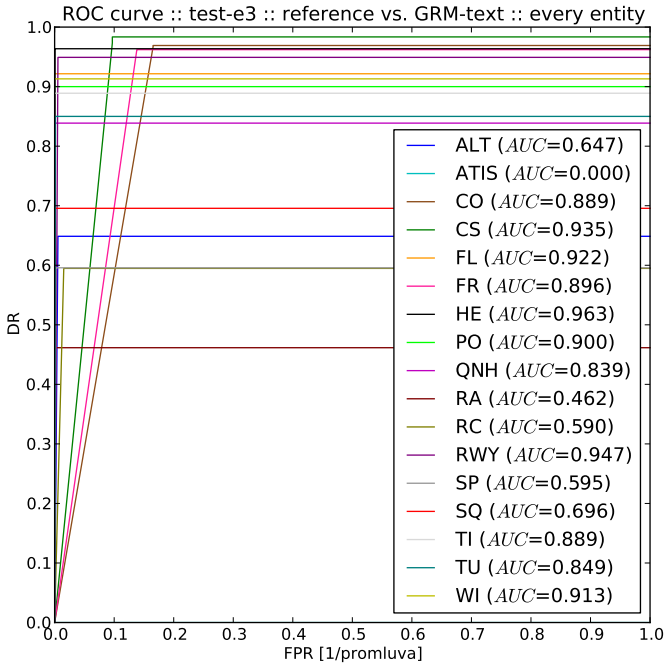
```
'e3_ACCL-0Ap7RG_00004': {'CS': 1.0, 'FL': 1.0}
```

Po použití již známého skriptu **entityFilter.py** máme vše potřebné k porovnání s referenčními daty.

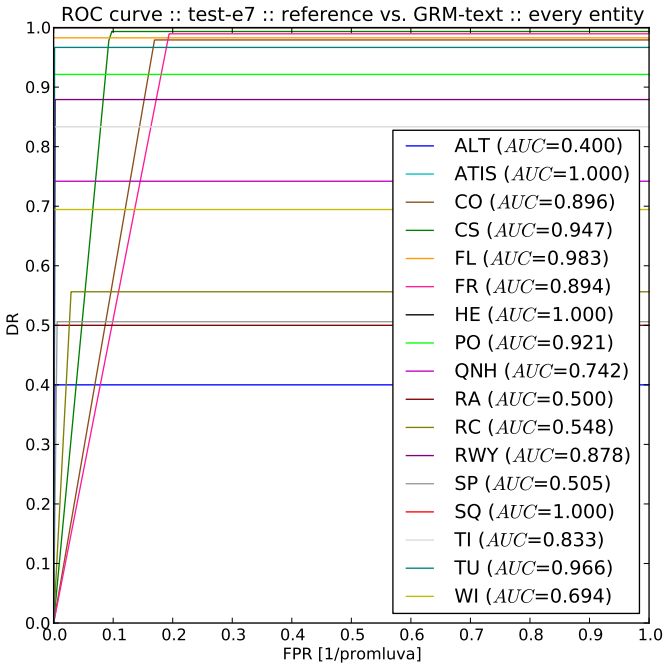
1. metoda GRM aplikovaná na ruční přepisy promluv



Obrázek 4.9: Gramatiky aplikované na ruční přepis promluv pro všechny entity dohromady - sada e3 a sada e7

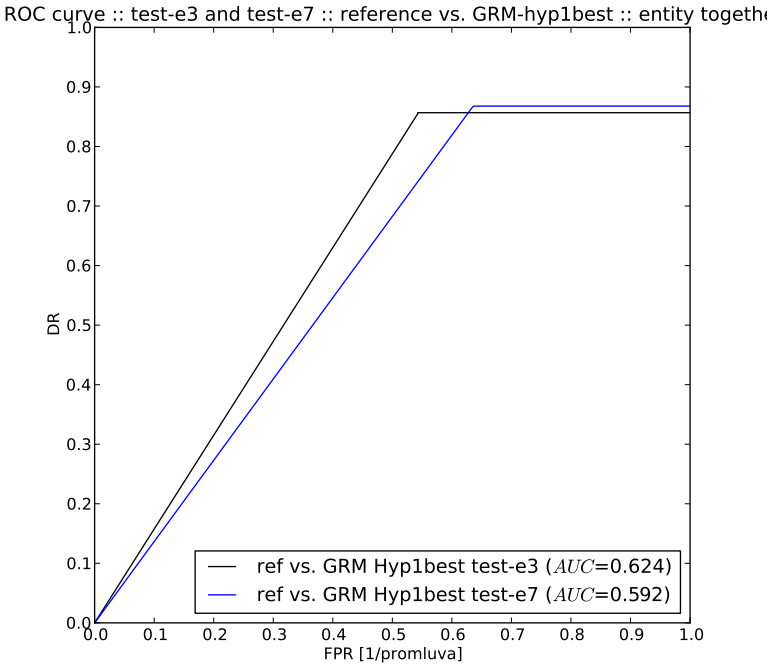


Obrázek 4.10: Gramatiky aplikované na ruční přepis promluv pro každou z entit zvlášť - sada e3

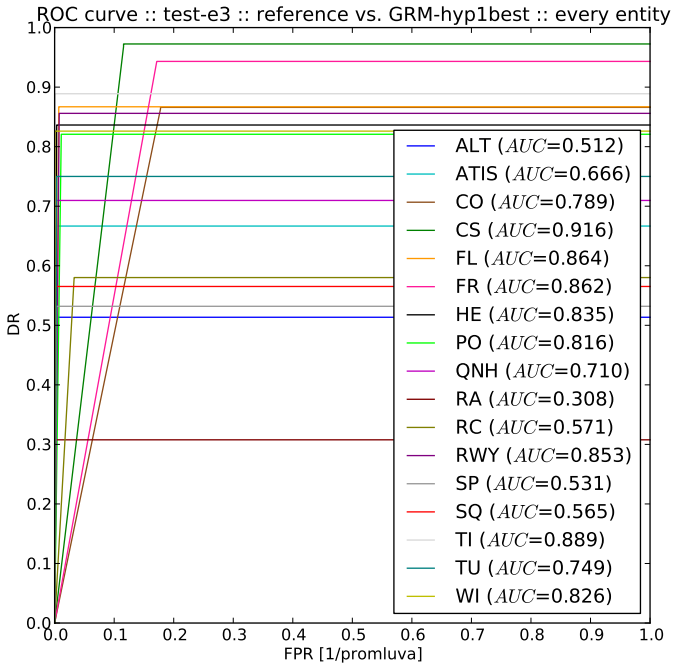


Obrázek 4.11: Gramatiky aplikované na ruční přepis promluv pro každou z entit zvlášť - sada e7

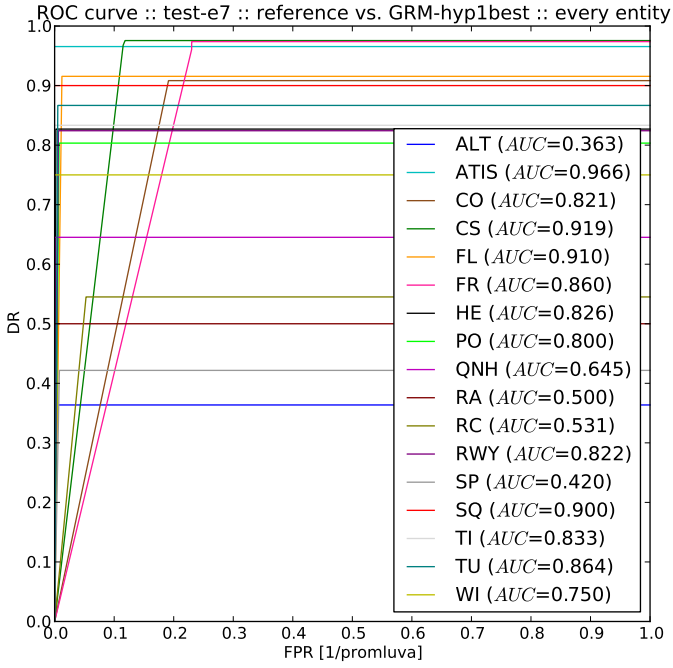
2. metoda GRM aplikovaná na ASR 1-best hypotézu



Obrázek 4.12: Gramatiky aplikované na vstup ve formě nejlepší hypotézy ASR pro všechny entity dohromady - sada e3 a sada e7

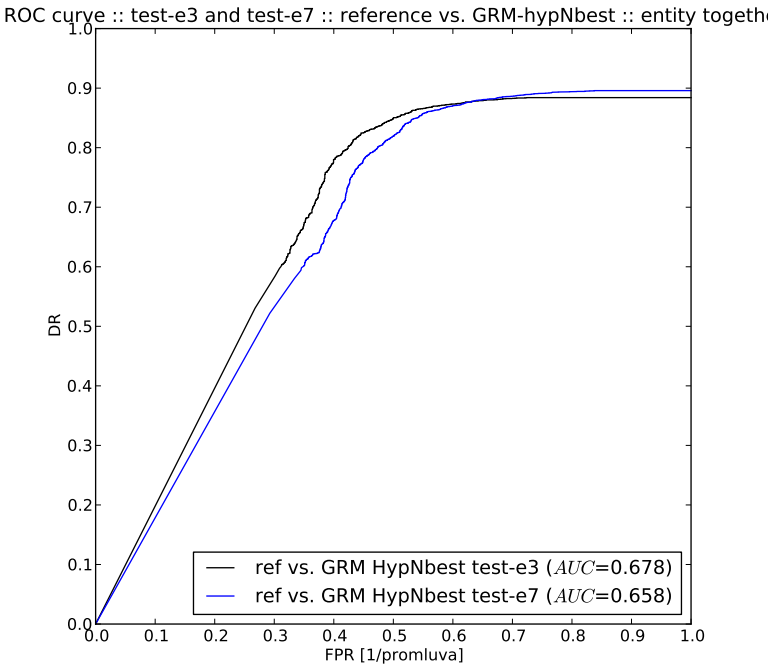


Obrázek 4.13: Gramatiky aplikované na vstup ve formě nejlepší hypotézy ASR pro každou z entit zvlášť - sada e3

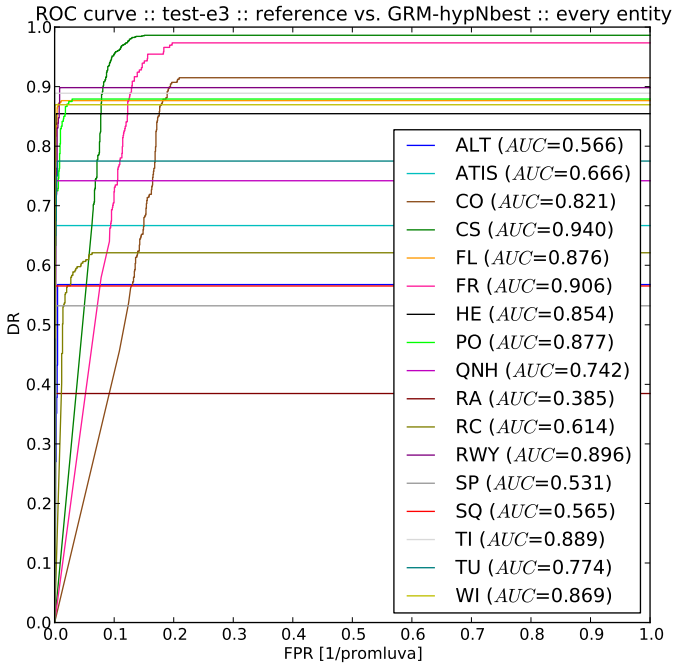


Obrázek 4.14: Gramatiky aplikované na vstup ve formě nejlepší hypotézy ASR pro každou z entit zvlášť - sada e7

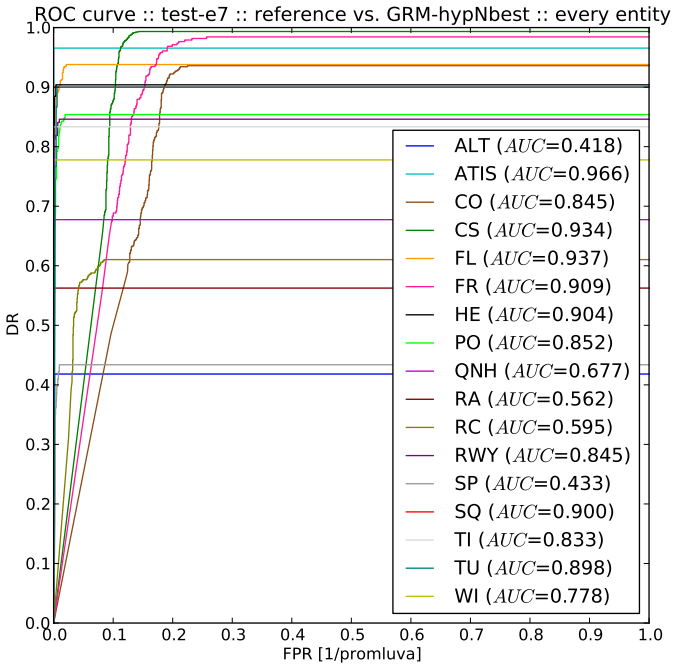
3. metoda GRM aplikovaná na ASR mřížku



Obrázek 4.15: Gramatiky aplikované na vstup ve formě ASR mřížky (N-best hypotéz) pro všechny entity dohromady - sada e3 a sada e7



Obrázek 4.16: Gramatiky aplikované na vstup ve formě ASR mřížky (N-best hypotéz) pro každou z entit zvlášť - sada e3



Obrázek 4.17: Gramatiky aplikované na vstup ve formě ASR mřížky (N-best hypotéz) pro každou z entit zvlášť - sada e7

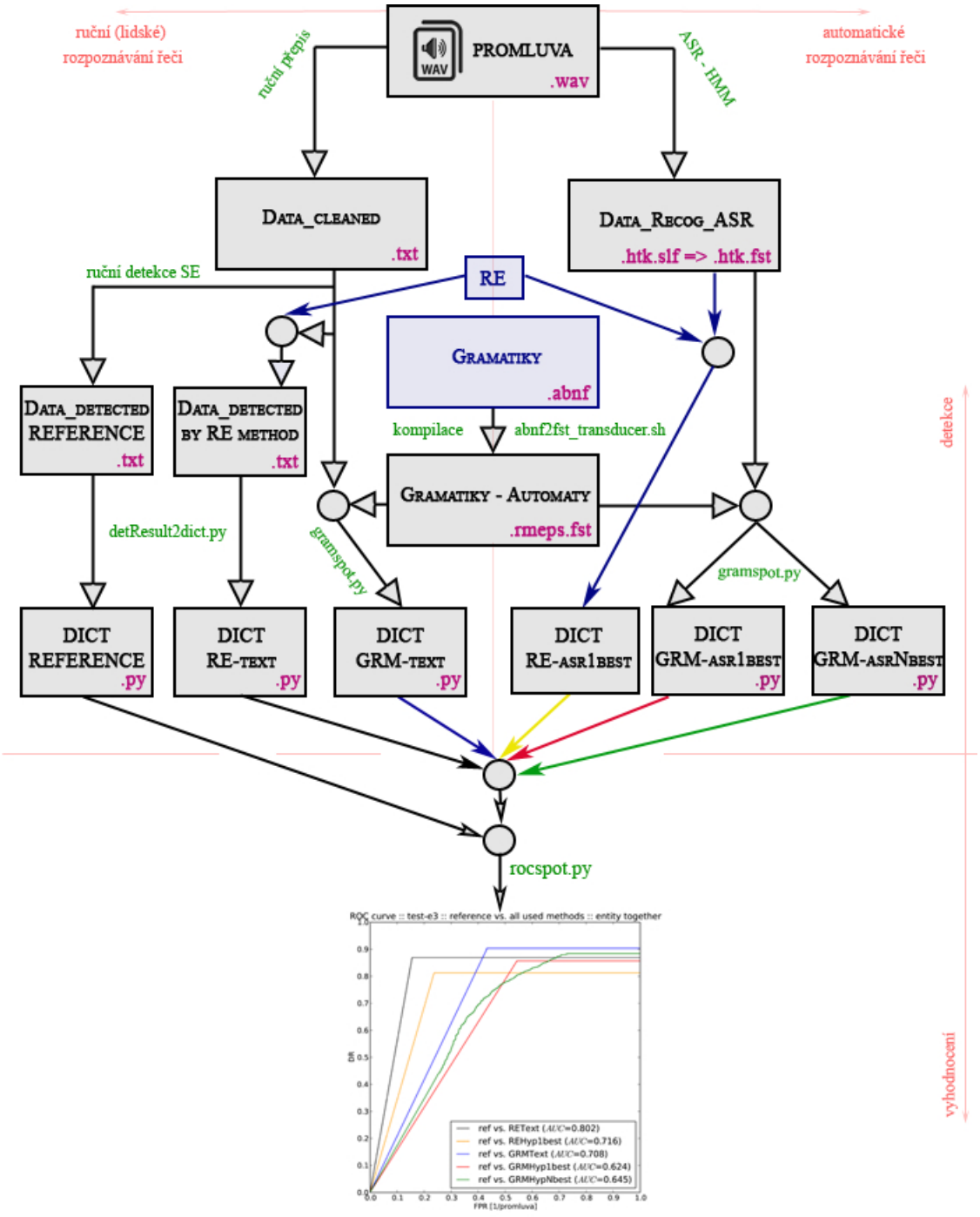
Z výsledných grafů můžeme usuzovat na mírně horší kvalitu detekce v porovnání s metodou RE, nicméně, jak již bylo několikrát zmíněno, gramatiky můžeme snadno upravovat. Z výsledků tedy odhalíme entity, které se detekují hůře (v této úloze jsou to například **RC** či **RA**), a iterativním vývojem, zahrnujícím nové konzultace s experty či studium oboru z nových zdrojů, můžeme kvalitu detekce postupně vylepšovat.

S detekcí většiny entit můžeme být téměř spokojeni. Nezaskočí nás, že výsledky detekce z ASR mřížky jsou obecně lepší než výsledky detekce z nejlepší hypotézy ASR. Samozřejmě nejlépe dopadla detekce z ručního přepisu, z kterého byla tvořena také referenční data a který neobsahuje nepřesnosti nedokonalého rozpoznávače řeči.

Detailnější grafy i text popisující porovnání metody gramatik s metodou RE najdeme níže v textu.

4.3.5 Souhrn postupu práce

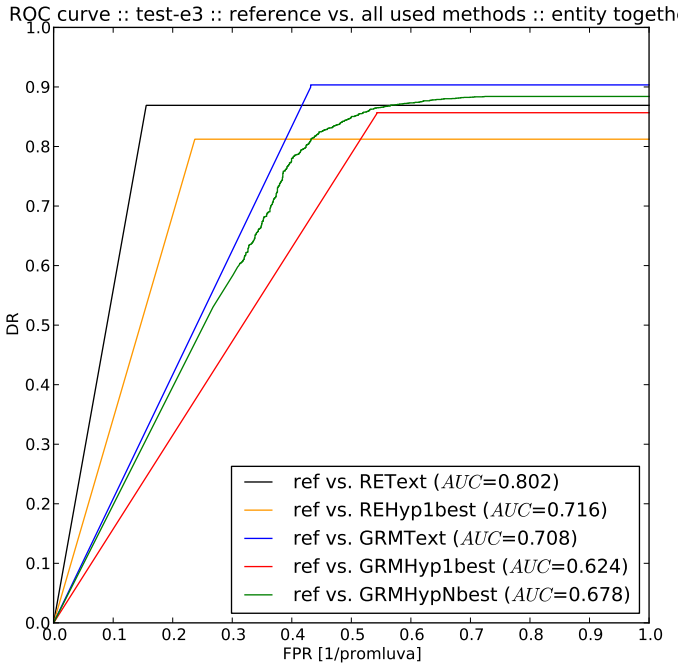
Celkový pohled na výše popsaný postup práce nabízí následující znázornění:



Obrázek 4.18: Detekce sémantických entit s vyhodnocením - postup

4.4 Porovnání použitých metod

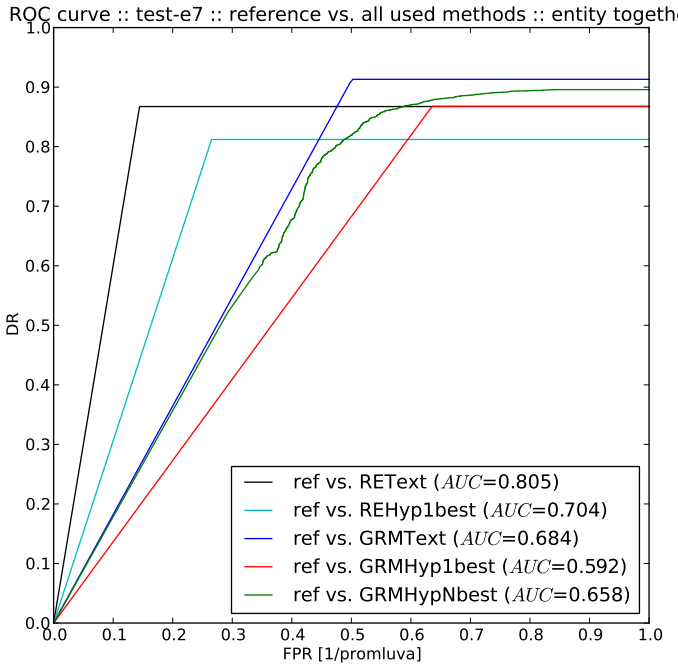
Následující grafy nabízejí porovnání kvality detekce jednotlivých metod postupně pro obě sady testovacích dat. V obou případech se jedná o výsledky uvažující detekci všech entit dohromady. Grafy pro jednotlivé entity zde pro zachování kompaktnosti práce neuvedeme, nicméně v adresáři s projektem jsou vygenerovány a připraveny k nahlédnutí.



Obrázek 4.19: Porovnání jednotlivých metod detekce při různých typech vstupu - sada e3

Podle zvoleného kritéria, plochy pod ROC křivkou, má metoda RE oproti gramatikám mírně navrh, ovšem z grafu také vyplývá, že gramatiky dokázaly správně detekovat více entit než RE. Celková kvalita detekce pomocí gramatik však „trpí“ na množství „falešných poplachů“, tedy označení entit na místech, kde nejsou.

Výsledky metod, které mají na vstupu výstup z automatického rozpoznávače řeči, mohou být zašuměné jeho nepřesnostmi, neboť se porovnává s referenčními daty vytvořenými z ručního přepisu promluv.

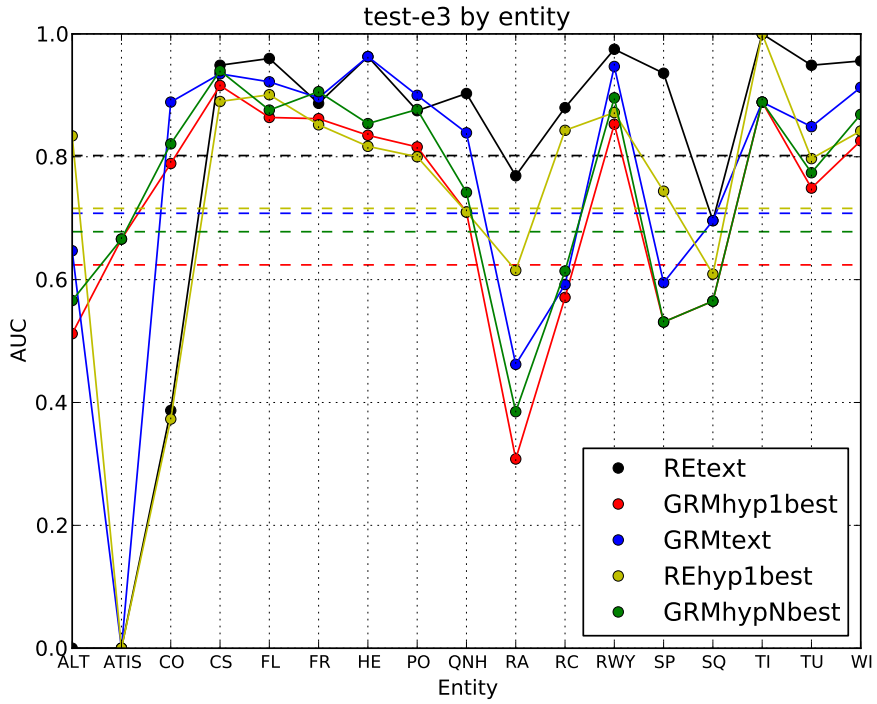


Obrázek 4.20: Porovnání jednotlivých metod detekce při různých typech vstupu - sada e7

Kompletní souhrn všech hodnot **AUC**:.

Sém. entita	RE-text	RE-ASR1best	GRM-text	GRM-ASR1best	GRM-ASRNbest
⟨**ALT/**⟩	0.000	0.834	0.647	0.512	0.566
⟨**ATIS/**⟩	0.000	0.000	0.000	0.666	0.666
⟨**CO/**⟩	0.387	0.373	0.889	0.789	0.821
⟨**CS/**⟩	0.949	0.890	0.935	0.916	0.940
⟨**FL/**⟩	0.960	0.901	0.922	0.864	0.876
⟨**FR/**⟩	0.887	0.852	0.896	0.862	0.906
⟨**HE/**⟩	0.963	0.817	0.963	0.835	0.854
⟨**PO/**⟩	0.875	0.800	0.900	0.816	0.877
⟨**QNH/**⟩	0.903	0.710	0.839	0.710	0.742
⟨**RA/**⟩	0.769	0.615	0.462	0.308	0.385
⟨**RC/**⟩	0.880	0.843	0.590	0.571	0.614
⟨**RWY/**⟩	0.975	0.872	0.947	0.853	0.896
⟨**SP/**⟩	0.936	0.744	0.595	0.531	0.531
⟨**SQ/**⟩	0.696	0.609	0.696	0.565	0.565
⟨**TI/**⟩	1.000	1.000	0.889	0.889	0.889
⟨**TU/**⟩	0.949	0.797	0.849	0.749	0.774
⟨**WI/**⟩	0.956	0.824	0.913	0.826	0.869
Dohromady	0.802	0.716	0.708	0.624	0.678

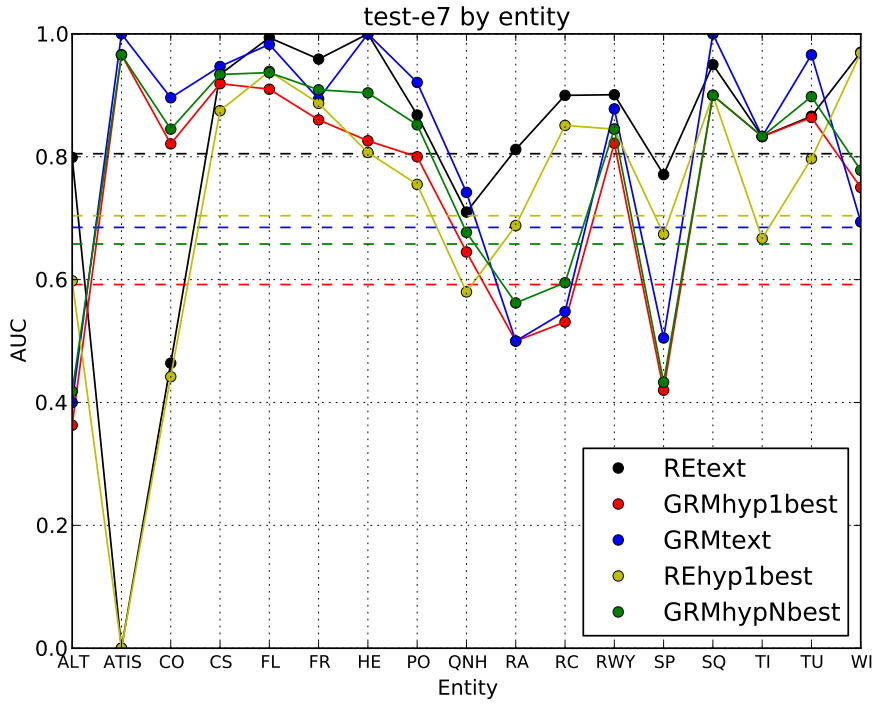
Tabulka 4.4: Kompletní souhrn AUC hodnot - sada e3 (1.0 :: dokonalá detekce / 0.0 :: nic nedetekováno správně)



Obrázek 4.21: Vyhodnocení jednotlivých metod na entitách, čárkovaně jsou znázorněny celkové hodnoty pro danou metodu - sada e3

Sémantická entita	RE-text	RE-ASR1best	GRM-text	GRM-ASR1best	GRM-ASRNbest
⟨**ALT/**⟩	0.799	0.598	0.400	0.363	0.418
⟨**ATIS/**⟩	0.000	0.000	1.000	0.966	0.966
⟨**CO/**⟩	0.464	0.442	0.896	0.821	0.845
⟨**CS/**⟩	0.934	0.875	0.947	0.919	0.934
⟨**FL/**⟩	0.994	0.939	0.983	0.910	0.937
⟨**FR/**⟩	0.959	0.887	0.894	0.860	0.909
⟨**HE/**⟩	1.000	0.807	1.000	0.826	0.904
⟨**PO/**⟩	0.868	0.755	0.921	0.800	0.852
⟨**QNH/**⟩	0.710	0.580	0.742	0.645	0.677
⟨**RA/**⟩	0.812	0.688	0.500	0.500	0.562
⟨**RC/**⟩	0.900	0.851	0.548	0.531	0.595
⟨**RWY/**⟩	0.901	0.845	0.878	0.822	0.845
⟨**SP/**⟩	0.771	0.674	0.505	0.420	0.433
⟨**SQ/**⟩	0.950	0.900	1.000	0.900	0.900
⟨**TI/**⟩	0.833	0.667	0.833	0.833	0.833
⟨**TU/**⟩	0.866	0.797	0.966	0.864	0.898
⟨**WI/**⟩	0.970	0.969	0.694	0.750	0.778
Dohromady	0.805	0.704	0.684	0.592	0.658

Tabulka 4.5: Kompletní souhrn AUC hodnot - sada e7 (1.0 :: dokonalá detekce / 0.0 :: nic nedetekováno správně)



Obrázek 4.22: Vyhodnocení jednotlivých metod na entitách, čárkovaně jsou znázorněny celkové hodnoty pro danou metodu - sada e7

4.5 Návrhy na vylepšení

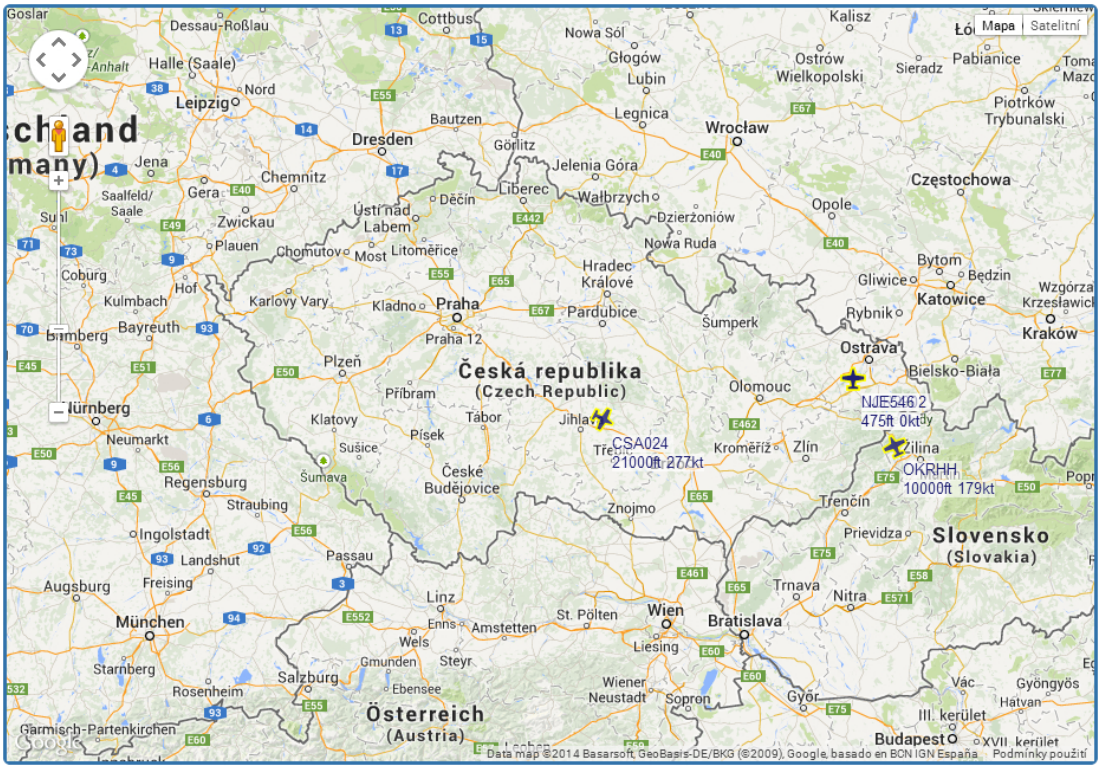
Vývoj gramatik je iterativní proces. Postupným zdokonalováním expertních znalostí jsme po několikanásobných aktualizacích gramatik dosáhli výrazného zlepšení detekce. Můžeme proto předpokládat, že ani výsledky zaznamenané v této práci nejsou nejlepší možné.

Jedním z plánů do budoucna by měl být test navržených gramatik na nových datech. Z většího množství dat jistě získáme nové poznatky o jednotlivých entitách. Vyloučeno není ani zavedení nových entit.

Výsledná ROC křivka pro detekci gramatikami se v grafu „vyšplhala“, v porovnání s RE, vysoko. Musíme si uvědomit, že kritérium podle kterého výsledky vyhodnocujeme, plocha pod ROC křivkou, pro nás nemusí mít úplně platnou vypovídající hodnotu. Metoda gramatik doplácí na množství „falešných poplachů“, což by ovšem nemusel být v konečném výsledku při globálním porozumění takový problém. Jedním z návrhů na vylepšení by tedy možná mohla být iniciativa při hledání alternativního způsobu vyhodnocení detekce.

5. Ukázková aplikace

V rámci řešeného projektu bylo vytvořeno webové demo, u kterého si uživatel může vyzkoušet roli operátora letového provozu.



Obrázek 5.1: Webové demo: mapa leteckého provozu

Na mapě (obrázek 5.1) nalezneme několik náhodně generovaných letadel, která mají dokonce svá označení odpovídající skutečným formátům entity „call sign“ (CS) - volacího znaku letadla. Dále si můžeme u každého letadla v popisku všimnout jeho aktuální letové hladiny a rychlosti letu. Z natočení letadla můžeme odhalit jeho aktuální směr.

Demo funguje na principu hlasové komunikace, kdy uživatel (operátor řízení) zavolá dané letadlo a dá mu nějaký pokyn (například pokyn k poklesu na určitou letovou hladinu). Počítač rozpozná uživatelovu promluvu, porozumí informaci, vybere správné letadlo podle volacího znaku a automatický pseudopilot smysluplně odpoví (většinou potvrzením informace). Dané letadlo poté na mapě skutečně splní uvedený pokyn.

Ukázka procesu celkového porozumění

Uvažujme promluvu operátora řízení směrem k pilotovi:

```
CSA 024 descend to flight level two zero zero please
```

Nejprve se provede detekce sémantických entit, jejíž výstup by pro uvedenou promluvu vypadal takto:

```
<CS/> descend to <FL/> <CO/>
```

Výstupem je informace, že promluva obsahuje volací znak letadla, letovou hladinu a nějakou zdvořilostní frázi. Tato data nyní zpracovává proces "vyššího"porozumění, který nalezne globální význam promluvy:

```
Pilote letadla CSA 024 klesej do letové hladiny 200
```

Znázornění přínosu gramatik

Předpokládejme nyní, že systém automatického rozpoznávání řeči nerozpozná správně volací znak „CSA 024“ a řekněme, že ho označí z 75% jako „CSA 04“ a pouze z 25% jako „CSA 024“. „CSA 04“ neodpovídá žádné sémantické entitě.

V případě detekce z nejlepší (1-best) hypotézy ASR mřížky se informace o 25% pravděpodobnosti výskytu hypotézy „CSA 024“ zcela ztratí, metoda s regulárními výrazy entitu nebude detekovat a tím dojde ke zmatení systému porozumění globální úrovně, neboť promluva najednou přestane dávat smysl.

Gramatiky, které dokáží detekovat z ASR mřížky, informaci o 25% pro hodnotu „CSA 024“ uchovají a k dalšímu zpracování pošlou přesné výsledky, tedy, že se jedná na 75% o „CSA 04“ a na 25% o „CSA 024“. Protože první varianta neoznačuje žádné reálné letadlo, systém zvolí alternativu v podobě „CSA 024“ a dokáže se tak s nepřesností v rozpoznání promluvy vypořádat.

Principiálně bude systém postupovat tak, jak by to v dané situaci dělal i člověk.

Závěr

Citát pana Dr Emersona Pugha zmíněný v úvodu jsme touto prací sice vyvrátit nedokázali, nicméně, stanovené cíle se nám podařilo naplnit. Měřítkem kvality implementované metody mělo být porovnání s člověkem a nyní, podpořeni získanými výsledky, můžeme konstatovat, že počítač řízený implementovanými algoritmy řeší danou úlohu na většině testovacích datech stejně dobře jako člověk.

Podíváme-li se do výsledkových tabulek, může se zdát, že navržené gramatiky fungují hůře než metoda s regulárními výrazy. Nesmíme však zapomenout zmínit způsob vyhodnocení, který se staví na velikosti plochy pod ROC křivkou. Prohlédneme-li si pozorně výsledné grafy, můžeme si všimnout, že gramatiky dokázaly detekovat více entit než RE, což je pro nás zásadní. S větším množstvím „falešných poplachů“ si již poradí systém „vyššího“ porozumění.

Možnost detekce z ASR mřížek umožní redukovat chyby při rozpoznávání řeči ve výsledném dialogovém systému. V budoucnu bude zajímavé testovat gramatiky na nových datech. Obecně se očekávají podobné výsledky, neboť na rozdíl od regulárních výrazů, gramatiky byly navrhovány zcela obecně.

V této práci byla představena metoda detekce sémantických entit na komunikaci v leteckém provozu, obecně však nabízí nevídaný potenciál. Sémantické entity můžeme najít prakticky v jakémkoli oboru či odvětví lidské komunikace, potom už tedy zbývá jen najít motivaci a důvod pro zapojení stroje do systému.

Právě tento smysl účasti umělé inteligence při lidských činnostech je zcela zásadním předpokladem úspěšného vývoje. Nikdy by se nemělo stát, aby člověk vlivem zapojení stroje v jakémkoliv slova smyslu ztrácel jistotu.

Literatura

[1] Jan Švec. *Diskriminativní model pro porozumění mluvené řeči*. Plzeň, 2013. Disertační práce. Západočeská univerzita.

[2] Adam Chýlek. *Optimalizace gramatik pro automatické rozpoznávání řeči*. Plzeň, 2011. Bakalářská práce. Západočeská univerzita.

[3] J. Psutka, L. Müller, J. Matoušek, and V. Radová. *Mluvíme s počítačem česky*. Academia, Praha, 2006.

[4] Openfst. <http://www.openfst.org/>, April 2011.

[5] Igor Szöke. Jak se počítač učí rozpoznávat mluvenou řeč. <http://www.osel.cz/index.php?clanek=5152>, July 2010.

[6] Miroslav Pecka. Regulární výrazy. <http://www.regularnivyrazy.info>, 2005.

[7] Estimation lemma. Estimation lemma — Wikipedia, the free encyclopedia. <http://en.wikipedia.org>, 2010. [Online; accessed 15-May-2014].

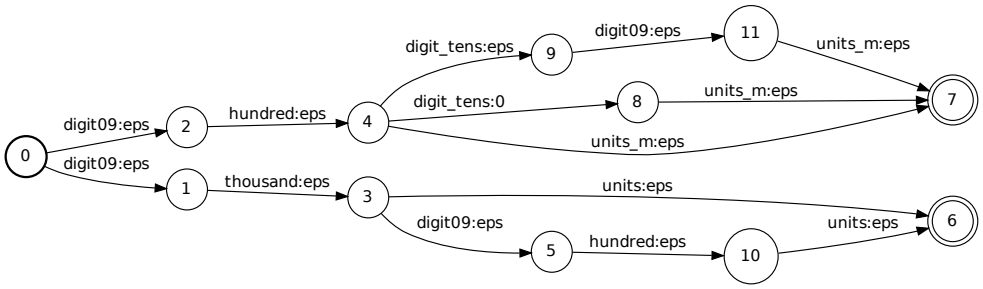
[8] Irena Diatelová. Nejúspěšnější chatboti a jak fungují? <http://nlp.fi.muni.cz/trac/research/wiki/OCemSeMluvi>, 2012.

[9] Pavel Svoboda. Systém řízení letového provozu v Čr. <http://www.aeroweb.cz/clanek.asp?ID=906&kategorie=3>, November 2007.

Příloha A - Návrhy gramatik

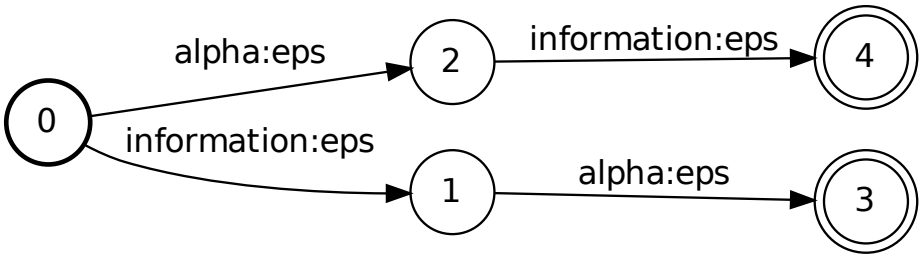
Následující grafy byly automaticky vygenerovány z gramatik ve formátu .abnf. Představují původní ”papírový”návrh. V budoucnu se budou tyto návrhy po každé úpravě některé z gramatik automaticky aktualizovat v pracovním adresáři aplikace.

Altitude (ALT)



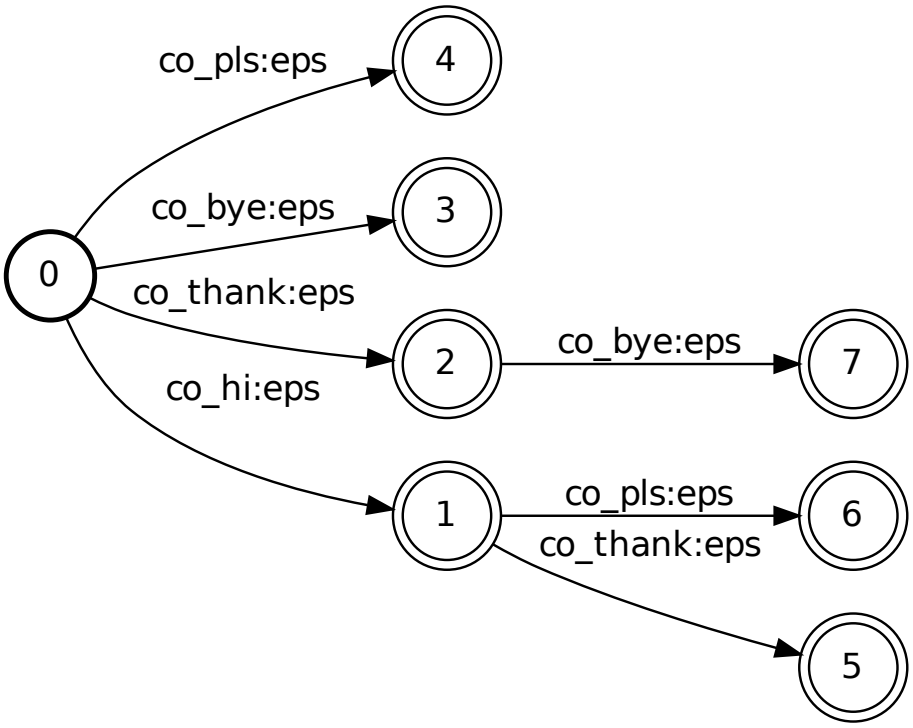
Obrázek 5.2: Gramatika pro sémantickou entitu ALT

Automatic Information Service (ATIS)



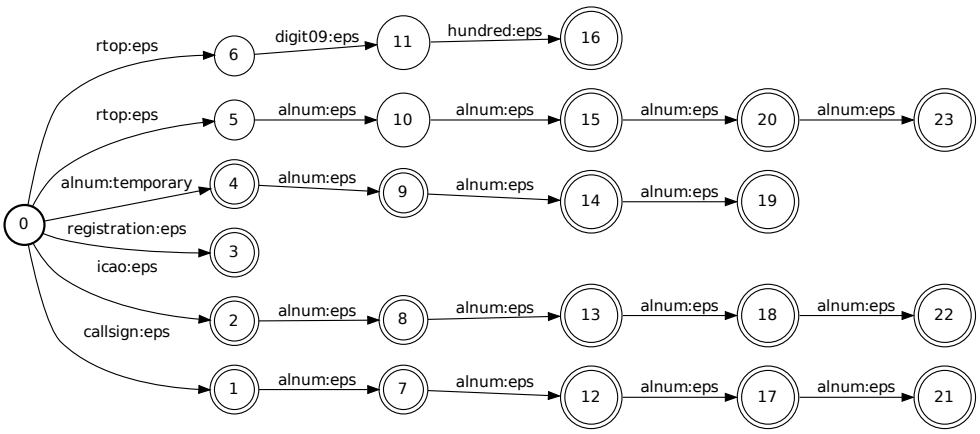
Obrázek 5.3: Gramatika pro sémantickou entitu ATIS

Courtesy (CO)



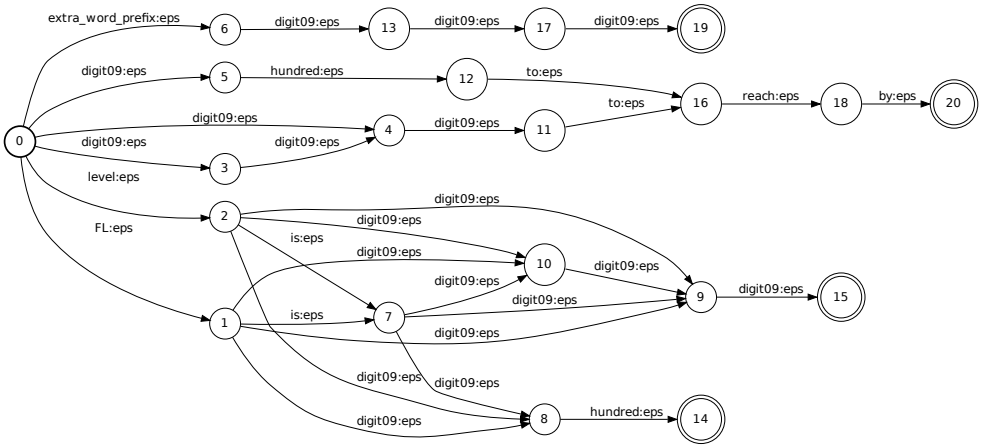
Obrázek 5.4: Gramatika pro sémantickou entitu CO

Call Sign (CS)



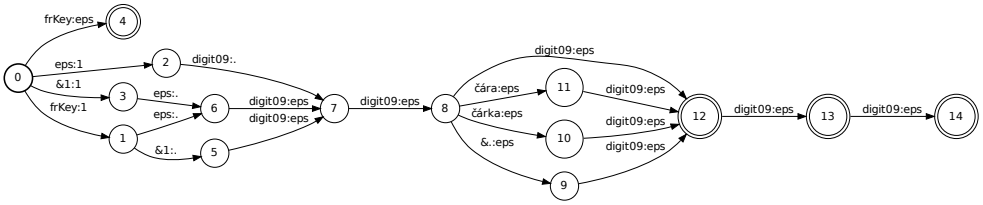
Obrázek 5.5: Gramatika pro sémantickou entitu CS

Flight Level (FL)



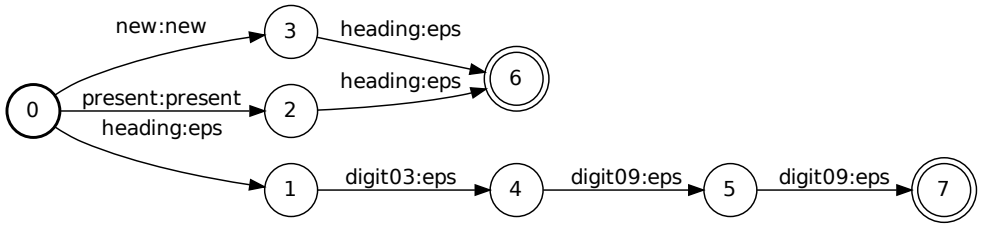
Obrázek 5.6: Gramatika pro sémantickou entitu FL

Frequency (FR)



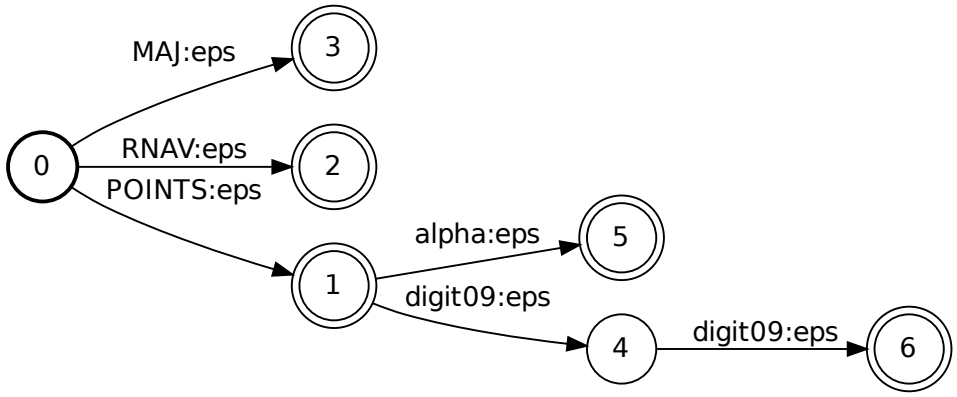
Obrázek 5.7: Gramatika pro sémantickou entitu FR

Heading (HE)



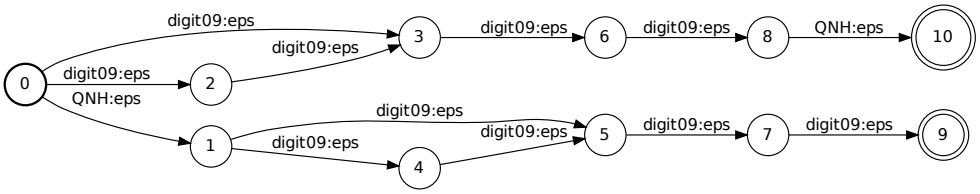
Obrázek 5.8: Gramatika pro sémantickou entitu HE

Point (PO)



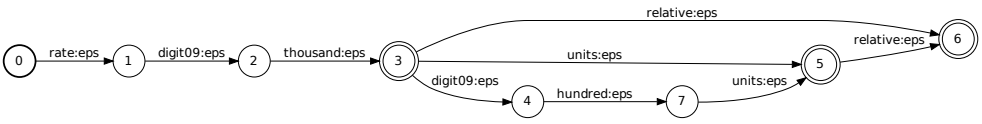
Obrázek 5.9: Gramatika pro sémantickou entitu PO

Q-code (QNH)



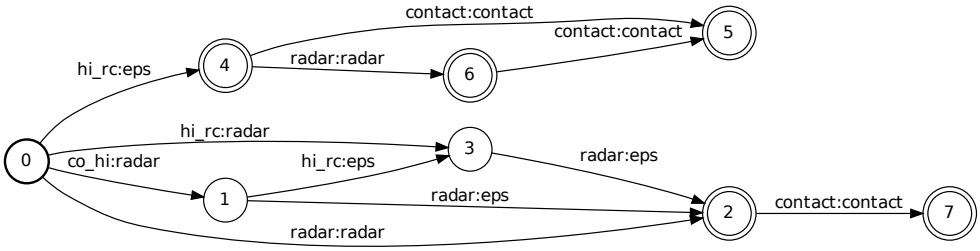
Obrázek 5.10: Gramatika pro sémantickou entitu QNH

Rate (RA)



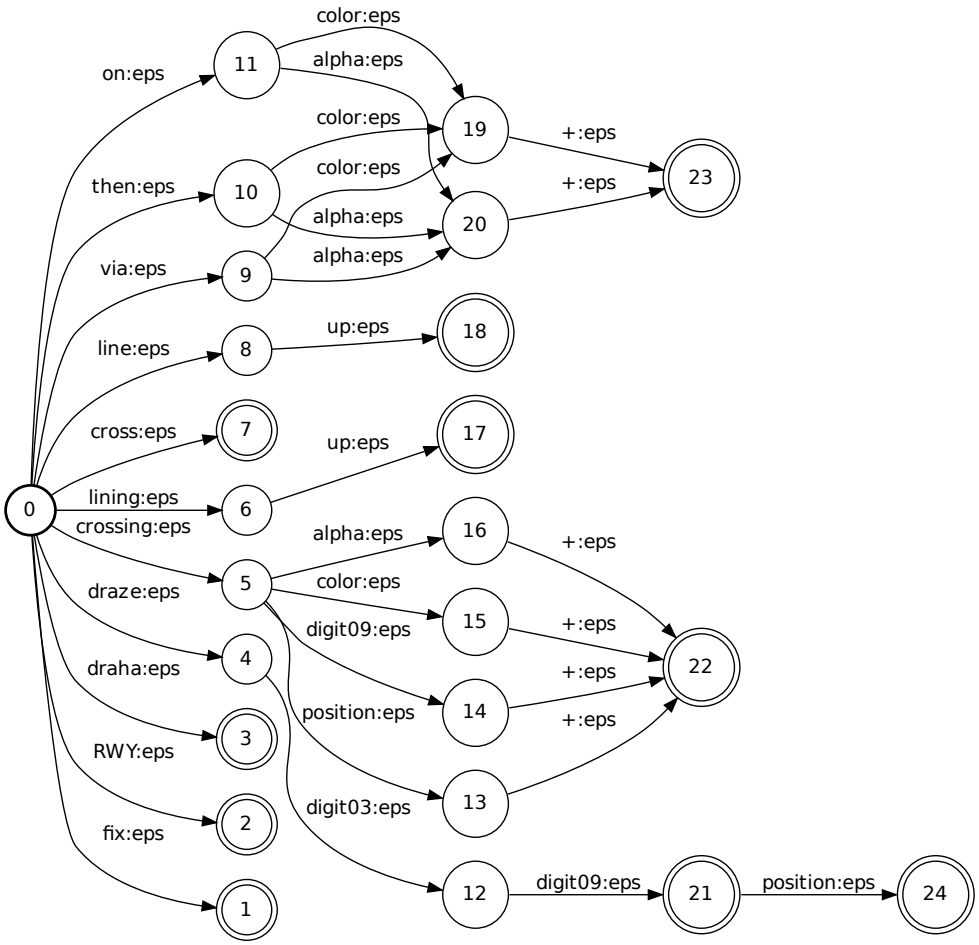
Obrázek 5.11: Gramatika pro sémantickou entitu RA

Radar Contact (RC)



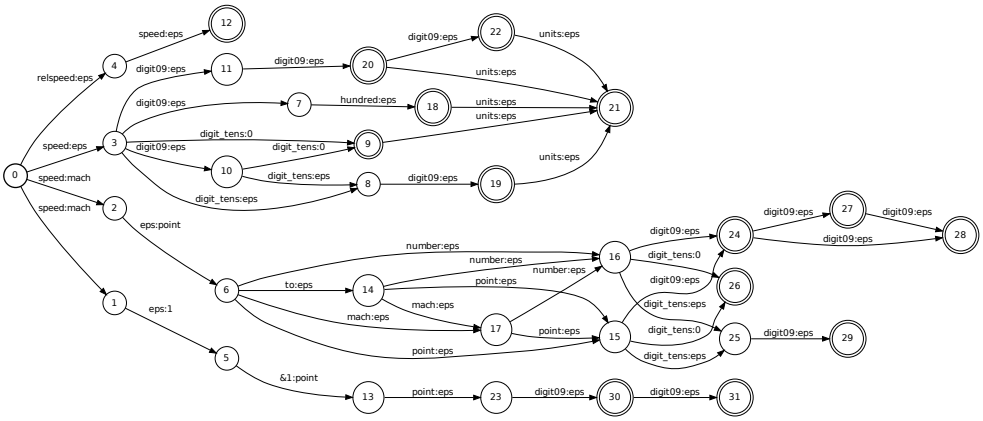
Obrázek 5.12: Gramatika pro sémantickou entitu RC

Runway (RWY)



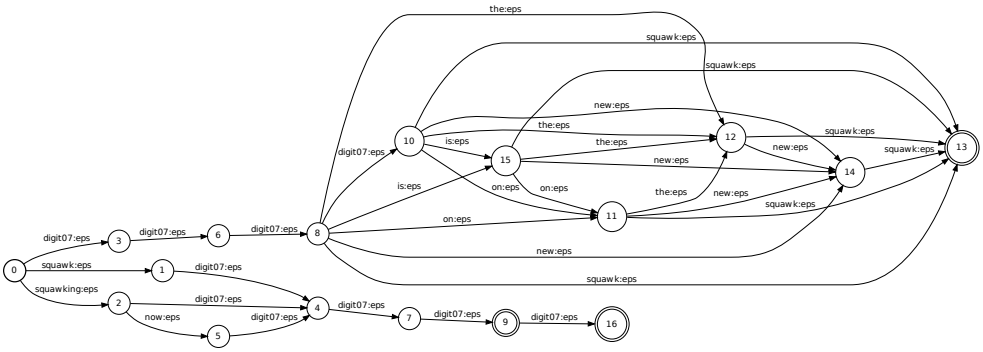
Obrázek 5.13: Gramatika pro sémantickou entitu RWY

Speed (SP)



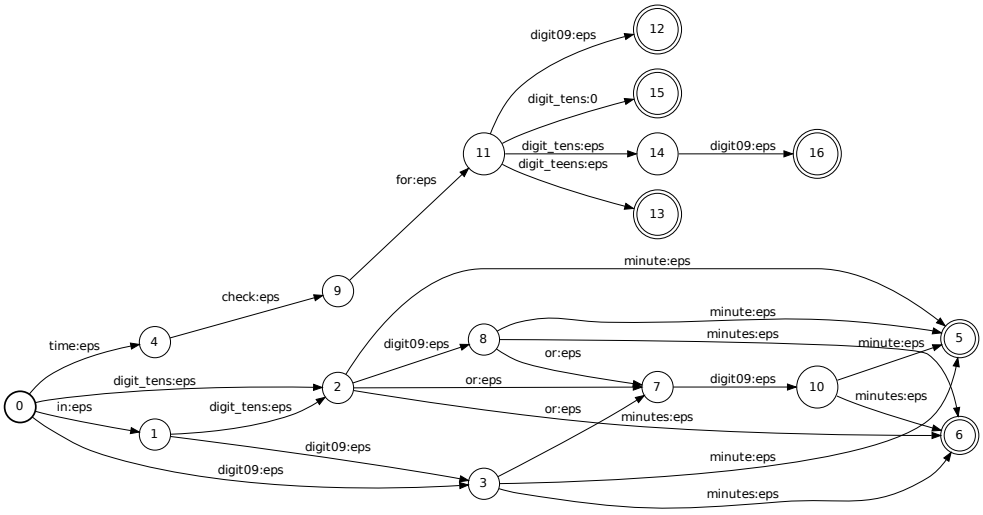
Obrázek 5.14: Gramatika pro sémantickou entitu SP

Squawk (SQ)



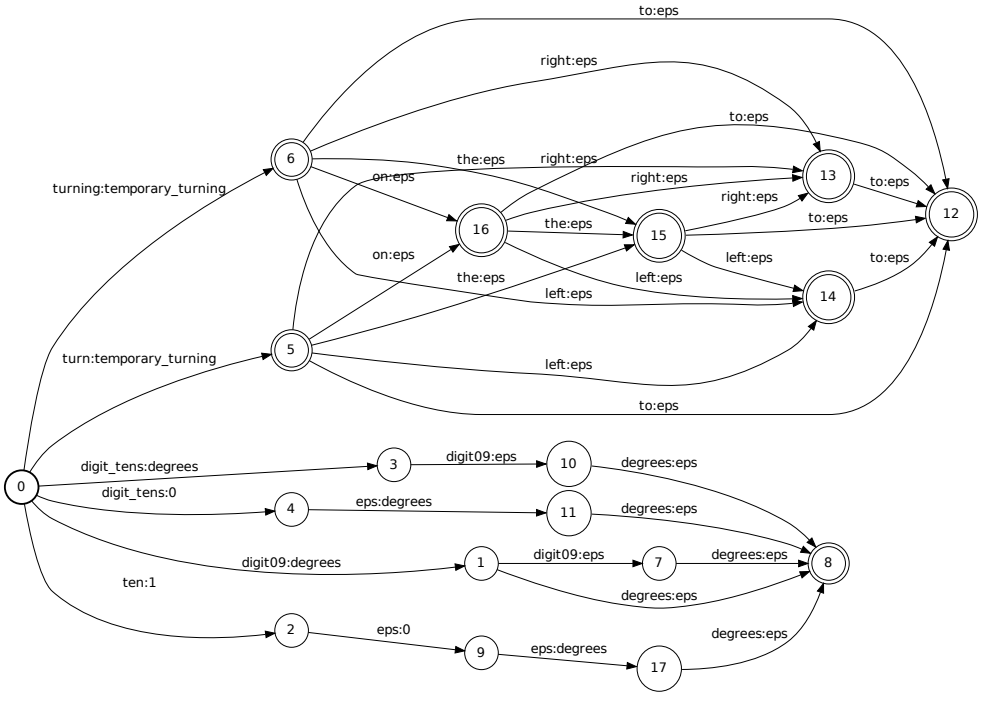
Obrázek 5.15: Gramatika pro sémantickou entitu SQ

Time (TI)



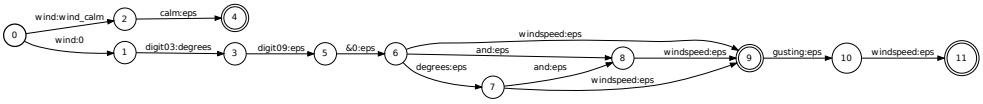
Obrázek 5.16: Gramatika pro sémantickou entitu TI

Turn (TU)



Obrázek 5.17: Gramatika pro sémantickou entitu TU

Wind (WI)



Obrázek 5.18: Gramatika pro sémantickou entitu WI