Laboratorij za analizu teksta i inženjerstvo znanja – TakeLab

Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva
Zavod za elektroniku, mikrolektroniku, računalne i inteligentne sustave
Unska 3, 10000 Zagreb, Hrvatska

© 2012

Autorska prava na sadržaj ovog dokumenta
zadržavaju njegov(i) autor(i) i TakeLab FER.

Niti jedan dio ovog dokumenta ne smije se
distribuirati, modificirati, umnožavati niti prevoditi na drugi jezik
bez prethodnog pismenog odobrenja.

SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 459

Generiranje tekstnog izvještaja na temelju strukturiranih podataka

Veljko Srdarević

Zagreb, lipanj 2012.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA
ODBOR ZA DIPLOMSKI RAD PROFILA

Zagreb, 5. ožujka 2012.

DIPLOMSKI ZADATAK br. 459

Pristupnik: **Veljko Srdarević**
Studij: Računarstvo
Profil: Računarska znanost

Zadatak: **Generiranje tekstnog izvještaja na temelju strukturiranih podataka**

Opis zadatka:

Generiranje prirodnog jezika podgrana je obrade prirodnog jezika koja se bavi generiranjem rečenica i tekstova na prirodnome jeziku temeljem strukturiranih podataka. Sustavi za generiranje prirodnog jezika koriste se za tekstovnu interpretacija informacija sadržanih u bazi podataka u svrhu prezentacije podataka krajnjem korisniku, podrške u odlučivanju, poboljšanju pristupa informacijama i slično.
U okviru diplomskog rada potrebno je proučiti postupke generiranja prirodnog jezika te postupke za automatsku indukciju gramatike iz korpusa. Razraditi postupak za generiranje tekstnog izvještaja na hrvatskome ili engleskom jeziku temeljem strukturiranih podataka, kao što su prometni podatci, vremenska prognoza i sl. Implementirati i ispitati različite metode generiranja, uključivo metode temeljene na predlošcima, jezičnim modelima i domenskoj probabilističkoj kontekstno neovisnoj gramatici. Provesti ručno vrednovanje razvijenog postupka i automatsko vrednovanje temeljem odgovarajućih ispitnih uzoraka i evaluacijske metrike kao što je BLEU. Radu priložiti izvorni programski kod, programsku dokumentaciju i ispitne uzorke.

Zadatak uručen pristupniku: 9. ožujka 2012.
Rok za predaju rada: 21. lipnja 2012.

Mentor:

Doc.dr. sc. Jan Šnajder

Djelovođa:

Prof.dr.sc. Domagoj Jakobović

Predsjednik odbora za diplomski rad profila:

Prof.dr.sc. Siniša Srbljić

Zahvaljujem mentoru doc. dr. sc. Janu Šnajderu na ukazanoj pomoći i strpljenju, te kolegici Sonji Grđan na strpljivom slušanju raznih ideja dok su još bile u zametcima

Sadržaj

1. Uvod

Generiranje prirodnog jezika (engl. *natural language generation, NLG*) grana je umjetne inteligencije i računalne lingvistike čiji je cilj izgradnja računalnih sustava koji mogu stvoriti tekstove na nekom prirodnom jeziku (Reiter i Dale, 2000).

Sustavi NLG su započeli svoj razvoj kao pomoć u pisanju izvješća i strukturiranih podataka. No osim pisanja takvih izvještaja, danas postoje i sustavi NLG koji teže proizvesti tekstove s izričitim elementima ljudskosti, poput humora (Binsted i Ritchie, 1994) ili pričanja priče (Callaway i c. Lester, 2002).

Većina današnjih sustava NLG-a temelje se na ručnoj izradi pravila dubinske generacije kojom je moguće točno odrediti o kakvoj se vrsti teksta radi. Nadalje, većina sustava radi za ograničeni skup podataka i ne može lako mijenjati domenu. Zapravo, za većinu sustava promjena domene zanči gradnju potpuno novog sustava. Osim toga, izgradnja je skup i dugotrajan proces koji često uključuje suradnju s stručnjacima za danu domenu i potpuno ručnu izgradnju cijelokupnog sustava. Postoje i sustavi opće namjene, no oni su ograničeni na tekstove koji opisuju neke strukturirane podatke .Većina takvih sustava za svaku prenamjenu zahtjeva ručno pisanje generacijskih pravila, što znači da jedini dio koji je ponovno koristan jest dio odgovoran za odabir pravila. Iznimka je sustav pCRU (Belz, 2008), koji dio generacijskih pravila piše automatski i skratio je vrijeme izgradnje sustava, no i dalje zahtjeva ručno pisanje viših, apstraktniji pravila. Vrijedi istražiti mogućnost potpuno automatske izgradnje generacijskih pravila. Takav sustav bi uistinu bio sustav opće namjene.

Pristup istražen u ovom radu temelji se na vjerojatnosnim kontekstno neovisnim gramatikama i grupiranju podataka hijerarhijskim aglomerativnim algoritmom grupiranja. Dok su vjerojatnosne kontekstno neovisne gramatike odgovorne za niža pravila sustava (ona pravila koja generiraju sam tekst), grupiranje i pronalažnje odgovarajuće grupe predstavlja viša pravila sustava (ona koja od-

lučuju kojeg će oblika biti generirani tekst). Za skup podataka odabran je skup vremenski prognoza za naftne platforme u Sjevernom moru SumTime-METEO (Sripada et al., 2003), čiji je fokus na prognozama za vjetar i valovitost mora. Ovaj skup podataka je korišten u nekoliko različitih radova poput (Belz, 2008) i (Reiter et al., 2005).

U poglavlju 2 dan je opis problema generiranja prirodnog jezika. U poglavlju 3 je dan opis kontekstno neovisnih gramatika i tehnika njihovog induciranja iz korpusa tekstova jer je njihovo raumijevanje teorijska osnova rada. U poglavlju 5 će biti prikazan pregled područja i predstavljena tri rada različite namjene i različitih tehnika generiranja. Poglavlje 6 opisuje sustav NLG-a koji je izgrađen u sklopu rada, a poglavlje 7 daje opis tehničke implementacije sustava. U poglavlju 8 opisani su dobiveni rezultati,a u poglavlju 9 iznesen je zaključak.

2. Generiranje prirodnog jezika

Sustavi NLG-a generiraju tekstove na nekom prirodnom jeziku temeljem nelingvističkih podataka (Reiter i Dale, 2000). Ti podaci mogu bili bilo kakvi strukturirani podaci, poput n-torki iz baze podataka, ili meta-podatci koji opisuju priču (Callaway i c. Lester, 2002).

Postoji niz sličnosti generiranja prirodnog jezika s drugim, sličnim granama obrade prirodnog jezika poput, strojnog prevođenja (engl. *machine translation, MT*). Dapače, za sustave NLG-a moguće je koristiti iste tehnike vrednovanja koje se tipično koriste za sustave obrade prirodnog jezika (Belz i Reiter, 2006). Osim sa sličnim granama, NLG je moguće usporediti i s razumjevanjem prirodnog jezika (engl. *natural language understanding, NLU*). Sličnosti između NLG-a i NLU-a su sljedeće (Mann, 1987):

1. Leksikon – pretpostavlja taksonomiju riječi, osnovno semantičko značenje i morfologiju riječi;
2. Gramatika – mora postojati barem minimalan skup gramatičkih pravila jezika;
3. Karakteristike diskursa – poput teme i fokusa (engl. *topic*). Također, u ljudskim jezicima nema potrebe da se sve eksplicitno navodi, implicitna materija je jednako važna kao i ona eksplicitno rečena;
4. Karakteristike situacije – Situacija u kojoj se jezik upotrebljava.

NLG se može svrstati u podgranu obrade prirodnog jezika (engl. *natural language processing, NLP*) i tada istražuje (Reiter i Dale, 2000):

- Kakva treba biti interakcija čovjeka i računala;
- što je jasan, razumljiv i čitljiv tekst;
- i kako pokazati računalne podatke da budu razumljivi čovjeku.

2.1. Razlika između generiranja i razumijevanja prirodnog jezika

Generiranje i razumijevanje prirodnog jezika mogu se promatrati kao inverzni problemi. Cilj NLU-a jest uspješno povezati ljudski jezik s odgovarajućom računalnom internom strukturom, a cilj NLG-a jest povezati internu računalnu strukturu s odgovarajućim tekstom ljudskog jezika (Reiter i Dale, 2000). Nažalost ne postoji postupak koji bi razrješavao problem NLU, a da njemu inverzan postupak razriješava problem NLG. To je stoga što se postupak NLU-a definira kao postupak biranja ispravne interpretacije ljudskog jezika, dok se postupak NLG-a definira kao postupak biranja najboljeg načina da se dođe do zadanog cilja (McDonald, 1992). No jedan i drugi postupak mogu koristiti iste resurse poput gramatika, riječnika itd.

2.2. Razlika u primjeni NLG-a

Primjena NLG-a može se podijeliti na dvije vrste: (1) na sustave gdje je izlazni tekst iz sustava podložan ljudskoj kontroli i sustav ne generira čitav tekst, već samo isječke poput sustava opisanih u (Ross Turner, 2010), (Ross Turner, 2006), te (2) na sustave gdje računalo samo generira cijeloviti tekst poput sustava opisanog u (Reiter et al., 2009).

2.3. Osnovna arhitektura sustava NLG

U radu (Reiter, 1994) analizirano je nekoliko sustava NLG-a i predloženi su koraci koje bi arhitektura sustava morala podržavati. Ti koraci su sljedeći:

- *Određivanje sadržaja* (engl. *content determination*) – preslikava početni ulazni sadržaj (npr. odgovor na pitanje ili izricanje neke namjere) na semantičku formu;
- *Planiranje rečenica* (engl. *sentence planning*) – preslikavanje konceptualnih struktura na lingvističke strukture što uključuje agregaciju, utvrđivanje referentnih izraza itd;

– *Površinska realizacija* (engl. *surface realization*) – koristeći apstraktne izlaze prijašnjih koraka izgrađuje se površinski oblik (bio to tekst, odlomak ili samo jedna rečenica);

– *Morfologija* (engl. *morphology*) – postojanje morfološkog koraka tj. modula za morfologiju ovisi o morfološkoj složenosti jezika, a modul je zadužen za provjeru i postavljanje riječi u tekstu u ispravne morfološke forme;

– *Formatiranje*(engl. *formating*) – postojanje ovog koraka je potpuno opcionalno, zadaća koraka je formatirati tekst u zadanom formatu (npr. odlomak teksta u *html*-u ispravno označiti s `<p>` i `<\p>` oznakama).

U Reiter i Dale (2000) na temelju Reiter (1994) i Reiter (1996) predstavljena je osnova arhitektura sustava NLG koja se smatra *de facto* standardom. Sustav se mora sastojati od sljedeća tri dijela: planera dokumenta (engl. *document planner*), mikro planera (engl. *microplanner*) i površinskog realizatora (engl. *surface realizer*).

Zadaća planera dokumenta jest odrediti sadržaj i strukturu teksta koristeći znanje o domeni i primjeni. Osim toga, mora se imati znanje o tome kako se pišu tekstovi u danoj domeni. Planer dokumenta najčešće ne određuje cjelokupnu strukturu dokumenta već određivanje pojedinih dijelova ostavlja mikro-planeru.

Mikro-planer ima zadaću urediti tekst, tj. planirati rečenice. Npr. planer dokumenta može označiti da u dokumentu ulaze informacije o broju (najčešćem, standardnom) žica gitare i bas gitare. Mikro-planer može odlučiti da tekst ima oblik:“Gitara ima 6 žica. Bas gitara ima 4 žice” ili “Gitara ima 6, dok bas gitara ima 4 žice.”. Osim agregacijskih odluka, mikro-planer određuje i referentne izraze, npr. ako imamo dvije rečenice:“Gitare imaju 6 žica. Također postoje i gitare s 7 ili 8 žica”, tada mikro-planer može odlučiti da umjesto riječi “*gitare*” drugi put koristi zamjenicu i tada tekst ima oblik: “Gitare imaju 6 žica. Također postoje i one s 7 ili 8 žica.” Izlaz iz mikro-planera je i dalje u apstraktnom obliku, dok se sama realizacija rečenica i odabir riječi vrši u površinskom realizatoru.

Sustav mora izvršiti sljedeće zadatke:

1. Određivanje sadržaja (engl. *content determination*), koje izvršava planer dokumenta;

2. strukturiranje dokumenta (engl. *document structuring*), koje također izvršava planer dokumenta;

3. leksikalizacija (engl. *lexicalization*), koju izvršava mikro-planer;

4. generiranje referetnih izraza (engl. *refering expression generation*), koje također izvršava mikro-planer;

5. agregaciju (engl. *aggregation*) također izvršava mikro planer;

6. lingvistička realizacija (engl. *linguistic realization*), koju izvršava površinski realizator;

7. i strukturna realizacija (engl. *structural realization*), koju izvršava površinski realizator.

2.3.1. Određivanje sadržaja

Određivanje sadržaja predstavlja problem odlučivanja koje je podatke potrebno iskomunicirati u tekstu (Reiter i Dale, 2000). Koji put će korisnici sustava specificirati koji podatak žele, npr. jedna rečenica o kemijskoj olovci kojom je pisan ovaj rad ("Kemijska olovka je crne boje.", može biti jedna od ponuđenih generiranih rečenica). Češće sam sustav mora zaključiti koje su informacije bitne i koje treba generirati. Naravno, odabir sadržaja ovisi o više čimbenika.

Ovisno o cilju komunikacije, razlike mogu biti drastične poput generiranja sažetka radnje epizode neke serije ili generiranja biografija likova u seriji, ili suptilnije poput sažetka radnje jedne epizode (više detalja) naspram sažetka radnje čitave sezone (manje detaljno).

Na sadržaj utječe to kome je tekst namjenjen, nije isto pisati osobi koja je stručnjak u domeni u kojoj pripada napisani tekst i osobi koja je laik za dotičnu domenu.

Na sadržaj utječu i razna ograničenja npr. duljina teksta može biti ograničena na neki broj riječi.

Izvor podataka igra ulogu u određivanju sadržaja jer je potrebno znati koje podatke treba iskoristiti.

2.3.2. Strukturiranje teksta

Da bi tekst bio razumljiv, ne može se nabaciti samo hrpa rečenica s informacijama, već mora postojati određeni redosljed pružanja informacija. Također postoje i pravila strukturiranja vezana uz žanr (poezija i znanstveni članci ne mogu biti identično strukturirani).

2.3.3. Leksikalizacija

Leksikalizacija je problem odabira riječi (imenica, glagola, priloga itd.). Osim odabira riječi, u leksikalizaciju spada problem korištenja sintaktičnih pravila, npr: "Veljkova trzalica" i "Trzalica kojoj je vlasnik Veljko".

2.3.4. Generiranje referentnih izraza

Referentni izrazi su oni izrazi koji čitatelju uvijek omogućuju da razumije o kojem se entitetu radi. To predstavlja problem za NLG jer postoji više načina za referiranje na neki entitet. No problem gledišta s kojeg novi entitet ulazi u tekst je (najčešće) izbjegnut jer sustavi NLG (najčešće) imaju vrlo ograničenu primjenu. Problem početnog referiranja je time izbjegnut, no problem svakog idućeg referiranja nije, jer je potrebno moći razlučiti na koji se entitet koji referentni izraz odnosi, npr. u rečenicama "Batman i Superman su prijatelji. On može letjeti", nije moguće odrediti na koga se zamjenica *on* odnosi i umjesto zamjenice sustav bi trebao koristiti imenicu *Superman*.

2.3.5. Agregacija

Agregacija je postupak u kojem se odlučuje u kojim će rečenicama, odlomcima ili poglavljima informacije biti izražene. Recimo da je potrebno predstaviti informacije o vrsti gitare (električna, akustična) i proizvođaču. Agregacijom se vrši odabir hoće li informacije biti izložene u npr. jednoj ili dvije rečenice. Agregacija je u bliskoj vezi s leksikalizacijom.

2.3.6. Lingvistička realizacija

Lingvistička realizacija je postupak stvaranja rečenice poštujući morfološka i sintaktička pravila. Lingvistička realizacija pokušava razriješiti problem primjene ovih pravila na neku apstraktnu strukturu, npr. neka imamo strukturu `[Superman, moći, letenje]` tada se lingvističkom realizacijom generira rečenica "Superman može letjeti".

2.3.7. Strukturna realizacija

Strukturna realizacija sastoji se od primjene pravila priređenih u prijašnjim koracima vezanih uz strukturu teksta. Primjena ovisi o tome gdje (tj. kako) će tekst biti izložen. Npr., ako se tekst iskazuje pomoću html-a tada će oznaka za novi odlomak biti `<p>`, a za kraj odlomka `</p>`, dok će u slučaju LaTeX-a dvostruki znak nove linije (`\n`) označavati kraj prethodnog i početak novog odlomka.

2.3.8. Kritika Reiterive i Daleove arhitekture

Gore predstavljena arhitektura prema nekim istraživanjima sadrži bitne greške. Tako Evans et al. (2010) kao najveću grešku ističu nejasno definiran ulaz u sustav. Ulaz je definiran kao četvorka $< k, c, u, d >$, gdje je k izvor znanja (engl. *knowledge source*), c je cilj komunikacije (engl. *communication goal*), u je korisnički model (engl. *user model*), a d je povijest diskursa (engl. *discourse history*). Problem definicije jest izvor znanja koji je domenski ograničen, zbog čega je nemoguće definirati općenit model ulaza u sustav NLG, budući da je svaki sustav specifičan po onom čime se bavi.

3. Kontekstno neovisna gramatika

Kontekstno neovisne gramatike, kada se primjenjuju u lingvistici, mogu se koristiti za opisivanje strukture rečenica i riječi nekog prirodnog jezika. Upravo zbog toga može ih se koristiti u generiranju prirodnog jezika.

Kontekstno neovisna gramatika (engl. *context-free grammar, CFG*) je četvorka (V, Σ, R, S) gdje je (Sipser, 2006):

1. V je konačni skup *varijabli* (engl. *variables*),
2. Σ je konačni skup *završnih znakova* (engl. *terminals*), pri čemu $\Sigma \cap V = \emptyset$
3. R je konačan skup *pravila* (engl. *rules*), gdje je svako pravilo dvojka sačinjena od varijable te niza varijabli i završnih znakova i
4. $S \in V$ je *početna varijabla*.

Primjer kontekstno neovisne gramatike je sljedeći:

Neka je $V = S, A, B$ konačni skup varijabli i neka je S ujedno i početna varijabla. Također, neka je $\Sigma = x, y, z$ skup završnih znakova i neka imamo pravila $R = (S \to xAB), (A \to yB), (B \to z)$. Tada je dano kontekstno neovisna gramatika, koju se lakše može prikazati tako da se prikažu samo njena pravila uz dogovor da velika slova abecede predstavljaju varijable, slovo S uvije poredstavlja početnu varijablu, a mala slova abecede predstavljaju završne znakove.

$$\begin{aligned} S &\to xAB \\ A &\to yB \\ B &\to z \end{aligned} \tag{3.1}$$

3.1. Vjerojatnosna kontekstno neovisna gramatika

Poseban oblik kontekstno neovisne gramatike jest vjerojatnosna konteksno neovisna gramatika (engl. *probabilistic context-free grammar, PCFG*).

Vjerojatnosna konteksno neovisna gramatika je četvorka (W, N, N_s, R), gdje je W skup završnih znakova, N je skup nezavršnih znakova, $N_s \in N$ je početni znak i $R = (r_1, p(r_1)) \dots (r_m, p(r_m))$ je skup produkcijskih pravila s pridruženim vjerojatnostima. Svako pravilo r_i je oblika $n \to \alpha$, gdje je $n \in N$ i α je niz završnih i nezavršnih znakova. Za svaki nezavršnih znak n vrijednost zbroja svih $p(n \to \alpha_i)$ je jednak 1, $\sum_{i:(n \to \alpha_i, p(n \to \alpha_i)) \in R} p(n \to \alpha_i) = 1$

U nastavku će biti opisane dvije normalne forme, Chomskyeva i Greibachina, te način generiranja niza iz gramatike.

3.2. Chomskyeva normalna forma

Prilikom rada s konteksno neovinsnim gramatikama praktičnije ih je imati u nekoj pojednostavljenoj formi. Jedna takva forma je *Chomskyeva normalna forma* (engl. *Chomsky normal form, CNF*).

Kontekstno neovisna gramatika je u Chomskyevoj normalnoj formi ako i samo ako je svako pravilo jednako jednom od dva oblika (Sipser, 2006):

$$\begin{aligned} A &\to BC \\ A &\to a \end{aligned} \tag{3.2}$$

gdje je a bilo koji završni znak, a A, B, C su bilo koje varijable, osim što B i C ne smiju biti početna varijabla. Dodatno se dopušta pravilo $S \to \epsilon$, gdje je S početna varijabla, a ϵ prazan niz.

3.3. Greibachina normalna forma

Druga forma u kojoj se često pojavljuje CFG je *Greibachina normalna forma* englGreibach normal form, GNF (Srbljić, 2004):

Neka je gramatika G kontekstno neovisna. Moguće je izgraditi istovjetnu gramatiku G' koja ima sve produkcije oblika $A \to a\alpha$ gdje je a završni znak gramatike, a α niz nezavršnih znakova gramatike koji može biti i prazan.

No, za daljna razmatranja bitniji je CNF.

3.4. Generiranje niza

Neka imamo sljedeću kontekstno neovisnu gramatiku:

$$\begin{aligned} S &\to XY \\ X &\to 1X \mid 1 \\ Y &\to 0Y \mid 0 \end{aligned} \tag{3.3}$$

Vrijedi napomenuti da se znak | koristi kako bi se skratilo pisanje, npr. iz varijable X postoje dvije produkcije, jedna u $1X$, a druga u 1.

Želimo dobiti niz 111000. Iz dane gramatike to je moguće na više različitih načina, dakle moguće je izgraditi više *generativnih stabala* (koja se grade tijekom generiranja niza), čime se stvara nejednoznačnost. Da bi se ona izbjegla, primjenjuju se sljedeća dva načina generiranja niza: *generiranje niza zamjenom krajnjeg lijevog nezavršnog znaka* i *generiranje niza zamjenom krajnjeg desnog završnog znaka* (Srbljić, 2004).

Primjenom prvog načina niz 111000 se dobiva ovako:

$$S \to XY \to 1XY \to 11XY \to 111Y \to 1110Y \to 11100Y \to 111000 \tag{3.4}$$

Primjenom drugog načina:

$$S \to XY \to X0Y \to X00Y \to X000 \to 1X000 \to 11X000 \to 111000 \tag{3.5}$$

3.5. Indukcija kontekstno neovisne gramatike iz korpusa

Kontekstno neovisnu gramatiku moguće je izgraditi na više načina, no ovdje će se proučiti njena indukcija iz korpusa.

Prvi način indukcije je predstavljen kao alternativa onome što je korišteno u radu. Drugi način je onaj koji je korišten u samom radu.

3.5.1. Indukcija pomoću označivača vrste riječi

Prije početka indukcije tekstovi u korpusu su označeni označivačem vrste riječi.

Označivač vrste riječi (POS-označivač) (engl. *part-of-speech tagger, POS tagger*) je računalni program koji riječima u rečenici dodaje oznaku koje je vrste riječ (engl. *part-of-speech*), poput imenice, glagola itd. Tako je moguće gramatiku izgraditi nad sintaksom, a ne nad sintagmama.

Prije same indukcije potrebno je obaviti sljedeće radnje nad tekstovima (Chen i Tseng, 2006):

1. Tekstovi se razlome na pojedinačne rečenice, koristeći znakove točke, upitnika i uskličnika;
2. Zagrade se uklanjanju;
3. Riječi u rečenici označe se POS-označivačem

Potom, nakon obavljenih pripremnih radnji postupa se na sljedeći način za svaku rečenicu:

1. Izračuna se broj pojavljivanja za svaki bigram (gdje se bigram izgrađuje nad pos oznakama, a ne nad samim riječima)
2. napravi se pravilo s bigramom s najvećim brojem pojavljivanja na desnoj strani i proizvoljnim nezavršnim znakom na lijevoj strani, a koji se do sad nije pojavio u skupu nezavršnih znakova.
3. osvježi se skup znakova i pravila dobivenim pravilima i znakovima

Npr. neka se tekst sastoji samo od jedne rečenice “Sky is blue, sun is yellow”. Ta rečenica Nakon pripremnih radnji izgleda ovako:

```
Sky/NN is/VBZ blue/JJ sun/NN is/VBZ yellow/JJ
```

Iz rečenice je vidljivo da se nekoliko bigrama pojavljuje isti, maksimalni, broj puta pa se stoga uzima prvi po redu (`NN VBZ`). Tada se gradi pravilo `P0`→ `NN VBZ`. Osim tog pravila dodaju se pravila `NN`→`Sky`, `NN`→`sun` itd. U skup varijabli se dodaju `{P0,NN,VBZ}`. Potom rečenica poprima oblik:

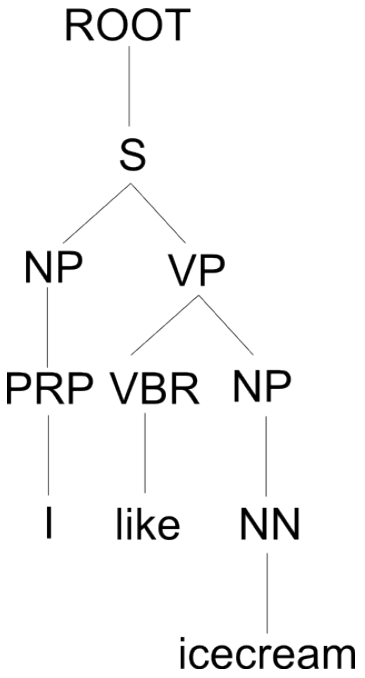
`P0 blue/JJ P0 yellow/JJ`

Iz toga se dobiva novo pravilo `P1`→`P0 JJ` i još dodatno pravila iz `JJ` u `blue` i `yellow`, a u skup varijabli se dodaje `{P1,JJ}`. Konačno dodaje se pravilo `S`→`P1 P1`, a `S` se osim toga označava kao početni znak.

3.5.2. Indukcija pomoću rečeničnog parsera

Drugi način indukcije temelji se na rečenicama parsiranim rečeničnim parserom (engl. *sentence parser*).

Izlaz iz rečeničnog parsera je banka stabala (engl. *treebank*) gdje pojedinačno stablo izgleda kao ono prikazano na slici 3.1



Slika 3.1: Primjer parsirane rečenice

Najlakši način indukcije bi bio preuzeti pravila direktno iz stabla npr:

$$S \rightarrow NPVPNP \rightarrow PRP|NN \tag{3.6}$$

Gramatike nastala na takav način, tzv. gramatike iz banke stabala (engl. *treebank grammars*), dugo su smatrane lošim no u (Charniak, 1996) pokazao je drugačije.

Mora se napomenuti da su gramatike nastale iz banke stabala poprilično velike, pa se i preporuča neki način odsijecanja broja pravila, npr. odsijecanje svih pravila koja se pojavljuju samo jednom (samo u jednom stablu).

3.5.3. Indukcija vjerojatnosne kontekstno neovisne gramatike

Indukcija vjerojatosne konteksno neovisne gramatike razlikuje se utoliko što računa vjerojatnost svakog pravila prema formuli (Charniak, 1996):

$$p(r) = \frac{|r|}{\sum_{r' \in \{r' | \alpha(r') = \alpha(r)\}} |r'|} \tag{3.7}$$

gdje je r pravilo, $|r|$ broj pojavljivanja pravila r u korpusu i $\alpha(r)$ varijabla koju pravilo r proširuje (koja se nalazi na lijevoj strani pravila). Takav način procjene vjerojatnosti naziva se procjena maksimalne izglednosti (engl. *maximum likelihood estimation, MLE*).

4. Tehnike vrednovanja

Jedan od većih problema generiranja prirodnog jezika jest problem vrednovanja sustava: kako provjeriti kvalitetu generiranih tekstova? Jedan od najranijih pristupa bilo je ručno ocjenjivanje od strane većeg broja ljudi, laika za dano područje ili manjeg broja stručnjaka na danom području. Osim problema logističke prirode i velikog vremenskog troška, u slučaju laika postoji i problem koliko ta osoba može kvalitetno ocijeniti tekst.

Zbog toga su brojni istraživači potražili rješenje u tehnikama evaluacije iz donekle srodnog područja strojnog prevođenja (engl. *machine translation, MT*), no to je, u većini slučajeva, obavljeno bez evaluacije kvalitete tih tehnika za NLG. Međutim u radu (Belz i Reiter, 2006) obavljena je evaluacija nekoliko tehnika vrednovanja.

U radu je uspoređeno ocjenjivanje stručnjaka, ne-stručnjaka i evaluacijskih tehnika NIST (Doddington, 2002), BLEU (Papineni et al., 2002) i ROUGE (Lin i Hovy, 2003), dok je kao osnovna mjera uzeta Levenshteinova udaljenost. Uz pretpostavku da će stručnjak najbolje ocjenjivati generirane tekstove, Pearsonovom korelacijom izračunato je da ocjena koja je najbliža ocijeni stručnjaka ocjena nestručnjaka, potom NIST-5 (NIST mjera koja koristi 5-grame), BLEU-4, ROUGE-4 i konačno Levenshtienova udaljenost.

U nastavku će biti pobliže opisana mjera NIST i mjera BLEU na kojoj se temelji mjera NIST.

4.1. Vrednovanje BLEU

BLEU (engl. *bilingual evaluation understudy*) (Papineni et al., 2002) je, kao i ostale spomenute metode, originalno rađen za strojno prevođenje, tj. za evaluaciju kvalitete prijevoda.

Da bi BLEU i slične metrike radile autori su odredili dva potrebna elementa: numeričku mjeru sličnosti prijevoda i korpus kvalitetnih ljudskih prijevoda.

Osnova BLEU je modificirana preciznost unigrama (engl. *modified unigram precision*). Da bi se ona izračunala prvo brojimo broj pojavljivanja svake riječi kandidata (računalno prevedene, tj. generirane) u bilo kojem referentnom (tj. ljudski generiranom) tekstu. Potom brojimo broj pojavljivanja svake riječi u kandidatu i odsijecamo je na broju pojavljivanja u referentnom tekstu. Preciznost dobivamo dijeljenjem odsječenog broja s ukupnim brojem pojavljivanja kandidata. Npr., ako imamo referentnu rečenicu: *"Ja volim plavo i volim zeleno."* i rečenicu kandidat: *"Ja volim volim volim volim volim"*, tada je modificirana unigramska preciznost riječi *volim* jednaka 2/5. Modificirana unigramska preciznost, izražena formulom, je:

$$p_n = \frac{min(broj\_pojavljivanja, maksimalan\_broj\_ref\_pojavljivanja)}{broj\_pojavljivanja} \quad (4.1)$$

Slično se računa i za ostale n-grame.

Naravno, najkvalitetnija metrika dobiva se kombiniranjem svih modificiranih n-gramskih preciznosti (do nekog razumnog broja n). No modificirana n-gramska preciznost gubi na kvaliteti otprilike eksponencijalno porastom veličine n-grama. Da bi se izrazio dobar prosjek, ovaj ekspoenecijalni gubitak mora se odraziti na način računanja. Stoga je prosjek izražen kao logaritamski prosjek s uniformnim težinama. Ovako izražen prosjek zapravo je jednak izražavanju prosjeka kao geometrijske sredine. Pokazano je da sustav najbolje radi uz maksimalnu duljinu n-grama postavljenu na vrijednost 4.

Osim modificirane n-gramske preciznosti, BLEU uzima u obzir i duljinu teksta kandidata. Kandidati duži od referentnih tekstova već su kažnjeni modificiranom n-gramskom preciznošću, stoga se uvodi kažnjavanje kraćih kandidata. S ovako definiranom mjerom BLEU bolje rangira kandidate koji su iste dužine, imaju isti odabir riječi i njihov redoslijed kao referentni tekstovi.

Konačna formula za BLEU je stoga:

$$BLEU = BP \times *exp(\sum_{n=1}^{N} logp_n) \quad (4.2)$$

gdje je BP kazna za dužinu (engl. *brevity penalty*) i jednaka je:

$$BP = \begin{cases} 1 & akoje\text{k>r} \\ e^{1-r/k} \text{ ako je } cr \end{cases} \quad (4.3)$$

Konačno, BLEU izražen u logaritamskoj formi daje bolji uvid u rangiranje:

$$logBLEU = min\left(1 - \frac{r}{c}, 0\right) + \sum_{n=1}^{N} w_n logp_n \tag{4.4}$$

4.2. Vrednovanje NIST

Vrednovanje NIST (engl. *National Institute of Standards and Technology*) (Doddington, 2002) je zasnovano na BLEU vrednovanju.

Analizom vrednovanja BLEU utvrđeno je da geometrijska sredina nije najbolje riješenje, jer je rezultat jednako osjetljiv proporcionalnim razlikama u pojavljivanju n-grama za sve n. Zbog toga može doći do nepoželjne varijance zbog nižeg broja pojavljivanja velikih n-grama. Kao alternativa je predložena aritmetička sredina pojavljivanja n-grama. Drugi problem koji je primjećen jest da je bolje više značenje (veći iznos težinskog faktora) pridodati n-gramima koji nose veću informaciju, tj. n-gramima s manjim brojem pojavljivanja.

Uzevši u obzir ove spoznaje, vrednovanje NIST definirano je kao:

$$NIST = \sum_{n=1}^{N} \frac{\sum_{\text{svi } w_1 \ldots w_n \text{ koji se pojavljuju}} log_2\left(\frac{N(w_1 \ldots w_{n-1})}{N(w_2 \ldots W_n)}\right)}{\sum_{\text{svi } w_1 \ldots w_n \text{ u izlazu sustava}}(1)} exp\left(\beta log^2\, min\left(\frac{L_{sys}}{L_{ref}}, 1\right)\right.$$

uz $N = 5$, $\beta = 0.5$ ako je broj riječi u izlazu sustava 2/3 prosječnog broja riječi u referentnom prijevodu po prosjeku na svim referentnim prijevodima i L_{sys} je broj riječi u prijevodu koji se ocjenjuje.

5. Srodni radovi

Kada je riječ generatorima prirodnog jezika ne postoji konsenzus što je dobar pristup izgradnji sustava i što je dobar način generiranja pa stoga postoji i mnoštvo radova s različitim tehnikama. Prvi rad koji će biti predstavljen je Binsted i Ritchie (1994), tj. generator viceva JAPE-1, zasnovan na predlošcima. Drugi rad je Callaway i c. Lester (2002), tj. generator priča o Crvenkapici StoryBook, zasnovan na automatima konačnih stanja (engl. *finite state machine, FSM*). Treći rad je Belz (2008), tj. općenit generator pCRU s pristupom zasnovanim na kontekstno neovisnim gramatikama, točnije kontekstno neovisnim gramatikama s nekoliko postroženja i preinaka.

5.1. JAPE-1

Generator viceva JAPE-1 (Binsted i Ritchie, 1994) generira viceve u obliku kratkih pitanja i odgovora, npr. “What do you call a murderer that has fibre? A cereal killer” (generirani primjer).

Vicevi ovog oblika mogu nastati na tri različita načina (postoje tri podtipa):

- *Zamjena slogova* – gdje se mijenjaju samoglasnici sa sličnozvučećim ili istozvučećim riječima, npr. “What do short-sighted ghosts wear? Spooktacles.”;
- *zamjena riječi* – gdje se riječi mijenjaju istozvučećim ili sličnozvučećim riječima, slično zamjeni slogova (“How do you make gold soup? Put fourteen carrots in it.”);
- i konačno *metatezom*, gdje se zamjenom zvukova i riječi nastoji sugerirati sličnost između dvije semantički različite fraze (“What’s the difference between a very short witch and a deer running from hunters? One’s a stunted hag and the other’s a hunted stag.”).

JAPE-1 sustav radi samo viceve zamjenom riječi uz ograničenja da se zamjena vrši samo u poanti vica (ne u pitanju) i mijenja se nekom čestom imeničkom frazom.

U nastavku će biti detaljnije opisana arhitektura sustava.

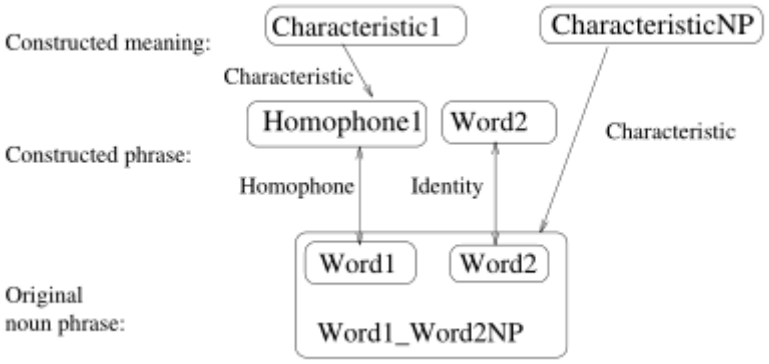
5.1.1. Arhitektura sustava JAPE-1

Sustav JAPE-1 se sastoji od tri dijela:leksikona, sheme i predloška.

Leksikon sadrži semantičke i sintaktičke informacije o riječima i imeničkim frazama koje su neovisne o humoru. Sintaktični podaci zapravo se minimalno koriste u ostatku sustava. Nadalje, leksikon ima upisane samo imenice, glagole, pridjeve i česte imeničke fraze. Sve ostale vrste riječi (prilozi, prijedlozi, veznici itd.) nalaze se u predlošcima. Kako je JAPE-1 ograničen samo na poante s imeničkim frazama, semantičke informacije postoje samo za imenice i pridjeve.

Homonimi su implementirani na odvojenoj listi riječi.

JAPE-1 ima šest *shema*. Jedna od njih prikazana je na slici 5.1 preuzetoj iz (Binsted i Ritchie, 1994). Predlošci se koriste jer ovaj tip vica često koristi slične



Slika 5.1: Primjer sheme sustava JAPE-1

forme. Primjer predloška dan je na slici 5.2 preuzetoj iz (Binsted i Ritchie, 1994).

Nakon generiranja vica još se vrše sitne provjere je li npr. korištena ista riječ za prvi i drugi fragment i sl.

What do you get when you cross [**text fragment generated from the first characteristic lexeme(s)**] with [**text fragment generated from the second characteristic lexeme(s)**]? *[the constructed noun phrase].*

Slika 5.2: Primjer predloška sustava JAPE-1

Evaluacija sustava izvršena je pomalo neuobičajeno. Započela je već kod prikupljanja leksikona, koji su radili dobrovoljci ne znajući čemu je leksikon namijenjen (da bi se izbjeglo "uključivanje" humora u leksikon). Potom je leksikon pregledan sa svrhom izbacivanja rijetkih riječi.

Sustav je generirao 188 viceva, koji su podijeljeni četrnaestorici sudaca. Vicevi su ocijenjeni ocjenama od 0 do 5, gdje nula označava da vic uopće nema smisla, a pet da je vic stvarno smiješan. Prosječna ocjena vica je bila 1.5 što je naznačavalo da vicevi nisu pretjerano smiješni. Svega je 9 viceva dobilo maksimalnu ocjenu, ali uvijek samo u jednog sudca.

5.2. StoryBook

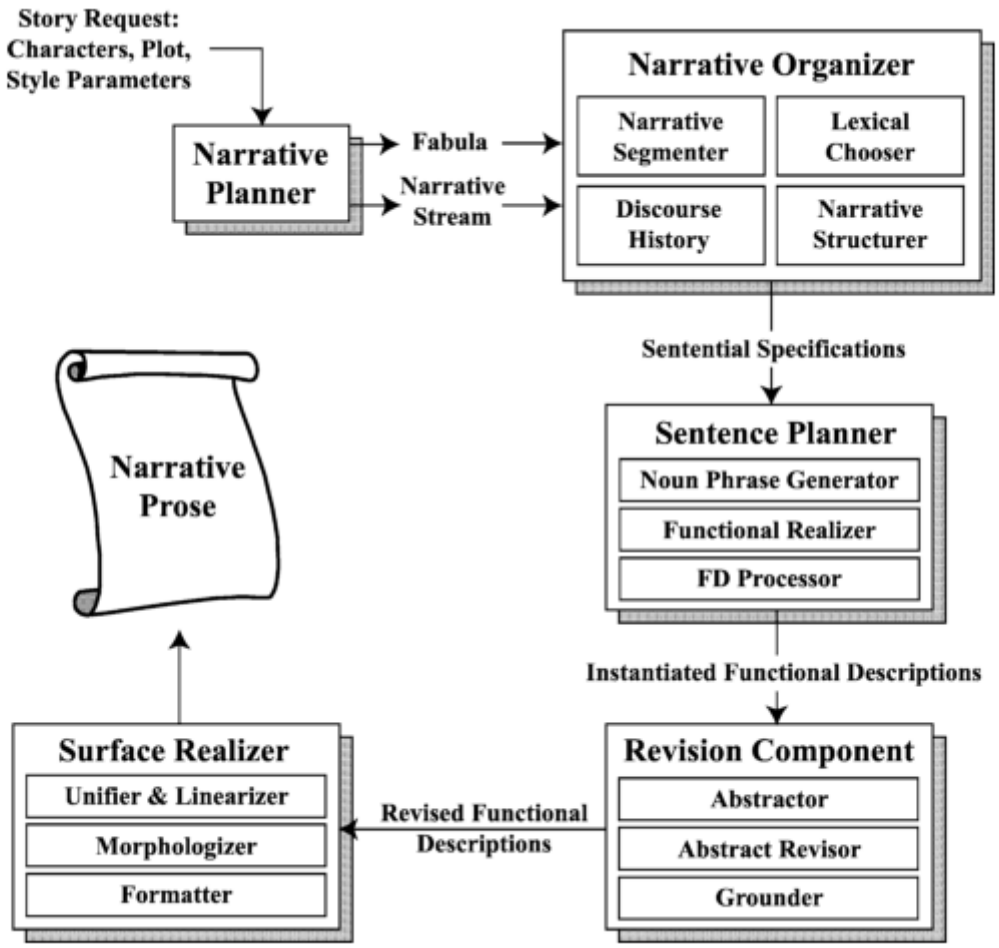
Generator narativne proze StoryBook (Callaway i c. Lester, 2002) generira priče, tj. narativnu prozu. Sustav je ograničen na priče o Crvenkapici, no arhitektura sustava je općenita za bilo koju narativnu prozu.

U nastavku je dan detaljni opis arhitekure sustava.

5.2.1. Arhitektura sustava StoryBook

Arhitektura sustava StoryBook prikazana je na slici 5.3 preuzetoj iz (Callaway i c. Lester, 2002).

Arhitektura se sastoji od tri dijela: narativnog planera (engl. *narrative planner*), planera rečenica (engl. *sentence planner*) i površinskog realizatora (engl. *surface realizer*). No osim tri osnovna dijela koriste se još i moduli povijesti diskursa (engl. *discourse history*), modul leksičkog odabira (engl. *lexical choice*), modul revizije (engl. *revision*) i modul narativnog formatiranja (engl. *narrative formatter*).



Slika 5.3: Arhitektura sustava StoryBook

Modul povijesti diskursa prati koji su entiteti već uvedeni u priču i nudi ostatku sustava njihovo imenovanje, način referiranja itd. *Modul leksičkog odabira* brine da narativna proza nije ponavljajuća, tj. nudi alternativnu leksalizaciju. *Modul revizije* provodi premještanje rečenica, agregaciju podataka i sl. tj. pokušava učiniti prozu sličnijom ljudskoj (složenijom). *Modul narativnog formatiranja* uveden je zbog specifičnosti narativne proze poput dijaloga, točnije provodi formatiranje teksta u ispravne formate.

U nastavku će biti detaljnije opisana tri glavna dijela arhitekture.

Narativni planer

Narativni planer ima četiri glavna zadatka:

1. *Oblikovanje likova* – planer mora odlučiti koliko likova ima i koju ulogu igraju u priči (glavni junak, negativac itd.);

2. *izgradnja radnje* – planer mora izgraditi osnovnu radnju i utvrditi ciljeve, likove, rekvizite i lokacije koje omogućuju likovima da ostvare svoje ciljeve, kao i redoslijed događaja koji vode od početka priče prema vrhuncu i kraju;

3. *izgradnja scena* – planer mora odrediti redoslijed događaja u radnji u neku razumljivu strukturu i odrediti prisutnost i odsutnost likova i rekvizita kako radnja napreduje;

4. *generiranje fabule i narativnog toka* – planer mora ugraditi likove i rekvizite u fabulu i stvoriti narativni tok koji reflektira redoslijed događaja, opisa i stanja.

Narativni planer sustava StoryBook sastoji se od automata s konačnim stanjima u kojima putanje do završnog stanja predstavljaju priču. Uzduž puta svaki čvor doprinosi *fabularnim operatorima* i *primitivama narativnog toka* rastućem narativnom planu. Ovakva procedura eksplicitno izvršava samo generiranje fabule i narativnog toka, dok implicitno izvršava preostala tri zadatka.

Fabularni operatori su `AddScene` (gradi hijerarhijsku strukturu scena), `AddActor` (stvara novi imenovani lik), `AddProp` (stvara koncept koji predstavlja fizički ili mentalni objekt kojim likovi mogu manipulirati itd.), `AddLocation` (stvara koncept kao lokaciju) i `AddAlias` (povezivanje koncepata jednog s drugim).

Primitive narativnog toka mogu se podjeliti u tri veće skupine:

- *primitive ograničenja* (engl. *delimiting primitives*) koje stvaraju narativni kontekst, određuju scene, likove, pripovjedače i sl;
- *primitive osnove* (engl. *base primitives*) koje predstavljaju golu s trukturu sadržaja koji se koristi za stvaranje rečenica u opisima i dijalozima;
- *modificirajuće primitive* koje predstavljaju informacije koje mijenjaju sadržaj ili dodaju detalje primitivama osnova.

Slika 5.4 prikazuje primjer fabularnih operatora, a na slici 5.5 dan je primjer narativnih primitiva. Obje su slike preuzete iz (Callaway i c. Lester, 2002).

```
(NewNarrative MyLRRH-Narrative001 Narrator001)
(AddActor Woodman001 Woodman Person Male)
(AddActor Wife001 Wife Person Female)
(AddLocation Cottage001 Cottage)
(AddLocation Forest001 Forest)
(AddActor Little-Red-Riding-Hood001 Little-Red-Riding-Hood
          Person Female "Little Red Riding Hood")
(AddAlias Daughter001 Daughter Little-Red-Riding-Hood001)
(AddProp Cloak001 Cloak)
(AddProp Hood001 Hood)
```

Slika 5.4: Primjer fabularnih operatora sustava Storybook

Narativni organizator

Narativni organizator mora podijeliti semantičku kontinuiranu strukturu iz narativnog planera na niz djelova veličine jedne rečenice. To postiže u četiri koraka:

1. Narativna segmentacija koja razdvaja kontinuiranu strukturu narativnog planera na jednu ili više narativnih primitiva koje tvore rečenice ili odlomke;
2. modul povijesti diskursa, koji mora paziti da se određene riječi zamjene zamjenicama i sl.;
3. modul leksičkog odabira zamjenjuje riječi sinonimima ili skupovima riječi istog značenja (opisima);
4. i modul narativnog strukturiranja, koji uzima skupove narativnih primitiva iz prvog koraka i iz njih započinje izgradnju rečenica.

Planiranje rečenica i revizija

Modul planera rečenica uzima niz stvoren u narativnom organizatoru i potom iz njega tvori funkcijske opisnike (hibridne semantičko-sintatičke entitete) koji se mogu koristiti izravno za stvaranje teksta.

Funkcijski opisnici prosljeđuju se modulu za reviziju, koji iz njih tvori apstraktan narativni plan.

Apstraktan narativni plan sastoji se od osnovnih leksičkih, semantičkih i sintaksnih jedinki koje tvore fabulu. Modul za reviziju potom prolazi kroz iteracije i

```
((narration-mode historical mixed complex ascii narrated english)
 (narrator-mode narrator001 third-person disembodied)
 (new-scene scene001)
 (new-actor woodman001)
 (new-actor mother001)
 (new-actor little-red-riding-hood001)
 ;;; "Once upon a time there was a woodman and his wife."
 (actor-property exist-being woodman001)
 (refinement and-along-with woodman001 wife001)
 (refinement belonging-to wife001 woodman001)
 (specification exist-being process-step-type once-upon-a-time)
 ;;; "The woodman and his wife lived in a pretty cottage."
 (prop-relationship living-in woodman001 cottage001)
 (refinement and-along-with woodman001 wife001)
 (refinement belonging-to wife001 woodman001)
 (detail cottage001 pretty-appearance)
 ;;; "The cottage was on the borders of a great forest."
 (prop-relationship location-on cottage001 border)
 (refinement container-of border forest001)
 (refinement discourse-reference border multiple-quantity-reference)
 (refinement discourse-reference border principle-subregion-reference)
 (detail forest001 great-sized)
 ;;; "The woodman and his wife had one little daughter."
 (actor-relationship having woodman001 daughter001)
 (refinement and-along-with woodman001 wife001)
 (refinement belonging-to wife001 woodman001)
 (refinement quantifier-value daughter001 unique-one)
 (detail daughter001 little-sized)
 ;;; "The girl was a sweet child."
 (actor-relationship identity girl001 child001)
 (detail child001 sweet-natured)  ...
```

Slika 5.5: Primjer narativnih primitiva sustava StroyBook

mijenja apstraktnan narativni plan koristeći operatore revizije (poput spajanja rečenica i sl.).

Površinski realizator

Površinski realizator iz dobivenih podataka od modula planera rečenica može generirati rečenice. Površinski realizator dodaje samo one riječi koje su navedene u apstraktnom planu, pazi na gramatičku ispravnost rečenica, prilagođava riječi da budu u skladu s rodom, brojem i padežom, uvodi točke, zareze i ostale simbole i pazi na formatiranje teksta.

Primjer teksta koji je nastao sustavom StoryBook prikazan je na slici 5.6 preuzetoj

iz (Callaway i c. Lester, 2002).

[**P03**] Her grandmother was a very old lady, who lived in the heart of a neighboring wood. She was delighted at being sent on this errand, for she liked to do kind things. It was a very long time since she had seen her grandmother, and so she had almost forgotten what the old lady looked like.

[**P04**] The sun was shining brightly, but it was not too warm under the shade of the old trees. Little Red Riding Hood went on her way singing and gathering great bunches of wild flowers. She wanted to give them to her grandmother. She sang so sweetly that a cushat dove flew down from a tree and followed her.

Slika 5.6: Primjer teksta nastao sustavom StoryBook

Vrednovanje sustava StoryBook

Vrednovanje sustava prevodeno je tako da je skupina ljudi davala ocjene od 0 do 4 (A, B, C, D i F). Prije toga je sustav generirao dvije priče u deset varijacija (ovisno o uključivanju i isključivanju pojedinih modula). Ocjene su davane za kategorije stila, gramatike, točnost teksta, dikcije, čitkosti, logičnosti, detaljnosti, vjerodostojnosti i za sveukupni dojam. Najbolje ocijene su priče dobile u varijaciji s svim uključenim modulima, prva 3 a druga 3.5 za sveukupni dojam.

5.3. pCRU

Sustav pCRU (Belz, 2008) je općenit generator koji se može prilagoditi za generiranje različitih vrsta tekstova.

U nastavku će biti izložena teorijska osnova sustava pCRU i njegova arhitektura.

5.3.1. Kontekstno neovisna reprezentacijska podspecifikacija

Kontekstno neovisna reprezentacijska podspecifikacija (engl. *context-free representational underspecification, CRU*) (Belz, 2004) predstavlja srž sustava pCRU.

CRU je temeljen na kontekstno neovisnim gramatikama i predstavlja njihovo postroženje. Svrha CRU, u smislu sustava pCRU, jest automatizirati dubinsku generaciju i donošenje odluka na višoj, konceptualnoj razini. CRU se definira kao klasična CFG četvorka (V, Σ, R, S) uz dodatnan zahtjev da ne postoje prijelazi u prazan znak, $A \rightarrow \epsilon$. Ono zbog čega je CRU bitan jest definicija podspecificiranog jezika (engl. *underspecified language*) za gramatiku CRU. Podspecificirani jezik se može prikazati na sljedeći način. Neka postoji rečenica " Ovo je tekst primjera". Za takvu rečenicu se može kazati da je potpuno specificirana. Ako bi se sve riječi u rečenici zamjenile pos oznakama, tada bi to bio podspecificirani oblik te rečenice jer je iz njega moguće dobiti više različitih rečenica. Formalna definicija podspecificiranog jezika jest:

Neka je $U(G)$ podspecificirani jezik za CRU gramatiku G i neka je to skup svih rečeničnih formi α iz G, takvih da $\alpha \rightarrow \beta$, $\alpha \neq \beta$, $\beta \in L(G)$, gdje su elementi od $U(G)$ podspecificirani izrazi pod G, a elementi od $L(G)$ potpuno specificirani izrazi iz G.

Kažemo da je relacija podspecifikacije definirana između elemenata od $U(G)$ i $L(G)$.

Upravo podspecifikacija omogućava dobru dubinsku generaciju u sustavu i veću varijabilnost u površinskoj realizaciji.

Stoga je osnovna ideja u sustavu pCRU (Belz, 2008) sljedeća: ako su generacijska pravila oblika

$$\text{relacija}_1(arg_1, ..., arg_n) \rightarrow relacija_2(arg_1, ..., arg_p) \rightarrow ... \rightarrow relacija_m(arg_1, ..., arg_q)$$

uz $m \geq 1, n, p, q \geq 0$, onda se skup svih generacijskih pravila može promatrati kao da tvori konteksno neovisnu gramatiku, tj. gramatiku CRU. Jedinstveni vjerojatnosni model može se odrediti na označenim ili neoznačenim tekstovima kojima se može voditi generacijski proces, tj. biranje produkcija gramatike.

5.3.2. Generiranje tekstova vremenskih prognoza sustavom pCRU

Osnovni generator pCRU-a izgrađen je polu automatski. Automatski dio sastojao se od pisanja jednostavnih pravila fragmentiranja (engl. *chunking rules*), koja su fragmentirala tekst na smjer vjetra, brzinu vjetra, snagu naleta, izjave o naletima, vremenske izraze, tranzicijske izraze (npr. *incresing to*) predmodifikatore (engl. *pre-modifiers*) i postmodifikatore (engl. *post-modifiers*). Ovakvi fragmenti automatski su postali dio gramatike CRU. Potom su ručno napisana viša,apstraktnija pravila zadužena za dubinsku generaciju, tj. pravila koja su pozivala jednostavna automatski izgrađena pravila.

Dio CRU gramatike prikazan je na slici 5.7 i mogu se primjetiti viša pravila u prvim redovima i automatski izrađena pravila u nižim. preuzetoj iz (Belz, 2008).

$Gusts(Nv_1, N_2, n) \rightarrow GustCore(Nv_1, N_2)$
$Gusts(Nv_1, N_2, ST) \rightarrow GustCore(Nv_1, N_2)\ GustPostMod(ST)$

$GustCore(Nv, n) \rightarrow GustTrans\ Num(Nv)$
$GustCore(Nv_1, Nv_2) \rightarrow GustTrans\ Num(Nv_1) - Num(Nv_2)$

$GustTrans \rightarrow$ gusting
$GustTrans \rightarrow$ gusts
$GustTrans \rightarrow$ gusts to
$GustTrans \rightarrow$ in gusts
$GustTrans \rightarrow$ risk gusts to
$GustTrans \rightarrow$ with gusts
$GustTrans \rightarrow$ with gusts to

$GustPostMod(s) \rightarrow$ in any showers
$GustPostMod(s) \rightarrow$ in or near showers
$GustPostMod(s) \rightarrow$ in showers
$GustPostMod(t) \rightarrow$ in any thunderstorm
$GustPostMod(t) \rightarrow$ in any thunderstorms
$GustPostMod(t) \rightarrow$ in any thundery showers

Slika 5.7: Dio gramatike CRU sustava pCRU

Viša pravila osnova su za interpretaciju rečenica o vjetru predstavljenih kao nizovi neovisnih elemenata informacija tzv. segmenata, koji se minimalno sastoje od raspona brzine vjetra, a maksimalno od svih gore spomenutih fragmenata.

Na slici 5.8 prikazan je segment i pet rečenica (tj. rečeničnih ulomaka) koje su generirane iz njega.

$$Segment(2, -1, up, counterclock, slow, ssw, 22, 28, n, n, n) \stackrel{*}{\Longrightarrow}$$

gradually backing SSW and gradually increasing 22-28
gradually backing and increasing SSW 22-28
backing/increasing gradually SSW 22-28
backing SSW increasing 22-28
backing gradually SSW 22-28

Slika 5.8: Primjer segmenta sustava pCRU

Na tako izgrađenim pravilima pokrenuto je učenje kako bi se odredile vjerojatnosti pojavljivanja. Na slici 5.9, preuzetoj iz (Belz, 2008), prikazan je dio pravila i njihovih vjerojatnosti.

$-0.1513\ GustTrans \rightarrow$ gusts
$-2.3978\ GustTrans \rightarrow$ with gusts to
$-4.1026\ GustTrans \rightarrow$ risk gusts to
$-4.1026\ GustTrans \rightarrow$ with gusts
$-4.7957\ GustTrans \rightarrow$ gusts to
$-5.4889\ GustTrans \rightarrow$ gusting
$-5.4889\ GustTrans \rightarrow$ in gusts

Slika 5.9: Dio pravila gramatike CRU sustava pCRU

Procedura procjenje vjerojatnosti ponovljena je pet puta zbog unakrsne provjere. Malena varijacija između ponavljanja i mala razlika u rezultatima nad skupovima ta učenje i skupovima za ispitivanje ukazali su da je pet ponavljanja bilo dovoljno.

Površinska realizacija izvedena je (klasičnim) PCFG-om i bigramskim jezičnim modelom.

Statistički jezični model dodjeljuje vjerojatnost nizu od m riječi $P(w_1, ..., w_m)$ iz vjerojatnosne razdiobe. Generiranje pomoću jezičnog modela se provodi maksimizacijom vjerojatnosti niza riječi (Manning et al., 2008).

5.3.3. Evaluacija sustava pCRU

Evaluacija sustava izvedena je pomoću mjera BLEU-4 i NIST-5.

Sustav je generirao rečenice pomoću pohlepne metode (engl. *greedy*), Viterbijevim algoritmom (Viterbi, 1967). i odabirom pomoću ruletskog kola (engl. *roulette wheel selection*). Osim toga, implementirana je nasumična metoda i bigramski jezični model. Očekivano je da će Viterbijeva metoda pokazati najbolje rezultate, no to se nije pokazalo točnim. Takvo očekivanje bilo jest stoga što Viterbijeva metoda služi pronalaženju najboljeg (najvjrojatnijeg) puta između stanja (u ovom slučaju najboljeg odabira produkcija). Najbolje rezultate je pokazala pohlepna metoda za obje mjere. Razlog tome je što je Viterbijeva metoda imala tendenciju pisati kraće rečenice i stoga imala veću kaznu zbog kratkih rečenica. Prosječan broj riječi po rečenici u korpusu bio je 11.28, dok je prosjek za Viterbijevu metodu 7.54, a za pohlepnu 11.51. Druga po kvaliteti bila je metoda odabira pomoću ruletskog kola.

Osim opisanog načina vrednovanja, izvedena su dodatna ispitivanja s dva ljudska ocjenjivača koji su ocjenjivali tekstove nastale s dvije najbolje metode pCRU (pohlepna i ruletsko kolo) i tekstove nastale iz potpuno ručno izrađenog specijaliziranog sustava SumTime-Hybrid (Reiter et al., 2005). Prvi ocjenjivač dao je malu prednost sustavu SumTime-Hybrid, dok je drugi dao prednost pohlepnoj metodi pCRU, no u oba slučaja je razlika između ocjena pre-mala, a da bi bila statistički značajna.

6. Generiranje teksta iz strukturiranih podatka

IZ opisa srodnih radova može se primjetiti da dubinska generacija uvijek zahtijeva ručno pisanje pravila. Nadalje, dubinska generacija može se promatrati kao klasifikacija ulaznog niza prema vrsti izlaznog teksta. U slučaju sustava pCRU, to je riješeno ručnom izgradnjom viših pravila gramatike CRU, tj. preslikavanjem ulaznih segmenata na rečenice.

No, iako je sustav pCRU skratio vrijeme izrade sustava, on i dalje zahtijeva veliki ljudski upliv, proučavanje korpusa i neki oblik stručne analize.

Ono što se u ovom radu pokušava proučiti jest izgraditi sustav koji može prihvatiti bilo kakve strukturirane podatke uz pripadajuće im tekstove i uz minimalni ljudski upliv naučiti preslikavanje sa strukturiranih podataka na tekstove.

Nadalje, ako se računa na minimalan ljudski upliv, tada proučavanje korpusa ne može doći u obzir i ne može se računati da će se na bilo koji način dodatno obraditi i ručno pripremiti podatke. Bez proučavanja korpusa nije moguće znati broj vrsta tekstova, što znači da nije moguće znati broj klasa u smislu preslikavanja ulaza u odgovarajuću klasu, tj. vrstu teksta.

Uz takva ograničenj problemu je jedino moguće pristupiti pomoću nenadziranog učenja, tj. grupiranjem ulaznih podataka. Jedini ulazi u takav sustav su skup strukturiranih ulaznih podataka i njima pripadajućih tekstova te funkcija udaljenosti među podacima.

U nastavku će biti opisani skup podataka nad kojim se vršila generiranje, grupiranje podatka, način generiranja gramatike i način generiranja teksta.

6.1. Skup podataka

Za skup podataka korišten je korpus SumTime-METEO (Sripada et al., 2003) sačinjen od 1045 vremenskih prognoza pisanih za naftne platforme u Sjevernom moru. Osim tekstova prognoze, pridruženi su svi strukturirani podaci temeljem kojih meteorolozi pišu prognoze. Fokus prognoza su vjetar i valovitost mora, a sam tekst prognoze je vrlo tehnički. Prognoze su pisala tri različita meteorologa, što se pokazalo kao problem zbog određenih razlika u interpretaciji podataka o vremenu. Tekst prognoze može se vidjeti na slici 6.1 preuzetoj iz (Somayajalu G. Sripada, 2005). Zbog visoko tehničke prirode korpusa, ovakvi tekstovi predstavljaju idealne primjere za generiranje, jer je riječnik ograničen, a veza podataka i teksta je jasna. Iz tih razloga nekoliko radova ga je sustava koristilo, poput sustava pCRU (Belz, 2008) ili se gradilo nad njim poput sustava SumTime-Hybrid (Reiter et al., 2005). Sustav pCRU je postigao,za BLEU-4 mjeru, 0.636 za svoju najbolju metodu generiranja, a sustav SumTime-Hybrid je postigao 0.527 za istu mjeru.

6.2. Grupiranje podataka

Za grupiranje podataka izabrano je hijerarhijsko grupiranje, točnije algoritam hijerarhijskog aglomerativnog (Sibson, 1973) grupiranja s mjerom udaljenosti izgrađenom nad podacima za vremensku prognozu. Aglomerativno grupiranje postepeno grupira podatke, gdje su početne grupe svaki primjer za sebe, grupirajući uvijek dvije najbliže grupe dok se u konačnici sve grupe ne stope u jednu.

Kako količina podataka za pojedinu vremensku prognozu nije uvijek ista, nad podacima su rađene određene transformacije, tj. dodani su metapodaci koji opisuju pojedine podatke, a potom je nad metapodacima definirana funkcija udaljenosti. Metapodaci su definirani tako da pronalaze trend u podacima. Zbog jednostavnosti i mogućnosti usporedbe s ostalim radovima, odabrano je samo generiranje izjava (prognoza) o vjetru. Podaci za vjetar uključuju smjer, brzinu, snagu naleta na 10m visine i snagu naleta na 50m visine. Svaki je podatak prikazan u vremenskoj seriji od po tri sata za jedan dan (osam unosa po danu za svaki od podataka). Iz tih osam unosa (koji put je za prognozu korišteno i manje) stvara se metapodatak o stabilnosti stanja za sve vrste podataka i metapodatak

WINDIRSTB	podatak o stabilnosti smjera vjetra (1 ako stabilan, 0 ako nestabilan)
GUST10STB	podatak o stabilnosti snage vjetra na 10 m visine (1 ako stabilna, 0 ako nestabilna)
GUST10CHG	podatak o rastućem ili padajućem trendu snage vjetra na 10 m visine (1 ako raste ili pada, 0 ako je konstantna)
GUST50STB	podatak o stabilnosti snage vjetra na 50 m visine (1 ako stabilna, 0 ako nestabilna)
GUST50CHG	podatak o rastućem ili padajućem trendu snage vjetra na 50 m visine (1 ako raste ili pada, 0 ako je konstantna)
WINSTRSTB	podatak o stabilnosti brzine vjetra (1 ako stabilna, 0 ako nestabilna)
WINSTRCHG	podatak o rastućem ili padajućem trendu brzine vjetra (1 ako raste ili pada, 0 ako je konstantna)

Tablica 6.1: Metapodaci i njihovo objašnjenje.

o rastućem ili padajućem trendu za sve podatke osim za smjer vjetra. Stanje se smatra stabilnim ako se isti podatak pojavi više od polovice broja puta u vremenskoj seriji. Trend se smatra rastućim ako je barem polovica odnosa između dva susjedna podatka u seriji rastuće prirode.

Ovako je dobiveno ukupno 7 metapodataka koji se mogu prikazati u obliku binarnog broja. Metapodaci su prikazani u tablici 6.1 Udaljenost se potom promatra kao broj različitih mjesta između dva binarna broja. Dendrogram za jednostruku i za potpunu povezanost pokazao je slične rezultate što daje naznaku da je mjera dobro odabrana. Ovim načinom dobiveno je trinaest grupa.

6.3. Generiranje gramatike

Nad grupiranim podacima može se pristupiti izradi gramatike PCFG za pojedine grupe. Ovime se omogućava generiranje teksta koji pripada točno određenoj grupi.

Izgradnja PCFG-a provodi se u tri koraka:

1. Svi tekstovi koji pripadaju grupi parsiraju se rečeničnim parserom,

2. u banci stabala sva eksplicitna spominjanja podataka iz vremenskih prognoza zamjenjuju se s odgovarajućim oznakama,

3. iz dobivene banke stabala izravno se inducira vjerojatnosna kontekstno neovisna gramatika uz izračun vjerojatnosti pomoću procjene najveće izglednosti (engl. *maximum likelihood estimation, MLE*).

Prvi je korak neovisan o grupi kojoj gramatika pripada i može se izvršiti globalno, odjednom nad svim tekstovima. Ova pripremna radnja izvršena je koristeći Stanfordov parser za engleski jezik (Klein i Manning, 2003).

U drugom koraku se koristi nekoliko jednostavnih napisanih pravila koja pronalaze spominjanja podataka u tekstu i mijenjaju ih oznakama, numerirajući oznake koje označavaju podatke iste vrste. Oznake su za smjer vjetra (`WINDIR`), brzinu vjetra (`WINSTR`) i nalete (`GUSTS`). To vrijedi pokazati na primjeru jedne rečenice.

```
N-NNW 28-32 soon increasing 35-40 gusts 55, easing 25-30 gusts 45 later
```

Koristeći pravila dobiva se rečenica:

```
WINDIR1 WINSTR1 soon increasing WINSTR2 gusts GUST1, easing WINSTR3
gusts GUST2 later
```

Vrijedi napomenuti da je jednak rezultat rada i nad bankama stabala.

Nad tako pripremljenim skupom obavljeno je grupiranje i generirane su pojedinačne gramatike (sve ukupno 13. gramatika).

Osim gramatike, izgrađen je i bigramski jezični model za pojedine grupe.

6.4. Generiranje teksta

Generiranje teksta izvedeno je pomoću pohlepne metode i metode ruletskog kola za PCFG i klasičnog bigramskog jezičnog modela.

Te metode su odabrane zbog rezultata prikazanih u (Belz i Reiter, 2006). Proces generiranja pomoću pohlepne metode vrlo je jednostavan: počevši od početnog

znaka gramatike uvijek se bira ono pravilo s najvećom mogućom vjerojatnosti. U slučaju metode ruletskog kola pristup je nešto drugačiji i koristi nasumične brojeve. Vjerojatnosti produkcija mogu se prikazati kao ruletsko kolo gdje svaka moguća produkcija s istom lijevom stranom zauzima onoliko od ruletskog kola kolika je njena vjerojatnost. Generator slučajnih brojeva (loptica u ruletu) generira slučajni broj koji upada u raspon nekog područja i to područje (tj. produkcija) biva izabrano. Npr., ako postoje pravila $S \to a$ uz $p = 0.5$, $S \to b$ uz $p = 0.3$, $S \to c$ uz $p = 0.12$ i $S \to d$ uz $p = 0.08$, tada, ako ih se posloži na ruletsko kolo (zbog jednostavnosti ga se može i razmotati na pravac od 0 do 1) tako da produkcija u a zauzima brojeve od 0 do 0.5, produkcija u b zauzima brojeve od 0.5 do 0.8, produkcija u c zauzima brojeve od 0.8 do 0.92 i produkcija u d zauzima brojeve od 0.92 do 1, generiranje slučajnog broja 0.23 upada u područe produkcije u a i upravo se ta produkcija onda koristi.

Vrednovanje dobivenih tekstova je provedeno mjerom BLEU-4.

```
1.INFERENCE 0300 GMT, TUESDAY,          25-Dec   2001
LOW 968MB OVER SOUTHERN SWEDEN WILL MOVE EASTWARDS. LOW 976MB
WEST OF BERGEN WILL MOVE SOUTHEASTWARDS TO BE OVER SOUTHERN
DENMARK BY EVENING. A DEPRESSION WILL FORM OFF THE DENMARK STRAIT
AND MOVE SOUTHEAST TO BE OVER THE NORTH OF SCOTLAND BY THURSDAY
AFTERNOON.

2.FORECAST 06-24 GMT, TUESDAY,          25-Dec   2001

-----WARNINGS: WINDS ABOVE 40 KNOTS                          -------

WIND(KTS)       CONF   HIGH
  10M:          N-NNW 28-32 SOON INCREASING 35-40 GUSTS 55, EASING
                25-30 GUSTS 45 LATER
  50M:          N-NNW 35-40 SOON INCREASING 45-50 GUSTS 65, EASING
                30-38 GUSTS 45 LATER
WAVES(M)        CONF   HIGH
  SIG HT:       4.5-5.0 RISING 6.0--7.0, LATER FALLING 5.0-6.0 LATER
  MAX HT:       7.5-8.0 RISING 9.5-11.0, LATER FALLING 8.0-9.5 LATER
  PER(SEC):     8-13
  WEATHER:      SQUALLY WINTRY SHOWERS
  VIS(NM):      OVER 10 REDUCED TO 1-3 IN SHOWERS
  TEMP(C):      3-4 FALLING 1 OR LESS IN SHOWERS
  CLOUD:        3-5 CUSC 1200-2000 BECOMING 5-7 CUCB 600-1000
(OKTAS/FT)      IN SHOWERS
  LIGHTNING RISK : OVER 60 PERCENT IN SHOWERS

3.FORECAST 00-24 GMT, WEDNESDAY,        26-Dec   2001
  WIND(10M):    N-NNW 25-30 GUSTS 45 GRADUALLY EASING NW  8-12
      (50M):    N-NNW 30-38 GUSTS 55 GRADUALLY EASING NW 10-15
  SIG WAVE:     5.0-6.0 FALLING 3.5-4.0
  MAX WAVE:     8.0-9.5 FALLING 5.5-6.5
  WEATHER:      WINTRY SHOWERS GRADUALLY DYING OUT
  VIS:          GOOD EXCEPT IN SHOWERS

4.FORECAST 00-24 GMT, THURSDAY,         27-Dec   2001
  WIND(10M):    NW  8-12 EASING  8 OR LESS FOR A TIME, INCREASING
                NW-N 15-20 LATER
      (50M):    NW 10-15 EASING 10 OR LESS FOR A TIME, INCREASING
                NW-N 18-25 LATER
  SIG WAVE:     3.5-4.0 FALLING 3.0-3.5
  MAX WAVE:     5.5-6.5 FALLING 5.0-5.5
  WEATHER:      SCATTERED WINTRY SHOWERS

5A.LONG RANGE OUTLOOK:  FRI  28-Dec 2001, AND  SAT    29-Dec 2001,
  WIND(10M):    N-NW 15-20 SOON INCRAESING 20-25, EASING 10-15
                SATUDAY MORNING, INCREASING 20-25 LATER
  SIG WAVE:     3.0-3.5 RISING AROUND 5.5, LATER FALLING AROUND 4.0


N.B. THE HIGHEST INDIVIDUAL WAVE THAT MAY BE EXPERIENCED IS OF THE
       ORDER OF TWICE THE SIGNIFICANT WAVE HEIGHT.
```

Slika 6.1: Primjer teksta vremenske prognoze

7. Tehnička implementacija

Za jezik implementacije odabran je programski jezik Haskell zbog brzine izvršavanja i lake izrade prototipova. Haskell je standardizirani, čisti funkcijski programski jezik, opće namjene s lijenom evaluacijom i strogim statičkim tipskim sustavom. Vrijedi spomenuti i objasniti još tipske razrede. Tipski razredi omogućuju definiranje instanci neke klase za neki tip podataka i definicija funkcija (operacija) pridružene toj klasi. Tipski razredi, dakle sadrže operacije koje određuju tu klasu.

Program pisan u Haskellu se sastoji od kolekcije modula. Modul služi kontroli prostora imena i kreiranju apstraktnih tipova podataka. Izgrađeno je ukupno osam modula:

- `Data.Wind` – modul koji sadrži tip podataka pojedine prognoze za vjetar i pomoćni je modul,
- `Data.Grammar` – modul koji sadrži tip podataka vjeorjatnosne konteksno neovisne gramatike i funkcije koje se mogu izvršiti nad njom,
- `Data.Grammar.PCFGinduction` – modul koji sadrži funkcije potrebne za indukciju PCFG-a iz banke stabala,
- `Clustering.Hierarchical` – modul odgovoran za grupiranje podataka,
- `Database.DatabaseAccess` – pomoćni modul koji služi pristupu bazi podatka,
- `NLG.Generation` – modul koji sadrži funkcije odgovorne za generiranje tekst.

.

Osim Haskella, korišten je i programski jezik Java za predobradu tekstova parserom, jer je Stanfordov parser implementiran u Javi. Svi su podaci pohranjeni u bazi podataka Microsoft SQL Server 2008, jer su podaci iz korpusa SumTime-METEO

došli u obliku pogodnom za tu bazu podatka (tablice Microsoft Access). Osim toga, Java je korištena i za vrednovanje sustava, gdje je korišten Stanfordov paket za strojno prevođenje Phrasal i njegova gotova implementacija mjere BLEU (Cer et al., 2010).

U nastavku će svaki od modula biti nešto detaljnije predstavljen, uz opis svake javne funkcije ili tipa podataka modula, dok će one privatne funkcije i tipovi podatka biti preskočeni (jer ionako predstavljaju pomoć glavnim funkcijama).

7.1. Data.Wind

Pomoćni modul `Data.Wind` sastoji se od novog tipa podatka `Wind` u kojem su sadržani podaci preuzeti iz baze podatka za pojedini vjetar. Svi podaci, osim identifikacijskog broja prognoze prikazani su kao lista nizova znakova. Osim toga, u modulu je implementirana funkcija udaljenosti između dva vjetra, `distanceBetweenWinds` onako kako je to opisano u poglavlju **??**.

7.2. Data.Grammar

U modulu `Data.Grammar` implementiran je novi tip podatka `CFGrammar` koji predstavlja jednu produkciju gramatike i sastoji se od njene vjerojatnosti, nezavršnog znaka s lijeve strane pravila i skupa završnih i nezavršnih znakova s desne strane. Osim toga implementiran je tip `Symbol` koji je zapravo samo sinonim za tip podatka `String`, no takva je implementacija bila potrebna zbog primjenjivanja i implementacije tipske klase `CFGSymbol` nad tipom podatka `Symbol`. Tipska klasa `CFGSymbol` predstavlja funkcije koje vrše provjeru je li znak završni (`isTerminal`) ili nije završni (`isNonTerminal`) i mogu se cirkularno definirati (`isTerminal = not . isNonTerminal` i obrnuto). Osim toga implementirane su još metode `countProductions`, koja vraća broj produkcija u gramatici i funkcija `normalizePCFG`, koja postavlja produkcije na pretpostavljenu normalnu formu, dakle nema višestrukih produkcija i sve su vjerojatnosti ispravno izračunate i postavljene koristeći procjenu maksimalne izglednosti.

7.3. Data.Grammar.PCFGInduction

Modul `Data.Grammar.PCFGInduction` ima samo jednu javnu funkciju, `PCFGInduction`, koja služi indukciji vjerojatnosne konteksno neovisne gramatike iz banke stabala.

7.4. Clustering.Hierarchical

Modul `Clustering.Hierarchical` služi zapravo samo kao omotač oko modula `Data.Clustering.Hierarchical`[1], koja implementira algoritam hijerarhijskog aglomerativnog grupiranja. Tako se modul sastoji samo od funkcije `clusterWinds`, koja preuzima listu vremenskih prognoza o vjetru i vraća listu listi prognoza o vjetru, tj. grupirane podatke.

7.5. Database.DatabaseAccess

Pomoćni modul `Database.DatabaseAccess` služi za pristup bazi podatka i implementira funkcije `queryParsedFields`, koja vraća banku stabla za daljnu indukciju u PCFG, `queryClusteringData`, koja vraća skup vremenskih prognoza za vjetar potrebnih za grupiranje, te funkciju `insertClusterData`, koja u bazu upisuje grupirane podatke.

7.6. NLG.Generation

Modul `NLG.Generation` sastoji se od implementacije različitih metoda generiranja teksta. Tako funkcija `greedyPCFG` implementira pohlepnu metodu za PCFG, a funkcija `roulettePCFG` implementira metodu ruletskog kola za PCFG.

[1] Implementacija modula se nalazi na web adresi http://hackage.haskell.org/packages/archive/hierarchical-clustering/0.4.2/doc/html/Data-Clustering-Hierarchical.html

8. Rezultati

Puštanjem sustava u rad dobili su se rezultati drugačiji od očekivanih. U nastavku je dan primjer za pohlepnu metodu, koja se pokazala najbolja, dok je metoda ruletskog davala ili jako duge rečenice ili jako kratke rečenice nad kojima nije moglo biti napravljeno konzistentno mjerenje.

```
WINDIR1 WINSTR1 later WINDIR2 WINSTR2
```

Iz podataka u 8.1 dobiva se sljedeća rečenica, koristeći gornji primjer:

```
SSW 10 later SW 16
```

`WINDIR` predstavlja smjer vjetra, a `WINSTR` snagu vjetra. Može se primjetiti da ovaj tekst predstavlja grupu tekstova u kojima je početno bio jedan smjer vjetra s nekom snagom, a potom je došlo do promjene u drugi smjer s drugom snagom. Nažalost, standardni tekst prognoze sadrži još neke podatke, poput izraza o vremenu dana u kojem je došlo do promjene, npr. *by evening*, *by midday* i sl. Postoji nekoliko mogućih interpretacija zašto je došlo do ovakvog rezultata što će biti raspravljeno u nastavku.

TIME	WINDDIR	WINDSPEED	GUST10m	GUST50m
03	SSW	10	8	10
06	SSW	12	8	10
09	SSW	12	9	11
12	SSW	12	7	9
15	SW	16	13	15
18	SW	17	14	21
21	SW	16	10	17

Tablica 8.1: Primjer podataka o vremenu iz baze podataka.

Grupa	BLEU-4
1	0.11
2	0.05
3	0.17
4	0.0006
5	0.007
6	0.09
7	0.09
8	0.03
9	0.01
10	0.12
11	0.64
12	0.45
13	0.09
14	0.21

Tablica 8.2: Rezultati vrednovanja pohlepne metode BLEU-4 mjerom po grupama

8.1. Rasprava

Za loše rezultate razlog se može tražiti na nekoliko mogućih mjesta:

- Potreba za većim stručnim znanjem,
- grupiranje provedeno nad podacima umjesto nad tekstovima,
- loše definirana funkcija udaljenosti
- i loše definirani ulazni podaci.

U tablici 8.2 su dani rezultati za pohlepnu metodu za BLEU-4 mjeru na skupu za vrednovanje.

Može se primjetiti da najbolji rezultat ima grupu 10, gdje 60% rečenica je istog oblika kao i rečenica generirana pohlepnom metodom. Ta rečenica se sastoji od samo 2 podatka, jedan o smjeru vjetra i drugi o snazi vjetra, npr.:

```
SSE 10-16
```

Ostale rečenice te grupe predstavljaju varijaciju na temu.

Prosječna ocjena sustava je 0.1475, što nije usporedivo s ocjenama sustava pCRU (0.64) i SumTime-Hybrid (0.53) čija je evaluacija također rađena nad istim korpusom. No, grupe 10 i 11 su usporedive s ovim rezultatima.

Razloge loših rezultata valja potražiti u sljedećem: (1) potrebi za većim stručnim znanjem, (2) provođenju grupiranja nad tekstovima, a ne nad podacima, (3) loše definiranoj funkciji udaljenosti, te (4) lošom definicijom ulaznih podataka.

8.1.1. Potreba za većim stručnim znanjem

Proučavajući skup podatka može se primjetiti određena nekonzistentnost između podataka o vremenu i samog teksta vremenske prognoze. To se može objasniti time što meteorolozi koriste, osim podataka prikazanih u skupu, i dodatne podatke (poput prognostičkih karata) i osobno iskustvo. Dapače, često se može dogoditi da će meteorolog potpuno zanemariti neki podatak i intuitivno nešto zaključiti (Somayajalu G. Sripada, 2005). Tu vrstu ekspertne intuicije nije moguće pronaći i izraziti u programskom kodu bez pomoći stručnjaka.

8.1.2. Provođenje grupiranja nad tekstovima

Proučavajući radove koji su koristili isti skup podataka poput Belz (2008) i Reiter et al. (2005) dolazi se do zaključka da grupiranje i klasifikacija nisu provedeni nad skupom podataka o vremenu, već nad samim tekstom prognoze. Nadalje, grupiranje i klasifikacija su provedeni potpuno ručno u slučaju sustava SumTime-Hybrid (Reiter et al., 2005) ili polu-automatski u slučaju sustava pCRU (Belz, 2008). Još vrijedi primjetiti da se automatski dio sustava pCRU odnosi na odluke niske razine, poput odluke da je vremenski izraz za vrijeme dana “nakon 12 sati popodne”. Odluke visoke razine, koje odlučuju o obliku teksta na temelju ulaznih podatka, napisane su ručno, što je značilo da je određeno vrijeme provedeno u analizi korpusa. Dapače, u slučaju sustava SumTime-Hybrid analiza je provedena uz pomoć stručnjaka.

8.1.3. Loša definicija funkcije udaljenosti

Loše definiranje funkcije udaljenosti može se nadovezati na potrebu za većim stručnim znanjem, tj. nemogućnosti ispravnog interpretiranja podataka o vremenu. Nadalje, moguće je da se funkcija udaljenosti trebala definirati nad tekstovima, no tada bi se ponovno zahtjevala još veća ručna analiza korpusa i sustav bi izgubio smisao da generira tekstove na temelju strukturiranih ulaznih podataka.

8.1.4. Loša definicija ulaznih podataka

Pogledom na radove pCRU, SumTime-Hybrid i sl., može se zaključiti da je ulaz u sustav drugačiji nego što je predviđen u ovom radu. Iako jest u pitanju neka vrsta strukturiranih podataka, to nisu podaci koji opisuju vremensku prognozu, već točno diktiraju kako mora izgledati tekst prognoze. Dapače, u slučaju sustava pCRU, ulaz se sastoji od skupa *segmenata* koji u sebi nose podatke što treba biti prikazano u vremenskoj prognozi. Svaki se segment preslikava na točno jedan komad teksta. Svi segmenti nisu nužno iste dužine (ne nose istu količinu podataka) i jedini podatak koji segment minimalno treba imati jest smjer vjetra. Takva vrsta ulaza prilagođena je stručnjaku koji prethodno interpretira podatke o prognozi i piše ulaz. Može se postaviti pitanje koliko je isplativo pisati takve vrste ulaza ako su već sami tekstovi kratki.

9. Zaključak

Generiranje prirodnog jezika grana je umjetne inteligencije i računalne lingvistike čiji je cilj izgradnja računalnih sustava koji mogu stvoriti tekstove na nekom prirodnom jeziku. Danas postoji mnoštvo različitih sustava NLG od kojih je većina potpuno ručno izrađena i ograničena na određenu domenu i određeni skup podataka. Postoje i sustavi opće namjene, no većina njih zahtjeva velike pripremne radnje i vrijedi se zapitati koliko se vrijeme gradnje sustava skratilo. Iznimka među sustavima opće namjene je sustav pCRU koji je bitno skratio vrijeme gradnje sustava za određenu domenu, no i dalje se zahtjeva veliki upliv čovjeka. Vrijedi istražiti može li se napraviti sustav kojim se može potpuno automatski generirati tekst, uz minimalan upliv čovjeka, a čiji su ulaz skup strukturiranih podataka i pripadnih im tekstova.

U ovom radu je istražena mogućnost potpuno automatskog generiranja teksta koristeći strukturirane podatke kao ulaz i grupiranje podataka kao proces donošenja odluka dubinske generacije. Za generiranje je korištena vjerojatnosna konteksno neovisna gramatika. Nad ulaznim podacima je obavljeno hijerarhijsko aglomerativno grupiranje da bi se dobili različiti tipovi teksta. Evaluacija je provedena mjerom BLEU-4 i rezultati su se pokazali bitno lošiji od rezultata sličnih radova.

Razlog loših rezultata se može pronaći u više mogućih stvari, no vrijedi primjetiti da se većina može ispraviti ili ručnim pisanjem klasifikacijskih pravila ili suradnjom s ekspertom za područje s kojeg je korpus.

Kao potencijalno riješenje problema se nameće korištenje manje ekspertnih tekstova s podacima koje je lakše interpretirati, no to se može pokazati kao dvosjekli mač zbog toga što manje ekspertni tekstovi imaju manje ograničen jezik što ne pogoduje generiranju ni na koji način.

Rad sustava se svakako može poboljšati u suradnji s ekspertom za dane podatke, u ovom konkretnom slučaju meteorologom, npr. tako da se pomoću stručnjaka

napiše funkcija udaljenosti između dva skupa podatka.

Literatura

Anja Belz. Context-free representational underspecification for nlg, 2004.

Anja Belz. Automatic generation of weather forecast texts using comprehensive probabilistic generation-space models. *Natural Language Engineering*, 14:431–455, October 2008. ISSN 1351-3249. URL `http://dl.acm.org/citation.cfm?id=1520025.1520026`.

Anja Belz i Ehud Reiter. Comapring automatic and human evaluation of nlg systems. U *Proceedings of EACL 2006*, 2006.

Kim Binsted i Graeme Ritchie. An implemented model of punning riddles. U *AAAI '94*, stranice 633 – 638, 1994.

Charles B. Callaway i James c. Lester. Narrative prose generation. *Artificial Intelligence*, 139:213–252, 2002.

D. Cer, M. Galley, D. Jurafsky, i C.D. Manning. Phrasal: A toolkit for statistical machine translation with facilities for extraction and incorporation of arbitrary model features. U *Proceedings of the NAACL HLT 2010 Demonstration Session*, stranice 9–12. Association for Computational Linguistics, 2010.

E. Charniak. Tree-bank grammars. U *Proceedings of the National Conference on Artificial Intelligence*, stranice 1031–1036, 1996.

Tai-Hung Chen i Chun-Han Tseng. Automatic learning of context-free grammar, 2006.

George Doddington. Automatic evaluation of machine translation quality using n-grams co-occurrence statistics. U *HLT '02 Proceedings of the second international conference on Human Language Technology Research*, stranice 138–145, 2002.

Roger Evans, Paul Piwek, i Lynne Cahill. What is nlg?, 2010.

Dan Klein i Christopher D. Manning. Automatic unlexicalized parsing, 2003.

C.Y. Lin i E. Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. U *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, stranice 71–78. Association for Computational Linguistics, 2003.

C. William Mann. What is special about natural language generation research? Theoretical Issues in Natural Language Processing 3, 1987.

C.D. Manning, P. Raghavan, i H. Schutze. *Introduction to information retrieval*, svezak 1. Cambridge University Press Cambridge, 2008.

D. McDonald. Type-driven suppression of redundancy in the generation of inference-rich reports. *Aspects of automated natural language generation*, stranice 73–88, 1992.

K. Papineni, S. Roukos, T. Ward, i W.J. Zhu. Bleu: a method for automatic evaluation of machine translation. U *Proceedings of the 40th annual meeting on association for computational linguistics*, stranice 311–318. Association for Computational Linguistics, 2002.

E. Reiter, S. Sripada, J. Hunter, J. Yu, i I. Davy. Choosing words in computer-generated weather forecasts. *Artificial Intelligence*, 167(1):137–169, 2005.

Ehud Reiter. Has a consensus nl generation architecture appeared, and is it psycholinguistically plausible? U *Proceedings od the 7th International Workshop on Natural Language generation*, INGLW-1994, stranice 163–170, 1994.

Ehud Reiter. Building natural language generation sytems, 1996.

Ehud Reiter i Robert Dale. *Building Natural Language Generation Systems*. Cambridge University Press, 2000.

Ehud Reiter, Ross Turner, Norman Alm, Rolf Black, Martin Dempster, i Annalu Waller. Using nlg to help language-impaired users tell stories and participate in social dialogues. U *Proceedings of the 12th European Workshop on Natural Language Generation*, ENLG '09, stranice 1–8, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

Ehud Reiter Ross Turner, Somayjalu Sripada. Empirical methods in natural language generation, 2010.

Ehud Reiter Ian P. Davy Ross Turner, Somayajalu Sripada. Generating spatio-temporal descriptions in pollen forecasts, 2006.

R. Sibson. Slink: an optimally efficient algorithm for the single-link cluster method. *The Computer Journal*, 16(1):30–34, 1973.

M. Sipser. *Introduction to the Theory of Computation*. Thomson Course Technology, 2006.

Ehud Reiter Somayajalu G. Sripada. Sumtime-meteo: Parallel corpus of naturally occuring forecast texts and weather data, 2005.

Siniša Srbljić. *Jezični procesori 1.* 2004.

S.G. Sripada, E. Reiter, J. Hunter, i J. Yu. Exploiting a parallel text-data corpus. *ENE*, 15(18):23, 2003.

A. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *Information Theory, IEEE Transactions on*, 13(2):260–269, 1967.

Generiranje tekstnog izvještaja na temelju strukturiranih podataka

Sažetak

Generiranje prirodnog jezika je grana umjetne inteligencije i računalne lingvistike čiji je cilj izgradnja računalnih sustava koji mogu stvoriti tekstove na nekom prirodnom jeziku. Danas postoji mnoštvo sustava NLG, no većina ih je ograničena na točno određenu domenu i točno određen ulazni skup podataka. Nekolicina radova opće namjene obično zahtjevaju dugotrajne pripremne radnje da bi promijenili domenu. U radu je istražena mogućnost potpuno automatskog generiranja teksta na temelju strukturiranih podataka, pomoću vjerojatnosne kontekstno neovisne gramatike i grupiranja podataka hijerarhijskim aglomerativnim algoritmom grupiranja. Dobiveni generirani tekstovi su evaluirani pomoću BLEU-4 mjere koja je dala rezultat 0.14. Rezultat se može smatrati lošim u usporedbi sa sličnim sustavima rađenim na istom skupu podataka. Može se zaključiti da je ipak potreban veći ljudski upliv.

Ključne riječi: NLG,PCFG,kontekstno neovisna gramatika, generiranje izvjesca, prirodni jezik, racunalna lingvistika, umjetna inteligencija, generiranje prirodnog jezika

Text Report Generation Based on Structured Data

Abstract

Natural language generation is a field in artificial intelligence and computer linguistics. The goal of NLG is building computer systems that can generate texts in a natural language. There are many NLG systems today, but most of them are restricted to one domain, and one type of input. There are general systems that can change domains, but the change, most of the time, is hard to make. This paper explores the posibility of completely automatic text generation based on structured data. Generation is done with probibalistic context-free grammar and hiearchical aglomerative clustering algorithm. Generated texts were evaluated using BLEU-4, and the result was 0.14. This is a poorer result than in similar system which used the same corpus. In conclusion, it can be said that bigger human input is necessary.

Keywords: natural language generation, context-free grammar, PCFG, report generation, natural language, computer linguistics, artificial intelligence

Dodatak A
Izgrađeni Haskell moduli i njihove javne funkcije

```
module Clustering.Hierarchical (
    clusterWinds
  ) where
```

`clusterWinds :: [Wind] -> [[Wind]]`

Function clusters winds using `distanceBetweenWinds` function form `Data.Wind` module

```
module Database.DatabaseAccess (
    queryParsedFields,  queryClusteringData,  insertClusterData
  ) where
```

`queryParsedFields :: String -> Int -> IO [String]`

Returns the prepared parsed fileds which are used in PCFG induction.

```
queryClusteringData :: IO [Wind]
```

Returns wind data for clustering

```
insertClusterData :: [[Wind]] -> IO ()
```

Inserts data about clusters into db

```
module Data.Grammar (
    CFGrammar(CFG, p, leftSide, rightSide),
    CFGSymbol(isTerminal, isNonTerminal),  Symbol(Symbol, unSymbol),
    countProductions,  normalizePCFG
  ) where
```

```
data CFGrammar
```

```
=            CFG
p ::  Float  leftSide ::  Symbol  rightSide ::  [Symbol]
```

Represents a (P)CFG.

```
instance Eq CFGrammar
instance Show CFGrammar
```

```
class CFGSymbol a where
```

Simple class wich encapsulates a Context-Free Grammar Symbol. It can determin if a Symbol is terminal or non terminal. The minimum required definition is isTerminal (or isNonTerminal, but positive example is encouraged)

Methods

```
isTerminal :: a -> Bool
```

```
isNonTerminal :: a -> Bool
```

```
instance CFGSymbol Symbol

newtype Symbol
     =                    Symbol
     unSymbol ::  String

instance Eq Symbol
instance Show Symbol
instance CFGSymbol Symbol

countProductions :: Symbol -> [CFGrammar] -> Float
```

Counts the total production count for a variable

```
normalizePCFG :: [CFGrammar] -> [CFGrammar]
```

Normalizes the grammar to correct PCFG form

```
module Data.Grammar.PCFGInduction (
    pcfgInduction
  ) where
```

PCFG Induction from treebanks

```
pcfgInduction :: [String] -> [CFGrammar]
```

creates the grammar from a list of parsed sentences

```
module Data.Wind (
    Wind(Wind, windid, winddir, windspeed, gust10m, gust50m),  filterWinds,
    distanceBetweenWinds
  ) where
```

```
data Wind
     =                      Wind
     windid ::  Int         winddir ::  [String]  windspeed ::  [String]
     gust10m ::  [String]   gust50m ::  [String]

instance Eq Wind
instance Show Wind

 filterWinds :: Int -> String -> Int -> String -> Bool

distanceBetweenWinds :: Wind -> Wind -> Double
```

Calculates the distance between two wind prognosis

```
module NLG.Generation (
    greedyPCFG,  roulettePCFG
  ) where
```

```
greedyPCFG :: [CFGrammar] -> String
```

Returns a sentence generated using greedy method.

```
roulettePCFG :: [CFGrammar] -> IO String
```

Returns a sentence generated using roullette wheel method.

Dodatak B
Dio gramatike za jednu od grupa

```
["JJ"->["w-nw"](2.5641026e-2),
"ROOT"->["FRAG"](0.25531915),
"NNS"->[""](0.3548387),
"FRAG"->["ADJP","S"](0.625),
"WINSTR1"->["nil"](1.0),
"S"->["VP"](0.51282054),
"NN"->["easing"](4.7058824e-2),
"VP"->["VBG","NP","NP"](2.7272727e-2),
"GUST1"->["nil"](1.0),
"ADJP"->["JJ","NP"](0.6666667),
"CC"->["or"](0.75),
"NP"->["GUST1"](5.899705e-2),
"WINDIR1"->["nil"](1.0),
"NP"->["WINSTR2","NNS"](8.8495575e-3),
"NP"->["JJ","NNS"](8.8495575e-3),
"NP"->["WINSTR1"](6.4896755e-2),
"NP"->["WINSTR1","NN","NNS"](5.899705e-3),
"NNS"->["gusts"](0.48387095),
"QP"->["GUST1","CC","WINDIR1"](0.54545456),
"WINSTR2"->["nil"](1.0),
"NP"->["QP"](5.6047197e-2),
"VBG"->["increasing"](0.17105263),
"NP"->["NP","NP"](4.4247787e-2),
"JJ"->["s-se"](0.102564104),
"ROOT"->["NP"](0.24468085),
```

```
"ROOT"->["S"](0.30851063),
"NN"->["backing"](3.529412e-2),
"S"->["NP","VP"](0.2948718),
"WINDIR2"->["nil"](1.0),
"VP"->["VBD","NP","ADVP"](1.8181818e-2),
"IN"->["by"](0.70212764),
"NP"->["QP","NNS"](2.9498525e-3),
"NN"->["evening"](0.27058825),
"ADVP"->["FW","WINDIR2"](1.6949153e-2),
"NP"->["WINDIR1","WINSTR1","NN"](2.9498525e-3),
"QP"->["GUST1","CC","CD"](9.090909e-2),
"NP"->["WINDIR2","WINSTR2"](1.7699115e-2),
"NP"->["WINDIR1"](5.899705e-3),
"NP"->["NN"](8.5545726e-2),
"FW"->["becoming"](1.0),
"PP"->["IN","NP"](0.6909091),
"CD"->["less,"](0.21428572),
"NP"->["NP","PP"](3.539823e-2),
"VBD"->[""](0.75),
"VBZ"->["gusts"](0.9444444),
"ROOT"->["X"](5.319149e-2),
"VBG"->["easing"](0.25),
"X"->["X","S"](0.4),
"DT"->["this"](0.5),
"S"->["ADVP","VP"](6.410257e-2),
"NP"->["WINDIR1","WINSTR1"](9.7345136e-2),
"VP"->["VBG","NP","PP"](0.21818182),
"NP"->["WINSTR2"](9.439528e-2),
"X"->["ADVP","NP"](0.4),
"NP"->["DT","NN"](3.539823e-2),
"ADVP"->["RB"](0.6440678),
"VP"->["VBZ","NP","S"](2.7272727e-2),
"ADVP"->["WINDIR1"](5.0847456e-2),
"JJ"->["s-sse"](2.5641026e-2),
"RB"->["slowly"](7.272727e-2),
"RB"->["later"](0.2),
```