UNIVERZITET U BEOGRADU
MATEMATIČKI FAKULTET

Miloš Utvić

KONAČNI AUTOMATI U REGULARNOJ IMENSKOJ DERIVACIJI

Magistarska teza

BEOGRAD
2008.

Sadržaj

Predgovor

Rad je podeljen u četiri dela, uz uvodnu glavu u kojoj je dat kratak pregled problema prepoznavanja imenovanih entiteta u okviru razvoja sistema za razumevanje poruka i poseban osvrt na problem prepoznavanja klase vlastitih imena.

U prvom delu opisani su različiti modeli primene konačnih automata i transduktora u rešavanju problema morfološke analize prirodnih jezika. Zatim su izloženi osnovni pojmovi koji se koriste u daljem radu sa posebnim osvrtom na predstavljanje leksičkih informacija posredstvom acikličnih konačnih automata. Aciklični konačni automati se uvode kao struktura koja je podesna za efikasnu i kompaktnu reprezentaciju leksičkih informacija.

Drugi deo je posvećen fenomenu regularne derivacije i analizi primene konačnih transduktora na različite probleme koji se javljaju u prepoznavanju leme tokom procesa automatske morfološke analize. Najpre se posmatra pitanje nepoznate reči, tj. reči koja je neprepoznata u procesu leksičkog prepoznavanja i pokazuje kako se nepoznata reč može aproksimirati formalizovanjem procesa regularne derivacije. Razmatra se uloga regularne derivacije u dopuni sadržaja elektronskog rečnika i uvodi se pojam super-leme. Super-lema predstavlja uopštenje tradicionalne leme u jezicima sa bogatom morfologijom. Ovaj pojam omogućava da se u procesu leksičkog prepoznavanja sačuva rezultat tradicionalne analize, a istovremeno obezbeđuje kompaktan i sistematičan opis strukture rečničke odrednice tako da budu obuhvaćena pojedina derivaciona svojstva pored njenih flektivnih osobina.

U trećem delu se razmatra opis flektivnih i derivacionih svojstava toponima sa ciljem da se pojam super-leme primeni na opis njihovih flektivnih i derivacionih svojstava. Izloženi su principi klasifikovanja navedenih svojstava toponima i uvedeni pojmovi N-regularnog izraza i N-regularnog derivacionog pravila. Pokazano je na koji način se fenomen regularne derivacije može primeniti na izgradnju baze vlastitih imena tipa Prolex na primeru toponima.

Četvrti i završni deo, sastoji se iz tri celine. U prvoj su opisani izvori građe (rečnici i korpusi) korišćeni u istraživanju. Druga celina opisuje metode implementiranja morfoloških procesa u okviru formalizma sistema NooJ. Proučeni su i prikazani nedostaci NooJ-a u implementaciji procesa regularne derivacije. U trećoj celini, oslanjajući se na rezultate trećeg dela, izvršena je klasifikacija derivacionih paradigmi jednočlanih i višečlanih toponima.

Naposletku, u dodacima su prikazani rezultati primene izvršene analize na strukturiranje baze tipa Prolex, kao i rezultati primene generisanih rečnika u obradi teksta izvodi iz elektronskih rečnika toponima, kao i primeri rezultata obrade, a na samom kraju je navedena korišćena literatura.

Želim da se zahvalim članovima komisije za pregled i ocenu ove teze, dr Gordani Pavlović-Lažetić, dr Predragu Piperu i mentoru dr Dušku Vitasu, na pažljivom čitanju i primedbama koje su doprinele poboljšanju konačne verzije teksta. Takođe se zahvaljujem članovima Grupe za jezičke tehnologije na Matematičkom fakultetu u Beogradu, posebno dr Cvetani Krstev i mr Ranki Stanković, kao i dr Deniju Morelu sa Univerziteta Fransoa Rable u Turu, na ustupljenim resursima nad kojima su vršena istraživanja iz ovog rada.

Na kraju se zahvaljujem i svojim roditeljima, kolegama i prijateljima na podršci koju su mi pružili tokom izrade ove teze.

U Beogradu, januara 2008. Miloš Utvić

uvod

IMENOVANI ENTITETI

U ovom uvodnom delu razmatraju se pojam i značaj imenovanih entiteta, posebno vlastitih imena, u savremenoj obradi prirodnih jezika pomoću računara. Biće pomenuti svi značajniji faktori preko kojih imenovani entiteti utiču na efikasnost i kvalitet te obrade. Takođe će biti ukazano na svu složenost prepoznavanja i klasifikovanja imenovanih entiteta pomoću računara, na moguće pristupe rešavanju tog problema, kao i na njegove mnogobrojne primene.

Rezultati kvantitativnog merenja su pokazali da se među rečima u tekstu u značajnoj meri pojavljuju tri grupe entiteta koje su tek u novije vreme počele da privlače pažnju istraživača iz oblasti vezanih za automatsku obradu teksta. To su vlastite imenice, izrazi kojima se precizira datum ili vreme, i izrazi kojima se precizira neki novčani ili procentualni iznos.

Lingvistika uglavnom marginalizuje vlastite imenice, iako ih obilno koristi u svojim primerima. Međutim, istraživanja pokazuju da one predstavljaju više od 10% novinskih tekstova ([Coates 93]), ujedno i značajan deo "neprepoznatih reči"[1] u korpusu. Štaviše, u tekstu jednog romana, samo imena glavnih junaka mogu predstavljati više od 1% svih upotrebljenih reči ([Krstev 05a]).

Iako postoje rečnici vlastitih imena u papirnatom obliku, u računarskoj leksikografiji je, prilikom kreiranja morfoloških elektonskih rečnika, uglavnom zanemarivana potreba da se vlastita imena posebno obrade. Leksička analiza teksta pomoću računara korišćenjem takvih nepotpunih rečnika nailazi stoga na ozbiljne probleme. Razmotrimo sledeće primere:

(Pr1) *Devojka šeta pored reke*.

(Pr2) *Marina šeta pored Save*.

---

[1] Ovde pod "neprepoznatom rečju" u korpusu podrazumevamo reč koja se pojavila u korpusu, ali prilikom automatske analize teksta nije registrovana u rečniku

Lekseme u rečenici (Pr1) se analiziraju bez većih problema jer postoje u morfološkom elektronskom rečniku. Međutim, u (Pr2) se pojavljuju dve vlastite imenice (*Marina*, *Sava*) koje se ne mogu analizirati ukoliko ih rečnik ne sadrži. Osim toga, (Pr1) predstavlja uopšteni opis radnje, dok (Pr2) sadrži informacije koje su izuzetno značajne za analize višeg nivoa, pre svih semantičku. Time se ukazala potreba da se u računarstvu precizno definišu vlastita imena.

Sličan nedostatak precizne definicije je uočen i u slučaju datuma, izraza koji označavaju vreme na časovniku, novčanih iznosa, procentualnih iznosa i sl. Otuda je uveden pojam *imenovani entitet* kao zajednički imenitelj za elemente iz pomenute tri grupe, a sa njim i različiti prilazi u njihovoj identifikaciji u tekstu, analizi, i kategorizaciji kako bi se omogućila što efikasnija dalja analiza i obrada teksta.

Potreba za automatskim prepoznavanjem i klasifikovanjem imenovanih entiteta rezultirala je mnogobrojnim projektima i konferencijama, posebno od kraja osamdesetih godina prošlog veka pa sve do danas. Problem prepoznavanja i klasifikacije imenovanih entiteta je bitan za razne naučne discipline i oblasti: to je prvi od pet podzadataka **ekstrakcije informacija** (eng. Information Extraction, skr. **IE**) koja predstavlja specijalizovanu podoblast **pretraživanja informacija** (eng. Information Retrieval, skr. **IR**); s druge strane, u okviru **računarske lingvistike** (eng. Computational Linguistics, skr. **CL**), taj problem je od značaja za **obradu prirodnih jezika** (eng. Natural Language Prosessing, skr. **NLP**) i **automatsko prevođenje** (eng. Machine Translation).

Takođe, mnogobrojni projekti i konferencije su danas posvećeni imenovanim entitetima. Među projektima treba posebno pomenuti višejezične baze podataka vlastitih imena (npr. Prolex[2], o kome će biti više reči kasnije), kao i baze podataka geografskih entiteta i geografske informacione sisteme (eng. Geographic Information System, skr. **GIS**)[3]. Od konferencija su bitne ACE (Automated Content Extraction) i CoNNL (Conference on Computational Natural Language Learning). Pokrovitelj ACE-konferencija je NIST (National Institute of Standards and Technology, Nacionalni institut za standarde i tehnologiju). U svom programu, kao jedan od važnijih ciljeva, ACE-konferencije navode "pronalaženje i identifikovanje imenovanih entiteta" (Entity Detection and Tracking). CoNNL su godišnje konferencije, a 2002. i 2003. godine njihova tema je bila "Jezički nezavisno prepoznavanje i klasifikacija imenovanih entiteta" (Language-independent Name Entity Recognition and Classification). Međutim, pomenute konferencije svoje teme i ciljeve uglavnom vezuju za engleski jezik. Za srpski su značajne konferencije poput FASSBL[4] (Formal Approaches to South Slavic and Balkan Languages) i radionice BSNLP[5] (Balto-Slavonic Natural Language Processing) u okviru konferencije ACL (Association for Computational Linguistics).

Snažan temelj u tom smislu, pogotovo u definisanju standarda, postavile su Konferencije o razumevanju poruka (Message Understanding Conferences, ***MUC***).

Konferencije o razumevanju poruka (MUC)

Tokom poslednje dve decenije održano je sedam MUC-konferencija. Konferencije je organizovala DARPA (The Defense Advanced Research Projects Agency), agencija za

[2] http://www.cnrtl.fr/lexiques/prolex/

[3] Videti npr. Alexandria Digital Library na adresi http://www.alexandria.ucsb.edu/gazetteer

[4] Videti npr. http://dcl.bas.bg/Conference/home.html

[5] Videti npr. http://langtech.jrc.it/BSNLP2007/

odbranu američke vlade, koja, između ostalog, finansira i istraživačke projekte. Sve konferencije su imale takmičarski karakter: na njima su ocenjivani sistemi za ekstrakciju informacija koje su učesnici prethodno razvili. Neposredno uoči konferencije, učesnicima bi bio podeljen skup ulaznih testova koje su oni prosleđivali svojim sistemima za ekstrakciju informacija (posle prijema podataka, učesnici na svojim sistemima nisu smeli da vrše nikakve izmene u načinu obrade ulaza dok traje takmičenje). Izlaz koji su proizveli sistemi takmičara je potom evaluiran u odnosu na ručno pripremljene očekivane rezultate. Takođe, predstavnici organizatora konferencija, stručnjaci američke vlade, detaljno su opisivali probleme za koje su zainteresovani, dok su ostali učesnici konferencija, stručnjaci iz oblasti lingvistike, matematike i informatike, predstavljali svoja dostignuća u rešavanju tih problema. Na ovu temu postoji bogata bibiliografija u [Chinchor 97], [Chinchor 98], [Chinchor 99] itd[6].

Još tokom druge konferencije (MUC-2, 1989) se zadatak takmičara sveo na prepoznavanje određenih događaja u tekstu i popunjanje odgovarajućeg obrasca (formulara). Naime, za svaki prepoznati događaj u tekstu trebalo je automatski popuniti polja odgovarajućeg obrasca koja su označavala različite informacije o događaju: tip događaja (radnje), agenta, vreme i mesto dešavanja, posledice itd. Za takmičenje MUC-2 postojalo je 10 polja koje je trebalo automatski popuniti.

Takođe, na istoj konferenciji su razrađeni detalji u pogledu metrike koja će biti korišćena za evaluaciju sistema za ekstrakciju informacija takmičara. Naime, ako pretpostavimo da odgovarajući obrazac ima F polja koje treba popuniti, neka je K broj korektno popunjenih polja, a I broj polja koja nisu korektno popunjena. Može da se dogodi da sistem ne popuni pojedina polja usled čega je $K + I \leq F$. Tada se mogu definisati mere **odziv** (eng. **recall**) i **preciznost** (eng. **precision**) na sledeći način:

$$odziv = \frac{K}{F}$$

$$preciznost = \frac{K}{K + I}$$

Odziv ukazuje na to koliko je sistem sveobuhvatan tokom ektrakcije relevantnih informacija, dok preciznost ukazuje na tačnost s kojom sistem radi. Odnos između ovih mera je teško utvrditi[7], iako one pokazuju tendenciju da budu obrnuto proporcionalne (ako odgovor sistema teži da bude sveobuhvatan, to će uticati na smanjenje njegove preciznosti; i obrnuto, ako odgovor sistema teži da bude precizniji, to će negativno uticati na njegovu potpunost). U praksi se ove dve mere kombinuju kroz tzv. ***F*-meru**. Ako označimo preciznost, odziv i ***F***-meru redom sa ***P***, ***R***, tada se ***F***-mera može izraziti na sledeći način ([Jurafsky 00]):

$$F\text{-mera} = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}$$

Parametar β se koristi za održavanje ravnoteže između preciznosti i odziva. U slučaju kada je $\beta = 1$, preciznost i odziv imaju istu težinu (***F***-mera se tada svodi na njihovu harmonijsku sredinu); kada je $\beta > 1$, veću težinu dobija preciznost, dok za $\beta < 1$, veću težinu ima odziv.

---

[6] Videti npr. http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings/muc_7_toc.html#appendices

[7] Poređenjem kriva koje opisuju prosečnu preciznost dva sistema za dati nivo odziva, mogu se naći primeri da je jedan sistem precizniji za visok i nizak nivo odziva, dok je drugi sistem precizniji za srednji nivo odziva ([Jurafsky 00]).

Tokom poslednje dve MUC-konferencije (MUC-6 1995, MUC-7 1998) precizno je definisano šta su imenovani entiteti i postavljen je zadatak njihovog automatskog prepoznavanja. **Imenovani entiteti** (eng. **Named Entities**, skr. **NE**) su definisani kao vlastita imena i određeni izrazi za iznose ([Chinchor 98]). Zadatak prepoznavanja imenovanih entiteta sastoji se od tri podzadatka koja uključuju prepoznavanje:

- **imena** (eng. **Name**, npr. imena osoba, organizacija i lokacija),
- izraza kojima se opisuju datumi i vreme na časovniku,
- brojčanih izraza[8] (procentualnih i novčanih izraza).

Prema ovako postavljenim zadacima, sistem za prepoznavanje imenovanih entiteta mora proizvesti jedinstven, nedvosmislen rezultat za bilo koji relevantan niz karaktera u tekstu, čak i onda kad pravi odgovor nije očigledan na prvi pogled.

Navedene tri vrste entiteta se redom označavaju kao ENAMEX (imena), TIMEX (vremenski izrazi) i NUMEX (brojčani izrazi), a preuzete su iz smernica koje je dao TEI (Text Encoding Initiative) za kodiranje i razmenu tekstova u elektronskom obliku ([Sperberg 94])[9]. U smernicama se navodi da imena, datumi i brojevi mogu biti od posebne važnosti, kako istraživaču koji tretira tekst kao izvor za bazu podataka, tako i istraživaču koji je prvenstveno zainteresovan za jezičku analizu teksta.

U ([Chinchor 97]) je data detaljna MUC-specifikacija koja propisuje koji deo teksta spada u imenovane entitete, uz obilje definicija i primera. Prema ovoj specifikaciji, u imenovane entitete se ne ubrajaju:

- imena kolekcija i stvari nazvanih prema osobama (npr. Fildsova medalja),
- naslovi (npr. naslov knjige),
- pridevi izvedeni od imenovanih entiteta (npr. srpski),
- opšte imenice koje referišu na imenovane entitete (npr. preduzeće),
- brojevi koji nisu vremenski intervali, datumi, procenti ili novčani iznosi.

Prema dogovoru, jezik za obeležavanje MUC-tekstova je SGML (Standard Generalized Markup Language). Sistem za prepoznavanje imenovanih entiteta koji je učestvovao u takmičenju, trebao je da obeleži pronađene entitete u tekstu odgovarajućim elementima sa odgovarajućim atributima.

Na konferenciji MUC-7 su definisani sledeći dozvoljeni elementi i atributi:

1. Element ENAMEX. Značenja njegovih atributa su:

PERSON – lična imena i prezimena,

ORGANIZATION – preduzeća, državne institucije i druge organizacije,

LOCATION – imena politički ili geografski definisanih lokacija (gradovi, pokrajine, države, vode, planine itd.);

2. Element TIMEX. Značenja njegovih atributa su:

DATE – potpuni ili nepotpuni izrazi za datume,

TIME – potpuni ili nepotpuni izraz za vremena unutar dana;

3. Element NUMEX. Značenja njegovih atributa su:

MONEY – novčani izrazi,

[8] Oblik brojčanih izraza može biti numerički (26%) ili alfabetski (*milion dolara*).

[9] U trenutku pisanja ovih redova, aktuelne smernice za kodiranje i razmenu tekstova u elektronskom obliku se mogu naći na adresi http://www.tei-c.org.uk/P5/Guidelines/index.html (Burnard, Lou, and Syd Bauman, eds. *TEI P5*: *Guidelines for Electronic Text Encoding and Interchange*. TEI Consortium, 2005).

PERCENT – procenti.

Tako npr, imenovani entitet *New York* trebalo bi da bude obeležen sa **<ENAMEX TYPE="LOCATION">** New York **</ENAMEX>**.

MUC-konferencije su dale snažan podstrek intenzivnijem proučavanju imenovanih entiteta. Standardi koje su one postavile su danas opšeprihvaćeni u svetu.

Problem prepoznavanja imenovanih entiteta

Prepoznavanje imenovanih entiteta (Name Entity Recognition, skr. **NER**), prema savremenom shvatanju, podrazumeva postupak identifikacije imenovanih entiteta u tekstu, kao i njihovo eksplicitno obeležavanje. Tokom ili posle prepoznavanja, često se obavlja i **klasifikovanje imenovanih entiteta** (Name Entity Classification, skr. **NEC**) u unapred dogovorene odgovarajuće kategorije (imena osoba, organizacija, lokacija itd.). Kada se obavljaju obe radnje, tada se postupak naziva **prepoznavanje i klasifikacija imenovanih entiteta** (Name Entity Recognition and Classification, skr. **NERC**).

Problem prepoznavanja i klasifikovanja imenovanih entiteta je izuzetno složen, i može mu se pristupiti na razne načine. Neki od tih pristupa se oslanjaju na krajnje jednostavne ideje. Tako npr. primenjuju se jednostavne heurističke metode zasnovane na pravopisnoj tradiciji određenog jezika (npr. zapisivanje vlastitih imena velikim početnim slovom). Jedan drugi prilaz polazi od toga da postoje iscrpne liste imenovanih entiteta u rečnicima geografskih imena (eng. **gazetteer**), enciklopedijama, telefonskim imenicima itd, na osnovu kojih može da se napravi sveobuhvatna lista imena (osoba, organizacija, lokacija itd); upoređivanjem te liste sa zadatim tekstom identifikuju se odgovarajući imenovani entiteti u tekstu ([Mikheev 99]).

Obe pomenute ideje imaju svoje slabosti. Prvo, treba naglasiti da vlastita imena imaju morfološka (flektivna i derivaciona) svojstva jezika u kojima se pojavljuju. Stoga, iako postoje pozitivni rezultati primene ovih metoda na tekstove na engleskom jeziku, to nije slučaj sa jezicima sa bogatom morfologijom, posebno slovenskim. Drugo, čak i kada je u jednojezičnom tekstu moguće uočiti pojavu vlastite imenice korišćenjem pomenutih jednostavnih metoda, u višejezičnom tekstu je identifikacija koncepta predstavljenog tim imenom moguća samo izuzetno. Jedan ilustrativan primer daju prisvojni i relacioni pridevi: sintagma u srpskom *šopenovska tradicija* može da varira u engleskom kao *the Chopin tradition* ili *Chopinian tradition* ili *Chopin's tradition*, u poljskom *Chopinowska tradycja*, u francuskom *la tradition chopinienne*. Ovaj primer takođe pokazuje da se vlastita imena ne pišu uvek velikim početnim slovom ([Krstev 05a]).

Lista imenovanih entiteta nije dovoljna za klasifikaciju imenovanih entiteta: bez dodatne analize, program koji identifikuje imenovane entitete, koristeći samo poređenje liste i teksta, najverovatnije će pogrešno identifikovati imenovani entitet *Nikola Tesla* kao **osobu** u naslovu

Podrška Aerodroma "Nikola Tesla" Beograd predstavnici naše zemlje na "Evroviziji".[10]

umesto da prepozna da je u pitanju **organizacija** *Aerodrom "Nikola Tesla" Beograd.* Stoga su neophodne dodatne informacije kao unutrašnji i spoljašnji dokazi (eng. **internal and external**

[10] http://www.airport-belgrade.co.yu/code/navigate.php?Id=6

evidences) ([McDonald 96]). Unutrašnji dokaz se izvodi iz sekvence reči koje sačinjavaju imenovani entitet (npr. reči koje ukazuju da se radi o osobi poput *dr* ili *mr*, skraćenice koje ukazuju da se radi o organizaciji poput *KBC* (*Kliničko-bolnički centar*) ili *a.d.* (*akcionarsko društvo*), *d.d. (deoničarsko društvo*), *d.o.o.* (*društvo sa ograničenom odgovornošću*) itd). Spoljašnji dokazi se pak izvode iz konteksta u kome se javlja imenovani entitet i omogućavaju njihovu identifikaciju na osnovu toga što se imenovanim entitetima referiše na pojedinačne objekte koji imaju karakteristična svojstva i učestvuju u karakterističnim događajima (u gornjem primeru upravo *Aerodrom* predstavlja spoljašnji dokaz koji omogućava identifikaciju imenovanog entiteta). Npr. u rečenici

Republičko takmičenje učenika muzičkih škola Srbije održaće se od 26. marta do 5. aprila u muzičkim školama "Josip Slavenski", "Dr Vojislav Vučković".[11]

imamo primer i unutrašnjeg (*Dr*) i spoljašnjeg dokaza (*muzičke škole*), i kao što vidimo, spoljašnji dokaz je presudan u pravilnoj identifikaciji, tj. kao u prethodnom primeru ne radi se o osobi (što sugeriše unutrašnji dokaz), već o organizaciji (na šta ukazuje spoljašnji dokaz).

Treba primetiti i da sveobuhvatan popis imenovanih entiteta nije moguće sastaviti. Naime, broj imenovanih entiteta nije zanemarljiv, pri čemu postoje neograničene mogućnosti stvaranja novih; neki od njih vremenom izlaze iz upotrebe. Imenovanih entiteta, po pravilu, nema u opštim rečnicima, kako onim u papirnatom obliku, tako i elektronskim, namenjenim automatskoj obradi teksta. Istraživanja (videti npr. za geografske entitete [Mikheev 99]) su pokazala da je od kvantiteta mnogo bitniji kvalitet liste imenovanih entiteta; razvijene su aproksimativne metode koje koriste minimalnu listu imenovanih entiteta i na osnovu unutrašnjih i spoljašnjih dokaza "obezbeđuju dovoljno dobru preciznost i odziv". Ispostavlja se da minimalna lista pre svega zahteva prisustvo lokacija, kao i opšte poznatih imena, zato što se u njihovom kontekstu najčešće ne pojavljuju "suvišne" informacije (tipa *Beograd je glavni grad Srbije*) koje bi omogućile prepoznavanje i klasifikaciju imena. Druga istraživanja ([Maurel 04]), pokazuju da je zbog homografije i višečlanih imenica primena minimalnih lista imenovanih entiteta korektna samo u 50% slučajeva. Stoga, aproksimativne metode nisu dovoljne, te je neophodna sistematska obrada vlastitih imena u računarskoj leksikografiji.

Današnji sistemi koji pokušavaju da reše problem NERC-a koriste jedan od tri prilaza:

a) prilaz zasnovan samo na pravilima (eng **rule based**), ili
b) prilaz zasnovan samo na mašinskom učenju (eng. **machine learning**), ili
c) kombinaciju prethodna dva pristupa (tzv. hibridni sistemi).

Pristup zasnovan na pravilima više uvažava specifičnosti jezika i znanje o jeziku, dok se mašinsko učenje uglavnom oslanja na statistiku, tačnije na izračunavanje verovatnoće pojavljivanja kolokacija (niz reči ili pojmova koji se zajednički pojavljuju češće nego što bi se to očekivalo kod slučajnog zajedničkog pojavljivanja). Za prvu grupu sistema treba utrošiti više vremena za razvoj, konstrukcija pravila zahteva lingvističko znanjem o jeziku (lingviste), a za prilagođavanje sistema različitim domenima tekstova treba uložiti dodatni napor. S druge strane, sistemi koji koriste mašinsko učenje zahtevaju velike količine prethodno obeleženog teksta na kome sistem "uči", dok su tako dobijena "pravila" nečitljiva za čoveka (matrice verovatnoća). Rezultati evaluacije pokazuju da sistemi zasnovani na pravilima imaju bolje rezultate, a još bolje rezultate postižu hibridni sistemi, kombinujući pravila i statistiku (na poslednjoj MUC-konferenciji upravo je pobedio hibridni sistem LTG sa ***F***-merom 93.39).

[11] http://www.beograd.org.yu/cms/view.php?id=1269239&print=y

Budući da je kreiranje sistema za prepoznavanje i klasifikaciju imenovanih entiteta izuzetno obiman zadatak, pri čemu su mnogi istraživački timovi veoma napredovali u rešavanju izuzetno specifičnih podzadataka, postavlja se sledeće pitanje: da li uopšte ima smisla da se pri razvijanju takvog sistema za srpski jezik "počne od nule", tj. da li neki od postojećih sistema može da se, uz izvesna prilagođavanja, iskoristi za rešavanje pomenutog problema u srpskom jeziku?

S obzirom na ogroman uticaj koji ima engleski jezik, većina pomenutih sistema se obično razvija i testira na tom jeziku. Pri tome se često zanemaruju jezičke karakteristike koje ne postoje u engleskom. Kod jezika sa bogatom morfologijom (kao npr. kod slovenskih jezika, a samim tim i u slučaju srpskog) se ovakvim metodama obično dobijaju grube aproksimacije. Što se tiče sistema za jezike bliske srpskom (pre svega za slovenske jezike), izuzimajući hrvatski sistem NERC ([Bekavac 05]), i bugarski modul u okviru sistema GATE[12] ([Paskeleva 02]), takvih sistema praktično i nema.

Međutim, čak i u slučaju srodnih jezika, pravopisna pravila koja se tiču vlastitih imena mogu veoma da se razlikuju. Tipičan primer je odnos srpskog i hrvatskpg pravopisa. Ove razlike su ilustrovane na primerima toponima Njujork (New York) u [Pavlović 04]. Ovaj toponim se, u osnovi, zapisuje na isti način u zapadnoevropskim jezicima. Međutim, u srpskom jeziku, zbog bogate morfologije, isti toponim se beleži na daleko više načina. Navešćemo samo neke varijacije:

- *New York* se može zapisati korišćenjem različitih pisama (ćirilica ili latinica): *Њујорк* ili *Njujork*. Pri tom ova dva pisma koriste različite kodne sheme (ISO 646 IRV, ISO-8859-2 ili ISO-8859-5, Win CP 1250 ili Win CP 1251, Unicode itd). Štaviše, digraf *Nj* ima dve interpretacije u latiničnoj varijanti: kao jedna grafema koja odgovara ćiriličnom *Њ*, ili kao konsonantska grupa *n* + *j*. U kodnom rasporedu Unicode je ova višeznačnost sačuvana time što se slovo *Nj* može predstaviti kao digraf korišćenjem dva kodna mesta ili kao ligatura korišćenjem jednog kodnog mesta. Štaviše, veliko slovo *Nj* se može pisati korišćenjem oba velika slova (*NJ*) ili korišćenjem samo prvog slova kao velikog (*Nj*). Može se zaključiti da Unicode dopušta četiri mogućnosti da se zabeleži veliko latinično slovo *Nj* (isto važi i za slova *Lj* i *Dž*).
- *New York* se obično transkribuje s obzirom na fonetski princip srpskog pravopisa, mada se upotrebljava i originalni oblik[13]. Ovaj fenomen predstavlja i jednu od osnovnih razlika između srpske i hrvatske varijante srpsko-hrvatskog: u hrvatskom je praksa potpuno suprotna, tj. obično se navodi originalni oblik, a transkribovani oblik samo retko. Noviji hrvatski pravopis uvodi još jednu bitnu razliku između srpskog i hrvatskog: etimološki princip se u hrvatskom poštuje i u izvedenicama i u oblicima flektivne paradigme. Tako se relacioni pridev *njujorški* (koji se doskoro izražavao na isti način u oba jezika, poštujući fonetsko načelo) u hrvatskom može pisati i **newyor*ški***, i to u svim oblicima njegove flektivne paradigme.
- Na nivou fleksije, imenica *New York* ima svoju flektivnu paradigmu koja uključuje čak i oblik vokativa: *Njujorče*.
- Na nivou derivacije, ova imenica ima svoj relacioni pridev (koji ima 14 flektivnih oblika, pri čemu ne računamo moguće grafemske varijacije, kao ni sve moguće kombinacije osnovnih gramatičkih kategorija - 7 padeža, 2 broja, 3 roda, što daje

[12] GATE (General Architecture for Text Engineering) je nastao unutar NLP-grupe na Univerzitetu u Šefildu, a zamišljen je kao opšte razvojno okružje za obradu prirodnih jezika (videti npr. [Cunningham 02]).

[13] "Zašto je odbacio sve što je njegovom talentu i veštini darežljivo nudio grad New York, novi kontinent i potpuno novi život koji mu se pružao pod nogama?" (Momo Kapor, *Zoe*, Znanje, Zagreb, 1984)

7*3*2=42). Pri tom pravopis nalaže da se relacioni pridev piše malim slovima, što uspostavlja složenu derivacionu relaciju između sinonima "*dokovi Njujorka*" i "*njujorški dokovi*"[14]. Takođe se izvode i imena muških i ženskih stanovnika: *Njujorčanin* i *Njujorčanka*, a oni imaju svoja sopstvena flektivna i derivaciona svojstva (deminutive, augmentative, prisvojne i relacione prideve itd).

Pomenute varijacije su ilustrovane na korpusu savremenog srpskog jezika ([Korpus 06]). U ovom korpusu od približno 23 miliona reči, pojavilo se 68 različitih formi toponima *New York* sa ukupnom učestanošću 2455, ili 0,01% ukupnog broja prostih reči u korpusu. Od tog broja 68, imenica *Njujork* se pojavljuje u 14 različitih flektivnih i grafemskih varijanti, relacioni pridev *njujorški* u 36, a stanovnik Njujorka, *Njujorčanin*, u 9. Rezultate za imenicu *Njujork* prikazuje Tabela 1.

Oblici u jednini		Učestanost	Varijante
Njujork	**nominativ akuzativ**	**1086**	**Njujork NJUJORK NjUJORK Njujork njujork**
Njujorka	**genitiv**	**33**	**Njujorka NJUJORKA NjUJORKA**
Njujorku	**dativ lokativ**	**93**	**Njujorku NJUJORKU NjUJORKU Njujorku**
Njujorkom	**instrumental**	**11**	**Njujorkom NJUJORKOM**

Tabela 1

Uticaj korišćenih kodnih shema neutralizovan je internom kodnom shemom. Interna kodna shema koristi standardan ASCII kôd koji nema specijalne karaktere za naša slova sa dijakriticima, kao ni pojedinačne karaktere koji bi označavali digrafne simbole. Stoga je uveden zapis slova koji omogućava predstavljanje slova sa dijakriticima, kao i da se jednoznačno razlikuju digrafi od konsonantskih grupa. Ova kodna shema, korišćena u sistemu **aurora** ([Vitas 81]), preslikava dijakritičke karaktere (Tabela 2) i digrafe (Tabela 3) koji se koriste u srpskoj latinici.

Prepoznavanje i identifikacija imenovanih entiteta je od posebnog značaja za višejezične programe poput onih koji obavljaju paralelizaciju tektova[15] ili automatsko prevođenje jer predstavljaju značajan izvor srodnih reči (kognata, eng. **cognates**). Koliko je problem identifikacije kognata složen ilustruje upravo prethodni primer: *New York* je dvočlano ime, dok u srpskom *Njujork* predstavlja jednočlano ime; takođe, gore navedeni oblici flektivne i derivacione paradigme u srpskom dodatno usložnjavaju problem. Ista situacija je i sa već pomenutim primerom "*the docks of New York*" i "*njujorški dokovi*".

[14] Takve relacije se mogu razrešiti korišćenjem lokalnih gramatika (v. 1.2.4)

[15] Dva ili više tekstova, na više jezika, pri čemu je bitno da su svi tekstovi iste sadržine, dakle potekli od istog originalnog teksta, predstavljaju **paralelne** tekstove. Paralelni tekstovi se mogu upariti, tako što će se uspostaviti veze između odgovarajućih elemenata jednog i drugog teksta. To uparivanje se naziva **paralelizacija** tekstova i moguće ga je ostvariti na različitim nivoima (paragraf, rečenica, reč). Rezultat tog uparivanja je **paralelizovani** tekst.

Dijakritički karakteri			
Veliko slovo	**AURORA kod**	**Malo slovo**	**AURORA kod**
Č	**Cy**	**č**	**cy**
Ć	**Cx**	**ć**	**cx**
Đ	**Dx**	**đ**	**dx**
Š	**Sx**	**š**	**sx**
Ž	**Zx**	**ž**	**zx**

Tabela 2

Digrafi			
NJ, Nj	**Nx**	**nj**	**nx**
LJ, Lj	**Lx**	**lj**	**lx**
DŽ, Dž	**Dy**	**dž**	**dy**

Tabela 3

Eksperiment u kome je paralelizovan srpski i hrvatski prevod francuskog originala romana *Put oko sveta za 80 dana* Žila Verna ([Vitas 05e]) pokazao je da se već u prvoj rečenici pojavljuje priličan broj imenovanih entiteta. Tabela 4 ilustruje varijacije vlastitih imena pri prevođenju na hrvatski kao srodni jezik. U prvom redu je imenica *godina* izostavljena u hrvatskom, u drugom redu se promenio red reči, u trećem redu je izvršena transkripcija vlastitog imena u srpskom, ali ne i u hrvatskom (ali zato postoji flektivni nastavak za genitiv) itd. Nekoliko reči koje su ostale identične u sva tri teksta su predstavljene masnim slovima.

	francuski	**srpski**	**hrvatski**
1	l'année **1872**	Godine **1872,**	**1872**.
2	le numéro **7** de Saville-row	broj **7** u Ulici Sevilrou	U ulici Saville row (...) broj **7**
3	Burlington Gardens	Barlington Gardenz	Burlington-Gardena (*genitiv*)
4	Sheridan	Šeridan	Sheridan
5	**1816**	**1816.**	**1816.**
6	Phileas Fogg	Fileas Fog	Phileas Fogg
7	**Reform**-Club	**Reform**-kluba (*genitiv*)	Kluba **Reform**
8	de Londres	londonskog (*relacioni pridev*.)	u Londonu

Tabela 4

Iako vlastita imena prevashodno predstavljaju srodne reči, u mnogim slučajevima njihove grafičke varijacije mogu da onemoguće njihovo prepoznavanje približnim uparivanjem niski ([Krstev 05a]). Npr, ime prethodnog pape je *Giovanni Paolo II* u italijanskom, *Ιωάννης Παύλος II* u grčkom, *Jean Paul II* u francuskom, *Juan Pablo II* u

španskom, *John Paul II* u engleskom, *Jovan Pavle II* i *Јован Павле II* u srpskom (korišćenjem latinice i ćirilice), *Ivan Pavao II* u hrvatskom itd. Posebno je interesantan primer

Komunistička partija Francuske, ... čiji su generalni sekretari bili Moris Torez (1930-1964), Žorž Marše (1972-1994) i Rober I (1994-2001), je poslednjih dvadeset godina, iz godine u godinu, gubila na značaju.

Podvučena grafička varijacija je nastala transkripcijom vlastitog imena *Robert Hue*. Automatska morfološka analiza teksta koja bi koristila rečnik imena i prezimena (na kojoj se nalazi i pomenuto ime *Rober I*) i koja ne bi koristila rezultate dodatnih analiza (sintaksičke, semantičke itd), ne bi mogla da razreši da li je u pitanju prezime, veznik *i*, ili pak rimski broj *I* (jedan).

Varijacije vlastitih imena u srpskom utiču i na preciznost i odziv kod pretraživanja informacija. Tako Google, najpoznatiji pretraživač Interneta, za upit "*kuća*" pronalazi dokumente koji sadrže samo taj oblik, ali ne i ostale flektivne oblike poput "*kući*", "*kuću*" itd. Ovaj primer ilustruje teškoće izazvane morfološkim varijacijama[16]; međutim, mnogo ozbiljniji problemi su izazvani sinonimijom. Razmotrimo dva primera:

"Slonovi" i "lale" svoj meč igraju 16. juna u Štutgartu.
"Lale" minimalno, "Slonovi" nerešeno.

Prvi primer se odnosi na utakmicu sa Svetskog prvenstva u fudbalu održanog u Nemačkoj 2006, koju su odigrali fudbaleri Holandije ("Lale") i Obale Slonovače ("Slonovi"), dok se drugi primer odnosi na dva odvojena meča u kojima su učestvovale ove dve ekipe (minimalna pobeda reprezentacije Holandije i nerešen rezultat reprezentacije Obale Slonovače).

Koliko je problem složen pokazuje i primer da čak ni Google ne beleži nijedan dokument koji zadovoljava upit "*plate u republikama bivše Jugoslavije*", iako je iz upita potpuno jasno šta korisnika zapravo interesuje; naime, korisnik ne očekuje da u dokumentu pronađe bukvalno pojavljivanje te sintagme, već da dobije informaciju o platama u oblasti koju je nekad zauzimala SFRJ, dakle, u Srbiji, Makedoniji, Crnoj Gori, Hrvatskoj, Sloveniji, Bosni i Hercegovini.

Ovi problemi mogu se razrešiti proširivanjem upita ([Krstev 06], [Stanković 04], [Stanković 07]) korišćenjem morfološkog elektronskog rečnika (za morfološke varijacije) i **WordNet**[17]-a (za sinonimijske varijacije). Međutim, u slučaju sinonimije to može da dovede do "eksplozije" rezultata koji će uglavnom povećati odziv, ali će drastično smanjiti preciznost,

[16] Sačuvani upiti korisnika sistema za pretraživanje pokazuju da većina korisnika najčešće postavlja upit koristeći nominativ jednine, zaboravljajući pritom da relevantni dokumenti mogu sadržati ključne reči u morfološki modifikovanom obliku.

[17] WordNet je elektronska baza leksičkih relacija razvijena najpre za engleski jezik na Univerzitetu u Prinstonu 1985. godine (v. Fellbaum, Christiane (editor). 1998. *WordNet:An Electronic Lexical Database*. The MIT Press, Cambridge, MA). Danas se razvijaju verzije WordNet-a za mnoge evropske i svetske jezike, između ostalih i za srpski (v. npr. [Krstev 04a]). Moć WordNet-a leži u leksičkim relacijama nezavisnim od domena koje se uspostavljaju između sinsetova (eng. synset). Elementi sinset-a nisu lekseme već oblici reči iste vrste (imenice, glagoli ili prilozi) koji imaju neko zajedničko značenje; na taj način se sinsetom opisuje neki koncept, pojam, odnosno značenje, a WordNet u svojoj bazi uspostavlja između tih značenja leksičke relacije (hiperonimije i hiponimije, meronimije, holonimije, antonimije). Detalji o WordNet-u prevazilaze okvire ovog rada, videti dodatna objašnjenja i primer u sledećem odeljku.

jer dodati sinonimi najčešće imaju i druga, bliska značenja (polisemija) ili značenja koje nemaju nikakve veze sa onim značenjem koje se zapravo traži (homonimija). Tako, u slučaju fudbalera Holandije, kao rezultat možemo dobiti dokumente o cveću (*lale*), dokumente u kojima se pominju stanovnici Banata (Banaćani[18], *Lale*), ili pak dokument u kome se referiše na nekoga sa nadimkom *Lale*[19].

Prolex – leksikografska obrada imenovanih entiteta

Projekat Prolex[20] je započeo sredinom devedesetih godina prošlog veka ([Maurel 96]) sa relativno jednostavnim ciljem da proizvede bazu podataka francuskih toponima i naziva stanovnika, sa određenim lingvističkim informacijama za potrebe obrade prirodnih jezika. Glavni cilj projekta Prolex danas je da razvije višejezični rečnik vlastitih imena i relacija između njih. Projekat je podržan od programa RNTL-Technolangue[21] francuskog Ministarstva Industrije uz učešće kompanija Systran[22] (vodeći svetski proizvođač softvera za automatsko prevođenje) i Exalead[23] (razvija softver specijalizovan za pretraživanje Interneta). Poslednja verzija baze Prolex sadrži oko 323,000 jedinica i 55,000 relacijskih veza za francuski jezik. Zahvaljujući kompaniji Systran, u sistem je ugrađeno nekoliko višejezičnih listi (na engleskom, italijanskom, nemačkom, španskom, holandskom i portugalskom). Sem francuskog, postoje odgovarajući resursi i za nemački (13,000 jedinica povezanih sa odgovarajućim prevodnim ekvivalentima u francuskom), dok se odgovarajući resursi za engleski, poljski, ruski, srpski i bugarski tek razvijaju ([Tran 05a], [Maurel 06a], [Maurel 06b], [Maurel 07]).

Zamisao projekta Prolex je da opis vlastitih imena u višejezičnom kontekstu ne može da se svede na prostu konstrukciju višejezičnog e-rečnika, s obzirom na složenost semantičkih relacija koje ih povezuju. Odatle se izvodi zaključak da je u višejezičnom kontekstu pogodnije predstaviti vlastita imena kao ontologiju u smislu u kome je ovaj pojam definisao Gruber[24]. Analiza svojstava vlastitih imena pokazuje da takva ontologija mora imati bar četiri nivoa: dva jezički nezavisna nivoa: konceptualni i metakonceptualni, i dva jezički zavisna nivoa: lingvistički nivo i nivo instanci. Naknadno je pridodat i peti, jezički nivo ([Tran 04]).

Konceptualni nivo je organizovan oko pojma ***pivota*** (***konceptualno vlastito ime***), predstavljenog jedinstvenim identifikacionim brojem (ID). Pivot ima ulogu međujezičkog identifikatora, tj. omogućava povezivanje vlastitih imena koja predstavljaju iste koncepte u različitim jezicima. Konceptualna vlastita imena omogućavaju da se između pivota uspostave relacije na konceptualnom nivou, poput sinonimije, meronimije, predikacije. Relacija sinonimije se realizuje na različite načine; npr. *Francuska* i *Republika Francuska* su sinonimi

[18] Postoji tendencija da se ovaj naziv pogrešno koristi i u proširenom značenju "Vojvođanin"; ostali Vojvođani su zapravo Sremci i Bačvani.

[19] Npr. aligator, junak crtanog filma Wally Gator, se u srpskoj sinhronizovanoj verziji zove Lale Gator.

[20] http://www.cnrtl.fr/lexiques/prolex/

[21] http://www.recherche.gouv.fr/technolangue

[22] http://www.systransoft.com

[23] http://beta.exalead.fr/search

[24] Prema Gruberu, formalna reprezentacija celokupnog znanja zasnovana je na konceptualizaciji kao "apstraktnom, pojednostavljenom pogledu na svet koji želimo da predstavimo iz nekog razloga", a "*ontologija je eksplicitna specifikacija konceptualizacije*". Dakle, ontologija predstavlja neposredan i precizan opis koncepata. Na osnovnom nivou se opisuju *objekti*; zatim *klase* kao skupovi, kolekcije ili tipovi objekata; potom *atributi* kao svojstva, obeležja ili parametri koje objekti mogu da dele; i na kraju *relacije* kojima su objekti međusobno povezani (videti npr. Gruber T. R. 1995. *Toward Principles for the Design of Onthologies Used for Knowledge Sharing*. International Journal of Human-Computer Strudies 43: 907-928).

samo u političkom kontekstu (naime, neuobičajeno je da se kaže da smo "*proveli odmor na plažama Francuske republike*"). U slučaju sinonimije u dijahronijskom registru (npr. *Zair* je preimenovan u *Demokratska Republka Kongo* posle državnog udara) uvedena je posebna dijahronijska relacija *Is_Renamed* (*Je_Preimenovan*). Relacija meronimije između pivota predstavlja odnos dela i celine (npr. *Pariz* ⊂ *Francuska* ⊂ *Evropa*). Predikacija je relacija koja može da se uspostavi između dva vlastita imena koji su argumenti istog predikata (u logičkom smislu) [25]. Tipični primeri se sreću kod imenskih predikata (u gramatičkom smislu), npr, *Pariz je prestonica Francuske*, *Mocart je kompozitor Čarobne frule*. Ovde su *prestonica*, *kompozitor* instance logičkog predikata koji povezuje odgovarajuća vlastita imena, tj. **prestonica**(*Pariz*, *Francuska*) i **kompozitor** (*Mocart*, *Čarobna frula*). Relacija predikacije je inspirisana Mjelčukovom leksičkom funkcijom *Cap*[26].

Na ovom nivou je uspostavljena veza sa resursima *WordNet*-a. Svaki koncept opisan u *WordNet*-u ima jedinstven identifikator - međujezički indeks (eng. Inter-Lingual Index, skr. ILI), prvi put predstavljen u okviru projekta EuroWordNet ([Vossen 98]). Na taj način, svakom pivotu (konceptualnom vlastitom imenu) odgovara neka vrednost međujezičkog indeksa (ILI). U okviru engleskog *WordNet*-a uspostavljena je leksička hijerarhija koju su preuzeli kasnije izgrađeni primerci *WordNet*-a za druge evropske jezike. U toj leksičkoj hijerarhiji, zahvaljujući uspostavljenoj vezi pivot-ILI, može se definisati pozicija svakog (konceptualnog) vlastitog imena. Tako je koncept *Pariz*, predstavljen u engleskom *WordNet*-u kao sinset (skup sinonima) <Pariz, Grad svetlosti, francuska prestonica, glavni grad Francuske>, i njegov ILI je 0558236-n. Pozicija ovog koncepta u hijerarhiji WordNet-a izgleda ovako (simbol ↑ ukazuje na hiperonim tekućeg koncepta):

entitet
↑ ***lokacija***
↑ ***region***
↑ ***oblast, zemlja***
↑ ***centar, središte, srce***
↑ ***sedište***
↑ ***glavni grad***
↑ ***glavni grad države***
↑ ***Pariz, Grad svetlosti, ...***

Metakonceptualni nivo omogućava homogenu klasifikaciju vlastitih imena zasnovanu na super-tipu i tipu koji su pridruženi svakom vlastitom imenu, gde supertip klasifikuje vlastita imena na osnovu njihovih sintaksičkih i semantičkih svojstava, dok tip daje finiju klasifikaciju super-tipa. Npr, za super-tip *toponim* (ime mesta), mogući tipovi su *astronim* (ime zvezde; opštije ime nebeskog tela), *geonim* (geografsko ime, npr. ime kontinenta), *hidronim* (ime vodene površine, npr. reke, jezera itd), itd. Takođe se pravi razlika između istorijskih, religijskih i fiktivnih imena. Trenutno postoje tridesetak tipova i svega četiri super-tipa: *antroponimi* (imena ljudi, lična imena i prezimena), *toponimi* (imena mesta),

[25] Ovde će reč *predikat* biti korišćena u dva različita značenja: logičkom (predikat kao relacija) i gramatičkom (predikat kao deo rečenice). Da ne bi bilo zabune, kad god postoji dvosmislenost, u zagradi će biti izričito navedeno koje se značenje koristi.

[26] Mel'cuk I. 1984-I, 1988-II, 1992-III. *Dictionnaire explicatif et combinatoire du français contemporain*. Les presses de l'Université de Montréal.

ergonimi (imena apstraktnih ili konkretnih stvari koje je proizveo čovek, npr. *Linux*, *Odiseja*, *Titanik*, *Legija časti*, *Fiat*) i *pragmonimi* (imena događaja, npr. *Francuska revolucija*, uragan *Vilma*, *Olimpijske igre*, *Uskrs*). Kompletnu tipologiju poslednje verzije baze Prolex prikazuje Tabela 5, a detaljni primeri se mogu naći u [Maurel 07].

Lingvistički nivo opisuje realizacije vlastite imenice u posmatranom jeziku. Na ovom nivou se uvodi pojam **prolekseme** koji opisuje kanonski oblik (lemu) svih instanci nekog vlastitog imena. Svakoj proleksemi je pridružen identifikacioni broj (ID) odgovarajućeg jezika. Proleksema za *glavni grad Francuske* je *Paris* u francuskom i engleskom[27], *Pariz* i *Париз* u srpskom, *Paryz* u poljskom, i *Париж* u ruskom i bugarskom. Sa proleksemom su povezani aliasi (alternativna imena, "nadimci") koji opisuju pravopisne varijacije, skraćene oblike imena, akronime. Npr, u engleskom aliasi za *Ivo Andric* su *Ivo Andrich,* i *Ivo Andrics* itd. Takođe, kod toponima *Ljubljana* konsonantska grupa *lj* može biti zabeležena kao digraf i kao jedan karakter. Upravo su aliasi najčešći unutrašnji dokazi pri prepoznavanju imenovanih entiteta.

Vlastito ime						
Antroponim			Toponim		Ergonim	Pragmonim
Individualni	Kolektivni					
		Grupa		Teritorija		
Slavna ličnost Lično ime Patronim Pseudo-antroponim	Dinastija Etnonim	Društvo Ansambl Firma Institucija Organizacija	Astronim Zgrada Grad Geonim Hidronim Dromonim	Država Region Nadnacionalna	Objekat Produkt Misao Plovni objekat Delo	Katastrofa Događaj Svetkovina Istorija Meteorologija

Tabela 5

Na ovom nivou se takođe uspostavljaju relacije između prolekseme i njenih derivacionih oblika, kao i relacije između njenih aliasa i njihovih derivacionih oblika. Primeri ovih tipova relacija su imena muških i ženskih stanovnika toponima, prisvojni i relacioni pridevi izvedeni od toponima i stanovnika, itd. Npr, u engleskom *Parisian* je stanovnik toponima *Paris*, dok u srpskom, *Parižanin* je muški stanovnik toponima *Pariz* (sa aliasom *Parizlija*), a *Parižanka* je ženski stanovnik (stanovnica) toponima *Pariz*.

Na lingvističkom nivou se definišu i druge relacije, kao što su:

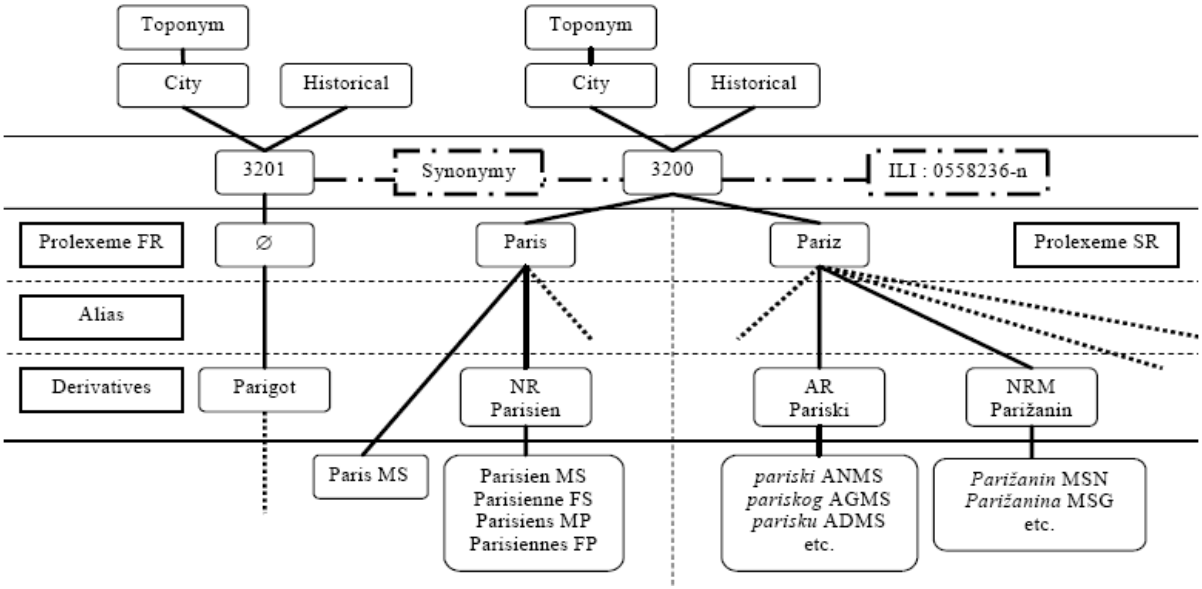
a) Relacija Blark (skr. od eng. Basic LAnguage Ressources Kit, osnovni alati za jezičke resurse) koja povezuje vlastito ime sa nekim vremenskim periodom ili regionom sa ciljem da ukaže na njegovu relevantnost u odnosu na taj period, odnosno region (Cucchiarini et al. 2000).
b) Relacija antonomazija, kojom se vlastita imenica pretvara u zajedničku. Npr, u francuskom je vlastita imenica *kleenex* postala zajednička imenica koja označava papirne maramice, u engleskom *biro* označava naliv-pero, a u srpskom *žilet* (dobijen transkripcijom od *Gillette*) predstavlja uobičajen naziv za nožić za brijanje.

[27] Homografi u različitim jezicima su duplirani, tj. svaki jezik ima svoju proleksemu.

c) Relacija sortiranja pruža informacije o tome kako da se klasifikuju višečlane vlastite imenice. Naime, mnogi rečnici uređuju višečlane vlastite imenice tako što prave inverziju[28] njihovih konstituenata ([Tran 05b]).

Nivo instanci sadrži lingvistički opisane flektivne oblike vlastitih imena (npr, navedena su njihova flektivna svojstva). Odnos između lema na lingvističkom nivou i njihovih oblika na nivou instanci može se definisati preko koda flektivne klase. Kod mnogih evropskih jezika ovaj kôd odgovara kodu koji je pridružen svakoj lemi u rečniku tipa DELA[29].

Kao primer višejezične ontologije vlastitih imena naveden je pojednostavljen model implementacije vlastite imenice *Pariz* u francuskom i srpskom (Slika 1).



Slika 1

Na konceptualnom nivou, jedinstveni pivot (konceptualno vlastito ime) odgovara vlastitoj imenici *Paris*, predstavljenoj vrednošću 3200 identifikatora ID. Relacija ILI ga povezuje sa *WordNet*-om (odgovarajući međujezički indeks ILI ima vrednost 0558236-n), i tako specificira njegovu poziciju u hijerarhiji hiperonima i hiponima. Njegov tip na metakonceptualnom nivou je *Grad* sa super-tipom *Toponim*.

Na lingvističkom nivou, konceptualna vlastita imenica ID=3200 realizuje se u francuskom kao proleksema *Paris*, bez aliasa, ali sa izvedenicom *Parisien* (stanovnik *Pariza*). *Parisien* je relacijom sinonimije povezan sa leksemom *Parigot* (stanovnik *Pariza* u francuskom slengu). Iako su sinonimi, postoje sociolingvističke razlike između ovih leksema, a razlikuju se i po komunikacijskim situacijama u kojima se koriste. Treba naglasiti da je proleksema druge lekseme *Parigot* prazna (ø), a vrednost identifikatora ID (3201) njenog konceptualnog vlastitog imena se razlikuje od vrednosti koju za ID ima *Parisien* ((3200).

[28] Npr, *Palanka*, *Bela* umesto *Bela Palanka* itd.

[29] O tome videti detaljnije u odeljku 1.1.2

Na nivou instanci postoji samo jedna instanca (*Paris*) koja odgovara proleksemi *Paris*, označena kao imenica muškog roda (*M*) u jednini (*S*), dok četiri instance odgovaraju izvedenici *Parisien*, definišući njenu flektivnu paradigmu (*F* označava ženski rod, a *P* množinu). Proleksema *Paris* ima, kao izvedenicu, relacioni pridev *parisien* sa svojim sopstvenim instancama (Slika 1 ne prikazuje ovu izvedenicu i njene instance).

U srpskom, proleksema koja odgovara istom konceptualnom imenu sa ID=3200 je *Pariz*, a njen alias je ćirilični zapis *Париз*. Derivacioni procesi u srpskom su složeniji nego u francuskom. Osim relacionog prideva *pariski*, i imena za muškog stanovnika, *Parižanin*, postoji poseban oblik za ženskog stanovnika (stanovnicu), *Parižanka*. Na osnovu naziva za stanovnike, izvedeni su relacioni pridev *parižanski* (koji se odnosi na stanovnike *Pariza*), i prisvojni pridev *Parižaninov* (koji pripada *Parižaninu*) i *Parižankin* (koji pripada *Parižanki*) – ove izvedenice nisu prikazane na slici. Na nivou instanci je prisutan skup flektivnih oblika koji odgovaraju proleksemi, njenim aliasima, i svim izvedenicama; korespondencija između flektivnih oblika i lema je uspostavljena korišćenjem pogodnog regularnog izraza.[30]

Treba naglasiti da sama derivacija u srpskom ima dva nivoa: na prvom nivou su oblici izvedeni neposredno od prolekseme ili njenih aliasa, dok su na drugom nivou oblici koji su sistematski proizvedeni od prethodnih izvedenica mehanizmom regularne (strukturne) derivacije[31]. Kao u francuskom, i u srpskom postoji alternativni naziv za stanovnika Pariza, *Parizlija*, koji odgovara francuskom *Parigot*, te je njegov konceptualni ID takođe 3201. Takvi oblici ne postoje ni u nemačkom ni u engleskom.

S obzirom na na višejezičnu dimenziju Prolex-a, za kodiranje unosa se koristi Unicode. Model je implementiran kao relaciona baza podataka, a XML sheme se koriste za razmenu podataka. Od glavnih budućih primena takve baze podataka treba spomenuti prevođenje, kako automatsko, tako i prevođenje pomoću računara (eng. computer aided translation) pretraživanje informacija, paralelizaciju tekstova, kao i proveru pravopisa (eng. spelling checking). Po završetku, baza bi trebala da pokrije većinu evropskih jezika, a poslednja verzija je dostupna na adresi http://www.cnrtl.fr/lexiques/prolex/.

Umesto zaključka

Iz svega gore navedenog može se zaključiti da prilikom prepoznavanja i klasifikacije imenovanih entiteta, posebno imena, izuzetno značajnu ulogu imaju raspoloživi resursi, ali i da nije bitna samo količina informacija koju oni poseduju već i način na koji su te informacije organizovane i međusobno povezane raznim leksičkim i semantičkim relacijama.

Ovaj rad je inspirisan višejezičnom bazom vlastitih imena Prolex. Njegova prvobitna motivacija je bila da pojednostavi kreiranje i održavanje resursa za srpski jezik u okviru baze Prolex; a krajnji cilj je da se započne opis klasa regularne derivacije u srpskom jeziku (pre svega za vlastita imena), nezavisno od baze Prolex ili bilo koje druge implementacije.

[30] Detalji, uključujući i primere, se mogu naći u odeljcima 1.1.2, 1.2.2.1 i 1.2.3.2.

[31] Definicija i primeri regularne derivacije su opisani u glavi 2.

1

KONAČNI AUTOMATI U OBRADI PRIRODNIH JEZIKA

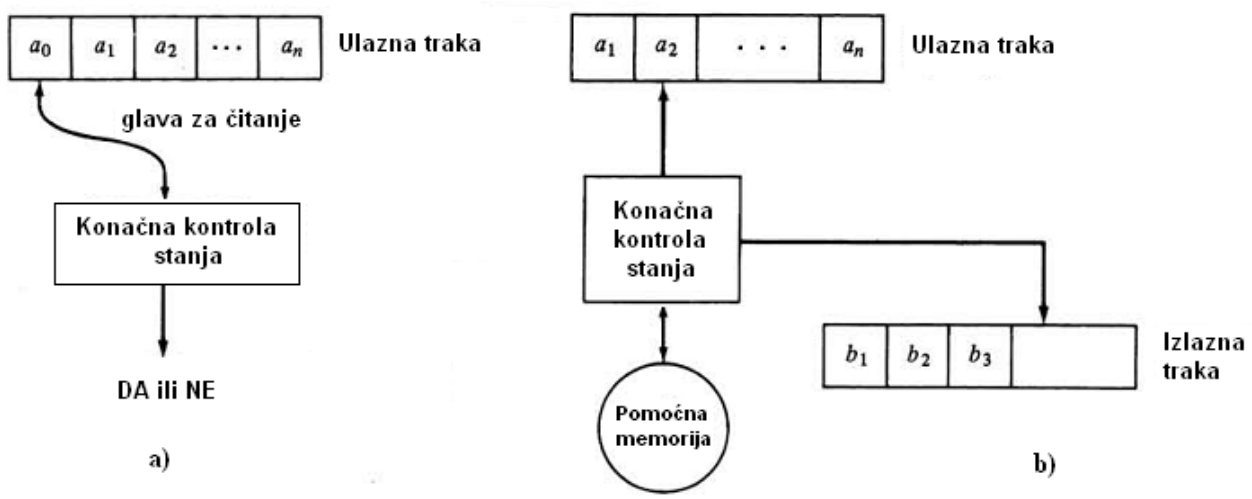
U ovom delu razmatra se formalizam konačnih automata i konačnih transduktora, ali iz ugla obrade prirodnih jezika. Posebno će biti istaknuta uloga acikličnih konačnih automata (i transduktora) u implementaciji savremenih morfoloških elektronskih rečnika. Takođe će biti ukazano na očiglednu prednost u korišćenju istog formalizma (konačnih automata i transduktora) za modeliranje morfološkog i sintaksičkog modula u okviru aplikacije namenjene obradi prirodnog jezika.

Tjuringova mašina, model algoritamskog izračunavanja koji se smatra jednim od temelja savremenog računarstva, inspirisao je 1943. godine model neurona[32] umnogome nalik današnjem konačnom automatu. Na osnovu tog modela, pedesetih godina prošlog veka Klini je definisao formalizam **konačnih automata** (eng. Finite-state automata, skr. **FSA**), kao i formalizam **regularnih izraza,** i pokazao njihovu ekvivalentnost[33]. Konačne automate možemo zamisliti kao mašine sa konačno mnogo stanja, koje prepoznaju da li zadata niska pripada određenom skupu niski (Slika 1.1a). Ulaz mašine je proizvoljna niska, a njen izlaz je odgovor "da" ili "ne" u zavisnosti od toga da li je automat zadatu ulaznu nisku prepoznao ili

[32] Videti npr. McCulloch, W. S. and Pitts, W., *A logical calculus of ideas immanent in nervous activity*, Bulletin of Mathematical Biophysics, pp.115-133. Reprinted in Neurocomputing: Foundations of Research, ed. by J. A. Andersen and E. Rosenfeld, MIT Press 1988.

[33] Klini je zapravo definisao formalizam regularnih skupova; regularni izrazi predstavljaju matematičku notaciju za opis regularnih skupova. Tvorci operativnog sistema Unix, pre svih Ken Tompson, su ugradili tu notaciju u editore i razne druge alate za rad sa tekstom, što ju je učinilo izuzetno popularnom. Danas su regularni izrazi sastavni deo najznačajnijih programskih jezika i alata za obradu prirodnog jezika. Međutim, njihova sveprisutnost je dovela do varijacija u pogledu sintakse. Stoga će u odeljku 1.2.1 biti definisani samo regularni skupovi, a izuzimajući povremene primere, precizna sintaksa regularnih izraza (koja je od značaja za ovaj rad) biće opisana u odeljcima 3.2.2 i 4.2.2.

ne. Samo prepoznavanje teče tako što mašina počinje rad u nekom početnom stanju, "čita" nisku sleva nadesno simbol po simbol i, u zavisnosti od pročitanog simbola, eventualno menja stanje u kome se nalazi. Jedino ako je cela niska "pročitana", a automat se našao u završnom stanju, onda je niska prepoznata. Detalji o teoriji konačnih automata i njihovoj implementaciji se mogu naći u [Aho 72], [Vitas 06]. Njihovu formalnu definiciju navešćemo u odeljku 1.2.2.



Slika 1.1

Iako je formalizam **konačnih transduktora** (eng. Finite-state transducers, skr. **FST**) matematički blizak formalizmu konačnih automata, razvoj ta dva modela je imao donekle različit put. Pedesetih godina prošlog veka pojavile su se dve ekvivalentne verzije konačnih transduktora: najpre Murova mašina ([Moore 56]), a zatim, kao njeno proširenje i sinteza sa radom Hafmana ([Huffman 54]), i Milijeva mašina ([Mealy 55]). To su bile mašine sa konačno mnogo stanja, koje su, pored azbuke ulaznih simbola (prisutne i kod konačnih automata), uvele i azbuku izlaznih simbola. Milijeva mašina koristi **tabelu prelaza** (eng. **state-transition table**) koju je prvi definisao Hafman, baveći se modeliranjem rada sekvencijalnih kola.

Za razliku od konačnih automata koje zamišljamo kao "mašine za prepoznavanje", konačne transduktore možemo zamisliti kao "mašine za prevođenje". Konačni transduktor na osnovu proizvoljne ulazne niske generiše na izlazu novu nisku, "prevod" ulazne niske (Slika 1.1b). Otuda su transduktori dobili svoje ime kao "najjednostavniji prevodioci". Najočiglednija primena transduktora je da prepoznatoj niski pridruži odgovarajuću informaciju. Kakva će informacija biti pridružena zavisi od specifične primene: to može biti informacija o morfološkim svojstvima niske (npr. vrsta reči), ali i daleko složeniji oblik informacije (npr. “prevod” na drugi jezik). O konačnim transduktorima se može više naći u [Aho 72], [Roche 97]. Formalnu definiciju konačnih transduktora navešćemo u odeljku 1.2.3.

Zbog svoje jednostavnosti i mogućnosti da se efikasno implementiraju, konačni automati i transduktori su se pokazali naročito pogodnim za rešavanje dva slična problema: leksičke analize u kompilaciji programskih jezika i **morfološke analize teksta na prirodnom jeziku**. Prvi problem je praktično rešen tako što se pri konstrukciji programskog jezika vodi računa o tome da se njegove leksičke klase mogu relativno lako opisati regularnim izrazima. Na osnovu svakog regularnog izraza može se automatski konstruisati minimalni konačni automat koji prepoznaje sve niske opisane tim regularnim izrazom i nijednu drugu nisku. Kombinovanjem automata koji prepoznaju pojedinačne leksičke klase programskog jezika

može se automatski konstruisati odgovarajući leksički analizator. To je jedna od primena konačnih automata gde su postali nezamenljivi.

Drugi navedeni problem, morfološka analiza teksta na prirodnom jeziku, je kudikamo teži, i pojavljuje se u nizu drugih oblasti, poput automatskog prevođenja, računarske leksikografije, pretraživanja i ekstrakcije informacija, veštačke inteligencije i sl. Automatska analiza prirodnih jezika je od značaja za sve one koji se bave rečima i tekstom, bilo da je to u kontekstu učenja stranog jezika, književnih studija, lingvističke analize, istorijskih istraživanja, elektronskog poslovanja ili marketinga, i mnogih drugih zanimljivih oblasti naučnog i praktičnog rada.

Iako su pomenuti problemi leksičke i morfološke analize slični, jezici na koje se odnose se bitno razlikuju: programski jezici su veštački stvoreni i njihova struktura (morfološka i sintaksička) je dovoljno "jednostavna" (o čemu se mora voditi računa kad se dizajniraju). Njihova semantika je takva da proizvoljna konstrukcija programskog jezika uvek ima tačno jedno značenje. S druge strane, iako je donekle podložna pravilima, struktura prirodnih jezika je daleko složenija i dosadašnji modeli su samo grube aproksimacije. Stoga se može reći da problem morfološke analize teksta na prirodnom jeziku još uvek nije rešen na zadovoljavajući način.

Upravo zbog težine problema i raznovrsnih oblasti za koje je značajan, rešavanju ovog problema se pristupalo na razne načine. Mnogi od tih pokušaja su se pokazali ekvivalentni, ili su predstavljali nadgradnju dotadašnjih pokušaja. Danas možemo razlikovati nekoliko osnovnih pristupa morfološkoj analizi, o kojima će biti više reči u odeljku 1.1. Iako će svaki od tih pristupa biti ukratko razmotren, posebna pažnja biće posvećena samo jednom, i to onom koji za analizu koristi isključivo morfološke elektronske rečnike (v. odeljak 1.1.2); za implementaciju tih rečnika se koriste upravo konačni automati i transduktori. Trenutne implementacije ogromnih elektronskih rečnika pomoću konačnih automata imaju brzinu pretraživanja od milion do nekoliko miliona reči u sekundi, a algoritmi za konstruisanje tako velikih automata su izvanredno unapređeni poslednjih godina ([Friburger 04]). Od 1989. godine, kombinovanjem raznih pristupa, priličan broj akademskih institucija i komercijalnih poduhvata popločao je put sve efikasnijem i potpunijem leksičkom opisu mnogih jezika. Npr, pomenimo, sem konzorcijuma RELEX, kompanije Teragram, Xerox, Lingsoft, Connexor ili Exorbyte i mnoge druge organizacije koje su uvidele vrednost efikasno predstavljenih leksičkih resursa kao najbolje polazne tačke za naprednije primene računara na polju obrade prirodnih jezika.

Ovaj rad će se fokusirati upravo na primenu konačnih automata i transduktora u računarskoj leksikografiji; ali pre nego što taj deo detaljnije izložimo, samo ukratko ćemo se osvrnuti na neke druge bitnije primene konačnih automata i transduktora, a koje su vezane i za upravo spomenuti problem morfološke analize teksta na prirodnom jeziku.

Od trenutka skidanja vela vojne tajne sa elektronskih računara četrdesetih godina prošlog veka, pa sve do sadašnjih dana, živo je aktuelan **problem automatskog prevođenja sa jednog prirodnog jezika na drugi**. Taj problem još uvek nije rešen na zadovoljavajući način iako se tokom poslednjih šezdeset godina pojavilo nekoliko generacija sistema za automatsko prevođenje. Prva dva nivoa analize tih sistema odgovaraju leksičkoj i sintaksičkoj analizi u teoriji kompilacije, a to su nivo morfološke analize i nivo sintaksičke analize. Konačni automati i transduktori su najpre našli "živu" primenu u morfološkoj analizi, a nešto kasnije su započeli i prvi pokušaji sintaksičke analize pomoću konačnih transduktora.

I pre pojave Internet-a, državne agencije širom sveta suočili su se sa problemom skaniranja i obrade ogromnih količina teksta dobavljenog iz različitih izvora. Internet je svojim resursima dostupnih informacija samo produbio problem **pretraživanja i ekstrakcije željenih informacija**. Tako su se pojavili specijalizovani sistemi za pretraživanje i ekstrakciju

informacija koji u svojoj osnovi koriste konačne automate i transduktore. Tipičan primer takvog sistema je FASTUS (Finite State Automata-based Text Understanding System, sistem za razumevanje teksta zasnovan na konačnim automatima) ([Jurafsky 00]). Uprkos pretencioznom imenu, to je zapravo sistem za ekstrakciju informacija iz proizvoljnog teksta, a ne sistem za razumevanje teksta. FASTUS radi kao niz uzastopno primenjenih konačnih automata, odnosno transduktora. Njegove primene se svode na anotiranje delova teksta koji su od interesa (npr. imena ljudi ili preduzeća), ili pak na popunjavanje obrazaca (formulara) ekstrahovanim informacijama koje se potom mogu iz formulara uneti u relacionu bazu podataka. Osim FASTUS-a, mnogi drugi sistemi za ekstrakciju informacija koriste konačne automate i transduktore, i to tako da su ili zasnovani na njima (uz izvesna proširenja), ili ih koriste samo u fazi predobrade, ili ih kombinuju sa drugim komponentama zasnovanim na stablima odlučivanja.

Definitivno, najprivlačniji primeri primene konačnih automata i transduktora su upravljači dijalozima (eng. **dialogue manager**) kod konverzacionih agenata (eng. **conversational agent**). Konverzacioni agent je program koji komunicira sa korisnicima na prirodnom jeziku i sposoban je da na inteligentan način odgovara na korisničke zahteve. Savremeni primeri konverzacionih agenata uključuju informacione sisteme avionskih prevoznika koji odgovaraju na pitanja o letovima i omogućavaju rezervaciju avionske karte. Tu su informacioni sistemi koji predstavljaju "turističke vodiče", odgovarajući npr. na pitanja o lokalnim restoranima; ili telefoni koji imaju prirodno-jezički interfejs za korišćenje elektronske pošte i kalendara itd. Pored interakcije agenta i korisnika koja se odvija kucanjem teksta, razvijeni su i konverzacioni agenti kod kojih se interakcija obavlja verbalno, korišćenjem prepoznavanja i generisanja govora. Upravljač dijalozima je komponenta konverzacionih agenata, koja kontroliše tok dijaloga, odlučujući kako treba odgovoriti korisniku, koja pitanja mu treba postaviti, kao i u kom trenutku treba to uraditi. Najjednostavniji upravljači dijalozima su zasnovani na konačnim automatima; stanja automata odgovaraju pitanjima koje upravljač dijalozima postavlja korisniku, dok prelazi iz jednog u druga stanja automata odgovaraju akcijama koje će se izvršiti zavisno od toga šta je korisnik odgovorio.

1.1 Morfološka analiza korišćenjem FSA i FST

Iako termin morfološka analiza ima mnogo šire značenje, ovde ćemo pod tim podrazumevati isključivo morfološku analizu računarski pohranjenih prirodno-jezičkih dokumenata. Pre definicije i opisa najvažnijih tipova morfoloških analizatora, precizno uvodimo nekoliko pojmova.

Morfologija predstavlja deo nauke o jeziku koji se bavi rečima – njihovim vrstama, oblicima i tvorbom (građenjem, izvođenjem), a često se isti termin upotrebljava da označi sistem oblika jednog jezika ([Stanojčić 99]). Morfologija određuje reči polazeći od njihovog grafemskog sastava, od njihovog značenja (pojedinačno značenje reči) i od njihove službe (funkcije) u jeziku i u rečenici (gramatičko značenje reči). **Flektivna** morfologija se bavi vrstama i oblicima reči, dok se **derivaciona** (tvorbena, leksička) morfologija bavi tvorbom reči (detaljnije o tome u glavi 2).

Morfeme su najmanje jezičke jedinice koje su nosioci značenja (leksičke morfeme) ili gramatičke službe u rečenici (gramatičke morfeme). Reč (u lingvističkom smislu) se sastoji od jedne ili više morfema. Konkretne grafemske realizacije morfema se nazivaju **morfi**. U zavisnosti od konteksta, jedna te ista morfema (npr. *iz-*) može konkretno da se ostvari na

različite načine (kao *iz-* u *izdati*, kao *is-* u *ispitati*, kao *iš-* u *iščitati*, kao *iž-* u *iždžikljati*), i svako takvo različito (kontekstualno uslovljeno) ostvarenje morfeme se naziva **alomorf**[34].

Prirodno-jezički dokument predstavlja tekst u opštem, lingvističkom smislu, kreiran konkretnim prirodnim jezikom i zapisan određenim pismom i u skladu sa određenim pravilima. To je apstraktna jedinica jezika čijom interpretacijom se može dobiti nameravano značenje. Njegov osnovni cilj je da prenese određenu informaciju namenjenu međuljudskoj komunikaciji.

Elektronski tekst je zapis prirodno-jezičkog dokumenta u računarskom obliku. Računarski tekst je proizvoljna niska nad konačnom azbukom čiji su simboli predstavljivi u računaru. Ta azbuka se naziva skup karaktera ili karakterski skup, a njeni elementi karakteri.

Simboli pisma kojim se realizuje prirodno-jezički dokument, kao i karakteri kojim su ti simboli predstavljeni u računaru, mogu da se podele na alfabetski skup (čije elemente dogovorno nazivamo slovima) i separatorski skup (koga čine beline, znaci interpunkcije, cifre i bilo kakvi neslovni simboli).

Pod **tekstuelnim** (**formalnim**) rečima podrazumevaćemo konstituente prirodno-jezičkog dokumenta, gde konstituenti predstavljaju alfabetske niske u smislu teorije formalnih jezika, međusobno razdvojene karakterima separatorskog skupa[35].

S druge strane, **leksička** reč predstavlja element rečnika (leksikona), uključujući informacije pridružene tom elementu (leksičko značenje, pridružena vrsta reči u skladu sa tim značenjem, vrednosti prisutnih gramatičkih kategorija (rod, broj), oznake realizovanih oblika (padež, lice) itd). Pridružene informacije (**morfosintaksički opis**) ističu da se leksička reč može upotrebiti kao deo rečenice (sintaksička jedinica) i da zato ima dati gramatički oblik (morfološke karakteristike). Ovde pod rečnikom ne podrazumevamo uobičajene rečnike u papirnatom obliku; apstraktni element takvih rečnika je **leksema**, jezička jedinica leksičkog nivoa sa svim njenim gramatičkim oblicima (**paradigma**) i frazeološkim proširenjima ([Bugarski 95]), predstavljena samo svojim kanonskim oblikom (**lema**)[36]. U ovom radu rečnik predstavlja skup **morfosintaksički reči (**[Stanojčić 99]**)**, gde je morfosintaksička reč uređen par oblika (leksička reč, morfosintaksički opis) [37].

Morfološka analiza elektronskog teksta (eng. **morphological parsing**) predstavlja proces kojim se konstituentima tog teksta pridružuju odgovarajuće informacije (morfosintaksički opis) korisne za dalju obradu (posebno za sintaksičku analizu). Faze morfološke analize su:

- leksikalizacija (leksematizacija), tj. prepoznavanje i izdvajanje tekstuelnih (formalnih) reči kao niski simbola koje su se pojavile u tekstu.
- lematizacija, tj. povezivanje izdvojenih formalnih reči sa potencijalnim pripadajućim jezičkim informacijama (morfosintaksički opis), uključujući i lemu.
- otklanjanje višeznačnosti, tj. proces izbora samo onih jezičkih informacija (morfosintaksičkih opisa) koje mogu biti relevantne za konkretnu realizaciju date formalne reči. Npr, za tekstuelnu reč *knjigama* relevantne su sledeće tri morfosintaksičke reči: {*knjigama*, *knjiga*, imenica ženskog roda u dativu množine}, {*knjigama*, *knjiga*, imenica ženskog roda u instrumentalu množine}, {*knjigama*, *knjiga*, imenica ženskog roda u lokativu množine}; međutim, zavisno od konteksta u

[34] Tako *iz-*, *is-*, *iš-*. *iž-* predstavljaju alomorfe morfeme *iz-*

[35] Granica između alfabetskog i separatorskog skupa nije uvek jasna, jer pojedine reči sadrže i separatore (npr. sever-severozapad, G7+1)

[36] Npr, leksema RADITI ima za lemu infinitiv *raditi*, a ostali oblici paradigme su *radiću, radili, radeći, radi* itd.

[37] Npr, {*knjigom, knjiga, imenica ženskog roda u instrumentalu jednine*} je jedan element rečnika.

kome se pojavljuje tekstuelna reč *knjigama*, samo jedna od navedenih morfosintaksičkih reči je odgovarajuća. Npr, u rečenici *On je posvećen knjigama* treba pridružiti prvu morfosintaksičku reč, u rečenici *On se druži sa knjigama* treba pridružiti drugu morfosintaksičku reč, dok u rečenici *On traži u knjigama* treba pridružiti treću morfosintaksičku reč.

Daljoj računarskoj obradi posle morfološke analize prirodno-jezičkog dokumenta često nije potrebna prethodno precizno određena morfološka struktura tekstuelnih reči; umesto toga, ispostavlja se da je dovoljno da se prepoznaju i formalizuju procesi morfološke prirode na osnovu kojih se može uspostaviti relacija između leksičkih reči, takva da omogućava da se algoritamski "izračuna" veza između tekstuelnih i leksičkih reči. U [Vitas 93b] su detaljno opisani razni pristupi modeliranju pomenutih morfoloških procesa.

Razmotrićemo samo nekoliko osnovnih tipova morfološke analize elektronskog teksta[38], u čijoj implementaciji su iskorišćeni upravo konačni automati i transduktori. Navedeni tipovi analize se razlikuju po načinu predstavljanja suštinskih komponenti jezičkog sistema: gramatičkih pravila (kao strukturnog okvira jezika) i rečnika (kao sadržinskog dela jezika).

1.1.1 Kimov model

Prvi tip koristi gramatička pravila pri svođenju tekstuelne reči na leksičku reč. Tipičan primer za ovaj tip morfološke analize je tzv. **dvorazinski morfološki model** (eng. Two-level morphology) opisan u [Koskenniemi 83]. Prema svom autoru (Kimmo Koskenniemi) se još naziva **Kimov model**. Model je realizovan preko dva nivoa: nivoa leksičkih reči (leksički nivo, eng. lexical level) i nivoa tekstuelnih reči (površinski nivo, eng. surface level). Veza između ovih nivoa se uspostavlja pravilima poput

$$x : y \Leftrightarrow \alpha : \beta\_\lambda : \mu$$

sa sledećim značenjem: ako je α levi, a λ desni kontekst leksičkog simbola x, onda je y tekstuelna realizacija za x, ako i samo ako je β levi i μ desni kontekst površinskog simbola y. Npr, delimično pravilo sibilarizacije bi moglo da se opiše pravilom[39]

$$\boldsymbol{k} : \boldsymbol{c} \Leftrightarrow \_ \boldsymbol{:i}$$

sa značenjem: leksičkom 'k' sa kraja morfeme (npr. *junak*) odgovara površinsko 'c', ako za njim ('c') sledi površinsko 'i' (tako spoju morfema *junak* i *i* odgovara površinska reč *junaci*).

Najvažnija novina Kimovog modela u odnosu na ranije modele zasnovane na pravilima je bila ideja da se pravila primenjuju paralelno umesto sekvencijalno. Takođe, skupom pravila Kimovog modela uspostavlja se dvosmerna koresondencija između leksičkih i tekstuelnih reči. S obzirom da ovakva pravila mogu da opišu morfološke procese, ona se

[38] Iz okvira ovog rada izlaze formalizmi za morfološku notaciju poput DATR (Keller B, *DATR theories and DATR models*, 33rd Annual Meeting of the Association for Computational Linguistics, p. 55-62, 1995; Evans R, Gazdar G, *A language for lexical knowledge representation*, Computational Linguistics 22.2, 167-216, 1996) ili MULTEXT, odnosno MULTEXT-East (Erjavec T, Krstev C, Petkevič V, Simov K, Tadić M, Vitas, D, *The MULTEXT-East Morphosyntactic Specifications for Slavic Languages*, in Proceedings of the EACL 2003 Workshop on Morphological Processing of Slavic Languages, p. 25-32, Budapest, 2003).

[39] U slučaju kada ne postoje ograničenja za levi, odnosno desni kontekst leksičkog, odnosno tekstuelnog simbola, odgovarajući kontekst je predstavljen praznom niskom, a metasimbol ':' je naveden samo kada je to neophodno, tj. kada je prisutan (levi ili desni) kontekst bar jednog (tekstuelnog ili leksičkog) simbola.

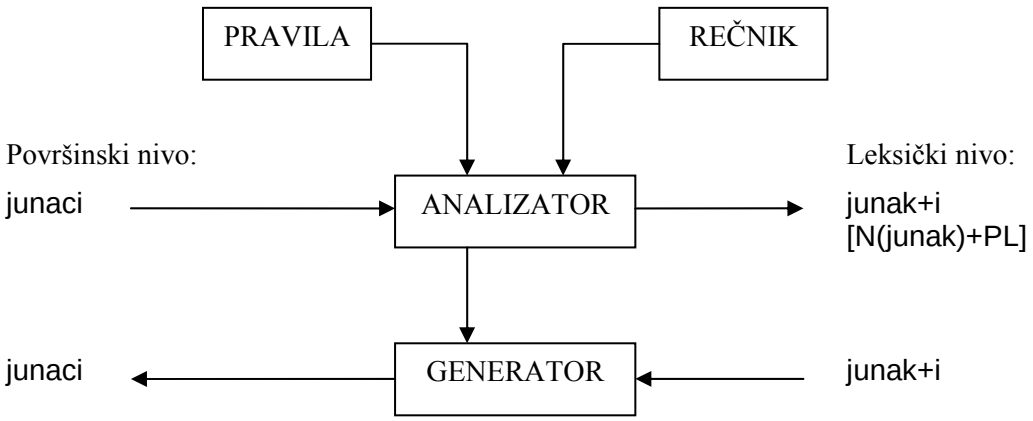
mogu koristiti ne samo za analizu (svođenje površinske reči na leksičku), već i za generisanje (površinske reči na osnovu leksičke, Slika 1.2).

Iako akcenat stavlja na gramatička pravila, ovaj model zahteva i konsultovanje odgovarajućeg rečnika alomorfa da bi mogao da vrši analizu.

Algoritmi za implementaciju pravila i rečnika dvorazinskog modela preko konačnih transduktora su detaljno opisani u [Karttunen 87] i [Antworth 90]. Istorijat Kimovog modela je dat u [Karttunen 01].

Problemi sa pristupom koji koristi dvorazinski morfološki model su višestruki.

- Komplikovaniji je u odnosu na modele koji koriste samo rečnike (videti odeljak 1.1.2).
- Održavanje konzistentnosti i čitljivosti pravila je složeno.
- Izbor "kanonskog oblika" leksičke reči (leme) na koju će površinska reč biti svedena nije jednoznačan.
- Proširivanjem skupa pravila kako bi se opisao što veći deo jezika, višestruko se povećavaju zahtevi za detaljnijom specifikacijom konteksta. Međutim, iako opšta pravila postaju obimnija i detaljnija, njima i dalje nisu obuhvaćeni izuzeci.
- Vremenska složenost prepoznavanja proizvoljne niske korišćenjem pravila dvorazinskog modela eksponencijalno zavisi od broja pravila ([Ritchie 92]).[40]



Slika 1.2

Iako je dvorazinski model zamišljen kao nezavisan od jezika i načina implementacije, pomenuti problemi se posebno pojavljuju kod jezika sa razvijenom morfologijom i velikim brojem alternacija. Zbog svih pomenutih razloga ovaj model nije razmatran "kao alat podesan za izgradnju robusnog sistema za morfološku obradu zapisa teksta" srpskog jezika ([Vitas 93b]).

[40] Prvo razmatranje vremenske složenosti algoritama dvorazinskog morfološkog modela je dato u G. Edward Barton, *Computational complexity in two-level morphology*, Proceedings of the 24th annual meeting on Association for Computational Linguistics, p.53-59, July 10-13, 1986, New York. Iako priznaju formalnu korektnost tog razmatranja, zastupnici Kimovog modela smatraju da je Barton previdео da u praksi "inherentna svojstva prirodnih jezika sprečavaju kreiranje skupa pravila koje bi dovelo do eksponencijalne složenosti" (videti npr. Kimmo Koskenniemi and Kenneth W. Church. *Complexity, two-level morphology and Finnish*. In Proceedings of the 12th International Conference on Computational Linguistics (COLING'88), p. 335-340, Association for Computational Linguistics, 1988).

1.1.2 Morfološki elektronski rečnici

Drugi pristup morfološkoj analizi prirodnog jezika je zasnovan samo na rečnicima. Tipičan primer predstavlja sistem morfoloških elektronskih rečnika DELA. Ovde treba naglasiti da postoji suštinska razlika između elektronskih rečnika i mašinski čitljivih rečnika. Mašinski čitljivi rečnici predstavljaju zapis papirnatih rečnika na elektronskom medijumu nadgrađen sistemom za pretraživanje. Samim tim, mašinski čitljivi rečnici predstavljaju specifične baze podataka namenjene ljudskom korisniku. S druge strane, elektronski rečnici su, pre svega, resursi koje konsultuju sistemi za automatsku obradu prirodno-jezičkih tekstova kao svoje baze znanja, i nisu neposredno namenjeni ljudskom korisniku[41].

Morfološki elektronski rečnici su rečnici u kojima su predstavljene morfosintaksičke informacije. U laboratoriji LADL[42] pod rukovodstvom profesora Morisa Grosa, jednog od pionira u primeni konačnih automata i transduktora u obradi prirodnih-jezičkih fenomena ([Gross 88a]), razvijen je sistem elektronskih rečnika DELA[43]. Ovaj sistem rečnika je najpre razvijen za francuski ([Gross 89a]), a potom i za mnoge evropske jezike (engleski, nemački, italijanski, norveški, finski, ruski, poljski, španski, portugalski, grčki, starogrčki itd), ali i za pojedine azijske jezike poput arapskog, korejskog i tajskog; specijalno, kad su slovenski jezici u pitanju, srpski je prvi jezik za koji je započet razvoj elektronskih rečnika po uzoru na rečnike tipa DELA ([Vitas 93b], [Vitas 01]).

Sistem rečnika DELA sačinjava više komponenti (podrečnika):

- DELAS[44] – podrečnik prostih reči[45] (eng. simple words), koji odgovara rečnicima u papirnatom obliku. Svakoj odrednici u rečniku pridružen je odgovarajući morfološki i gramatički opis što omogućava da se automatski generišu oblici njene paradigme.
- DELAF[46] – podrečnik flektivnih oblika prostih reči. Za svaku prostu reč u DELAS-u, koja ima svoju flektivnu paradigmu, elementi te paradigme, zajedno sa odgovarajućom odrednicom iz DELAS-a (lemom) i odgovarajućim morfosintaksičkim opisom se pohranjuju u podrečniku DELAF. DELAF se automatski generiše na osnovu DELAS-a i odgovarajućih opisa flektivnih paradigmi pomoću regularnih izraza.

---

[41] U slučaju elektronskog rečnika sve je podređeno automatskoj obradi teksta tako da ona bude kvalitetnija i efikasnija. U praksi to znači da sadržaj elektronskog rečnika može da bude i nešto što nikad ne bi bilo element uobičajenih papirnatih rečnika, ako se time poboljšava rad sistema za automatsku obradu prirodno-jezičkih tekstova. Npr, neki najučestaliji parovi formalnih reči poput "mi je" predstavljaju izvor višeznačnosti u tekstu. Naime, tokom lematizacije se prvoj formalnoj reči "mi" mogu pridružiti dva morfosintaksička opisa, a drugoj formalnoj reči "je" – tri, što predstavlja ukupno šest različitih mogućnosti. Međutim, u stvarnosti se samo od jedna od tih 6 mogućnosti realizuje. Stoga ima smisla dodati sistemu elektronskih rečnika i rečnik takvih (prividno višeznačnih, a učestalih) parova formalnih reči sa pridruženim odgovarajućim morfosintaksičkim opisom, jer bi se time otklonio jedan značajan izvor učestanosti (videti npr. Cvetana Krstev, Duško Vitas, *How to find the right path*? (*On the morphological disambiguation of sentence in Serbian*), 7th European Conference on Formal Description of Slavic Languages FDSL-7, University of Leipzig, 2007).

[42] Laboratoire d'Automatique Documentaire et Linguistique, Université Paris VII

[43] Dictionaire électronique du LADL

[44] DELA de formes simples

[45] Ovde je upotrebljena terminologija koja nije u skladu sa stručnom lingvističkom terminologijom; po svom značenju termin prosta reč nalikuje lemi jednočlane lekseme. U prirodno-jezičkom dokumentu prosta reč se reprezentuje kao jedna tekstuelna (formalna) reč.

[46] DELA de formes Fléchies

- DELAC[47] – podrečnik složenih reči[48] (eng. compound words). Pod složenim rečima se podrazumevaju nizovi od dve ili više prostih reči čije se značenje razlikuje od značenja komponenti (tj. prostih reči). Npr. *okrugli sto* je složena reč koja najčešće označava "sastanak" a ne "sto koji je je okruglog oblika". Pošto složene reči treba posmatrati kao nedeljivu celinu, neophodno je da se one opišu u posebnom rečniku. Njima se takođe pridružuje odgovarajući morfosintaksički opis, što omogućava generisanje oblika flektivne paradigme[49].
- DELACF[50] – podrečnik oblika složenih reči. U tom smislu, igra sličnu ulogu kao rečnik DELAF, tj. ono što DELAF predstavlja za DELAS, to DELACF predstavlja za DELAC.

Sem osnovnih podrečnika, sistem DELA još sadrži podrečnike vlastitih imena, lokalne gramatike[51], a za francuski jezik postoje i tablice leksikon-gramatika[52].

Od početka devedesetih godina prošlog veka postojao je samo jedan programski sistem koji je integrisao ove rečnike, a koji ih je koristio kao leksičke resurse za obradu korpusa. To je sistem **Intex** koga je razvio Maks Silberštajn, najpre u okviru laboratorije LADL, a kasnije nezavisno od nje ([Silberztein 94], [Silberztein 03]). Laboratorija LADL je potom razvila **Unitex**, programski sistem sa sličnim mogućnostima, čiji je grafički deo implementiran korišćenjem programskog jezika Java, i sa potpunom podrškom za kodnu shemu Unicode. U međuvremenu je Silberštajn odustao od daljeg razvoja sistema Intex i razvio je potpuno novi sistem - **NooJ**. NooJ je implementiran korišćenjem programskog jezika C#, samim tim ima potpunu podršku za Unicode, ali koristi drugačiji format za elektronske rečnike. Delimično je podržana automatska konverzija rečnika čiji je format DELA u rečnike sa formatom NooJ, tako da korisnici sistema NooJ uglavnom moraju sami da izvrše tu konverziju.

Svi ovi pomenuti programski sistemi koriste konačne automate i transduktore za implementaciju rečnika.

1.1.3 Određivanje osnove reči (stemming)

Treći pristup morfološkoj analizi prirodnog jezika, za razliku od prethodna dva, ne koristi rečnike. On je naročito prisutan kod zadataka vezanih za pretraživanje informacija. Tipičan primer je kada korisnik traži relevantne dokumente (preko servisa na Internet-u, ili pretraživanjem baze podataka digitalne biblioteke, itd) postavljanjem upita sistemu za pretraživanje; upit predstavlja opis najvažnijih karakteristika željenih dokumenata. Tipičan primer upita sadrži bulovsku kombinaciju relevantnih **ključnih reči** (eng. **keyword**) ili

---

[47] DELA de formes composées

[48] Ovde je upotrebljena terminologija koja nije u skladu sa stručnom lingvističkom terminologijom; po svom značenju termin složena reč nalikuje lemi višečlane lekseme.

[49] Videti npr. [Savary 00] i [Savary 05].

[50] DELA de formes Composées Fléchies

[51] Lokalne gramatike su opisane u odeljku 1.2.4.

[52] Leksikon-gramatika je model sintakse na nivou elementarnih rečenica prirodnog jezika. Glavni princip ovog modela tvrdi da jedinica značenja nije na nivou reči već na nivou elementarnih rečenica. Leksikon-gramatika opisuje sintaksu glagola u njihovom "minimalnom okruženju": subjekat i "esencijalni komplementi" (videti npr. [Gross 88b]). Polazeći od 6 hiljada frekventnih glagola u francuskom, Gros i njegov tim u okviru LADL su uspeli da opišu 12 hiljada glagola u francuskom i klasifikuju ih prema njihovim sintaksičkim svojstvima. Rezultati te klasifikacije su dostupni na adresi http://infolingu.univ-mlv.fr/DonneesLinguistiques/Lexiques-Grammaires/Presentation.html u obliku tablica leksikon-gramatika.

sintagmi (npr. *Pančevo AND Dunav*, gde *AND* predstavlja logičku operaciju konjunkcije). Sistem pronalazi dokumente koji sadrže ključne reči upita (u našem primeru, to su *Pančevo*, *Dunav*) i vraća ih korisniku. Pri tom se može dogoditi da dokument sadrži *Pančevac*, *Dunavom*, ali ne sadrži ključne reči *Pančevo*, *Dunav*, te ga sistem može oceniti nerelevantnim. Stoga prilikom morfološke analize, sistem za pretraživanje dokumenata uspostavlja jednu relaciju ekvivalencije između reči: dve reči se smatraju ekvivalentnim ukoliko imaju istu osnovu; time se zanemaruju završeci (sufiksi i nastavci za oblik, *-ac*, *-om*) i uzima se u obzir samo osnova (*Pančev*, *Dunav*).

Tipičan primer je Porterov algoritam (videti npr. u [Jurafsky 00]). Treba napomenuti da je ovaj algoritam razvijen za engleski jezik čija je morfologija relativno jednostavna. Program koji određuje osnove (eng. **stemmer**) pomoću ovog algoritma nije savršen, ali pokazalo se da donekle poboljšava pretraživanje informacija, pogotovo kada su u pitanju manji dokumenti (naravno, na engleskom jeziku). Jednostavan i efikasan, algoritam je zasnovan na kaskadnim pravilima koja se jednostavno implementiraju pomoću konačnih transduktora. Pravila zamenjuju jednu podnisku drugom. Za srpski bi takva pravila mogla da izgledaju ovako:

OVANJE → OVATI (npr. kovanje → kovati)

NOST → AN (npr. tačnost → tačan)

Porterov algoritam za engleski jezik ima svega 60-tak pravila. S obzirom na broj sufiksa u srpskom jeziku, kao i na raznolikost njihove realizacije, broj odgovarajućih pravila za srpski jezik bi bio daleko veći. Ilustrujmo to navodeći samo neka moguća pravila za derivacioni sufiks *-ost*:

OST → ε (npr. mladost → mlad)

OST → AN (npr. gadost → gadan)

LOST → O (npr. zrelost → zreo)

ROST → AR (npr. hrabrost → hrabar)

SKOST → SKI (npr. ljudskost → ljudski)

SKOST → ZAK (npr. drskost → drzak)

PKOST → BAK (npr. gipkost → gibak)

TKOST → DAK (npr. jetkost → jedak)

TKOST → TAK (npr. krotkost → krotak)

Broj mogućih pravila, kao i teškoće sa njihovim održavanjem i primenom[53], sugerišu da ovo nije najbolje rešenje za morfološku analizu tekstova srpskog jezika.

1.1.4 Statistički pristup morfološkoj analizi

Primena statističkih metoda u obradi prirodnih jezika je započela još 40-tih i 50-tih godina dvadesetog veka, da bi potom te metode bile potisnute sve do početka devedesetih, pre svega zbog uticaja radova Noama Čomskog[54]. Stav Čomskog prema primeni statističkih

[53] Primetimo da bi eventualni program za određivanje osnova prilikom korišćenjem navedenih pravila za sufiks – *ost* morao da poseduje dodatno znanje, kako bi mogao da izabere odgovarajuće pravilo u slučaju kada ima više mogućnosti (npr. da se odluči za pravilo SKOST → SKI umesto za pravilo SKOST → ZAK u slučaju ključne reči *ljudskost*).

[54] Videti npr. Chomsky N, *Syntactic Structures*. The Hague: Mouton,1957. Reprint. Berlin and New York 1985.

metoda u obradi prirodnih jezika se najbolje može iskazati sledećom često citiranom rečenicom: "…'verovatnoća rečenice' je potpuno beskoristan pojam ma kako interpretiran."[55].

Početka devedesetih godina prošlog veka statističke metode ponovo postaju aktuelne u obradi prirodnih jezika. Ove metode su najviše rezultata dale kod automatskog morfološkog obeležavanja (eng. **tagging**) velikih korpusa. Pridruživanje odgovarajućih informacija (obeležja) nije toliko zasnovano na lingvističkim činjenicama, koliko na verovatnoći pojedinačnih ishoda morfološke analize; tekstuelnoj reči se pridružuje onaj rezultat analize koji ima najveću verovatnoću. Stoga se problem svodi na izračunavanje verovatnoća mogućih ishoda analize, a na osnovu prethodno dobijenih rezultata raznih statističkih analiza (određivanje liste učestanosti tekstuelnih reči, bigrama, trigrama, analiza prefiksa i sufiksa tekstuelnih reči itd).

Međutim, da bi statističke analize proizvele validan rezultat, neophodno je da postoji reprezentativan uzorak koji je prethodno precizno obeležen. Reprezentativnost se, između ostalog, odnosi na raznovrsnost tekstova. Naime, ako se uzorak sastoji samo od tekstova jedne određene vrste, onda se može očekivati da će preciznost anotacije teksta te vrste biti znatno veća nego što bi to bio slučaj sa nekim drugim tekstom. Što se tiče preciznog obeležavanja uzorka, jasno je da se to mora uraditi ručno. Tu se odmah otvara pitanje izbora skupa obeležja (eng. **tag**) koji će biti korišćen. Iskustvo pokazuje da mali broj mogućih obeležja obezbedi veću preciznost; s druge strane, skup obeležja ipak mora biti i dovoljno velik kako se ne bi izgubile informacije relevantne za primenu anotiranog teksta. Kod morfološki bogatih jezika (kakav je srpski) problem veličine skupa obeležja se odmah pojavljuje ako se, sem informacije o vrsti reči, ukaže potreba i za drugim morfološkim informacijama (rod, padež, broj i sl). Dodatni problem je što povećanje skupa obeležja iziskuje i povećanje veličine uzorka (ručno) obeleženih tekstova.[56]

Međutim, ručna anotacija je izuzetno zahtevna, teška i skupa. Tako dolazimo do "začaranog kruga": da bismo izbegli ručnu anotaciju, tražimo način da što efikasnije i preciznije rešimo problem automatske anotacije, a taj problem će imati preciznije rešenje ako se vratimo ručnoj anotaciji.

U [Maurel 06c] se navodi da je "značaj velikih lingvističkih resursa u analizi jezika, nažalost, često bio zanemaren, posebno tokom poslednje decenije, usled izrazito dominantnog uticaja statističkih metoda". Kako ističu autori, "primeniti statističke algoritme na tekstualne podatke je jedno, ali izabrati 'prave' entitete na koje to treba primeniti je nešto sasvim drugo ... Koliko nam je poznato, dosad nije bilo nikakvog suštinskog napretka u kreiranju bilo sofisticiranih lingvističkih baza podataka (npr. jednojezičkih ili dvojezičnih rečnika), ni sintaksičkih analizatora, ni sistema za prevođenje, zasnovanih na statističkom pristupu, a koji polaze od nule; oni nisu odmakli dalje od nabrajanja elementarnih lingvističkih činjenica".

1.2 Formalne definicije konačnih automata i transduktora

Konačni automati i transduktori se obično formalno definišu u okviru teorije formalnih jezika (skr. TFJ). Zbog toga ćemo pre samih definicija konačnih automata i transduktora, precizno uvesti neke neophodne osnovne pojmove iz TFJ.

---

[55] Chomsky N, *Quine's Empirical Assumptions*, in Words and Objections: Essays on the Work of W. V. Quine, pp. 53-68, D. Reidek, Dordrecht, 1969

[56] Videti npr. Milan S. Sečujski, Aleksandar D. Kupusinac, *Automatska morfološka anotacija tekstova na srpskom jeziku korišćenjem HMM*, 14. Telekomunikacioni forum TELFOR 2006, Beograd.

1.2.1 Osnovni pojmovi teorije formalnih jezika

Pod (formalnom) **azbukom** podrazumevamo ma kakav konačan neprazan skup simbola. Elemente azbuke nazivamo **slovima** te azbuke.

Označimo sa Σ proizvoljnu azbuku. Kažemo da je *x* **niska nad azbukom Σ** ako i samo ako *x* predstavlja konačan niz simbola azbuke Σ, tj. $x = (a_1, a_2, \ldots, a_n)$, gde je $n \geq 0$, a $a_i \in \Sigma^*$ za svako *i*, $1 \leq i \leq n$. Broj *n* se naziva **dužina** niske *x* i označava sa |*x*|. Ako je $n = 0$, niska *x* se naziva **prazna** niska i obeležava simbolom ε. Važi da je |ε| = 0. Skup svih niski nad azbukom Σ se označava sa Σ^*.

Neka su $x = (a_1, a_2, \ldots, a_n)$ i $y = (b_1, b_2, \ldots, b_m)$ niske nad Σ, tj. $x, y \in \Sigma^*$. **Proizvod dopisivanja** ili **konkatenacije** niski[57] *x* i *y* je niska $xy = (a_1, a_2, \ldots, a_n, b_1, b_2, \ldots, b_m)$. Dopisivanje je dobro definisana i asocijativna i nekomutativna operacija na skupu Σ^*. Prazna niska je neutralni element operacije dopisivanja. Stoga uređena trojka (Σ^*, dopisivanje, ε) predstavlja nekomutativni monoid.

Konačni proizvod dopisivanja niza niski x_i (u oznaci $\prod_{i=1}^{q} x_i$) definišemo na sledeći način:

$$\prod_{j=1}^{0} x_j = \varepsilon, \prod_{j=1}^{q+1} x_j = \left(\prod_{j=1}^{q} x_j \right) x_{q+1} \ (q \in \mathbb{N}).$$

Iz definicije konačnog proizvoda dopisivanja neposredno sledi da je $\prod_{j=1}^{q+1} x_j = (\ldots((x_1)x_2)\ldots x_q)x_{q+1}$. S obzirom da je operacija dopisivanja asocijativna, u zapisu $\prod_{j=1}^{q+1} x_j = (\ldots((x_1)x_2)\ldots x_q)x_{q+1}$ možemo izostaviti zagrade, tj. $\prod_{j=1}^{q+1} x_j = x_1 x_2 \ldots x_q x_{q+1}$.

Neka je $v \in \Sigma^*$. Za nisku $u \in \Sigma^*$ kaže se da je *faktor* (redom, *levi faktor*, *desni faktor*, *pravi faktor*) niske *v* ako postoje niske $v_1 \in \Sigma^*$ i $v_2 \in \Sigma^*$ takve da $v = v_1 u v_2$ (redom, $v = u v_2$, $v = v_1 u$, $v = v_1 u v_2$ i $v_1 \neq \varepsilon$ i $v_2 \neq \varepsilon$)[58]. Faktorizacija jedne niske je zapis te niske kao proizvod dopisivanja faktora. Kako je svaka neprazna niska proizvod dopisivanja niski dužine 1, to se niske dužine 1 i slova azbuke Σ mogu poistovetiti.

Ma koji skup $L \subseteq \Sigma^*$ naziva se (formalni) **jezik** nad azbukom Σ. Primetimo da ova definicija dozvoljava da formalni jezik bude i beskonačan skup, pošto dužina proizvoljne niske (iako je konačan broj) nije ograničena.

Nad jezicima nad azbukom Σ se mogu definisati uobičajene skupovne operacije unije, preseka i razlike:

$L_1 \cup L_2$ = {*x* | *x* pripada jeziku L_1 ili jeziku L_2}
$L_1 \cap L_2$ = {*x* | *x* pripada i jeziku L_1 i jeziku L_2}
L_1 - L_2 = {*x* | *x* pripada jeziku L_1 i ne pripada jeziku L_2}

---

[57] Obično se umesto *proizvod dopisivanja*, *konkatenacije* govori *dopisivanje*, *konkatenacija*

[58] Umesto termina *levi faktor*, *desni faktor*, *pravi faktor* koriste se redom i termini *prefiks*, *sufiks*, *infiks*

kao i operacija dopisivanja jezika i Klinijevo zatvorenje.

Dopisivanjem jezika L_1 i L_2 dobija se jezik koji sadrži sve niske koje se mogu dobiti dopisivanjem proizvoljne niske jezika L_2 na proizvoljnu nisku jezika L_1:

$$L_1 L_2 = \{xy \mid x \text{ pripada jeziku } L_1 \text{ i } y \text{ pripada jeziku } L_2\}$$

Definicija azbuke omogućava da niska nad jednom azbukom bude istovremeno i slovo druge azbuke. Ako jezik L_1 tretiramo kao azbuku, tj. ako svaku nisku jezika L_1 tretiramo kao jedan simbol te "azbuke", tada možemo posmatrati skup svih niski nad "azbukom" L_1: niske nastale dopisivanjem konačnog broja niski jezika L_1, uključujući i praznu nisku. Tako dobijeni skup se naziva **Klinijevim zatvorenjem** jezika L_1, u oznaci L_1^*. Formalna definicija je rekurzivna:

(i) Prazna niska pripada Klinijevom zatvorenju jezika L_1, tj. $\varepsilon \in L_1^*$;

(ii) Svaka niska x jezika L_1 pripada Klinijevom zatvorenju jezika L_1, tj. $x \in L_1^*$;

(iii) Ako je x proizvoljna niska jezika L_1 i y proizvoljna niska jezika L_1^*, tada niska xy dobijena dopisivanjem niske y na nisku x takođe pripada Klinijevom zatvorenju jezika L_1, tj. $xy \in L_1^*$;

(iv) Niske Klinijevog zatvorenja jezika L_1 se dobijaju isključivo konačnom primenom pravila (*i*)-(*iv*).

Prema tome, skup svih niski nad azbukom Σ (Σ^*) je zapravo Klinijevo zatvorenje azbuke Σ.

Klasa **regularnih jezika** nad azbukom Σ (u oznaci $\Re(\Sigma)$) definiše se rekurzivno na sledeći način:

(i) Prazan skup $\varnothing$ je regularan jezik nad azbukom Σ, tj. $\varnothing \in \Re(\Sigma)$;

(ii) Skup koji sadrži samo praznu nisku je regularan jezik nad azbukom Σ, tj. $\{\varepsilon\} \in \Re(\Sigma)$;

(iii) Ako je a proizvoljni simbol azbuke Σ, tada je i skup koji sadrži samo taj simbol takođe regularan jezik nad azbukom Σ, tj. $\{a\} \in \Re(\Sigma)$;

(iv) Neka su L_1 i L_2 regularni jezici nad azbukom Σ. Tada je regularan i:

a. jezik dobijen unijom jezika L_1 i L_2, tj. $L_1 \cup L_2 \in \Re(\Sigma)$;

b. jezik dobijen dopisivanjem jezika L_1 i L_2 , tj. $L_1 L_2 \in \Re(\Sigma)$;

c. jezik dobijen Klinijevim zatvorenjem jezika L_1, tj. $L_1^* \in \Re(\Sigma)$.

(v) Regularni jezici se dobijaju isključivo konačnom primenom pravila (*i*)-(*iv*).

Može se pokazati da su regularni jezici zatvoreni u odnosu na osnovne skupovne operacije: uniju, presek i razliku.

Regularni izrazi predstavljaju matematičku notaciju za opis regularnih jezika. Postoje razne varijacije u pogledu sintakse regularnih izraza. Stoga su u ovom odeljku definisani samo regularni jezici, dok će sintaksa regularnih izraza (od značaja za ovaj rad) biti precizno definisana u odeljcima 3.2.2 i 4.2.2.

1.2.2 Formalna definicija konačnih automata

Pod **(nedeterminističkim) konačnim automatom** nad azbukom Σ podrazumevamo uređenu petorku $A = (Q, \Sigma, \delta, q_0, F)$, pri čemu:

- Q predstavlja konačan neprazan skup stanja;
- Σ predstavlja ulaznu azbuku;
- δ predstavlja funkciju prelaza, $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to 2^Q$;[59]
- q_0 predstavlja početno stanje automata, $q_0 \in Q$;
- F predstavlja skup završnih stanja, $F \subseteq Q$.

Kažemo da automat A **prihvata** (**prepoznaje**) nisku $x = a_1 a_2 \ldots a_n$, gde su $a_1, a_2, \ldots, a_n$ slova azbuke Σ, ako postoje stanja $p_0, p_1, p_2, \ldots, p_n \in Q$, takva da važi

$$p_0 = q_0,$$
$$p_i \in \delta(p_{i-1}, a_i), i = 0 \ldots n-1,$$
$$p_n \in F.$$

Jezik koji **prihvata** (**prepoznaje**) automat A, u oznaci $L(A)$, definišemo kao skup svih niski koje prepoznaje konačni automat A.

Jezik $X \subseteq \Sigma^*$ je **prepoznatljiv** ako postoji konačni automat nad azbukom Σ takav da je $X = L(A)$. Klasa svih prepoznatljivih jezika nad azbukom Σ označava se sa $\mathfrak{P}(\Sigma)$.

Za dva konačna automata kažemo da su **ekvivalentni** ako i samo ako su jezici koje prihvataju identični.

U slučaju kada se δ svodi na funkciju $\delta : Q \times \Sigma \to Q \cup \{\varnothing\}$, automat A se naziva **deterministički** konačni automat. Deterministički konačni automat kod koga se funkcija prelaza δ svodi na funkciju $\delta : Q \times \Sigma \to Q$, naziva se **potpuni deterministički** konačni automat.

Može se pokazati (videti npr. [Aho 72]) da za svaki nedeterministički konačni automat postoji ekvivalentan deterministički konačni automat. Štaviše, među svim ekvivalentnim determinističkim konačnim automatima koji prepoznaju jezik L postoji jedinstven (do na preoznačavanje stanja) deterministički konačni automat sa najmanjim brojem stanja koji prepoznaje jezik L; takav automat nazivamo **minimalni** deterministički konačni automat.

Jedan od najvažnijih rezultata teorije formalnih jezika i teorije automata je:

Klinijeva teorema: *Neka je* Σ *azbuka*. *Tada se klase regularnih i prepoznatljivih jezika nad* Σ *poklapaju*, *tj*. *važi* $\mathfrak{R}(\Sigma) = \mathfrak{P}(\Sigma)$.

Klinijeva teorema je značajna jer daje dve različite karakterizacije klase regularnih jezika: preko konačnih automata i regularnih izraza ([Vitas 06]). Konačni automat se svodi na algoritam kojim se utvrđuje da li jedna niska pripada jeziku automata ili ne. S druge strane, regularnim izrazima je opisana sintaksička struktura tog jezika. Dokaz Klinijeve teoreme se sastoji iz dva dela. U prvom se pokazuje da je svaki regularni jezik prepoznatljiv, tako što se

[59] Sa 2^Q je označen skup svih podskupova skupa Q.

pokaže da je klasa prepoznatljivih jezika zatvorena u odnosu na regularne operacije (unija, dopisivanje, Klinijevo zatvorenje). Dokaz u suprotnom smeru se svodi na određivanje regularnog izraza koji predstavlja jezik konačnog automata.

Za slučaj kada je regularni jezik zadat regularnim izrazom, razvijeni su algoritmi (videti npr. [Vitas 06]) za konstrukciju minimalnog determinističkog konačnog automata koji prepoznaje taj jezik. Stoga ćemo u nastavku, bez umanjenja opštosti, pod pojmom "konačni automat" uvek podrazumevati minimalni deterministički konačni automat, osim ako ne naglasimo drugačije.

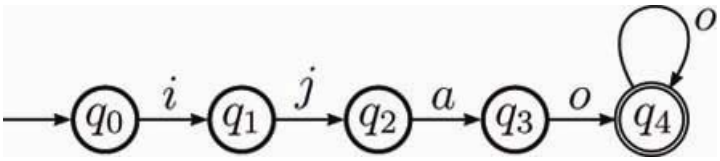
Konačni automati se često predstavljaju označenim orijentisanim grafovima. Čvorovi grafa odgovaraju stanjima, a orijentisane grane, označene simbolom ulazne azbuke, odgovaraju prelazima iz jednog stanja u drugo po odgovarajućem simbolu. Dopisivanjem oznaka grana proizvoljnog puta u grafu koji vodi od početnog do nekog završnog stanja dobijamo nisku koju automat prepoznaje.

Npr, automat *A* koji prepoznaje skup uzvika {*ijao*, *ijaoo*, *ijaooo*, *ijaoooo*,…} može se definisati na sledeći način: skup stanja ovog automata je $Q = \{q_0, q_1, q_2, q_3, q_4\}$, početno stanje je q_0, a skup završnih stanja je $F = \{q_4\}$; ulazna azbuka $\Sigma = \{i, j, a, o\}$, a funkcija prelaza δ definisana je tabelom prelaza (Tabela 1.1). Automat *A* može se predstaviti grafom (Slika 1.3).

δ	i	j	a	o
q_0	q_1	$\varnothing$	$\varnothing$	$\varnothing$
q_1	$\varnothing$	q_2	$\varnothing$	$\varnothing$
q_2	$\varnothing$	$\varnothing$	q_3	$\varnothing$
q_3	$\varnothing$	$\varnothing$	$\varnothing$	q_4
q_4	$\varnothing$	$\varnothing$	$\varnothing$	q_4

Tabela 1.1

Na osnovu definicije, automat prepoznaje nisku *ijaoo* jer je $\delta(q_0, i) = q_1$, $\delta(q_1, j) = q_2$, $\delta(q_2, a) = q_3$, $\delta(q_3, o) = q_4$, $\delta(q_4, o) = q_4$, a $q_4 \in F$. Ako uočimo put $q_0, q_1, q_2, q_3, q_4, q_4$ u grafu (Slika 1.3), dopisivanjem oznaka grana dobija se upravo niska *ijaoo*.



Slika 1.3

Primetimo da je jezik *L*(*A*) = {*ijao*, *ijaoo*, *ijaooo*, *ijaoooo*, …} regularan na osnovu definicije klase regularnih jezika date u odeljku 1.2.1 jer može da se predstavi u obliku $\{i\}\{j\}\{a\}\{o\}\{o\}^*$, tj. kao rezultat konačnog broja operacija dopisivanja i Klinijevog zatvorenja nad regularnim skupovima. Takođe, ovaj jezik je i beskonačan, jer iako je broj simbola *o* proizvoljne niske jezika *L*(*A*) uvek konačan, on nije ograničen. Automat ima

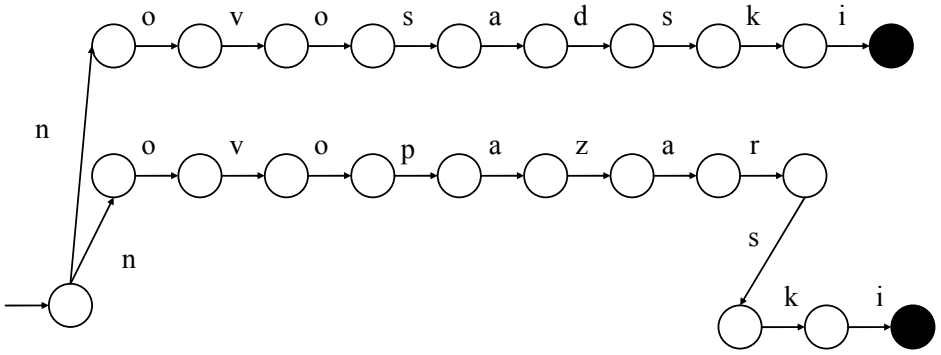
konačno mnogo stanja, ali, zahvaljujući ciklusu kojim se iz stanja q_4 po simbolu *o* ponovo prelazi u to isto stanje, može da prepozna ma koju nisku jezika *L(A)*, bez obzira koliko (konačno mnogo) puta se simbol *o* ponavlja.

Iz gornjeg primera vidimo da se jedna uprošćena interpretacija Klinijevog zatvorenja jednočlanog skupa (singltona) svodi na skup od 0 ili više "pojavljivanja" elementa tog singltona. Konkretno, u našem primeru, elemente skupa $\{o\}^* = \{\varepsilon, o, oo, ooo, ...\}$ čine 0 ili više "pojavljivanja" slova *o*.

1.2.2.1 Primeri predstavljanja leksičkih informacija konačnim automatima

Kao što se primećuje u [Gross 88a], u računarskoj i lingvističkoj literaturi se često, kao osnovni sintaksički modeli, koriste kontekstno slobodne gramatike ili rekurzivno nabrojive gramatike. Iako su to moćniji formalizmi, oni su istovremeno složeniji od formalizma konačnih automata. Ovde ćemo na nekoliko primera ilustrovati kako se konačnim automatima mogu predstaviti pojedine leksičke informacije. Takođe ćemo prikazati i specifičnosti nedeterminističkih, determinističkih i minimalnih konačnih automata.

U odeljku 1.1.2 je opisan sistem rečnika DELA u kome je razdvojen rečnik lema (DELAS) od rečnika njihovih flektivnih oblika (DELAF). Najprostiju varijantu DELAS-a, spisak lema bez pridruženog morfološkog i gramatičkog opisa, možemo implementirati preko konačnog automata. Jednostavnosti radi, pretpostavimo da "jezik" čiji rečnik razmatramo ima svega dve reči: *novosadski* i *novopazarski*. Na osnovu ove dve reči može se konstruisati automat A_1 (Slika 1.4).

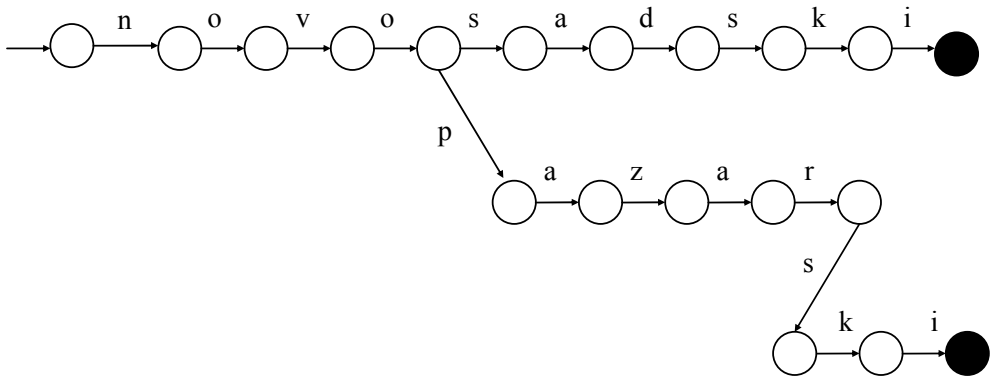


Slika 1.4

Jednostavnosti radi, stanja automata A_1 nisu numerisana, a završna stanja su predstavljena krugovima ispunjenim crnom bojom. Ovo je primer **nedeterminističkog** konačnog automata, jer iz prvog stanja može da se pređe u dva različita stanja po istom ulaznom simbolu *n*. Dati automat zaista prepoznaje reči *novosadski* i *novopazarski*, ali to čini na način koji nije najefikasniji. Naime, pretpostavimo da automat "čita" slovo po slovo reči *novopazarski.* Tada već kod prvog simbola *n* nije jasno koje je sledeće stanje automata. Ako automat izabere "pogrešno" stanje (tj. "pogrešan" put u grafu, Slika 1.4), posle čitanja simbola *o*, *v*, *o*, tek kod simbola *s* postaje jasno da izabrani put ne vodi do završnog stanja. Stoga je neophodno "vraćanje unazad" (eng. backtracking) i provera svih preostalih mogućih puteva u grafu da bi eventualno bio pronađen neki koji vodi do završnog stanja. Sve navedeno utiče na složenost algoritama za implementaciju nedeterminističkih konačnih automata, kako u pogledu memorijskog zauzeća (treba "pamtiti" sve napravljene izbore između više puteva u

grafu da bi "vraćanje unazad" bilo moguće), tako i u pogledu vremena utrošenog prilikom obilaska svih mogućih puteva u grafu.

Da bi se navedeni problemi izbegli, koriste se **deterministički** konačni automati. Svrha takvih automata je da se iz svakog stanja po jednom ulaznom simbolu može preći najviše u jedno stanje. Kao što smo već napomenuli u odeljku 1.2.2, za svaki nedeterministički konačni automat postoji ekvivalentan deterministički konačni automat. Postupak konstrukcije ekvivalentnog determinističkog konačnog automata za dati nedeterministički automat se naziva **determinizacija**. Označimo sa A_2 automat koji se dobija determinizacijom automata A_1 (Slika 1.5).

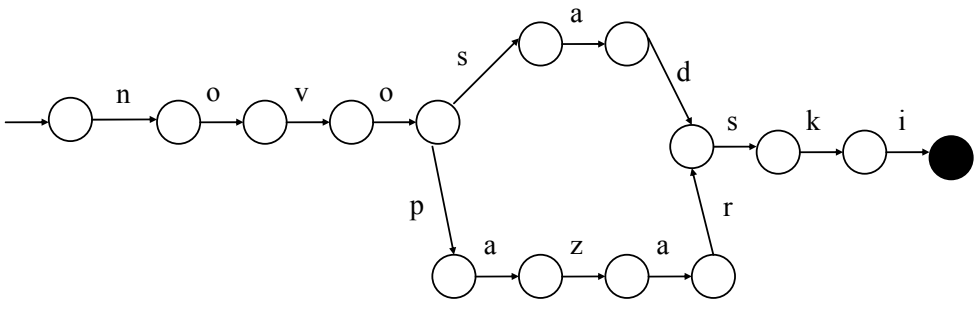


Slika 1.5

Iz svakog stanja automata A_2 se po jednom ulaznom simbolu može preći najviše u jedno stanje. Novodobijeni automat ima manji broj stanja, ali ne može se reći da je to odlika determinizacije. Naprotiv, ako je polazni automat imao *n* stanja, gornja granica broja stanja automata dobijenog determinizacijom je 2^n**.**

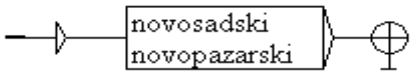
Determinizaciju automata možemo da posmatramo i kao faktorisanje niski koje on prepoznaje sleva, izvlačenjem zajedničkog levog faktora (prefiksa). U našem primeru prefiks je *novo*, a jezik koji prepoznaje automat može da se predstavi u obliku proizvoda dva jezika $\{novosadski, novopazarski\} = \{novo\}\{sadski, pazarski\}$.

Na isti način niske datog jezika mogu da se faktorišu zdesna, izvlačenjem zajedničkog desnog faktora (sufiksa). To zapravo i predstavlja suštinu **minimizacije** automata. U našem primeru, zajednički sufiks je *ski*, a minimizacijom automata A_2 dobija se automat A_3 (Slika 1.6).

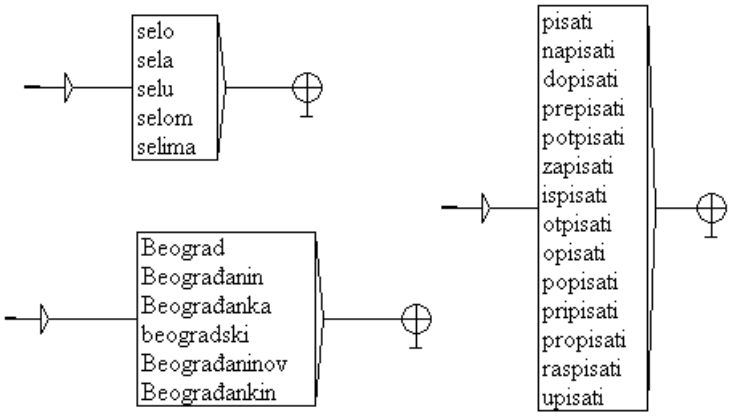


Slika 1.6

Programski sistemi Intex, Unitex i NooJ poseduju grafičke alate za predstavljanje konačnih automata (i transduktora) preko grafova. Prilikom konstruisanja grafova se stavlja akcenat na jednostavan opis jezika koji automat treba da prepozna, a programski sistem na osnovu tog opisa konstruiše ekvivalentni minimalni deterministički konačni automat. Npr, u programskom sistemu NooJ se automat A_3 (Slika 1.6) predstavlja grafom (Slika 1.7), a na sličan način se mogu opisati flektivne paradigme, kao i različiti derivacioni procesi (Slika 1.8).



Slika 1.7



Slika 1.8

1.2.2.2 Aciklični konačni automati

U svim prethodnim primerima automati nisu sadržali cikluse, a time ni puteve od početnog do završnog stanja proizvoljne dužine. Regularni jezici koje prepoznaju takvi automati mogu se opisati bez korišćenja Klinijevog zatvorenja jer su konačni. U takvim slučajevima govorimo o **acikličnim** automatima.

Iako postoje algoritmi za konstrukciju minimalnog deteminističkog automata koji se mogu primeniti na proizvoljne automate (determinizacija, minimizacija), ispostavlja se da postoje efikasniji algoritmi za konstruisanje minimalnih acikličnih automata[60] ([Maurel 06c]). Stoga proizilazi da u obradi prirodnih jezika konačnim automatima poseban značaj imaju aciklični automati. Oni su posebno pogodni za implementaciju rečnika s obzirom da se rečnici sastoje od konačnog broja morfosintaksičkih reči.

Ispostavlja se da za implementaciju konačnih automata nije pogodno postojanje više završnih stanja, odnosno postojanje završnih stanja iz kojih postoje prelazi u druga stanja automata. Taj problem se rešava uvođenjem posebnog karaktera koji označava kraj ulazne

[60] Videti npr. Revuz Dominique, *Minimization of acyclic deterministic automata in linear time*, Theoretical Computer Science, 1992 , ili Daciuk J., Mihov S., Watson B. W., Watson R. E., *Incremental construction of Minimal Acyclic Finite-state Automata*, Computational Linguistics, 26-1:3-16, 2000.

niske (#). Time se postiže da sve ulazne niske imaju zajednički sufiks #, pa će minimizovani automat (na osnovu izloženog u odeljku 1.2.2.1) imati samo jedno završno stanje u koje će postojati prelazi samo po tom karakteru.

Stoga uvodimo novu definiciju (reprezentaciju) konačnih automata koja će biti korišćena u ostatku rada.

Konačni automat je uređena petorka $A = (Q, \Sigma, \delta, q_0, q_f)$ pri čemu:

- Q predstavlja konačan neprazan skup stanja;
- Σ predstavlja ulaznu azbuku;
- δ predstavlja funkciju prelaza, $\delta : Q \times (\Sigma \cup \{\#\}) \to Q$;
- q_0 predstavlja početno stanje automata, $q_0 \in Q$;
- q_f predstavlja završno stanje automata, $q_f \in Q$.

Funkcija $\delta : Q \times (\Sigma \cup \{\#\}) \to Q$ se prirodno proširuje u funkciju δ^* definisanu na skupu $Q \times (\Sigma^* \cup \Sigma^*\{\#\})$ na sledeći način:

$$\delta^*(q, \varepsilon) = q,$$
$$\delta^*(q, xa) = \delta(\delta^*(q, x), a),$$
$$q \in Q, x \in \Sigma^*, a \in \Sigma \cup \{\#\}.$$

U ovoj notaciji, konačni automat A prihvata (prepoznaje) nisku $x \in \Sigma^*$ ako i samo ako važi $\delta^*(q_0, x\#) = q_f$, a jezik $L(A)$ koji prihvata automat A je $L(A) = \{x \in \Sigma^* \mid \delta^*(q_0, x\#) = q_f\}$.

1.2.3 Formalna definicija konačnih transduktora

Kao i u slučaju konačnih automata, postoji više definicija i više tipova konačnih transduktora. Pošto njihov detaljniji opis izlazi iz okvira ovog rada, a akcenat želimo da stavimo na njihovu primenu u obradi prirodnih jezika, navešćemo samo jednu definiciju, analognu poslednjoj definiciji konačnih automata datoj u odeljku 1.2.2.2.

Konačni transduktor[61] je uređena sedmorka $T = (Q, \Sigma, \Delta, \delta, \lambda, q_0, q_f)$ pri čemu:

- Q predstavlja konačan neprazan skup stanja;
- Σ predstavlja ulaznu azbuku;
- Δ predstavlja izlaznu azbuku;
- δ predstavlja funkciju prelaza, $\delta : Q \times (\Sigma \cup \{\#\}) \to Q \cup \{\varnothing\}$;
- λ predstavlja funkciju $\lambda : Q \times (\Sigma \cup \{\#\}) \to (\Delta \cup \{|\})^* \cup \{\varnothing\}$, tzv. **izlaznu** funkciju transduktora;
- q_0 predstavlja početno stanje transduktora, $q_0 \in Q$;
- q_f predstavlja završno stanje transduktora, $q_f \in Q$.

[61] Ova definicija se odnosi na konačni p-subsekvencijalni transduktor ([Maurel 06c]), ali pošto će samo on biti razmatran, upotrebljavaćemo prosto *konačni transduktor* ili samo *transduktor*.

Konačni automat $A = (Q, \Sigma, \delta, q_0, q_f)$ se naziva **baznim** automatom (eng. underlying automaton) transduktora *T*.

Neka je $\Lambda : (\Delta \cup \{\#, |\})^* \to (\Delta \cup \{\#, |\})^*$ preslikavanje definisano na sledeći način:

- ako su $s_1, s_2 \in (\Delta \cup \{\#\})^*$, tada je $\Lambda(s_1, s_2) = s_1 s_2$, tj. proizvod dopisivanja niski s_1, s_2;
- ako je $s_1 \in (\Delta \cup \{\#\})^*$, a $s_2 \in (\Delta \cup \{\#, |\})^*$, i postoje niske $t_0, t_1, t_2, \ldots, t_n \in (\Delta \cup \{\#\})^*$, takve da je $s_2 = t_0 \mid t_1 \mid t_2 \mid \ldots \mid t_n$, tada je $\Lambda(s_1, s_2) = s_1 t_0 \mid s_1 t_1 \mid s_1 t_2 \mid \ldots \mid s_1 t_n$;
- u svim ostalim slučajevima je $\Lambda(s_1, s_2) = \varepsilon$.

Kažemo da transduktor *T* **prepoznaje** nisku $x = a_1 a_2 \ldots a_n$, gde su $a_1, a_2, \ldots, a_n$ slova azbuke Σ, i pritom **generiše** nisku $s = \Lambda(s_1, s_2, \ldots, s_{n-1}, s_n, s_{n+1})$, gde su $s_1, s_2, \ldots, s_{n-1}, s_n, s_{n+1}$ niske nad izlaznom azbukom Δ, ako i samo ako postoje stanja $p_0, p_1, p_2, \ldots, p_n \in Q$, takva da važi:

$$
\begin{aligned}
&p_0 = q_0, \\
&\delta(p_{i-1}, a_i) = p_i, i = 0 \ldots n-1, \\
&\lambda(p_{i-1}, a_i) = s_i, i = 0 \ldots n-1, \\
&\delta(p_n, \#) = q_f, \\
&\lambda(p_n, \#) = s_{n+1}
\end{aligned}
$$

Pri tom, ako je niska *s* oblika $s = y_1 \mid y_2 \mid \ldots \mid y_m$, gde $y_1, y_2, \ldots, y_m \in \Delta^*$, svaka od niski se naziva **izlaznom niskom** transduktora *T* ("prevodom" ulaznom niske *x*), dok se sama niska *s* naziva **izlazom** transduktora *T*.

Simbol # se koristi kao oznaka kraja ulazne niske i uveden je da bi transduktor uvek imao samo jedno završno stanje. Simbol '|' koji se pominje u definiciji izlazne funkcije služi kao separator različitih "prevoda" iste prepoznate niske.

Ako neformalno interpretiramo gornju definiciju, vidimo da je konačni transduktor zapravo konačni automat koji ne samo što "prepoznaje" ulazne niske, već i na osnovu njih generiše odgovarajući izlaz ("prevod").

Na taj način konačni transduktori mogu da se posmatraju i kao "mašine za prepoznavanje", i kao "mašine za generisanje", i kao "mašine za prevođenje", odnosno kao "uspostavljanje relacije između skupova". Ova poslednja kvalifikacija na račun transduktora potiče od relacije R_T[62] koja se može uspostaviti između skupa niski koje transduktor *T* prepoznaje i skupa niski koje transduktor *T* generiše: niska *x* ulazne azbuke je u relaciji R_T sa niskom *s* izlazne azbuke ako i samo ako transduktor prilikom prepoznavanja niske *x* generiše nisku *s*.

---

[62] Ovakve relacije se nazivaju regularne relacije (videti npr. [Roche 97]). Može se pokazati da, kao što konačni automati i regularni jezici predstavljaju ekvivalentne formalizme, takav isti slučaj je i sa konačnim transduktorima i regularnim relacijama.

Za dva konačna transduktora T_1 i T_2 kažemo da su **ekvivalentni** ako i samo ako važi $R_{T_1} = R_{T_2}$.

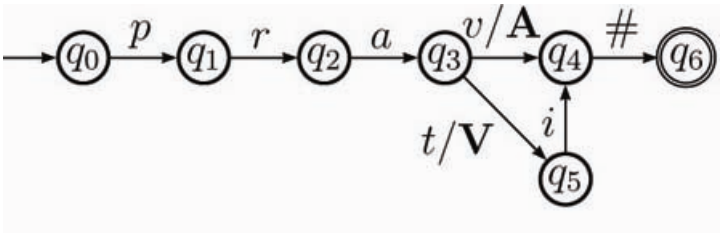
Funkcija prelaza δ i izlazna funkcija λ se prirodno proširuju u funkcije δ^* i λ^* definisane na sledeći način:

$$\delta^* : Q \times (\Sigma^* \cup \Sigma^*\{\#\}) \to Q \cup \{\varnothing\}, \qquad \lambda^* : Q \times (\Sigma^* \cup \Sigma^*\{\#\}) \to (\Delta \cup \{|\})^* \cup \{\varnothing\},$$
$$\delta^*(q, \varepsilon) = q, \qquad \lambda^*(q, \varepsilon) = \varepsilon,$$
$$\delta^*(q, xa) = \delta(\delta^*(q, x), a), \qquad \lambda^*(q, xa) = \lambda^*(q, x)\lambda(\delta^*(q, x), a),$$
$$q \in Q, x \in \Sigma^*, a \in \Sigma \cup \{\#\}.$$

Kao i u slučaju konačnih automata, i konačni transduktori se mogu predstaviti grafom. Čvorovi grafa odgovaraju stanjima, a orijentisane grane odgovaraju prelazima iz jednog stanja u drugo po odgovarajućem simbolu ulazne azbuke. Jedina razlika je u označavanju grana: oznaka je oblika a / s, gde je a ulazni simbol po kome se vrši prelaz, s je niska izlazne azbuke koja se pritom "emituje", a metasimbol '/' (kosa crta) je separator. U slučaju kada je oznaka grane oblika a / ε, koristićemo jednostavniju oznaku a.

Dopisivanjem delova levo od kose crte u oznakama grana proizvoljnog puta u grafu koji vodi od početnog do nekog završnog stanja, dobijamo nisku koju transduktor prepoznaje; dopisivanjem delova desno od kose crte u oznakama grana proizvoljnog puta u grafu koji vodi od početnog do nekog završnog stanja, dobijamo nisku koju transduktor generiše.

Razmotrimo primer konačnog transduktora T (Slika 1.9).



Slika 1.9

Transduktor T prepoznaje dve reči, *prav* i *prati*, i posle njihovog prepoznavanja generiše oznaku odgovarajuće vrste reči (**A** za pridev, **V** za glagol). Simbol # označava kraj ulazne niske i uveden je da bi transduktor uvek imao samo jedno završno stanje. Skup stanja ovog transduktora je $Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6\}$, početno stanje je q_0, a završno stanje je q_6; ulazna azbuka $\Sigma = \{p, r, a, v, t, i\}$, a funkcija prelaza δ i izlazna funkcija λ su definisane tabelama (Tabela 1.2 i Tabela 1.3).

Pošto je $\delta(q_0, p) = q_1$, $\delta(q_1, r) = q_2$, $\delta(q_2, a) = q_3$, $\delta(q_3, v) = q_4$, $\delta(q_4, \#) = q_6$, $\lambda(q_0, p) = \varepsilon$, $\lambda(q_1, r) = \varepsilon$, $\lambda(q_2, a) = \varepsilon$, $\lambda(q_3, v) = \mathbf{A}$, $\lambda(q_4, \#) = \varepsilon$, na osnovu definicije sledi da transduktor T prepoznaje nisku *prav* i generiše izlaz $\Lambda(\varepsilon\varepsilon\varepsilon\mathbf{A}\varepsilon) = \mathbf{A}$. Slično se pokazuje da T prepoznaje nisku *prati* i i generiše izlaz $\Lambda(\varepsilon\varepsilon\varepsilon\mathbf{V}\varepsilon\varepsilon) = \mathbf{V}$.

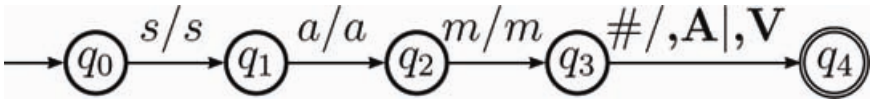
δ	p	r	a	v	t	i	#
q_0	q_1	∅	∅	∅	∅	∅	∅
q_1	∅	q_2	∅	∅	∅	∅	∅
q_2	∅	∅	q_3	∅	∅	∅	∅
q_3	∅	∅	∅	q_4	q_5	∅	∅
q_4	∅	∅	∅	∅	∅	∅	q_6
q_5	∅	∅	∅	∅	∅	q_4	∅
q_6	∅	∅	∅	∅	∅	∅	∅

Tabela 1.2

λ	p	r	a	v	t	i	#
q_0	ε	ε	ε	ε	ε	ε	ε
q_1	ε	ε	ε	ε	ε	ε	ε
q_2	ε	ε	ε	ε	ε	ε	ε
q_3	ε	ε	ε	**A**	**V**	ε	ε
q_4	ε	ε	ε	ε	ε	ε	ε
q_5	ε	ε	ε	ε	ε	ε	ε
q_6	∅	∅	∅	∅	∅	∅	∅

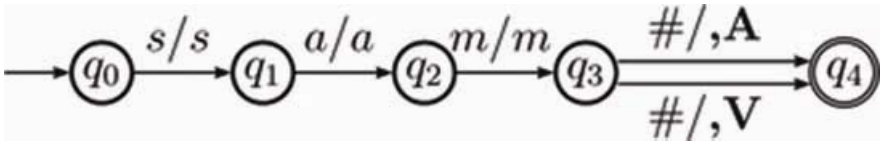
Tabela 1.3

Smisao simbola '|' u definiciji izlazne funkcije najbolje ilustruje primer transduktora koji za istu ulaznu nisku proizvodi više izlaznih niski (Slika 1.10). Ako se ponovo razmotri konstrukcija transduktora koji prepoznatoj niski pridružuje oznaku vrste reči, uočava se da homografi[63] (poput *sam*) zahtevaju poseban tretman, jer se njima pridružuje više "prevoda".



Slika 1.10

Transduktor (Slika 1.10) ne samo što prepoznaje nisku *sam*, već je i ispisuje sa pridruženim oznakama vrste reči (u ovom slučaju generisana su dva "prevoda"): *sam***,A** i *sam***,V**. Naime, da nema simbola '|' u definiciji, tada λ ne bismo mogli da definišemo kao funkciju pošto $\lambda(q_3,\#)$ ne bi bila jednoznačno određena vrednost (Slika 1.11).



Slika 1.11

Pošto je $\delta(q_0,s)=q_1$, $\delta(q_1,a)=q_2$, $\delta(q_2,m)=q_3$, $\delta(q_3,\#)=q_4$, $\lambda(q_0,s)=s$, $\lambda(q_1,a)=a$, $\lambda(q_2,m)=m$, $\lambda(q_3,\#)=$**,A|,V**, na osnovu definicije sledi da transduktor T prepoznaje nisku *sam* i generiše $\Lambda(s\ a\ m$ **,A|,V**$)=$ *sam***,A|***sam***,V**[64].

[63] Homografi su reči koje se zapisuju na isti način, a imaju različito značenje; npr. *sam* kao oblik glagola *jesam* i *sam* kao pridev.

[64] U navedenoj formuli argumenti funkcije Λ (tri jednoslovne niske i niska **,A|,V**) nisu razdvojeni zapetom već razmakom radi bolje preglednosti.

1.2.3.1 Operacije sa konačnim transduktorima

Kod konačnih transduktora se pojavljuju dve operacije kojih nema kod konačnih automata: inverzija i kompozicija.

Inverzija je unarna operacija; da bi mogla da se primerni na neki transduktor T neophodno je da se kod tog transduktora pri svakom prelasku po nekom ulaznom simbolu generiše tačno jedan izlazni simbol (uključujući tu i simbol prazne niske ε). Neformalno, rezultat inverzije (inverzni transduktor T^{-1}) dobija se od polaznog tako što u svakoj oznaci grane odgovarajućeg grafa ulazni i izlazni simbol zamene mesta. Relacija $R_{T^{-1}}$ se tada svodi na inverznu relaciju relacije R_T (R_T^{-1}), tj. $R_{T^{-1}} = \{(x,y) \mid (y,x) \in R_T\} = R_T^{-1}$. Stoga su sistemi zasnovani na konačnim transduktorima dvosmerni, tj. mogu da vrše ulogu i analizatora i generatora. Npr, kod Kimovog modela (odeljak 1.1.1), od konačnog transduktora koji se bavi morfološkom analizom (svođenjem površinske niske na leksičku, npr. *junaci* → *junak*) inverzijom se dobija konačni transduktor u ulozi generatora (površinske niske *junaci* na osnovu leksičke *junak*).

Kompozicija je binarna operacija, analogna kompoziciji u algebri: ako su T_1 i T_2 transduktori, tada primenom njihove kompozicija $T_1 \circ T_2$ na ulaznu nisku x dobijamo isti izlaz kao kad bismo najpre na ulaznu nisku x primenili tranduktor T_1, a zatim primenili transduktor T_2 na izlaz transduktora T_1, tj. važi $R_{T_1 \circ T_2} = R_{T_2} \circ R_{T_1} = \{(x,y) \mid (\exists z)(x,z) \in R_{T_1} \wedge (z,y) \in R_{T_2}\}$. Ako je $T_1 = (Q_1, \Sigma, \Delta, \delta_1, \lambda_1, i_1, f_1)$ i $T_2 = (Q_2, \Delta, \Omega, \delta_2, \lambda_2, i_2, f_2)$, transduktor $T_1 \circ T_2 = (Q, \Sigma, \Omega, \delta, \lambda, i, f)$ se definiše na sledeći način:

$$
\begin{aligned}
&Q = Q_1 \times Q_2,\ i = (i_1, i_2),\ f = (f_1, f_2),\\
&\delta((q_1,q_2),a) = (\delta_1(q_1,a), \delta_2^*(q_2, \lambda_1(q_1,a))),\\
&\lambda((q_1,q_2),a) = \lambda_2^*(q_2, \lambda_1(q_1,a)),\\
&q_j, i_j, f_j \in Q_j\ (j \in \{1,2\}),\\
&a \in \Sigma.
\end{aligned}
$$

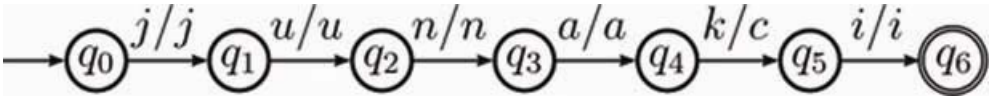
Definicija transduktora $T_1 \circ T_2$ omogućava da transduktor T_2 obrađuje izlazne niske transduktora T_1 i pre nego što su one potpuno generisane, tj, ne mora da "čeka" da T_1 završi sa obradom ulaza. Samim tim se niz serijski primenjenih tranduktora (tzv. **kaskadni transduktori**) može zameniti jednim složenim transduktorom (njihovom kompozicijom) sa istim efektom ([Mohri 97]).

1.2.3.2 Primeri primene transduktora u morfološkoj analizi

Ilustrovaćemo po jednim primerom kako se transduktori mogu iskoristiti tokom implementacije svakog od navedenih tipova morfološke analize.

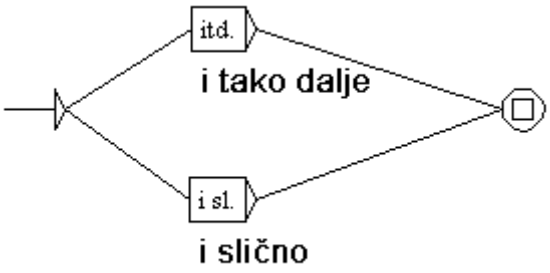
Dvorazinski morfološki model. Pravilo ***k : c*** ⇔ ***_ i*** kojim se može opisati delimična sibilarizicija u Kimovom modelu, može se implementirati konačnim transduktorom (Slika 1.12). Ovaj transduktor opisuje kako se od leksičke reči *junak* generiše tekstuelna reč

junaci; njegov inverzni transduktor opisuje kako se tekstuelna reč *junaci* svodi na leksičku reč *junak*.



Slika 1.12

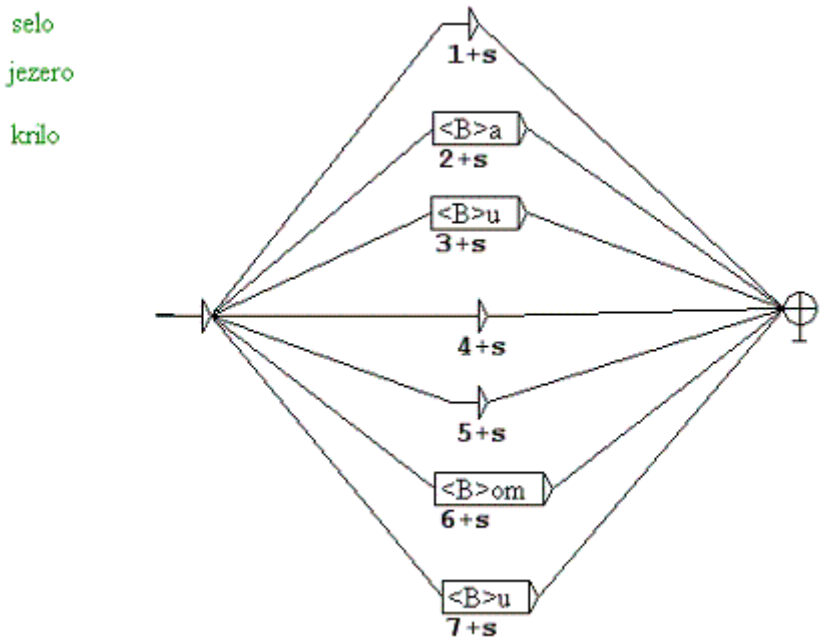
Morfološki elektronski rečnici. Pomenuti programski sistemi Intex, Unitex i NooJ poseduju grafičke alate za predstavljanje konačnih transduktora preko grafova. Prilikom konstruisanja grafova stavlja se akcenat na jednostavnost opisa transformacije koju transduktor treba da obavi, a programski sistem na osnovu tog opisa sam konstruiše minimalni konačni transduktor. Tako npr. transduktor koji "prevodi" skraćenice "*itd.*", "*i sl.*" u niske koje odgovaraju njihovom razvijenom obliku ("*i tako dalje*", "*i slično*") može da se predstavi grafom (Slika 1.13) u programskom sistemu Unitex.



Slika 1.13

Primer u odeljku 1.2.2.1 (Slika 1.8) predstavlja pojedinačni slučaj kada je u pitanju opis flektivne paradigme. Jedno uopštenje je dato u laboratoriji LADL pod rukovodstvom Morisa Grosa. Promenljive vrste reči (poput imenica, prideva, glagola) su klasifikovane tako da jednu klasu čine reči iste vrste čiji se oblici flektivne paradigme generišu na osnovu leme na istovetan način. Svakoj klasi je pridružena odgovarajuća oznaka tako da se na osnovu leme i njoj pridružene klase može automatski generisati njena flektivna paradigma. Na opisanoj klasifikaciji počivaju rečnici DELA iz odeljka 1.1.2. Kad je u pitanju analogna klasifikacija u srpskom jeziku, više se može naći u [Vitas 93b], [Krstev 97], [Vitas 01].

Pogledajmo jedan primer. Imenice *selo*, *jezero*, *krilo* imaju istovetne flektivne nastavke koji se mogu predstaviti transduktorom u sistemu NooJ (Slika 1.14). Jednostavnosti radi, predstavljeni su samo oblici jednine. Svaki čvor grafa odgovara jednom flektivnom obliku; **s** je oznaka za jedninu, a brojevi 1-7 odgovaraju redom padežima u srpskom. Oznaka **<B>** predstavlja poseban operator koji koristi NooJ, a koji briše jedan karakter niske i to zdesna. Npr. čvor **<B> *u*/ 3+s** opisuje da se oblik dativa jednine dobija tako što se obriše poslednji karakter leme zadate imenice i na ostatak se doda karakter ***u*** (tako se dobija *selu*, *jezeru*, *krilu*). Na ovaj način, znajući da imenica pripada određenoj klasi opisanoj odgovarajućim transduktorom (Slika 1.14), na osnovu transduktora klase i leme te imenice mogu se automatski generisati svi njeni flektivni oblici.



Slika 1.14

Na osnovu rečnika DELAS koji sadrži leme prostih reči (implementiranog kao automat) i na osnovu transduktora koji opisuju klase flektivnih paradigmi, automatski se generiše rečnik flektivnih oblika prostih reči DELAF (takođe implementiran kao automat). Poslednji automat se onda može u okviru morfološke analize primeniti na tekst.

Određivanje osnove reči. Slika 1.15 ilustruje implementaciju Porterovog algoritma za jedno konkretno pravilo

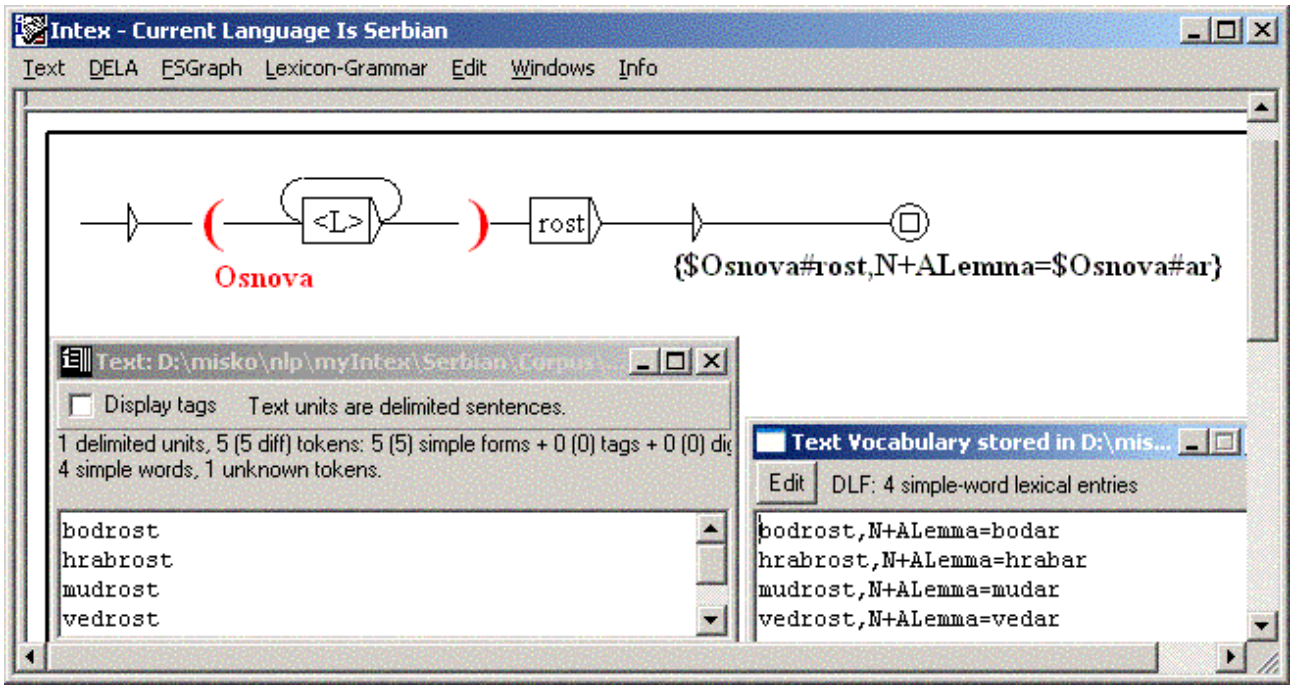
ROST → AR (npr. hrabrost → hrabar)

Ovde je transduktor konstruisan kao graf u okviru programskog alata Intex; u levom prozoru ispod grafa je prikazan tekst na koji je taj graf primenjen, a u desnom prozoru ispod grafa je prikazan rezultat te primene.

Deo grafa između zagrada predstavlja primer korišćenja promenljivih: niska koja bude prepoznata tim delom grafa (niz od jednog ili više slova[65]) biće memorisana u promenljivoj ***Osnova***. Kompletan graf funkcioniše tako što prepoznaje niske koje se završavaju sa *–rost*, a u promenjivu ***Osnova*** upisuje početni deo prepoznate niske bez završetka *–rost* (u navedenom primeru, za nisku *hrabrost* u promenljivoj ***Osnova*** biće memorisano *hrab*). Poslednji čvor u grafu služi za generisanje izlaza: na promenljivu se referiše tako što se ispred njenog imena navodi simbol '$' ($***Osnova***), a simbol # označava operaciju dopisivanja niski (u navedenom primeru, ako je u tekstu prepoznata niska *hrabrost*, tada je vrednost izraza **{$Osnova#rost,N+ALemma=$Osnova#ar}** niska {**hrabrost,N+ALemma=hrabar**}[66].

[65] Operator <L> označava proizvoljno slovo (eng. letter), a kako čvor grafa u kome se nalazi ima ciklus, taj deo grafa prepoznaje nisku od jednog ili više slova.

[66] Oznaka **N** ovde označava da je prepoznata niska *hrabrost* imenica, a **ALemma=hrabar** je oznaka za lemu prideva od koje je ta imenica izvedena.



Slika 1.15

Statistički pristup morfološkoj analizi. Na kraju pomenimo primer jedne modifikacije konačnih automata/transduktora koja se koristi kod statistički zasnovane morfološke analize, a posebno kod fonoloških, morfoloških i sintaksičkih modula sistema za prepoznavanje i generisanje govora ([Mohri 96]). U pitanju su tzv. **težinski** (**probabilistički**) automati/transduktori (eng. **weighted finite-state automata/transducers**). U pitanju je jednostavno proširenje konačnog automata/transduktora kod koga je svakom prelazu iz jednog stanja u drugo pridružen broj između nule i jedinice; taj broj predstavlja **verovatnoću** da taj put u automatu/transduktoru bude izabran. Zbir verovatnoća svih prelaza u automatu/transduktoru mora biti jednaka 1. Detaljnije razmatranje ove modifikacije izlazi iz okvira ovog rada[67].

1.2.4 Primena konačnih automata i transduktora u sintaksičkoj analizi

Iako su pioniri računarstva, neuronskih mreža i teorije informacija još pre pedesetak godina kreirali konačne automate i transduktore, uticaj radova Čomskog je odložio njihovu primenu u obradi prirodnih jezika. Zanimanje računarskih lingvista za konačne automate i transduktore počinje tek ranih sedamdesetih. Naime, i u lingvističkoj i u računarskoj literaturi se mnogo značaja pridavalo korišćenju kontekstno slobodnih gramatika kao glavnog formalizma za modelovanje sintakse jezika; "…lingvisti izgleda da su zaboravili da je mnoštvo lingvističkih fenomena i u sintaksi i u fonologiji takve prirode da se može predstaviti konačnim automatima" ([Gross 88a]).

[67] Videti npr. Pereira F, Riley M, Sproat R, *Weighted rational transductions and their application to human language processing*, in ARPA Workshop on Human Language Technology, 1994.

Problem sa primenom formalizma kontekstno slobodnih gramatika u sintaksičkoj analizi rečenica prirodnog jezika je u tome što to zahteva leksikalizovane gramatike ogromnih razmera; "leksikalizovane" znači da pravila moraju da sadrže konkretne reči jezika.

S druge strane, prema [Roche 97][68], konačni transduktori

- mogu da se iskoriste za generisanje efikasnijih sintaksičkih analizatora, pri čemu se analiza svodi na transformisanje niski, tačnije niza simbola koji predstavljaju rezultat morfološke analize.
- mogu da se iskoriste za transformaciju kontekstno slobodne gramatike jezika u odgovarajući transduktor (koji će vršiti sintaksičku analizu tog istog jezika).
- omogućavaju homogenu reprezentaciju i gramatike i ulazne rečenice i same analize. Naime, i kreiranje gramatike i sama analiza se svode na rad transduktora.

Ono što se najčešće navodi kao nedostatak konačnih automata i transduktora, kao sintaksičkih analizatora, jeste njihova "nepreciznost" kad je u pitanju obrada **rekurzije** i modeliranje složenijih konstituenata rečenice. Čomski je pokazao[69] da se kontekstno slobodni jezik može generisati konačnim automatom ako i samo ako postoji kontekstno slobodna gramatika koja generiše taj jezik i nema nijedno pravilo sa rekurzijom u sredini (tj., nema pravila oblika $A \rightarrow \alpha A \beta$).

Konačni automati i transduktori se mogu primeniti u sintaksičkoj analizi na više načina. Jedan primer smo već pomenuli na početku ove glave: algoritmi za ekstrakciju informacija (poput onih u sistemu Fastus) za sintaksičku analizu koriste kaskadno primenjene transduktore umesto kontekstno slobodne gramatike.

U [Roche 97][70] se može naći drugi pristup: opis algoritama za automatsko generisanje konačnog automata koji predstavlja aproksimaciju proizvoljne kontekstno slobodne gramatike. U osnovi gramatika koje koristi programski sistem Unitex, kreiran u LADL, se nalaze rekurzivne mreže prelaska (eng. **recursive transition networks**), koje predstavljaju proširenja konačnih automata, ali su zapravo ekvivalentne sa konteksno slobodnim gramatikama; međutim, pri kompilaciji takvih gramatika postoji opcija da se gramatika transformiše ili u ekvivalentni transduktor (ako on postoji) ili u transduktor koji predstavlja aproksimaciju polazne gramatike.

Još jedan doprinos rešavanju ovog problema je dat u [Gross 93]. Gros je prvi upotrebio pojam **lokalne gramatike** da opiše izraze za datume i vreme, kao i adrese u pismima. Lokalna gramatika je način da se opiše sintaksička struktura grupe pojedinačnih elemenata koji su međusobno povezani i predstavljaju gradivni deo veće gramatičke konstrukcije, a čije se sličnosti ne mogu lako izraziti. Sam naziv **lokalna** sugeriše da cilj takve gramatike nije opis sintaksičke strukture cele rečenice, već određenih lokalnih uslova i ograničenja koje treba da zadovolje susedne niske u prihvatljivim jezičkim iskazima.

Osnovna ideja njihove primene je da se najpre konstruišu lokalne gramatike za konstrukcije koje se često pojavljuju u tekstu (npr. izrazi za datume, [Maurel 87]), tako da se te gramatike mogu ponovo iskoristiti u opisu složenijih lingvističkih konstrukcija.

Jedna od osnovnih primena lokalnih gramatika jeste otklanjanje **višeznačnosti** (eng. **ambiguity**) tokom morfološke analize. Ovde se pod višeznačnošću podrazumeva mogućnost

---

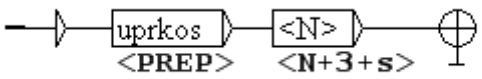
[68] Videti poglavlja Roche E, *Parsing with finite-state transducers*, str. 241-278, i Laporte E, *Rational Transductions for Phonetic Conversion and Phonology*, str. 407-429.

[69] Chomsky, N, *On certain formal properties of grammars*, Information and Control, 2, p. 137-167, 1959.

[70] Videti poglavlje Pereira F, Wright R, *Finite-state Approximation of Phrase-Structure Grammars*, str. 149-168.

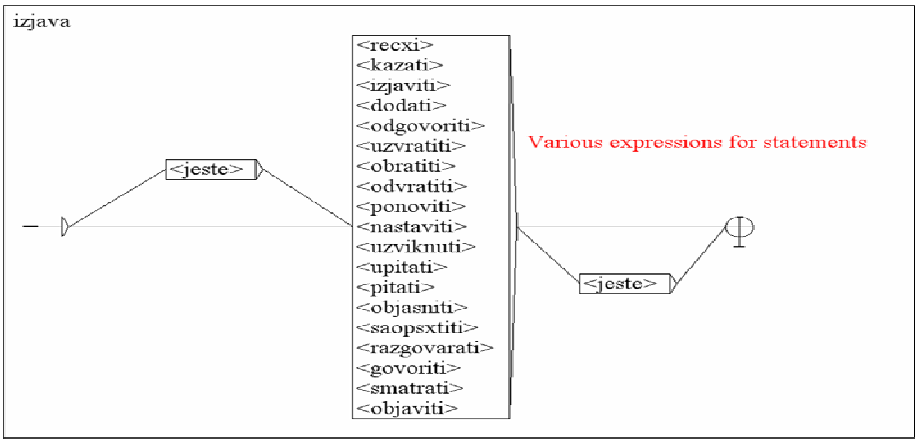
da se tokom morfološke analize istoj tekstuelnoj reči pridruži nekoliko morfosintaksičkih opisa; uzrok tome je pojava da tekstuelne reči sa različitim morfosintaksičkim opisima mogu imati istu grafemsku reprezentaciju[71].

Razmotrimo jedan primer. Pretpostavimo da je tokom morfološke analize u nizu tekstuelnih reči "*uprkos junaku*" prva tekstuelna reč označena kao predlog, dok su drugoj tekstuelnoj reči pridružene dve morfosintaksičke reči koje redom označavaju da se radi ili o dativu jednine ili o lokativu jednine imenice *junak*. Lokalna gramatika (Slika 1.16), kreirana u sistemu NooJ, uvodi ograničenje da, ako posle predloga *uprkos* sledi imenica, tada je uvek u pitanju padežni oblik te imenice u dativu. Primenom te lokalne gramatike na pretpostavljeni rezultat morfološke analize eliminiše se morfosintaksička reč koja koja označava lokativ jednine imenice *junak*, posle čega je višeznačnost otklonjena.



Slika 1.16

Ovakav prilaz ima dva nedostatka: (1) lokalna gramatika u obliku transduktora može biti vrlo složena i za najjednostavnije uslove i (2) transduktor menja rezultat leksičkog prepoznavanja, pa superpozicija ovakvih transduktora može dovesti do nekorektnog rezultata[72].



Slika 1.17

Lokalne gramatike se mogu koristiti i za prepoznavanje i klasifikaciju imenovanih entiteta. Primeri lokalnih gramatika koje opisuju izraze za datume, prezimena i lična imena u srpskom jeziku, i na osnovu kojih se mogu ekstrahovati informacije o određenim vrstama događaja (npr. o izjavama javnih ličnosti) se mogu naći u [Gucul 06] i [Krstev 05b]. Kao

[71] Pri tom to ne moraju biti različite reči već i flektivni oblici jedne iste reči (npr. *junaka* može biti genitiv jednine, akuzativ jednine i genitiv množine imenice *junak*).

[72] Stoga se za otklanjanje višeznačnosti u okviru sistema Unitex umesto lokalnih gramatika koriste ELAG-gramatike (skr. Elimination of Lexical Ambiguities bz Grammars). ELAG-gramatika se implementira kao automat kojim se iskazuje pravilo za otklanjanje višeznačnosti. Višeznačnost se otklanja konstruisanjem preseka automata teksta i automata ELAG-gramatike. Prednost ovakvog prilaza jeste to što rezultat superponirane primene automata ne dovodi do nepredviđenih efekata ([Laporte 99], [Paumier 06]).

primer navodimo pojednostavljeni deo lokalne gramatike (Slika 1.17)[73] iz [Gucul 06] koja opisuje izjave javnih ličnosti. Takođe su navedeni rezultati primene "glavne" gramatike u ekstrakciji informacija iz novinskih tekstova (Slika 1.18).

crnogorski premijer	Milo Đukanović smatra	da je postignuti ko
rže pod kontrolom,	rekao je Milorad Mračić	, direktor Direkcij
Centralnog registra	Vida Uzelac je rekla	da je na spisku Pri
sto digitalizacije,	izjavio je Draško Petrović	, generalni direkto
bu protiv korupcije	Verica Barać u sredu je izjavila	da očekuje da će S
i ekonomije Srbije	Božidar Đelić izjavio je u subotu, 13. marta,	da su
trgovine i turizma	Slobodan Milosavljević izjavivši	u Nišu da nije
da za razvoj Srbije	Olivera Božić izjavila je 27. juna	da je Fond u prv
sednik vlade Srbije	Miroljub Labus razgovarao je u subotu	sa predstavnicim
na evra. Ekonomista	Ljubomir Madyar dodao je	da je učešće subve
ministra finansija	Koviljka Mihailović je kazala	da je ostvaren cilj
inansijski direktor	Dyon Konors kaže	da u kompaniji oč
sna i Hercegovina",	objasnio je Dragoljub Mićunović	, koji predvodi d
na sudijskom poslu,	pita Vida Petrović Škero	, u situaciji primen
yela u novom roku",	kaže Vida Petrović Škero	. Beda "Sudovi
Srbije za naš list	govori Vladimir Kravčuk	, direktor Merkato

Slika 1.18

Na kraju ove glave dajemo kratak rezime prednosti i nedostataka korišćenja istog formalizma (konačnih automata i transduktora) u opisu jezičke obrade. Prednosti su:

- kompaktnost: reprezentacija rečnika pomoću automata smanjuje njegovu veličinu na način sličan LZW algoritmu (koji koristi pkzip)
- brzina: prepoznavanje reči konačnim automatom zahteva vreme proporcionalno veličini ulaza, i nezavisno je od veličine automata.
- jednostavna obrada: konačni automati se mogu konstruisati koristeći elementarne operacije slične operacijama sa skupovima. Vremenska i prostorna složenost tih operacija je polinomijalna te ih čini pogodnim za izračunavanja u realnom vremenu.
- modularnost: između različitih lingvističkih objekata, predstavljenih konačnim automatima i transduktorima, omogućena je interakcija; složene relacije se mogu modelirati preko niza jednostavnih relacija, zahvaljujući kaskadnoj primeni transduktora.
- optimizacija: za konačne automate i transduktore su razvijeni algoritmi sa efikasnošću koju drugačije implementacije ne mogu da dostignu.
- programski dizajn: zahvaljujući algoritmima koji se mogu iznova koristiti, kao i standardizovanim vezama između modula, složeni sistemi se lako modifikuju dodavanjem ili eliminisanjem pojedinih delova, a i manje su podložni greškama.

Nedostaci su:

- ograničena izražajna moć: klase jezika koje konačni automati i transduktori mogu da prepoznaju i generišu su ograničene memorijskim kapacitetom tih hipotetičkih mašina (tačnije, brojem stanja). Iz tog razloga je njihova upotreba ograničena na osnovne nivoe jezičke hijerarhije; međutim, ispostavlja se da su takva ograničenja ne samo prihvatljiva, već čak postoje i nove tehnike koje omogućavaju da taj formalizam bude

[73] Uglaste zagrade u grafu označavaju ne samo navedene leme (infinitive glagola) već i sve njihove oblike.

upotrebljen umesto daleko neefikasnijih klasičnih (teorijski boljih) rešenja (npr. sintaksička analiza u kojoj konačni transduktori zamenjuju formalizam kontekstno slobodnih gramatika)

- redosled operacija prilikom rada sa ogromnim konačnim automatima i transduktorima mora pažljivo da se izabere, da bi se izbegla izračunavanja preterane složenosti.

2

REGULARNA DERIVACIJA

U ovom delu razmatra se fenomen regularne derivacije u srpskom jeziku. Posebno se razmatraju neki modeli regularne derivacije imena koji mogu efikasno da se implementiraju u postojećim elektronskim rečnicima.

Dva osnovna morfološka procesa koja kombinuju morfeme su **fleksija** i **derivacija**[74]. Kombinovanje je uslovljeno samostalnošću morfema, jer postoje morfeme koje mogu da stoje samostalno (**slobodne** morfeme) i morfeme koje to ne mogu već samo u kombinaciji sa drugim morfemama (**vezane** morfeme); npr. reč *radnik* sadrži slobodnu morfemu *rad* i vezanu morfemu *-nik*. Prilikom kombinovanja morfema, jedna od njih predstavlja **koren**, tj. morfemu koja nosi "glavno" značenje rezultujuće reči, dok preostale (vezane) morfeme, **afiksi**, svojim "dodatnim" značenjima formiraju definitivno značenje reči ([Jurafsky 00]). Prilikom analize načina na koji se morfeme kombinuju, obično se umesto o korenu govori o **osnovi**, kao delu reči na koji se dodaje afiks (kao što ćemo u nastavku videti, osnova je širi pojam u odnosu na koren).

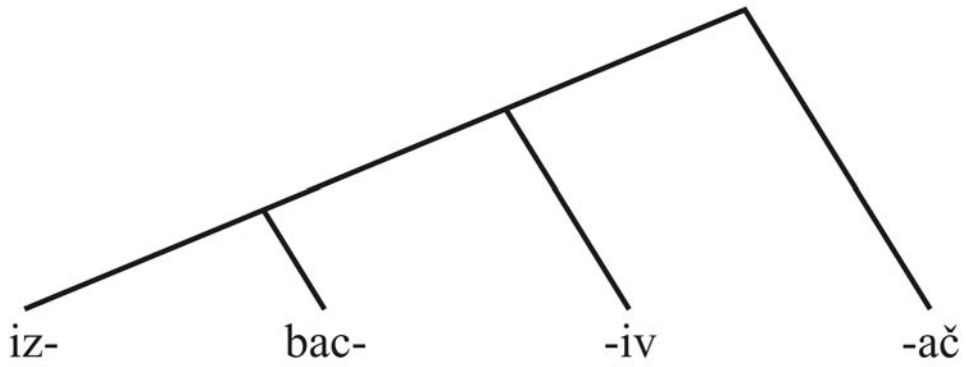
Afiksi se prema položaju u odnosu na osnovu mogu podeliti na:

- prefikse (stoje ispred osnove, npr. *pre-* u *precrtati*),
- sufikse (stoje iza osnove, npr. *-nje* u *crtanje*) i
- infikse (čine deo osnove, npr. *ov* u *sinovi*)

Analizirajmo npr. reč *izbacivač* (Slika 2.1). Najpre možemo da odvojimo sufiks *–ač* od osnove *izbaciv-*, posle toga sufiks *–iv* od njegove osnove *izbac-*, i na kraju prefiks *iz-* od

[74] U [Bugarski 95] nalazimo da su "tri osnovna morfološka procesa kompozicija, derivacija i fleksija. **Kompozicija** je kombinovanje slobodnih morfema, čiji su proizvod složenice (... *mimo* + *hod* – *mimohod* ...) ... Kompozicija i derivacija su leksički procesi, služeći za izvođenje novih reči (zbog čega se oba nekad obuhvataju pojmom derivacije u širem smislu) ..."

njegove osnove *bac-* (koja ujedno predstavlja i koren reči *izbacivač*). Ovaj primer pokazuje da osnova može biti i koren reči, ali da to nije uvek slučaj.



Slika 2.1

Fleksija predstavlja obrazovanje gramatičkih oblika reči kombinacijom (gramatičke) osnove i vezanih gramatičkih morfema (nastavaka za oblik i infiksa) ([Bugarski 95], [Stanojčić 99]). Rezultujuća reč pripada istoj vrsti reči (imenice, glagoli i sl.) kao i polazna osnova i ima tačno određeno značenje, odnosno ispunjava određenu gramatičku službu u rečenici, npr. slaganje (kongruenciju)[75]. Tipičan primer fleksije predstavljaju padežni oblici imenice (npr. *rad*, *rada*, *radu*, *radom*, *radovi* itd).

Derivacija (izvođenje, tvorba) predstavlja mehanizam za građenje novih reči kombinovanjem (tvorbene) osnove i afiksa ([Bugarski 95], [Stanojčić 99]). Reč koja je izvedena se obično naziva **motivisanom**, dok se reč od koje je izvedena naziva **motivnom**. U zavisnosti od vrste afiksa, derivacija obuhvata **prefiksaciju** (*iz-baciti*) i **sufiksaciju** (*doktor-ka*), a u novije vreme ([Klajn 02])[76] se u procese derivacije ubrajaju i **kompozicija** (slaganje reči, tj. formiranje složenica, npr. *belo-glavi*) i **konverzija** (ili pretvaranje, promena gramatičke vrste reči bez promene oblika, npr. konverzija prideva *mlada* u imenicu *mlada*). Kod derivacije motivisana reč ne mora pripadati istoj vrsti reči kao i motivna reč, a često je teško precizno predvideti njeno značenje.

2.1 Regularna derivacija

Među derivacionim procesima, posebno kod slovenskih jezika, od posebnog su značaja procesi kod kojih se značenje izvedene reči može izvesti iz značenja motivne reči. Ovu klasu derivacionih procesa nazivaćemo **regularna derivacija (**[Krstev 04b]**,** [Vitas 05d]**)**. Definicije korišćene u papirnatim rečnicima za leme izvedene regularnom derivacijom jasno pokazuju da se radi o posebnoj vrsti derivacije. Naime, u mnogim slučajevima, bilo gramatičkom referencom (npr. *računčić* - deminutiv od *račun*), bilo korišćenjem sinonimijskih parafraza u obliku regularnih obrazaca (npr. *kockarev – koji se*

[75] Slaganje (kongruencija) predstavlja formalnu usaglašenost međusobno povezanih delova rečenice u gramatičkim kategorijama ([Bugarski 95]). Npr. slaganje imenica i prideva u rodu, broju i padežu.

[76] U [Bugarski 95] se navode još i "srazmerno marginalni" tvorbeni procesi **skraćivanje** (npr. akronim *OUN*) i **slivanje** (kombinovanje prvog dela jedne reči i drugog dela druge reči, npr. *klinceza*), sa napomenom da se "koriste u ograničenoj meri i neretko uz posebna stilska obeležja".

odnosi na kockara), te definicije samo ukazuju da se radi o posebnoj, **pravilnoj** (regularnoj) vrsti derivacije. Razmotrićemo doprinos regularne derivacije rešavanju problema nepoznate reči u morfološkoj analizi, kao i njenu ulogu kao sistematičnog sredstva za upotpunjavanje sinonimije u smislu koji koristi Moris Gros[77].

2.1.1 Nepoznata reč i regularna derivacija

Prilikom automatske morfološke analize pomoću morfološkog elektronskog rečnika (definisanog u odeljku 1.1.2, skr. e-rečnik) poseban problem predstavljaju **nepoznate reči**, tj. tekstuelne reči kojih nema u e-rečniku. Kao takve, one dobijaju posebno obeležje, ali, budući da su bez pridruženog morfosintaksičkog opisa, ne mogu biti od koristi sintaksičkoj analizi.

Nepoznate reči se mogu podeliti u tri osnovne grupe ([Vitas 05f]). Prvu i najmanju grupu čine reči čije je pojavljivanje vezano za sam tekst (vlastita imena likova i mesta u književnom delu, sekvence iz stranih jezika, dijalektizmi).

Drugu grupu čine reči koje bi mogle da se pojave u rečniku, ali to iz nekog razloga nije slučaj (nedostajuće odrednice, vlastita imena uopšte, skraćenice, grafijske varijacije sa arhaizmima i regionalizmima, delovi višečlanih leksema[78]).

Treću grupu čine reči koje su rezultat regularne derivacije. Pri tom derivacione relacije mogu (ali ne moraju) da dovedu do promene vrste reči. Među njima, su posebno važni rezultati mocije roda i amplifikacije, tj. subjektivne ocene (deminutivi i augmentativi). Osim njih, za veliki broj imenica postoje prisvojni i relacioni pridevi, dok za praktično sve glagole nesvršenog vida postoje glagolske imenice ([Klajn 03]). Pravilnost regularne derivacije stvara utisak da ona stoji negde između fleksije i derivacije ([Vitas 07b]). Štaviše regularna derivacija je bliža fleksiji, pogotovo kada se prilikom izvođenja ne menja vrsta reči. Tako se u [Klajn 03] primećuje: "Čudno je da ni jedan gramatičar nije uočio bitnu osobinu po kojoj se (sufiks) *–nje* razlikuje od svih ostalih imeničkih sufiksa, a to je da za njegovu upotrebu, bar kod nesvršenih glagola, praktično nema ograničenja. Samim tim, reči na *nje* nisu deverbalne (deverbativne) imenice, kao one s drugim sufiksima, nego su glagolske imenice – glagolski oblik u imeničkoj funkciji... Mada nema sumnje da spadaju u delokrug tvorbe reči, ne bi ih trebalo zaobići ni u flektivnoj morfologiji, u odeljku o promeni glagola.".

Obrada nepoznatih reči zavisi od grupe kojoj pripadaju. Pri tom su posebno brojni imenovani entiteti (vlastita imena) i rezultati regularne derivacije. Stoga se u ovom radu razmatra obrada rezultata regularne derivacije, posebno izvedenica od imenovanih entiteta.

2.1.2 Regularna derivacija u elektronskom rečniku srpskog jezika

Razmotrimo primere ([Vitas 05d]) koje ilustruju Tabela 2.1 i Tabela 2.2, a koji su navedeni na osnovu jedinog eksplanatornog rečnika srpskog jezika (Rečnik srpskohrvatskog književnog jezika Matice srpske i Matice hrvatske, RMSMH)[79]. Svaka kolona u prvoj tabeli

---

[77] Gross, *Synonymie, morphologie, derivationnelle et transformations*, Languages, 128, Paris, Larousse, 1997.

[78] U [Vitas 05f] se još koristi izraz **leksički kompozit**. Odgovarajući engleski termin je **compound**.

[79] RMSMH. *Rečnik srpskohrvatskoga književnog jezika*. vol. 1–6, Beograd-Zagreb, Matica Srpska, Matica Hrvatska, 1967

sadrži imenicu koja predstavlja ljudsko biće, dok svaka kolona druge tabele sadrži imenicu koja ne predstavlja ljudsko biće. U redovima tabela dati su određeni primeri regularne derivacije. Leme u ćelijama tabele koje nisu zastupljene u RMSMH predstavljene su iskošenim slovima (italik), dok su reči koje nisu zastupljene u Korpusu savremenog srpskog jezika ([Korpus 06]) podvučene. U poslednjoj koloni je za svaki red tabele naveden zajednički morfosintaksički opis[80] lema u okviru jednog reda tabele.

osnovna lema	**protivnik**	**naslednik**	**vlasnik**	**N10+Hum**
Mocija roda (**Gen**)	protivnica	naslednica	*vlasnica*	N651+Hum
Rel. pridev (**Arel**)	protivnički	nasledn<u>ički</u>	*vlasnički*	A2+Rel
Prisv. pridev (**Apos**)	*protivnikov*	naslednikov	vlasnikov	A1+Pos
Rel. pridev (mocija) (**Gen**+**APos**)	protivn<u>ičin</u>	nasledn<u>ičin</u>	vlasn<u>ičin</u>	A1+Pos

Tabela 2.1

osnovna lema	**račun**	**stan**	**soliter**	**N1-Hum**
Deminutiv	r<u>ačunčić</u>	stančić	soliterčić	N27-Hum
Augmentativ	*računčina*	st<u>ančina</u>	*soliterčina*	N600-Hum

Tabela 2.2

Nesistematičnost obrade lema u uobičajenim leksikografskim opisima ilustrovana je primerima iz datih tabela. Osnovne leme, *protivnik*, *naslednik*, *vlasnik* regularnom derivacijom proizvode izvedenice na sličan način, kao što se vidi u prvoj tabeli, ali samo neke izvedene leme su predstavljene u RMSMH, i to ne nužno one koje se pojavljuju u korpusu. Npr, lema *naslednički* postoji u RMSMH, ali se ne pojavljuje u korpusu. S druge strane, prisvojni pridev *protivnikov* nije zastupljen u rečniku, ali jeste u korpusu. Slično, lema deminutiva *računčić* postoji u RMSMH, ali se ne pojavljuje u korpusu, dok se augmentativi *soliterčina* and *računčina* pojavljuju u korpusu iako nisu zastupljeni u rečniku (Tabela 2.2). Međutim, za one izvedene leme, zastupljene u rečniku, date su definicije preko regularnih obrazaca. Npr, opis značenja prisvojnog prideva imenice *X* se najčešće definiše sa "*koji pripada X*", deminutiv imenice *X* je definisan sa "*deminutiv od X*", dok za mociju roda postoje dve mogućnosti: "*žena koja je X*" ili "*supruga od X*".

[80] **N10**, **N651**, **N1**, **N27** predstavljaju oznake flektivnih klasa imeničkih lema, dok **A1**, **A2** predstavljaju oznake flektivnih klasa pridevskih lema. Ostatak morfosintaksičkog opisa predstavlja prisustvo (označeno znakom +) ili odsustvo (označeno znakom **-**) određene sintaksičke ili semantičke karakteristike leme (tako +**Hum** označava da je u pitanju ljudsko biće, dok –**Hum** označava da lema ne predstavlja ljudsko biće; +**Pos** označava prisvojni pridev, dok +**Rel** označava relacioni pridev).

Važno svojstvo regularne derivacije je da, ukoliko je u pitanju sufiksacija, tada sve osnovne leme koje pripadaju istoj flektivnoj klasi generišu izvedene leme koje pripadaju tačno određenoj flektivnoj klasi precizno definisanoj u sistemu rečnika DELA. Tako, sve osnovne leme iz klase **N10+Hum**, regularnom derivacijom grade, korišćenjem mocionog sufiksa *-ka*, izvedenice iz flektivne klase **N651+Hum**, a takođe i prisvojne prideve iz flektivne klase **A1**, i relacione prideve iz klase **A2** (Tabela 2.1). Takođe, imenice muškog roda iz klase **N1**, grade deminutive (takođe muškog roda) koji pripadaju klasi **N27**, kao i augmentative ženskog roda koji pripadaju klasi **N600**. Ukratko, ako imenica pripada klasi **N10+Hum**, na osnovu nje se mogu izvesti leme na način opisan u prvoj tabeli, pri čemu svaka izvedenica pripada unapred određenoj flektivnoj klasi koja zavisi samo od sufiksa, a ne i od motivne imenice.

Direktna posledica ovog svojstva je mogućnost da se regularna derivacija klasifikuje u sistemu elektronskih rečnika na sličan način kao što je to urađeno sa fleksijom. Naime, ono što po svaku cenu treba izbeći jeste ugradnja lema izvedenih regularnom derivacijom u e-rečnik. Time bi se proizvele sledeće neželjene posledice:

a) višestruko povećanje dimenzija e-rečnika,

b) složeniji proces održavanja e-rečnika,

c) povećanje obima višeznačnosti tokom morfološke analize.

S druge strane, ako u sastav rečnika uđu samo leme dobijene regularnom derivacijom koje su prisutne u papirnatim rečnicima, dolazi do ozbiljne nekonzistentnosti (Tabela 2.1 i Tabela 2.2). Na kraju, ako se leme dobijene regularnom derivacijom tretiraju kao posebne leme, u e-rečniku biće izgubljene veze između motivne reči i njenih izvedenica, što će učiniti nemogućom analizu relacija sinonimije pomoću e-rečnika.

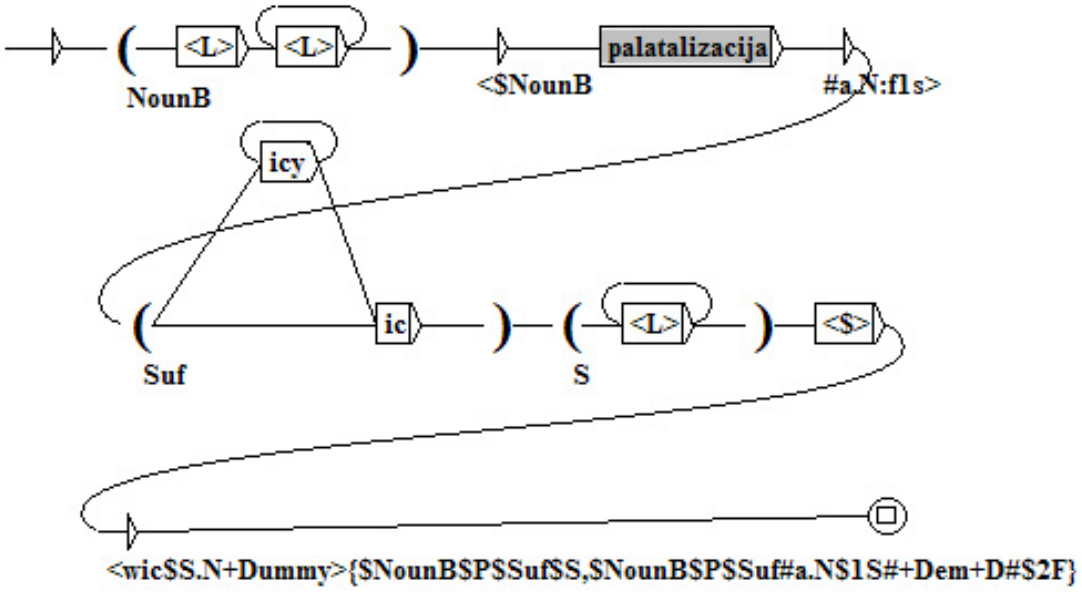
Da bismo ilustrovali ovo poslednje, razmotrimo primer paralelnog francusko-srpskog teksta Volterovog *Kandida* ([Vitas 05a]). Dok se u francuskom tekstu pojavljuju samo flektivni oblici leme *baron*, dotle se u srpskom prevodu, kao odgovarajuće reči, pojavljuju flektivni i derivacioni oblici od *baron* (*baronica*, *baronov*, *baroničica*).

Rešenje je da se konstruišu **lokalne morfološke gramatike** kojima bi se na osnovu sadržaja rečnika aproksimiralo prisustvo lema izvedenih regularnom derivacijom i tako omogućilo njihovo prepoznavanje.

Primer jedne lokalne morfološke gramatike koja prepoznaje neke deminutive imenica 3. grupe ilustruje Slika 2.2 ([Vitas 07a]). Cilj ove gramatike je da proba da svede nepoznatu reč, potencijalni deminutiv, na reč iz e-rečnika iz koje je taj deminutiv izveden. Najpre transduktor pokušava da prepozna segment derivacione osnove na koji ne utiču fonetske alternacije prilikom dodavanja sufiksa (i "pamti" ga u promenljivoj **NounB**), zatim eventualni alternirani segment (za to je zadužen poseban transduktor *palatalizacija*) i derivacioni sufiks (u ovom primeru *–ic* i *-icyic*), i na kraju flektivni nastavak ("pamti" se u promenljivoj **S**)., Pri tom transduktor *palatalizacija* zamenjuje alternirani segment derivacione osnove njegovim originalom (*č -> k*, *ž -> g*, *š -> h*). Kada tranduktor tako "rekonstruiše" osnovu motivne reči, dodaje flektivni nastavak za nominativ jednine imenica 3. grupe (*-a*) i proverava da li dobijena reč postoji u e-rečniku; ukoliko je to slučaj, tekstuelna reč će biti obeležena kao njen deminutiv.

Npr, tekstuelna reč *biljčice* se prepoznaje na sledeći način: u zagradama **NounB** se "pamti" niz slova (*bilj*), za kojim možda sledi palatalizirano *č*, koje treba vratiti u *k*. Ako se na tako dobijeni niz slova (*biljk*) dopiše *a* (*#a*), za dobijenu reč *biljka* se ispostavlja da je u e-rečniku obeležena kao imenica ženskog roda u nominativu singulara (**N:f1s**). Stoga, ostatak transduktora definiše da je *biljčice* flektivni oblik deminutiva od *biljka* i na osnovu

prepoznatog flektivnog nastavka (-*e*) pridružuje mu sve vrednosti gramatičkih kategorija koje bi imalo pojavljivanje flektivnog oblika *biljke* (dakle, isti rod, broj i padež).



Slika 2.2

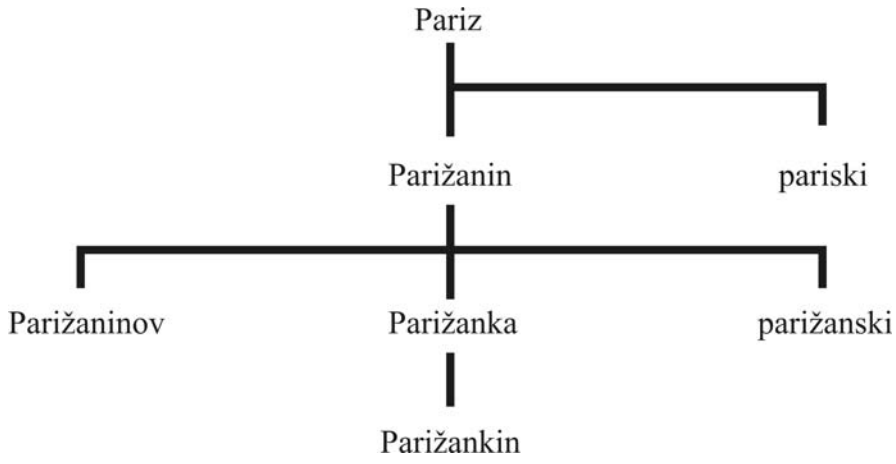
Takođe važna osobina regularne derivacije jeste da neki sufiksi, iako se koriste za izvođenje reči sa potpuno predvidljivim značenjem, mogu za neke leme da proizvedu izvedenice sa potpuno promenjenim značenjem. Npr, iako su *kašičica*, *svećica* izvedene od *kašika*, *sveća* (prvobitno kao deminutivi), danas imaju sasvim druga značenja koja ukidaju deminutivno značenje. Slično, *domaćica*, prvobitno mocioni parnjak reči *domaćin*, danas ima novo značenje – "nezaposlena udata žena". Ovo svojstvo određuje prioritete izvedenih oblika tokom prilikom morfološke analize. Naime, da bi se izbegla nepotrebna višeznačnost, lokalne gramatike koje opisuju regularnu derivaciju treba primeniti isključivo na nepoznate reči, dakle, na reči kojih nema u e-rečniku. Tako se izbegava da pomenuti primeri (*kašičica*, *svećica*, *domaćica*) budu pogrešno prepoznati kao deminutivi, odnosno rezultat mocije roda.

2.1.3 Regularna derivacija i superlema

Pristup koji koriste lokalne gramatike prilikom prepoznavanja rezultata regularne derivacije podrazumeva da se tekstuelna reč prilikom morfološke analize može svesti na više "lema" (kanonskih oblika), pri čemu sada, pored "flektivne leme" (tj. leme o kakvoj smo dosad govorili), možemo govoriti i o "derivacionoj lemi" (tj. o lemi motivne reči)[81]. Primetimo da "derivaciona lema" nije jedinstvena i da, u stvari, "derivacione leme" formiraju određenu hijerarhiju.

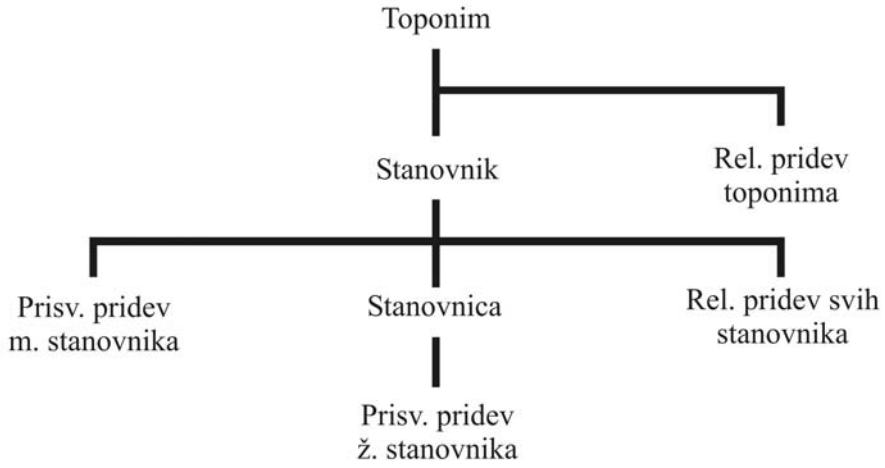
[81] U prethodnom primeru *biljka* je "derivaciona lema" za izvedenicu *biljčica*.

To se možda najbolje može ilustrovati na primeru regularne derivacije od toponima. Slika 2.3 prikazuje drvo hijerarhije koje uspostavljaju leme izvedene od imenice *Pariz* regularnom derivacijom.



Slika 2.3

Uopšteno drvo hijerarhije indukovano regularnom derivacijom toponima prikazuje Slika 2.4. Ovde treba pomenuti da postoje i drugi derivacioni procesi vezani za toponime (izvođenje deminutiva i augmentativa, npr. *Beograd – Beograđanin – Beograđančić*; izvođenje glagola *Nemačka – Nemac – ponemčiti*).



Slika 2.4

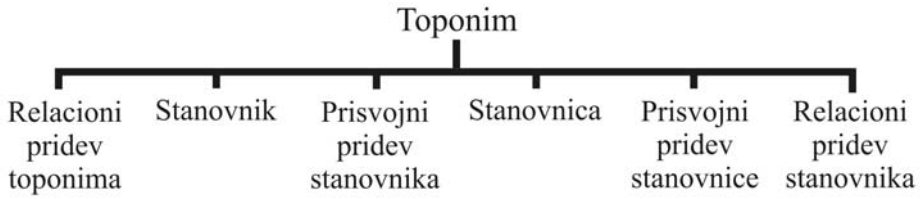
Problem sa ovim modelom je što u praksi nalazimo primere u kojima relacioni pridev nije izveden od toponima, već obratno: postoje mnogobrojni toponimi čija su imena nastala poimeničenjem prideva, pre svega u ženskom[82] rodu (*turski – Turska*; *grčki – Grčka*). Stoga ovde, umesto o hijerarhiji motivnih i motivisanih lema, pre ima smisla govoriti o hijerarhiji

[82] U [Klajn 03] se beleže i primeri toponima nastalih poimeničenjem prideva u srednjem rodu (*Brčko*, *Topusko*) i muškom rodu (*Imotski*), uz napomenu da u slučaju gradova osnove prideva mogu imati poseban oblik (*brčanski*).

njihovih značenja: na osnovu superleme sa značenjem "toponim *X*" se izvode leme sa značenjem "stanovnik od *X*", "koji se odnosi na *X*", "koji pripada stanovniku od *X*", "stanovnica od *X*", "koji pripada stanovnici od *X*", "koji se odnosi na stanovnike od *X*". Samim tim prestaje i potreba za prethodnom hijerarhijom (Slika 2.4), već se sve može prikazati mnogo jednostavnije (Slika 2.5).

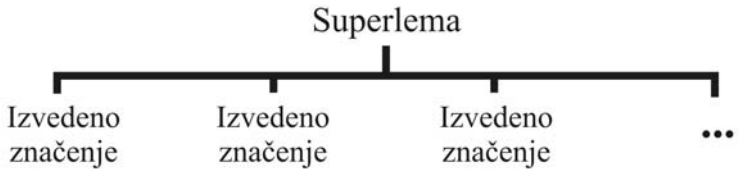
Tako dolazimo do pojma **superleme**, kao "osnovnog značenja" iz kog su izvedena sva ostala značenja: Slika 2.6 prikazuje koncept superleme u opštem slučaju. Njime se uopšte ne razmatra kojim nizom izvođenja je neka reč izvedena i koja je njena direktna motivna reč, već su jedino bitna izvedena značenja. U slučaju regularne derivacije ta izvedena značenja su uvek predvidljiva.

Treba istaći da koncept superleme nema za cilj da se suprotstavlja postojećim derivacionim modelima u morfologiji, niti da ih grubo pojednostavljuje. Glavno pitanje koje se ovde postavlja je kako bi taj koncept mogao da se iskoristi za modeliranje regularne derivacije u postojećim morfološkim elektronskim rečnicima.



Slika 2.5

Ideja na kojoj se temelji ovaj rad jeste da se u morfološki elektronski rečnik unose isključivo superleme sa pridruženim pravilom regularne derivacije koje opisuje generisanje odgovarajućih izvedenica. Tako bi se aproksimiralo prisustvo lema izvedenih regularnom derivacijom i omogućilo njihovo prepoznavanje tokom morfološke analize.



Slika 2.6

Modeliranje regularne derivacije u morfološkim elektronskim rečnicima korišćenjem koncepta superleme je moguće ukoliko se prethodno izvrši klasifikacija superlema na osnovu pravila regularne derivacije; jednu tako dobijenu klasu bi činile sve superleme koje na istovetan način (istim pravilom) formiraju svoje izvedenice regularnom derivacijom.

Naredne glave opisuju eksperiment sa klasifikacijom regularne derivacije od toponima i formalizuju postupak klasifikacije.

3

PRINCIPI KLASIFIKACIJE REGULARNE DERIVACIJE OD TOPONIMA

U ovom delu se najpre uvode definicije termina koji su u uskoj vezi sa toponimima i njhovim izvedenicama. Dat je i pregled njihovih osnovnih karakteristika u srpskom jeziku u slučajevima kada su izvedenice dobijenim regularnom derivacijom. Potom se razmatra moguća automatska klasifikacija toponima na osnovu pravila koja opisuju kako se regularnom derivacijom dobijaju njihove izvedenice. Tom prilikom se analiziraju principi koje treba uzeti u obzir pre nego što se pristupi samoj klasifikaciji. Takođe se formalno zasnivaju pravila regularne derivacije uvođenjem N-regularnih izraza i N-derivacionih pravila.

3.1 Toponimi, etnici i ktetici

Onomastika je posebna grana lingvistike koja proučava vlastita imena. Obuhvata **antropomastiku** (proučava imena, prezimena i nadimke ljudi - **antroponime**), i **toponomastiku** (proučava geografska imena – **toponime**). Posebne vrste toponima su **hidronimi** (imena vodenih površina), **oronimi** (imena planina), **oikonimi** (imena naselja, gradova, sela i sl.), **horonimi** (imena prirodnih ili administrativnih teritorija), **dromonimi** (imena puteva, autostrada), **agronimi** (imena poljoprivrednih prostornih objekata) itd.

Imena stanovnika naseljenog mesta se nazivaju **etnici** (npr. *Beograđanin*, *Beograđanka*), dok se relacioni/prisvojni pridevi izvedeni od etnika nazivaju **ktetici** (npr. *beogradski*, *beograđanski*, *Beograđaninov*, *Beograđankin*).

Izvedenice koje se dobijaju regularnom derivacijom od toponima su upravo one koje prikazuju Slika 2.4 (opšti slučaj) i Slika 2.3 (konkretan primer za toponim *Pariz*): ime stanovnika (*Parižanin*) i njemu odgovarajući prisvojni pridev (*Parižaninov*), ime stanovnice (*Parižanka*) i njoj odgovarajući prisvojni pridev (*Parižankin*), ktetik koji se odnosi na mesto (*pariski*) i ktetik koji se odnosi na stanovnike (*parižanski*).

Najveću pravilnost tokom derivacije pokazuju ktetici, kao i imena stanovnica. S druge strane, flektivne klase kojima pripadaju muška imena stanovnika su raznovrsne: N2 (*Mađar*), N10 (*Slovak*), N14 (*Čeh*), N42 (*Portugalac*), N60 (*Bugarin*), N741 (*Nišlija*) itd. Međutim, dva sufiksa za izvođenje etnika (a samim tim i njima pridružene flektivne klase) dominiraju po učestanosti: *-in* (pre svega njegovi alomorfi *-anin*/*-janin*, flektivna klasa N60) i *–ac* (N42).

Ženska imena stanovnika se završavaju alomorfom nekog od sufiksa *-ka*, *-inja*, *-ica*. Sufiks *-ka* kome odgovara flektivna klasa N661, je dominantan po učestanosti, pošto najčešće dolazi kao zamena za muške sufikse *-ac* i *-in* (*Portugalac – Portugalka*, *Bugarin – Bugarka*), ali se javlja i kao zamena za druge muške sufikse (*Nišlija – Nišlijka*). Sufiksi *-inja* (N601) i *-ica* (N651) su manje produktivni, ali i manje pravilni jer se u pojedinačnim slučajevima javljaju kao zamena različitih muških sufiksa (*Danac – Dankinja*, *Sremac – Sremica*; *Srbin – Srpkinja*; *Slovak – Slovakinja*, *Hrvat – Hrvatica*).

Relacioni pridevi završavaju se nekim od alomorfa sufiksa *-ski* (npr. *beogradski*, *niški*, *bečki*, *pećki*) i pripadaju flektivnoj klasi A2. Prisvojni pridevi se završavaju nekim od alomorfa sufiksa *-ov*/*-ev* (izvedeni od muškog imena stanovnika, npr. *Beograđaninov*, *Anadolčev*), odnosno *-in* (izvedeni od muškog i/ili ženskog imena stanovnika, npr. *Nišlijin*, *Nišlijkin*) i pripadaju flektivnoj klasi A1.

3.2 Automatska klasifikacija regularne derivacije od toponima

3.2.1 Principi klasifikacije

Glavni kriterijum prilikom određivanja etnika i ktetika za neki toponim jeste način na koji njegovi stanovnici nazivaju sami sebe; taj način ponekad odstupa od onog koji se očekuje na osnovu uobičajenih pravila derivacije, ali uvek ima prvenstvo (derivacione osnove mogu da se redukuju[83] ili čak promene, npr. *Prištevac* je etnik za toponim *Priština*, *Moskovljanin* je etnik za toponim *Moskva* itd).

Prilikom navođenja etnika u tekstu, posebno onih izvedenih od stranih toponima, često se oseća dilema autora koji od dva dominantna sufiksa *-(j)anin* i *-ac* da upotrebi. Tako se u Korpusu savremenog srpskog jezika pojavljuju i uobičajeniji *Izraelac*, ali i *Izraelićanin*, kao etnici toponima *Izrael*, pri čemu ovaj drugi u prevodima književnog (Jan Potocki, *Rukopis nađen u Saragosi*) i filozofskog dela (Mirča Eliade, *Mefistofeles i Androgin*). Slično, za toponim *Tuzla*, za koji su u papirnatim rečnicima potvrđeni etnici *Tuzlak* i *Tuzlanin*, Google daje primere i za *Tuzlanac* (jedan primer *Tuzlaci* se pojavljuje i u [Korpus 06]). Takođe, Google daje primere istovremeno i za *Jamajkanac*, *jamajkanski*, odnosno *Jamajčanin*, *jamajčanski*, itd.

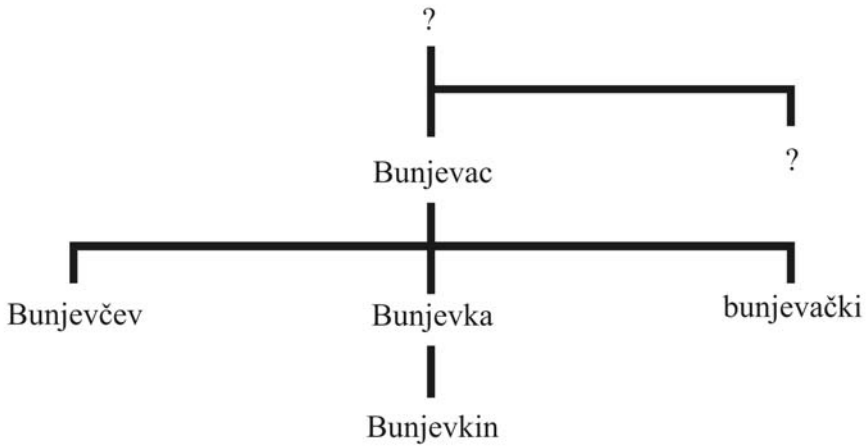
U slučaju toponima sa sufiksom *-ija* čiji etnik ima alomorf sufiksa *-ac* se takođe javljaju dileme ([Klajn 03]). Naime, sem uobičajenih sufiksa *-ac* (*Slovenija – Slovenac*) i *-anac* (*Austrija – Austrijanac*), pojavljuje se u novije vreme i *-ijac*, tako da se u tekstovima sem *Somalac*, *Tanzanac*, *Bolivijanac*, *Kolumbijanac* pojavljuju i *Somalijac*, *Tanzanijac*,

[83] Termin *redukovana osnova* je preuzet iz [Klajn 03] u vezi pojave da su izvedenice "izvedene od reči sa sufiksom, ali je taj sufiks otpao prilikom novog izvođenja. Takve su recimo *Vojvođanin* od *Vojvod-ina*, ... *Sarajlija* od *Saraj-evo* ...."

Bolivijac, *Kolumbijac*. Ovde treba spomenuti i da se u sredstvima javnog informisanja sve češće čuje *Baskijac* umesto *Bask*, kao i *Baskija* iako ovaj toponim ne postoji u originalu[84].

Postavlja se pitanje da li u rečnik treba unositi i "pogrešne" varijante etnika i ktetika, jer svojom učestalošću predstavljaju realnost u tekstu. Često se javlja dilema da li ih uzimati u obzir prilikom klasifikacije regularne derivacije.

S tim u vezi treba spomenuti i derivaciju jednog broja **etnonima**, tj. imena naroda, za koje ili ne postoji toponim od kog su izvedeni (*Bunjevac*, *Ciganin*, *Rom*, *Filistejac*, *Agarjanin*, *Danajac*) ili pak postoje motivni toponim(i), ali značenje izvedenica nije predvidljivo (*Indijanac*, *Samarićanin*, *Indoevropljanin*, *Afroamerikanac*, *Anglosaksonac*). Njihov model derivacije je gotovo istovetan sa modelom regularne derivacije od toponima, a glavna razlika je u tome što za njih ne postoji toponim koji bi bio odgovarajuća superlema (Slika 3.1).



Slika 3.1

U vezi sa hijerarhijskim opisom regularne derivacije od toponima (Slika 2.4) postoji problem koji zahteva diskusiju. Naime, kod većine toponima kod kojih se muški etnik završava sufiksom *-(j)anin*, a u nekim slučajevima i *-anac*, postoje dva različita relaciona pridevi na *-ski*: jedan izveden od toponima, a drugi od etnika (npr. *Banat – banatski* i *Banaćanin – banaćanski*; *Meksiko – meksički* i *Meksikanac – meksikanski*). U ostalim slučajevima postoji samo jedan relacioni pridev koji se odnosi i na toponim i njegove stanovnike. U prvom slučaju ne postoji saglasnost da li oba relaciona prideva imaju istovetna značenja, tj. da li se oba odnose i na toponim i na stanovnike ili se jedan odnosi samo na toponim, a drugi samo na etnik. Prvu tezu zastupa [Klajn 03] dok se potvrde za oba tvrđenja mogu naći u postojećim tomovima Rečnika Srpske akademije nauka i umetnosti (njih 17), kao i tri poslednja toma Rečnika Matice srpske:

a) svaki pridev ima posebno značenje, tj. jedan se odnosi na toponim, a drugi na stanovnike (*banatski*/*banaćanski*, *niški*/*nišlijski*, *egipatski*/*egipćanski*);
b) oba prideva imaju ista značenja, tj. oba se odnose i na toponim i na stanovnike (*sremski*/*sremački*, *indonezijski*/*indonežanski*, *norveški*/*norvežanski*);

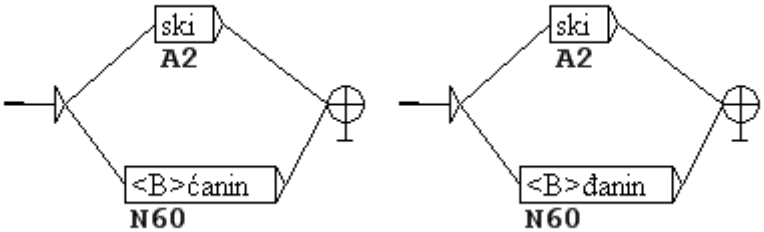
[84] Uobičajeni zajednički naziv za tri provincije na severu Španije koje naseljavaju Baski do ustava iz 1978. godine bio je *provincias vascongadas* (bukvalni prevod *baskijske provincije*), a danas one predstavljaju autonomnu zajednicu *Pais vasco* (bukvalni prevod *Baskijska zemlja*) što se u sredstvima javnog informisanja skraćuje u *Baskija* ([Klajn 03]).

c) jedan pridev se odnosi samo na toponim/etnik, a drugi na oba (*meksički* = "koji se odnosi na *Meksiko*"; *meksikanski* = "koji se odnosi na *Meksiko* i *Meksikance*");

Prilikom implementacije klasifikacije se postavlja pitanje kako tretirati slučajeve kada postoji samo jedan relacioni pridev: da li treba pretpostaviti da je u pitanju par ktetika gde jedan element nedostaje (tj. gde je pridev izveden ili samo od toponima ili samo od etnika, npr. *portugalski*/∅, ∅/*vojvođanski*) ili tretirati ktetike kao parove gde se elementi poklapaju (tj. prilikom izvođenja od toponima/etnika je dobijena istovetna izvedenica *portugalski*/*portugalski*; *vojvođanski*/*vojvođanski*).

Dodatne probleme prilikom automatske klasifikacije predstavljaju fonetske alternacije. Naime, sufiksi su podložni dubokim glasovnim promenama, pre svega jotovanju i palatalizaciji (*Beograd – Beograđanin*, *Lika – Ličanin*), ali ima i primera jednačenja po zvučnosti (*Zagreb – Zagrepčanin*), gubljenja suglasnika (*Perast – peraški*) itd. Ove promene zahvataju i osnovu i sufiks, stvarajući tako alomorfe sufiksa što je naročito primetno kod sufiksa *-ski* (*-ski*, *-ški*, *-čki*, *-ćki*).

Fonetske alternacije tokom automatske klasifikacije izazivaju mnogo finije diferenciranje toponima na klase nego što to čini derivaciona morfologija ([Utvić 06b]). Npr, sa stanovišta derivacione morfologije, od toponima *Banat* i *Tajland* se na isti način izvode etnici i ktetici sufiksima *–janin*, odnosno *-ski*, ali u slučaju etnika dolazi do jotovanja (*t* + *j* = *ć* i *d* + *j* = *đ*), te dobijamo *Banaćanin*, *banatski* i *Tajlanđanin*, *tajlandski*. Kada bismo to opisali transduktorima (analogno opisu flektivne paradigme datom u odeljku 1.2.3.2, Slika 1.14), dobili bismo različite rezultate (Slika 3.2). Suština ovog opisa je da se dodavanjem završetka *-ski* na lemu dobija relacioni pridev (iz flektivne klase **A2**), a da se u slučaju etnika (imenica iz flektivne klase **N60**) najpre briše poslednji karakter leme (operatorom **<B>** programa NooJ) i onda dodaje odgovarajući završetak *-ćanin*, odnosno *-đanin*. Postavlja se pitanje da li je moguće zameniti ova dva transduktora jednim ekvivalentnim transduktorom.



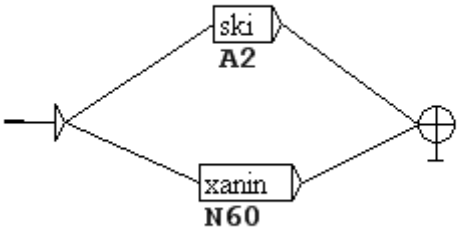
Slika 3.2

Jedna ideja je primeniti pogodno kodiranje teksta. U uvodnoj glavi smo opisali kodnu shemu, korišćenu u sistemu **aurora** ([Vitas 81]), koja preslikava dijakritičke karaktere i digrafe koji se koriste u srpskoj latinici na način koji samo delimično odslikava parove alternirajućih suglasnika prilikom jotovanja (Tabela 3.1). Eventualnom modifikacijom koda bi suglasnik *ć* mogao biti kodiran sa *tx* umesto sa *cx*. U tom slučaju se regularna derivacija od toponima *Banat* i *Tajland* opisuje istovetnim transduktorom (Slika 3.3). Međutim, ovakav pristup bi ugrozio princip da se elementarne operacije prilikom derivacije (i fleksije) uvek vrše nad slovima alfabeta[85] (*Banat -> Bana* + *ćanin* = *Banaćanin*) a ne nad karakterima (*Banat -> Banat* + *xanin* = *Banatxanin* = *Banaćanin*).

[85] Što se tiče programa kakvi su NooJ ili Unitex, elementi kodne sheme *sx*, *zx*, *cx* itd. su slova srpskog alfabeta.

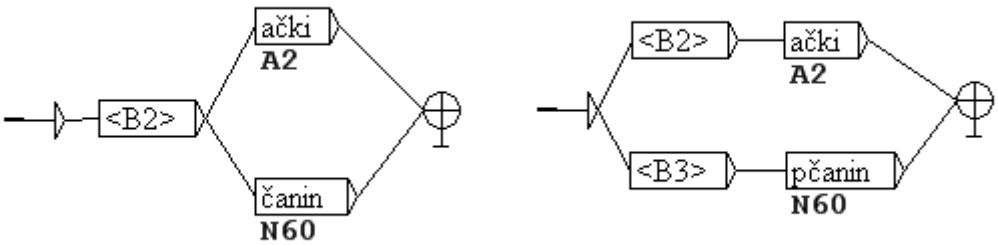
Nenepčani suglasnik	Prednjonepčani suglasnik	AURORA kôd prednjonepčanog suglasnika	Eventualna modifikacija koda
s	š	sx	
z	ž	zx	
n	nj	nx	
l	lj	lx	
d	đ	dx	
t	ć	cx	tx

Tabela 3.1



Slika 3.3

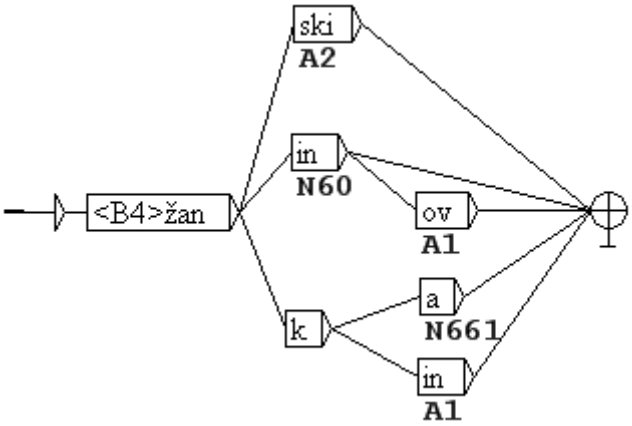
Druga ideja je eventualno proširivanje modula koji vrše automatsku fleksiju i derivaciju operatorima koji bi automatski realizovali pojedine glasovne promene. Naime, uobičajeni operatori nad lemama prilikom automatske fleksije i derivacije su uglavnom ograničeni na brisanje i dopisivanje karaktera, kao i na pozicioniranje u okviru niza karaktera koji čine lemu. Sem jotovanja, primer korisnog operatora bi bilo automatsko jednačenje po zvučnosti. Prema derivacionoj morfologiji, od toponima *Leskovac* i *Šabac* se na isti način izvode etnici i ktetici sufiksima *–janin*, odnosno *-ski*, ali u slučaju etnika *Šapčanin* dolazi do jednačenja po zvučnosti (*Šabac -> Šapc-janin -> Šapčanin*). Opis pomoću transduktora daje različite rezultate (Slika 3.4).



Slika 3.4

Svi prethodni primeri transduktora koji se koriste za opis flektivnih i derivacionih paradigmi se zasnivaju na tome da motivna lema i izvedena lema imaju zajednički početni segment; transduktor opisuje kako se na osnovu motivne leme "računa" taj segment i kako se dodavanjem odgovarajućih završetaka dobijaju odgovarajući oblici flektivne ili derivacione paradigme. Iz prethodnih primera se jasno vidi da se ti segmenti ne moraju poklapati sa onim što se u morfologiji naziva flektivna, odnosno derivaciona osnova, pre svega zbog fonetskih alternacija. Takođe, zajednički početni segment iste leme i različitih izvedenica ne mora biti iste dužine. To je posebno karakteristično za ktetik izveden od toponima i etnika (npr. *Egipat – egipat-ski*; *Egipat – Egip-ćanin*). Stoga je bolje za svaki od njih posebno računati zajednički početni segment, a time i odgovarajuće pravilo izvođenja, da se ne bi proizvodili veštački sufiksi (poput *Egip-atski*, *Egip-ćanin*).

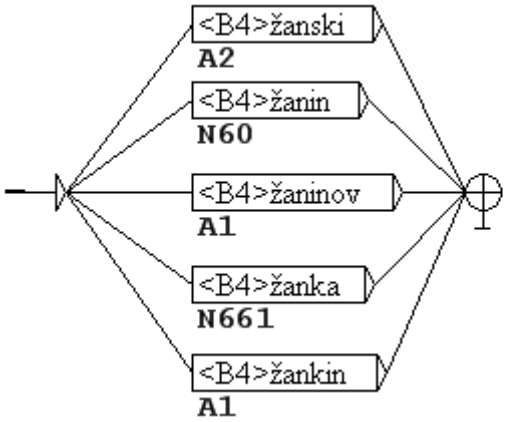
S druge strane, u slučaju jednog od najučestalijih sufiksa za izvođenje etnika -(*j*)*anin*, (a takođe i -*ac*, kad mu je odgovarajući ženski sufiks -*ka*) zajednički početni segment etnika (muškog i ženskog stanovnika), i njima odgovarajućih prisvojnih prideva je uvek isti (*Indonežan-in*, *Indonežan-ka*, *Indonežan -inov*, *Indonežan-kin*; *Valjev-ac*, *Valjev-ka*, *Valjev-čev*, *Valjev-kin*). Stoga ima smisla razmotriti mogućnost da se u takvom slučaju zajednički početni segment računa samo jednom, a da se potom na osnovu njega i lema izvedenica izračunaju odgovarajući završeci (Slika 3.5).



Slika 3.5

Ovakav pristup je u uskoj vezi sa determinizacijom i minimizacijom automata, odnosno transduktora. Današnji sistemi koji omogućavaju automatsku fleksiju i derivaciju (NooJ, Unitex) sadrže u sebi module koji automatski vrše determinizaciju i minimizaciju automata i transduktora, i njihovi autori naglašavaju da je bolje da korisnik kreira transduktor koji je pregledan i koji se lako održava, a da sistemu prepusti da obavi odgovarajuće optimizacije (Slika 3.6).

Postojeće implementacije takvih sistema ne podržavaju da se jednim transduktorom opiše celokupna derivaciona paradigma. Npr, sistem NooJ omogućava da se jednim transduktorom opiše derivacija više izvedenica isključivo pod uslovom da sve one imaju istovetan opis flektivne paradigme. Samim tim, u slučaju lema izvedenih od toponima regularnom derivacijom umesto jednog transduktora bilo bi potrebno kreirati četiri (po jedan za muškog i ženskog stanovnika, jedan za relacione prideve na -*ski*, i jedan za prisvojne prideve na -*ov*). Ovo otvara pitanje da li se treba vezati isključivo za postojeće implementacije i svaki tip izvedenice posebno klasifikovati (dakle, posebno svaki etnik i ktetik) ili treba klasifikovati opise celokupnih derivacionih paradigmi (i etnika i ktetika).



Slika 3.6

3.2.2 N-regularni izrazi i N-derivaciona pravila

Definicija 1. **N-regularni izrazi** nad azbukom Σ i skupom oznaka operatora $\mathcal{O}$[86] definišu se rekurzivno na sledeći način:

(i) Svaka oznaka operatora *o* iz skupa $\mathcal{O}$ je N-regularni izraz nad (Σ, $\mathcal{O}$);

(ii) Proizvoljni simbol *a* azbuke Σ je N-regularni izraz nad (Σ, $\mathcal{O}$);

(iii) Neka su r_1 i r_2 N-regularni izrazi nad (Σ, $\mathcal{O}$). Tada su

a. $r_1 \sqcup + \sqcup r_2$ (alternacija)[87],
b. $r_1 r_2$ (dopisivanje) i
c. (r_1) takođe N-regularni izrazi nad (Σ, $\mathcal{O}$);

(iv) N-regularni izrazi nad (Σ, $\mathcal{O}$) se dobijaju isključivo konačnom primenom pravila (*i*)-(*iii*).

Za sve N-regularne izraze *p*, *q*, *r* nad (Σ, $\mathcal{O}$) važi zakoni

$(pq)r = p(qr)$ (asocijativnost dopisivanja),
$(p \sqcup + \sqcup q) \sqcup + \sqcup r = p \sqcup + \sqcup (q \sqcup + \sqcup r)$ (asocijativnost alternacije),
$p \sqcup + \sqcup q = q \sqcup + \sqcup p$ (komutativnost alternacije),
$(p \sqcup + \sqcup q)r = pr \sqcup + \sqcup qr$ (distributivnost dopisivanja u odnosu na alternaciju).

Takođe važi da alternacija ima manji prioritet od dopisivanja. Prioritet se može izbeći korišćenjem zagrada, a pravilo (*iii*) c. asocijativni zakoni za dopisivanje i alternaciju omogućavaju da se zagrade oko izraza mogu izostaviti ukoliko to ne menja njegovo značenje.

---

[86] Radi kraćeg i jednostavnijeg izražavanja, još ćemo reći da je u pitanju N-regularni izraz nad (Σ, $\mathcal{O}$).

[87] Simbolom $\sqcup$ je naglašen karakter za razmak koji je sastavni deo N-regularnog izraza.

N-regularne izraze u kojima se ne pojavljuje ni alternacija ni zagrade nazivaćemo **atomičnim** N-regularnim izrazima. Atomični N-regularni izrazi su zapravo niske nad azbukom $\Sigma \cup \mathcal{O}$. Koristeći distributivnost dopisivanja u odnosu na alternaciju, kao i asocijativnost dopisivanja i alternacije, svaki N-regularni izraz r se može transformisati u ekvivalentan N-regularni izraz r' koji se sastoji od konačno mnogo atomičnih N-regularnih izraza, međusobno povezanih operatorom alternacije ($\sqcup + \sqcup$). Taj N-regularni izraz r', ekvivalentan izrazu r, nazivaćemo **normalnom formom** N-regularnog izraza r.

Definicija 2. Neka je r N-regularni izraz nad (Σ, $\mathcal{O}$), i $\hat{r}$ njegova normalna forma. Tada postoji konačno mnogo atomičnih N-regularnih izraza $a_1, a_2, \ldots, a_n$ takvih da je $\hat{r} = a_1 \sqcup + a_2 \sqcup + \sqcup \ldots \sqcup + \sqcup a_n$. Neka je, dalje, Φ skup morfosintaksičkih opisa[88] i $\varphi_1, \varphi_2, \ldots, \varphi_n \in \Phi$. Tada $a_1 / \varphi_1 \sqcup + \sqcup a_2 / \varphi_2 \sqcup + \sqcup \ldots \sqcup + \sqcup a_n / \varphi_n$ predstavlja **N-derivaciono pravilo** nad azbukom Σ, skupom oznaka operatora $\mathcal{O}$ i skupom morfosintaksičkih opisa Φ[89]; sam r se naziva N-regularnim delom pravila $a_1 / \varphi_1 \sqcup + \sqcup a_2 / \varphi_2 \sqcup + \sqcup \ldots \sqcup + \sqcup a_n / \varphi_n$.

N-derivaciona pravila možemo smatrati proširenjem N-regularnih izraza. Za njih takođe važe zakoni navedeni u *Definiciji 1* (asocijativnost dopisivanja, komutativnost alternacije i distributivnost dopisivanja u odnosu na alternaciju).

Primer 1. Neka je Σ skup karaktera koji u kodnoj shemi sistema aurora predstavljaju velika i mala slova srpskog latiničnog pisma; neka je skup oznaka operatora $\mathcal{O} = \{\langle \mathbf{B}n \rangle \mid n \in \mathbb{N}\}$, gde **<B***n***>** označava operator brisanja n karaktera levo od tekuće pozicije (podrazumevana tekuća pozicija je desni kraj niske); i neka je skup morfosintaksičkih opisa Φ = {**N60**, **N661**, **A1**, **A2**}. Tada je <B2>acyki primer jednog atomičnog N-regularnog izraza, dok su

<B2>cyanin/N60 ␣ + ␣ <B2>cyaninov/A1 ␣ + ␣ <B2>cyanka/N661 ␣ + ␣ <B2>cyankin/A1,

odnosno

<B2>(cyanin/N60 ␣ + ␣ cyaninov/A1 ␣ + ␣ cyanka/N661 ␣ + ␣ cyankin/A1)

ekvivalentna N-derivaciona pravila nad (Σ, $\mathcal{O}$, Φ). Njima je opisana derivaciona paradigma toponima *Leskovac*.

Definicija 3. Neka je $\mathbb{N}$ skup prirodnih brojeva[90], Σ azbuka, $\mathcal{O}$ skup oznaka operatora i Φ skup morfosintaksičkih opisa. Neka je svakom $o \in \Sigma \cup \mathcal{O}$ pridružena po jedna funkcija $\overline{o} : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$. **N-derivaciona paradigma** niske $x \in \Sigma^*$ indukovana N-derivacionim pravilom r je skup $DeriveAll(x, r)$, pri čemu važi:

$$DeriveAll(x, r) = DeriveAll(x, a_1 / \varphi_1 \sqcup + \sqcup a_2 / \varphi_2 \sqcup + \sqcup \ldots \sqcup + \sqcup a_n / \varphi_n) = \bigcup_{k=1}^{n} \{derive(x, a_k) / \varphi_k\}$$

$$derive(x, a_k) = derive(x, o_{k1} o_{k2} \cdots o_{km}) = \pi_1 \circ \overline{o}_{km} \circ \cdots \circ \overline{o}_{k2} \circ \overline{o}_{k1}(x, |x|)$$

---

[88] Videti odeljak 2.1.2.

[89] Radi kraćeg i jednostavnijeg izražavanja, još ćemo reći da je u pitanju N-derivaciono pravilo nad (Σ, $\mathcal{O}$, Φ).

[90] Kao što je to uobičajeno, podrazumevaćemo da je i nula prirodan broj.

$$r = a_1 / \varphi_1 \sqcup + \sqcup a_2 / \varphi_2 \sqcup + \sqcup \ldots \sqcup + \sqcup a_n / \varphi_n ,$$

$$a_k = o_{k1} o_{k2} \cdots o_{km} ,$$

$$o_{k1}, o_{k2}, \cdots o_{km} \in \Sigma \cup \mathcal{O},$$

$$\pi_1 : \Sigma^* \times \mathbb{N} \to \Sigma^*, \ \pi_1(x, n) = x \ \ (x \in \Sigma^*, n \in \mathbb{N}).$$

Primer 2. Neka je u prethodnom primeru svakom slovu azbuke $c \in \Sigma$ pridružena funkcija $\overline{c} : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$, koja u datu nisku x umeće slovo c na poziciju p, tj. $\overline{c}(x, p) = (ycz, p+1)$ gde je $x = yz, \ |y| = p,$ uz ograničenje $0 \le p \le |x|$. Neka je oznaci **<B*n*>,** $n \in \mathbb{N}$, pridružena funkcija $\overline{\mathrm{B}}n : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$, koja u datoj niski x briše n karaktera levo od pozicije p, tj. $\overline{\mathrm{B}}n(x, p) = (yw, p-n)$, gde je $x = yzw, \ |yz| = p, \ |z| = n,$ uz ograničenje $0 \le n \le p \le |x|$. Tada N-derivaciona paradigma niske *Leskovac* indukovana N-derivacionim pravilom

r = <B2>cyanin/N60 ⌴ + ⌴ <B2>cyaninov/A1 ⌴ + ⌴ <B2>cyanka/N661 ⌴ + ⌴ <B2>cyankin/A1

predstavlja skup

$DeriveAll(x, r) = \{$*Leskovcyanin*/N60, *Leskovcyaninov*/A1, *Leskovcyanka*/N661, *Leskovcyankin*/A1$\}$.

Definicija 4. Neka je P skup N-derivacionih pravila nad (Σ, $\mathcal{O}$, Φ) i $F : \Sigma^* \to P$ preslikavanje. Relacija ~ nad Σ^* definisana je na sledeći način: za $t_1, t_2 \in \Sigma^*$ važi da je $t_1 \sim t_2$ ako i samo ako je $F(t_1) = F(t_2)$. Neposrednom proverom se vidi da je ~ relacija ekvivalencije na Σ^*.

Klase ekvivalencije relacije ~ predstavljaju **N-derivacione klase** indukovane preslikavanjem *F*. Jedna N-derivaciona klasa obuhvata sve elemente skupa Σ^* kojima odgovara isto N-derivaciono pravilo nad (Σ, $\mathcal{O}$, Φ) kao slika pri preslikavanju *F*. Skup N-derivacionih klasa indukovanih preslikavanjem *F* (u oznaci K_F) predstavlja jednu particiju skupa Σ^*.

Za preslikavanje $[-]_F : \Sigma^* \to K_F, \ [-]_F : t \mapsto [t]_F$, $t \in \Sigma^*$, koje svakoj niski nad azbukom Σ dodeljuje njenu N-derivacionu klasu, reći ćemo da predstavlja **klasifikaciju** elemenata skupa Σ^* indukovanu preslikavanjem *F*.

Primer 3. Ako koristeći pretpostavke i definicije iz prethodna dva primera, definišemo preslikavanje *F* koje niskama *Leskovac* i *Lazarevac* pridružuje isto N-derivaciono pravilo *r*, te dve niske će pripadati istoj N-derivacionoj klasi.

4

IMPLEMENTACIJA AUTOMATSKOG KLASIFIKATORA REGULARNE DERIVACIJE OD TOPONIMA

U ovom delu je opisana implementacija automatskog klasifikatora regularnih derivacionih paradigmi toponima. Za formalno predstavljanje pravila kojima se opisuje derivaciona paradigma toponima korišćeni su regularni izrazi u formatu NooJ. Program je realizovan korišćenjem programskog jezika Visual C#, kako bi se tokom programiranja više pažnje posvetilo rešavanju problema klasifikacije, a ne konstruisanju pratećeg grafičkog okruženja. Posebno je obrađeno klasifikovanje derivacionih paradigmi višečlanih toponima. Navedeni su izvori za resurse korišćene za testiranje programa (toponimi, etnici i ktetici), kao i dobijeni rezultati (derivacione klase).

4.1 Izvori za opis derivacionih paradigmi toponima

Pre nego što se pristupi klasifikovanju toponima na osnovu njihovog mehanizma regularne derivacije, neophodno je najpre pripremiti odgovarajuće resurse (regularne derivacione paradigme toponima) u elektronskom formatu pogodnom za njihovu automatsku obradu. Stoga je za potrebe elektronskog formata resursa primenjena ista kodna shema koja je korišćena u sistemu **aurora** ([Vitas 81]) i koju koristi Grupa za jezičke tehnologije na Matematičkom fakultetu u Beogradu kao internu kodnu shemu za implementaciju sistema morfoloških rečnika srpskog jezika u formatu DELA i formatu NooJ ([Vitas 93b], [Vitas 01], [Vitas 03], [Krstev 05c]). Tu kodnu shemu ćemo nadalje nazivati **kôd aurora**.

Kodom aurora je neutralizovan uticaj različitih pisama (ćirilica ili latinica) i kodnih shema (ISO 646 IRV, ISO-8859-2 ili ISO-8859-5, Win CP 1250 ili Win CP 1251, Unicode itd) kojima se mogu zapisati lekseme srpskog jezika. To je postignuto korišćenjem standardnog ASCII koda (koji nema specijalne karaktere za naša slova sa dijakriticima, kao ni pojedinačne karaktere koji bi označavali digrafne simbole) i uvođenjem zapisa slova koji omogućava predstavljanje slova sa dijakriticima, kao i jednoznačno razlikovanje digrafa od

konsonantskih grupa. Aurora kôd preslikava dijakritičke karaktere i digrafe koji se koriste u srpskoj latinici (Tabela 2 i Tabela 3 uvodne glave).

Sami resursi su prikupljeni iz više izvora. Najpre su uzeti u obzir postojeći resursi u elektronskom obliku, dok su papirne verzije resursa korišćene uglavnom za kontrolu i dopunu prikupljenih podataka. Kao osnova resursa upotrebljeni su elektronski rečnici geografskih imena (*ascdelas-top*) Grupe za jezičke tehnologije na Matematičkom fakultetu u Beogradu ([Vitas 03], [Krstev 05c]) i Korpus savremenog srpskog jezika ([Korpus 06]). Za kontrolu i dopunu resursa korišćeni su [Klajn 02] (tvorba složenica) i [Klajn 03] (sufiksacija), a [Kristal 96] je konsultovan pre svega za kontolu i dopunu ktetika[91]. U obzir su isključivo uzimani podaci za koje je potvrđeno da se koriste u savremenom srpskom jeziku. Stoga su ređe konsultovani postojeći tomovi Rečnika Srpske akademije nauka i umetnosti (njih 17), kao i tri poslednja toma Rečnika Matice srpske. Naime, ovi rečnici teže da obuhvate sveukupnu leksiku srpskog jezika u vremenskom rasponu od Dositeja i Vuka do danas, te se u njima mogu naći primeri etnika i ktetika koji se u savremenom srpskom jeziku koriste retko (*Španjkinja*, *amerikanski*) ili se uopšte ne koriste (*Indijanin*, *Srbljanin*, *Jugoslavljanin*, *Ništevljanin*). Tabela 4.1 prikazuje raspodelu dostupnih i korišćenih resursa.

U postojećoj verziji rečnika toponima *ascdelas-top* korišćeni su podaci iz školskog geografskog atlasa i spiska naseljenih mesta u bivšoj Jugoslaviji Saveznog zavoda za statistiku ([Vitas 05c]). Za svaki region predstavljen u atlasu napravljen je izbor imena koja su uneta u rečnik i to: imena država, zvaničnih jezika, glavnih gradova, administrativnih jedinica od opšte važnosti (npr, savezne države u okviru SAD), gradova sa preko 10000 stanovnika u Srbiji i Crnoj Gori, odnosno sa preko 50000 stanovnika u bivšim jugoslovenskim republikama, odnosno sa preko 100000 stanovnika u ostalim oblastima; hidronima (jezera, močvare, reke), oronima (planine, vulkani) itd. U tekućoj verziji rečnika DELA-Top nalaze se i odgovarajući etnici i ktetici za pojedine toponime.

<table>
<tr><td colspan="4">Toponimi</td><td colspan="4">Derivacione paradigme</td></tr>
<tr><td colspan="2">Ukupni resursi (prisutni u el. rečniku i ostali)</td><td colspan="2">Elektronski rečnici geografskih imena ([Krstev 05c])</td><td colspan="4">Korpus savremenog srpskog jezika ([Korpus 06]), Tvorba reči u savremenom srpskom jeziku ([Klajn 02] i [Klajn 03])</td></tr>
<tr><td rowspan="2">jednočlani toponimi</td><td rowspan="2">višečlani toponimi</td><td rowspan="2">jednočlani toponimi</td><td rowspan="2">dvočlani toponimi</td><td colspan="2">jednočlani toponimi</td><td colspan="2">dvočlani toponimi</td></tr>
<tr><td>-ac</td><td>-anin</td><td>-ac</td><td>-anin</td></tr>
<tr><td rowspan="2">12788</td><td rowspan="2">2523</td><td rowspan="2">992</td><td rowspan="2">215</td><td>293</td><td>331</td><td>104</td><td>80</td></tr>
<tr><td colspan="2">624</td><td colspan="2">184</td></tr>
<tr><td colspan="2">15311</td><td colspan="2">1207</td><td colspan="4">806</td></tr>
</table>

Tabela 4.1

[Korpus 06] je korišćen za ekstrakciju etnika (na *–ac* i *-anin*) i ktetika. Kandidati za ekstrakciju su dobijeni korišćenjem proširenih regularnih izraza[92] podsistema CQP (eng. *Corpus Query Processor*) u okviru sistema za upravljanje korpusima IMS Corpus

[91] [Kristal 96] sadrži na kraju iscrpan indeks naziva jezika od kojih većina predstavlja ktetike

[92] Npr, prošireni regularni izraz **^[A-Z][a-z]*(ac|ca|cu|ce|cye|com|ci|aca|cima)$** je korišćen za ekstrakciju kandidata za etnike na *–ac*. Ovaj regularni izraz zadovoljavaju sve niske koje počinju velikim slovom, a završavaju se nekom od niski *ac*,*ca*,*cu*,*ce*,*cye*,*com*,*ci*,*aca*,*cima* (tj, završecima padežnih oblika etnika na *-ac*).

Workbench[93]. Identifikovanje etnika i ktetika među ekstrahovanim kandidatima je obavljeno ručno zbog toga što je *-anin* karakterističan sufiks za prezimena (*Bićanin*, *Gračanin*), a *-ac* i za toponime (*Lazarevac*, *Doljevac*) i za prezimena (*Pokrajac*, *Graovac*).

Prilikom klasifikacije u obzir su uzeti jedino jednočlani i dvočlani toponimi. Naime, derivacione paradigme toponima sa tri ili više članova (npr. *Bačko Petrovo selo*) je lakše direktno uneti u rečnik, pre svega zato što ih ima relativno malo u odnosu na ostale toponime. Sa druge strane, rezultat eventualne klasifikacije za *n*-točlane toponime ($n \geq 3$) bi verovatno bio takav, da skoro svaki od njih indukuje posebnu jednočlanu klasu (eng. singleton).

Derivacione paradigme toponima su podeljene na dve grupe u zavisnosti od toga da li se radi o jednočlanim i dvočlanim toponimima. Posle toga je svaka od grupa podeljena na tri podgrupe u zavisnosti od sufiksa kojim se izvodi odgovarajući muški etnik (*-ac*, *-anin* i ostali). Klasifikacija je obavljena nad dobijenim podgrupama.

4.2 Sistem NooJ

4.2.1 Rečnici toponima u formatu NooJ

U okvira rečnika u formatu NooJ leksičke odrednice se mogu povezati sa formalnim opisom svoje flektivne i/ili derivacione paradigme ([Silberztein 06]). Formalni opis paradigme proizvoljne leme ne nabraja njene oblike, odnosno izvedenice, već određuje kako se odgovarajuća paradigma može generisati na osnovu leme. To daje mogućnost da se leksičke odrednice klasifikuju u flektivne (odnosno derivacione) klase, tako da jednu klasu čine leme iste vrste reči (imenice, pridevi, glagoli itd) čiji je formalni opis fleksije (odnosno regularne derivacije) istovetan. U tom slučaju se (flektivna ili derivaciona) klasa može identifikovati sa formalnim opisom paradigme svojih elemenata. Ukoliko se tako dobijene klase pogodno označe, i svakoj lemi se pridruži oznaka klase kojoj pripada, na osnovu leme i njoj pridružene oznake klase može se automatski generisati njena paradigma.

Pridruživanje oznaka flektivnih i derivacionih klasa lemama je omogućeno posebnim predefinisanim svojstvima +**FLX** (za flektivne klase) i +**DRV** (za derivacione klase). Opšti oblik odrednice u rečniku može se predstaviti na sledeći način:

lema,PoS+FLX=Cxxx{+DRV=Dxxx[:Fxxx]}{+SynSem}

gde **PoS** predstavlja oznaku za vrstu reči (npr. **N** za imenicu, **A** za pridev, **V** za glagol itd), **Cxxx** je oznaka flektivne klase leme, **Dxxx** označava derivacionu klasu leme, a +**SynSem** predstavlja proizvoljni sintaksički ili semantički marker. **Fxxx** je oznaka zajedničke flektivne klase izvedenica koje se mogu generisati na osnovu leme i opisa derivacione paradigme **Dxxx**; vitičaste zagrade sugerišu da su pridružene derivacione paradigme i sintaksičko-semantički markeri neobavezni, ali i da odrednici može biti pridruženo više derivacionih paradigmi, odnosno markera.

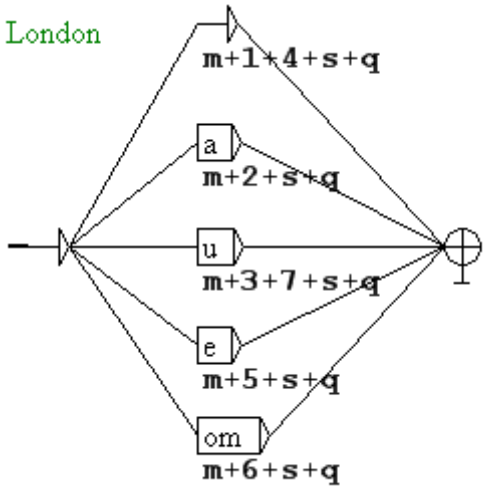
Evo jednog primera iz rečnika geografskih imena u formatu NooJ (*ascdelas-top.dic*) za srpski jezik ([Vitas 03], [Krstev 05c]):

London,N+FLX= N1001+DRV=AC:AcFlx+DRV=SKI:SkiFlx+NProp+Top+IsoUK+PGgr (1)

[93] Autori sistema su Oliver Christ i Bruno Maximilian Schulze iz Instituta za obradu prirodnih jezika (IMS) Univerziteta u Štutgartu. Videti npr. http://www.ims.uni-stuttgart.de/projekte/CorpusWorkbench/

Zapis (1) označava lemu *London*, imenicu (*N*) čija je oznaka flektivne klase N1001 (*+FLX=N1001*). Lema *London* pripada dvema derivacionim klasama (*AC* i *SKI*) koje opisuju kako se od leme regularnom derivacijom generišu izvedenice *Londonac*, odnosno *londonski*. *AcFlx* označava flektivnu klasu izvedenica (*Londonac*) generisanih na osnovu derivacione klase *AC* (*+DRV=AC:AcFlx*), dok *SkiFlx* predstavlja flektivnu klasu izvedenica (*londonski*) generisanih na osnovu derivacione klase *SKI* (*+DRV=SKI:SkiFlx*). Lemi *London* su pridružena četiri sintaksičko-semantička markera koji označavaju da se radi o vlastitoj imenici (*+NProp*), toponimu (*+Top*) koji pripada državi čija je međunarodna oznaka *UK* (*+IsoUK*), i koji je ujedno i njen glavni grad (*+PGgr*).

Opis paradigme se može kreirati u nekom od dva postojeća formata: grafičkom (datoteke tipa *.nof*) ili tekstuelnom (datoteke tipa *.flx*). Prvi format za opis koristi grafove (Slika 4.1 prikazuje graf *N1001.nof* koji opisuje flektivnu paradigmu označenu sa *N1001*). Tekstuelni format prilikom opisivanja koristi morfografemska pravila koja su detaljno objašnjena u odeljku 4.2.2.



Slika 4.1

U okviru rečnika mora postojati referenca na datoteke sa opisima (flektivnih i derivacionih) paradigmi korišćenih u zapisu leksičkih odrednica. Opšti oblici takvih referenci su

#use OpisParadigmeGrafom.nof
#use OpisParadigmePravilima.flx

gde se prvim oblikom referiše na datoteke sa grafovima (*.nof*), a potonjim na datoteke sa pravilima (*.flx*). Da bi se izbeglo referisanje na svaki opis paradigme ponaosob, takve datoteke mogu sadržati više opisa paradigmi, tj. datoteke tipa *.nof* mogu sadržati više grafova, a datoteke tipa *.flx* — više morfografemskih pravila.

Rezultat kompilacije rečnika u okviru sistema NooJ (*ascdelas-top.dic*) i opisa paradigmi (nezavisno od formata) je minimalni deterministički transduktor (*ascdelas-top.nod*). Generisani transduktor sadrži sve leksičke odrednice (leme) iz polaznog rečnika zajedno sa svim njihovim oblicima flektivne i derivacione paradigme, a kojima su

pridružene odgovarajuće informacije o lemi (kanonski oblik leme, vrsta reči, oznaka flektivne klase, semantički markeri), kao i dodatne morfološke informacije (npr. rod, broj, padež itd). Time se postiže isti efekat kao kada bi skup zapisa polaznog rečnika (*ascdelas-top.dic*) bio proširen zapisima poput

Londonom,London,N+FLX=N1001+DRV=AC:AcFlx+DRV=SKI:SkiFlx+NProp+Top+IsoUK+PGgr+m+6+s+q (2)

Zapis (2) predstavlja oblik *Londonom* leme *London* sa pridruženim informacijama čije je značenje isto kao u zapisu (1), uz dodatne informacije da je lema muškog roda (+*m*) i da označava nešto neživo (+*q*), kao i da njen oblik *Londonom* predstavlja instrumental (+*6*) jednine (+*s*).

Semantički markeri koji se koriste u postojećoj verziji rečnika toponima *ascdelas-top.dic* usklađeni su sa odgovarajućim kodnim rečima sistema Prolintex ([Maurel 96]). Sem opštih markera, pomenutih u primerima (1) i (2), u upotrebi su i specifični semantički markeri:

- +Top toponim (npr. Valjevo)
- +Hyd hidronim (npr. Dunav)
- +Oro oronim (npr. Jahorina)
- +Lang jezik (npr. gelski)
- +PDrz država (npr. Francuska)
- +Ppust pustinja (npr. Sahara)
- +PRav ravnica (npr. Panonija)
- +Preg region (npr. Banat)
- +PGgr glavni grad države (npr. Kairo)
- +PGr1, +PGr2, +PGr3,+PGr4 različite (rastuće) veličine gradova (npr. Atina za +PGr4)
- +PDgr gradska četvrt (npr. Palilula)
- +POps opština (npr. Zemun)

Izvedenicima se mogu pridružiti i dodatni markeri, npr. +*Hum* (označava da je u pitanju ljudsko biće), +Inh (označava da je u pitanju stanovnik):

Londonac,London,N+FLX=AcFlx+DRV=AC:AcFlx+DRV=SKI:SkiFlx+NProp+Top+IsoUK+PGgr+Hum+Inh

4.2.2 Morfografemska pravila sistema NooJ

U okviru sistema NooJ ([Silberztein 06]) tekstuelni format za opis flektivne i derivacione paradigme koristi morfografemska pravila. U zavisnosti od tipa paradigme (flektivna ili derivaciona), pravila za njen opis se delimično razlikuju. Naime, oblici flektivne paradigme pripadaju istoj vrsti reči kao i njihova lema, pa je ta informacija suvišna prilikom opisivanja paradigme. S druge strane, izvedenice derivacione paradigme ne moraju pripadati istoj vrsti reči kao njihova motivna lema, i stoga se informacija o vrsti reči kojoj pripada izvedenica uvek mora navesti u odgovarajućem pravilu. U sistemu NooJ definicija morfografemskog pravila (nezavisno od tipa paradigme) ima sledeći oblik:

nazivOpisaParadigme = **morfografemskoPravilo**;

gde je **nazivOpisaParadigme** proizvoljni identifikator, tj. niska karaktera koji mogu biti velika i mala slova engleske abecede, dekadne cifre i podvlaka ('_'), pri čemu prvi karakter identifikatora mora biti slovo ili podvlaka. Svaka definicija pravila se mora završiti simbolom ';'.

Morfografemska pravila za opis flektivne paradigme biće ilustrovana primerom u odeljku 4.2.2.1. Stroga definicija morfografemskih pravila za opis derivacione paradigme biće navedena u odeljku 4.2.2.4.

4.2.2.1 Opis flektivne paradigme morfografemskim pravilom

Primer (3) definiše morfografemsko pravilo *N1001* kojim se, između ostalog, opisuje flektivna paradigma toponima *London*:

N1001 =
<E>/m+1+4+s+q␣+␣a/m+2+s+q␣+␣u/m+3+7+s+q␣+␣e/m+5+s+q␣+␣om/m+6+s+q; (3)

Na osnovu zapisa leksičke odrednice *London* u rečniku (odeljak 4.2.1, zapis (1)) i pravila *N1001* sistem NooJ generiše odgovarajuće flektivne oblike sa pridruženim informacijama:

London,London,N+FLX= N1001+NProp+Top+IsoUK+PGgr+m+1+4+s+q
Londona,London,N+FLX= N1001+NProp+Top+IsoUK+PGgr+m+2+s+q
Londonu,London,N+FLX= N1001+NProp+Top+IsoUK+PGgr+m+3+7+s+q
Londone,London,N+FLX= N1001+NProp+Top+IsoUK+PGgr+m+5+s+q
Londonom,London,N+FLX= N1001+NProp+Top+IsoUK+PGgr+m+6+s+q

Pravilo *N1001* opisuje koje završetke treba dopisati na lemu (*a*, *u*, *e*, *om*; <E> označava praznu nisku) da bi se generisao odgovarajući oblik i koje mu morfološke informacije treba pridružiti (*+m* označava muški rod, oznake od *+1* do *+7* označavaju odgovarajuće padeže, *+s* predstavlja jedninu, a *+q* da je u pitanju nešto neživo); završeci i morfološke informacije su odvojene simbolom kosom crtom ('/'), a opisi pojedinačnih oblika simbolom '+' oko koga moraju postojati beline (karakter za razmak ␣ ili znak za novi red).

4.2.2.2 Operatori sistema NooJ

Mnoge leme nisu levi faktori (prefiksi) svojih oblika flektivne i derivacione paradigme. Stoga su, pored dopisivanja, neophodne i druge morfografemske transformacije polazne leme kojima bi se generisali odgovarajući oblici paradigme. NooJ koristi desetak[94] takvih transformacija koje se nazivaju operatorima ([Silberztein 06]). Ovi operatori se mogu

[94] Ovde su navedeni opšti operatori sistema NooJ koji se koriste od strane većine (evropskih) jezika. Međutim, sistem dopušta i definisanje korisničkih operatora specifičnih za pojedine jezike (npr. operatori koji slovima dodaju odgovarajući akcenat ili pak uklanjaju akcenat). Detalji su objašnjeni u [Silberztein 06].

posmatrati kao funkcije dva argumenta, tj. funkcije oblika $\Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$, gde je Σ azbuka koja se koristi za zapis leksičkih odrednica u rečniku, a koja zavisi od izabranog jezika i pisma[95]. Prvi argument ovih funkcija predstavlja nisku x karaktera nad), a drugi tekuću poziciju u okviru te niske izraženu prirodnim brojem p, $0 \le p \le |x|$. Vrednost funkcije je uređen par čiji prvi član predstavlja transformisanu nisku y nad azbukom Σ, a drugi tekuću poziciju u okviru transformisane niske izraženu prirodnim brojem q, $0 \le q \le |y|$. Svaki operator je označen odgovarajućim velikim slovom latinice između uglastih zagrada ('<' i '>'). Slede objašnjenja pojedinačnih operatora ($x, y, z \in \Sigma^*$, $c \in \Sigma$, $0 \le p \le |x|$):

- <E> dopisivanje prazne niske (eng. **E**mpty string); tj. $\text{<E>}(x, p) = (x, p)$;
- <B> brisanje prethodnog karaktera u odnosu na tekuću poziciju (eng. **B**ackspace); tj. $\text{<B>}(x, p) = (yz, p-1)$, gde je $x = ycz,\ |yc| = p$;
- <S> brisanje narednog karaktera u odnosu na tekuću poziciju (eng. **S**upress); tj. $\text{<S>}(x, p) = (yz, p)$, gde je $x = ycz,\ |y| = p$;
- <D> dupliranje prethodnog karaktera u odnosu na tekuću poziciju (eng. **D**uplicate); tj. $\text{<D>}(x, p) = (yccz, p+1)$, gde je $x = ycz,\ |yc| = p$;
- <L> pomeranje levo za jedan karakter u odnosu na tekuću poziciju (eng. go **L**eft); tj. $\text{<L>}(x, p) = (x, p-1)$, gde je $1 \le p \le |x|$;
- <R> pomeranje desno za jedan karakter u odnosu na tekuću poziciju (eng. go **R**ight); tj. $\text{<R>}(x, p) = (x, p+1)$, gde je $0 \le p < |x|$;
- <C> zamena velikog slova odgovarajućim malim i obrnuto (eng. change **C**ase); tj. $\text{<C>}(x, p) = (y\tilde{c}z, p)$, gde je $x = ycz,\ |yc| = p,$ a c i $\tilde{c}$ predstavljaju par odgovarajućih slova (veliko i malo slovo, odnosno malo i veliko slovo).

Posebnu grupu operatora čine operatori za rad sa niskama koje ili predstavljaju višečlane lekseme (leksičke kompozite) ili se dobijaju njihovom transformacijom korišćenjem operatora sistema NooJ. Takve niske nazivaćemo **višečlane niske** (eng. **Multi Word Unit**, skr. **MWU**) nasuprot jednočlanim niskama (eng. **Single Word Unit**, skr. **SWU**)**.** Razlika između višečlanih i jednočlanih niski se svodi na to da prva grupa niski sadrži jedan ili više karaktera za razmak, dok druga grupa niski ne. Karakter za razmak će nadalje biti označen simbolom ␣ radi jasnog isticanja. Delovi višečlane niske, međusobno razdvojeni separatorom ␣ predstavljaju **članove** niske (eng. **word forms**). U skladu sa definicijama iz odeljka 1.1, u prirodno-jezičkom dokumentu članovi višečlane niske se svode na formalne reči. Za brzo pozicioniranje u okviru niski koja se sastoji od dve ili više formalnih reči, koriste se sledeći operatori ($x, y, z, w \in \Sigma^*$, $0 \le p, q \le |x|$):

- <P> pomeranje na (desni) kraj prethodne formalne reči u odnosu na tekuću poziciju (eng. **P**revious word form); tj. $\text{<P>}(x, p) = (x, q)$, ako je $x = y\text{␣}zw, |y\text{␣}z| = p, |y| = q$ i niska z ne sadrži ␣. U slučaju da prethodna formalna reč ne postoji, $\text{<P>}(x, p) = (x, 0)$;

[95] Konkretno, za srpski jezik postoje tri verzije leksičkih resursa od kojih svaka koristi različitu azbuku: 1) karaktere srpske latinice kodne sheme Unicode; 2) karaktere srpske ćirilice kodne sheme Unicode; 3) karaktere koda aurora. Odgovarajući zapisi leksičke odrednice *Kruševac* simbolima ovih azbuka su: 1) *Kruševac*; 2) *Крушевац*; 3) *Krusxevac*.

- <N> pomeranje na (desni) kraj sledeće formalne reči u odnosu na tekuću poziciju (eng. **N**ext word form); tj. $\text{<N>}(x,p)=(x,q)$, gde je $x=yz␣w␣v, |y|=p, |yz␣w|=q$ i niske z, w i v ne sadrže ␣. U slučaju da sledeća formalna reč ne postoji, $\text{<N>}(x,p)=(x,|x|)$.

S obzirom da operatori sistema NooJ imaju isti domen i kodomen, otvorena je mogućnost njihove kompozicije. Umesto uobičajen matematičke notacije za kompoziciju $\text{<Y>}\circ\text{<X>}=\text{<Y>}(\text{<X>}(x,p))$, gde <X>, <Y> predstavljaju proizvoljne operatore sistema NooJ, koristićemo jednostavno označavanje <X><Y>.

Navedene definicije operatora sistema NooJ su strogo formalne. Međutim, ovi operatori su u [Silberztein 06] definisani opisno preko primera, uz korišćenje drugačije terminologije. Na osnovu te terminologije, operatori NooJ-a mogu imati samo jedan od dva moguća opciona argumenta. Prvi takav opcioni argument može biti prirodan broj $n\geq 1$ koji označava da se dejstvo operatora ponavlja n puta (u slučajevima kada to ima smisla). Npr. <Bn> označava brisanje prethodnih n karaktera[96]. Takođe, pojedini operatori kao opcioni argument mogu imati karakter W (od eng. whole word), a značenje zavisi od konkretnog operatora ($x,y,z,w\in\Sigma^*, 0\leq p,q\leq|x|$):

- <BW> brisanje svih karaktera levo u odnosu na tekuću poziciju, tj. $\text{<BW>}(x,p)=(z,0)$, ako je $x=yz, |y|=p$;
- <SW> brisanje svih karaktera desno u odnosu na tekuću poziciju, tj. $\text{<SW>}(x,p)=(y,p)$, ako je $x=yz, |y|=p$;
- <LW> pomeranje na početak (levi kraj) tekuće formalne reči, tj. $\text{<LW>}(x,p)=(x,q)$, ako je $x=y␣zw, |y␣z|=p, |y␣|=q$ i niska z ne sadrži ␣; odnosno, $\text{<LW>}(x,p)=(x,0)$, ako $x=yz, |y|=p$ i niska y ne sadrži ␣. Primetimo da operator <LW> nije ekvivalentan sa <P><R>. Naime, ako niska x ne sadrži ␣, tada je $\text{<P><R>}(x,p)=(x,1)\neq(x,0)=\text{<LW>}(x,p)$ s obzirom da je na osnovu definicije $\text{<P>}(x,p)=(x,0)$;
- <RW> pomeranje na (desni) kraj tekuće formalne reči, tj. $\text{<RW>}(x,p)=(x,q)$, ako je $x=yz␣w, |y|=p, |yz|=q$ i niska z ne sadrži ␣; odnosno, $\text{<RW>}(x,p)=(x,|x|)$, ako $x=yz, |y|=p$ i niska z ne sadrži ␣;
- <PW> pomeranje na (desni) kraj prve formalne reči, tj. $\text{<PW>}(x,p)=(x,q)$, ako je $x=y␣zw, |y␣z|=p, |y|=q$ i niska y ne sadrži ␣;
- <NW> pomeranje na kraj poslednje formalne reči (a time i na sam kraj niske), tj. $\text{<NW>}(x,p)=(x,|x|)$.

Neki zanimljivi primeri kompozicija operatora su:

- <LW><R><C> promena početnog velikog slova tekuće formalne reči u odgovarajuće malo i obrnuto.<LW><BW> briše sve karaktere levo od tekuće formalne reči;
- <PW><SW> briše sve osim prve formalne reči;

[96] Iako se tada operator <B> može posmatrati kao funkcija tri argumenta, s obzirom na način na koji smo definisali kompoziciju operatora NooJ-a, zgodnije je posmatrati {**<Bn>**| $n\in\mathbb{N}$ } kao familiju operatora.

- <PW><LW> pomeranje na početak niske koja se sastoji iz više formalnih reči, a tekuća pozicija nije unutar prve od njih;

4.2.2.3 Funkcije umetanja (slova)

Svakom slovu azbuke $c \in \Sigma$ može se u sistemu NooJ pridružiti funkcija $\overline{c} : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$, koja u datu nisku *x* umeće slovo *c* na poziciju *p*, tj. $\overline{c}(x, p) = (ycz, p+1)$ -gde je $x = yz,\ |y| = p,$ uz ograničenje $0 \le p \le |x|$ $(x, y, z \in \Sigma^*)$. Takve funkcije ćemo nazivati funkcije umetanja nad azbukom Σ i označavaćemo ih isto kao i odgovarajuće slovo.

S obzirom da operatori NooJ-a i funkcije umetanja nad azbukom Σ imaju isti domen i kodomen, dobro je definisana njihova kompozicija. Za označavanje kompozicije funkcija umetanja i/ili NooJ operatora *f* i *g*, umesto standardne matematičke notacije $g \circ f$ koristićemo oznaku *fg*. Tabela 4.2 prikazuje kako se računa vrednost kompozije operatora <P><B>o<S><R><C><RW><B>ac za nisku $Crna \sqcup Gora$ i njenu dužinu (9); radi bolje preglednosti, tekuća pozicija u okviru niski (međurezultata) je podvučena, a krajnji rezultat je predstavljen podebljanim fontom.

4.2.2.4 Opis derivacione paradigme morfografemskim pravilom

Neka je Σ[97] skup karaktera kodne sheme Unicode; neka je $\mathcal{O}$ skup oznaka operatora NooJ-a kojima su pridružene odgovarajuće funkcije $\overline{o} : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$ (operatori NooJ-a). U skladu sa pojmovima uvedenim u odeljku 3.2.2, NooJ-regularni izrazi se mogu definisati kao N-regularni izrazi nad $(\Sigma, \mathcal{O})$[98].

Neka je *P* skup oznaka vrsta reči u srpskom jeziku[99] koje koristi sistem NooJ za azbuku Σ, a *S* skup odgovarajućih sintaksičko-semantičkih markera[100]. Oba skupa su konačna i stoga regularna. Regularan skup $\Phi = P(\{+\}S)^*$ predstavlja skup morfosintaksičkih opisa koje koristi sistem NooJ[101]. Na osnovu pojmova uvedenih u odeljku 3.2.2, morfografemska pravila za opis derivacione paradigme u sistemu NooJ se mogu definisati kao N-derivaciona pravila nad $(\Sigma, \mathcal{O}, \Phi)$.

---

[97] Leksički resursi srpskog jezika u sistemu NooJ koriste kao azbuku sledeće podskupove karaktera kodne sheme Unicode: 1) skup karaktera srpske latinice; 2) skup karaktera srpske ćirilice; 3) skup karaktera koda aurora. Za svaku od ovih azbuka postoji posebna verzija resursa.

[98] Sistem NooJ koristi regularne izraze ne samo za opis flektivnih i derivacionih paradigmi, već i za opis sintaksičkih konstrukcija, kao i za zadavanje upita kojima se pretražuju kolekcije tekstova indeksiranih tim sistemom (na nivou karaktera i na nivou tekstuelnih reči). S obzirom da kompletna definicija takvih regularnih izraza izlazi iz okvira ovog rada, ovde je data definicija koja određuje samo onaj njihov deo koji sistem NooJ koristi za opis paradigmi. Stoga se u definiciji nigde ne pominje Klinijevo zatvorenje, što sugeriše da su transduktori generisani na osnovu morfografemskih pravila uvek aciklični.

[99] Neke od tih oznaka su *N* (imenica), *A* (pridev), *V* (glagol) itd.

[100] Npr. marker *Hum* označava da je u pitanju živo biće, *Inh* označava stanovnika, *Pos* ukazuje da se radi od prisvojnom pridevu itd. Neki od sintaksičko-semantičkih markera su detaljno opisani u odeljku 4.2.1.

[101] Neki elementi skupa Φ su *N*, *A*, *V*, *N+Hum+Inh*, *A+Pos* itd.

Skup izvedenica koje sistem NooJ generiše na osnovu leme $x \in \Sigma^*$ i morfografemskog pravila *r* je upravo N-derivaciona paradigma niske *x* indukovana N-derivacionim pravilom *r*, tj. skup $DeriveAll(x, r)$ (videti odeljak 3.2.2).

<P><B>o<S><R><C><RW><B>ac(*Crna␣Gora*, 9)				
Sadržaj niske pre primene operatora	Tekuća pozicija pre primene operatora	Komanda NooJ-a	Sadržaj niske posle primene operatora	Tekuća pozicija posle primene operatora
Crna␣Gora_	9	operator <P>	Crna␣Gora	4
Crna␣Gora	4	operator <B>	Crn␣Gora	3
Crn␣Gora	3	umetnuto slovo *o*	Crno␣Gora	4
Crno␣Gora	4	operator <S>	CrnoGora	4
CrnoGora	4	operator <R>	CrnoGora	5
CrnoGora	5	operator <C>	Crnogora	5
Crnogora	5	operator <RW>	Crnogora_	8
Crnogora_	8	operator <B>	Crnogor_	7
Crnogor_	7	umetnuto slovo *a*	Crnogora_	8
Crnogora_	8	umetnuto slovo *c*	**Crnogorac_**	**9**

Tabela 4.2

Npr, neka u rečniku *ascdelas-top.dic* postoji odrednica

Crna Gora,N+FLX=CGFlx+DRV=AC2:AcFlx+DRV=SKI2:SkiFlx+NProp+Top (4)

i referenca na datoteku koja sadrži definicije morfografemskih pravila

AC2 = <P><B>o<S><R><C><RW><B>ac/N ; (5)
SKI2 = <P><LW><R><C><N><P><B>o<S><R><C><N><B>ski/A; (6)

kao i reference na datoteke sa opisima flektivnih klasa *CGFlx* i *AcFlx*. Tokom kompilacije rečnika, sistem Nooj automatski generiše derivacionu paradigmu {*Crnogorac*/*N, crnogorski/A*} i sve odgovarajuće flektivne oblike (Lista 4.1) i ugrađuje ih u rezultujući transduktor *ascdelas-top.nod* ([Utvić 06a]):

Crnogorac,Crna Gora,N+Inh+Hum+FLX=CGFlx+DRV=AC2:AcFlx+DRV=SKI2:SkiFlx+NProp+Top+m+s+1
Crnogorca,Crna Gora,N+Inh+Hum+FLX=CGFlx+DRV=AC2:AcFlx+DRV=SKI2:SkiFlx+NProp+Top+m+s+2+4

Crnogorcu,Crna Gora,N+Inh+Hum+FLX=CGFlx+DRV=AC2:AcFlx+DRV=SKI2:SkiFlx+NProp+Top+m+s+3+7
Crnogorče,Crna Gora,N+Inh+Hum+FLX=CGFlx+DRV=AC2:AcFlx+DRV=SKI2:SkiFlx+NProp+Top+m+s+5
Crnogorcem,Crna Gora,N+Inh+Hum+FLX=CGFlx+DRV=AC2:AcFlx+DRV=SKI2:SkiFlx+NProp+Top+m+s+6
Crnogorci,Crna Gora,N+Inh+Hum+FLX=CGFlx+DRV=AC2:AcFlx+DRV=SKI2:SkiFlx+NProp+Top+m+p+1+5
Crnogoraca,Crna Gora,N+Inh+Hum+FLX=CGFlx+DRV=AC2:AcFlx+DRV=SKI2:SkiFlx+NProp+Top+m+p+2
Crnogorcima,CrnaGora,N+Inh+Hum+FLX=CGFlx+DRV=AC2:AcFlx+DRV=SKI2:SkiFlx+NProp+Top+m+p+3+6+7
Crnogorce,Crna Gora,N+Inh+Hum+FLX=CGFlx+DRV=AC2:AcFlx+DRV=SKI2:SkiFlx+NProp+Top+m+p+4
crnogorski,Crna Gora,A+Pos+FLX=CGFlx+DRV=AC2:AcFlx+DRV=SKI2:SkiFlx+NProp+Top+m+s+1
crnogorskog,Crna Gora,A+Pos+FLX=CGFlx+DRV=AC2:AcFlx+DRV=SKI2:SkiFlx+NProp+Top+m+s+2+4
crnogorska,Crna Gora,A+Pos+FLX=CGFlx+DRV=AC2:AcFlx+DRV=SKI2:SkiFlx+NProp+Top+f+s+1+5
…

Lista 4.1

Prilikom automatske morfološke analize teksta korišćenjem generisanog transduktora *ascdelas-top.nod*, tekstuelnoj reči koja predstavlja flektivni oblik lekseme *crnogorski* biće pridružene informacije koje se odnose:

- na derivacionu lemu *Crna␣Gora* (kanonski oblik leme *Crna Gora*, oznaka flektivne klase leme *CGFlx*, oznake derivacionih klasa (*AC2*, *SKI2*) i njihovih flektivnih paradigmi (*AcFlx*, *SkiFlx*), sintaksičko-semantički markeri *NProp*, *Top*). Lista 4.1 prikazuje te informacije podvučenim fontom;
- na flektivnu lemu *crnogorski* (vrsta reči *A*, sintaksičko-semantički marker *Pos*, morfološke informacije o rodu (*m*, *f*), broju (*s*, *p*), padežu (*1*-*7*)).

4.2.2.5 Nedostaci modela derivacije sistema NooJ

Model derivacije koji koristi sistem NooJ ima nekoliko nedostataka. Prvi nedostatak je što se generisanim oblicima derivacione paradigme na takav način pridružuju informacije o odgovarajućoj flektivnoj i derivacionoj lemi, da ih nije moguće jasno razlikovati. To se posebno odnosi na sintaksičko-semantičke markere. Lista 4.1 sugeriše da se u zapisima flektivnih oblika izvedenica pre svojstva *+FLX* navode sintaksičko-semantički markeri koji se odnose na flektivnu lemu, a posle svojstva *+FLX* sintaksičko-semantički markeri koji se odnose na derivacionu lemu. Međutim, u trenutku navođenja ovih redova, za tako nešto ne postoji izričita potvrda u dokumentaciji sistema NooJ ([Silberztein 06]).

Takođe, nije sasvim jasno kojoj flektivnoj klasi pripada flektivna lema. Taj podatak nije neophodan sistemu, zato što generisani oblik izvedenice sadrži sve neophodne morfološke informacije (oznake prisutnih gramatičkih kategorija i realizovanih oblika).

Sintaksa definicija morfografemskih pravila i način na koji se referiše na njih direkno utiču na metodologiju klasifikacije regularne derivacije od toponima. Naime, sistem NooJ omogućava da se jednim pravilom opiše derivacija više izvedenica isključivo pod uslovom da sve one imaju istovetan opis flektivne paradigme, tj. da pripadaju istoj flektivnoj klasi. Samim tim, u slučaju lema izvedenih od toponima regularnom derivacijom umesto jednog pravila bilo bi potrebno kreirati četiri (po jedan za muškog i ženskog stanovnika, jedan za relacione prideve na *-ski*, i jedan za prisvojne prideve na *-ov*). Ovo otvara pitanje da li se treba prikloniti postojećoj implementaciji sistema NooJ i svaki tip izvedenice posebno klasifikovati (dakle, posebno svaki etnik i ktetik) ili treba klasifikovati opise celokupnih derivacionih paradigmi (i

etnika i ktetika) koju bi podržala implementacija nekog budućeg sistema (koji ne mora biti NooJ). U prvom slučaju svakom toponimu se pridružuje onoliko derivacionih klasa koliko ima različitih flektivnih klasa njegovih izvedenica, a u drugom slučaju svakom toponimu se pridružuje tačno jedna derivaciona klasa koja opisuje njegovu kompletnu (regularnu) derivacionu paradigmu.

4.3 Klasifikacija regularne derivacije od toponima

Automatski klasifikator na ulazu očekuje tekstuelnu datoteku čiji je sadržaj kodiran shemom aurora. Format ulazne datoteke je definisan na sledeći način (Tabela 4.3):

- svaka linija datoteke predstavlja jedan toponim i njegovu regularnu derivacionu paradigmu (ktetik koji se odnosi na mesto, ime stanovnika i njemu odgovarajući prisvojni pridev, ime stanovnice i njoj odgovarajući prisvojni pridev, i ktetik koji se odnosi na stanovnike);
- u slučaju da za neki toponim postoji više različitih varijanti derivacione paradigme, svaka varijanta je predstavljena posebnom linijom;
- elementi derivacione paradigme toponima predstavljaju kolone u okviru datoteke. Kolone su međusobno razdvojene tabulatorom;
- ukoliko je lema dvočlana, tj. sastoji se iz dve formalne reči, onda su te formalne reči razdvojene razmakom.

London	londonski	Londonac	Londoncyev	Londonka	Londonkin	londonski
Crna Gora	crnogorski	Crnogorac	Crnogorcyev	Crnogorka	Crnogorkin	crnogorski
Banxa Luka	banxalucyki	Banxalucyanin	Banxalucyaninov	Banxalucyanka	Banxalucyankin	banxalucyanski
Banxa Luka	banxolucyki	Banxolucyanin	Banxolucyaninov	Banxolucyanka	Banxolucyankin	banxolucyanski
Novi Pazar	novopazarski	Novopazarac	Novopazarcyev	Novopazarka	Novopazarkin	novopazarski
Novi Pazar	pazarski	Pazarac	Pazarcyev	Pazarka	Pazarkin	pazarski

Tabela 4.3

Derivacione paradigme toponima su podeljene na dve grupe u zavisnosti od toga da li se radi o jednočlanim i dvočlanim toponimima. Posle toga je svaka od grupa podeljena na tri podgrupe u zavisnosti od sufiksa kojim se izvodi odgovarajući muški etnik (*-ac*, *-anin* i ostali). Svaka podgrupa je predstavljena posebnom tekstuelnom datotekom u opisanom formatu. Za svaku podgrupu ponaosob je obavljena klasifikacija na sledeći način:

a) koristeći da su kolone ulazne datoteke razdvojene tabulatorom, prilikom učitavanja svake linije se jednostavno izdvajaju lema toponima i njegove izvedenice;
b) za svaki toponim se, na osnovu njegove leme i izvedenica dobijenih regularnom derivacijom, automatski konstruiše morfografemsko pravilo sistema NooJ. Tim pravilom se opisuju transformacije koje je potrebno izvršiti nad lemom toponima kako bi se generisala odgovarajuća N-derivaciona paradigma.

c) svi toponimi kojima je pridruženo isto morfografemsko pravilo sistema NooJ, smatraju se elementima jedne derivacione klase koja je jednoznačno određena tim pravilom.

S obzirom na ograničenje da se u sistemu NooJ jednim morfografemskim pravilom može opisati derivacija više izvedenica isključivo ako sve one pripadaju istoj flektivnoj klasi (odeljak 4.2.2.5), automatski klasifikator je implementiran tako da se prethodno mogu izabrati kolone (izvedenice) ulazne datoteke, koje će tokom klasifikacije biti uzete u obzir. Kao posledica toga, klasifikator može da generiše derivacione klase koje se mogu iskoristiti u postojećoj implementaciji sistema NooJ (svakom toponimu se pridružuje onoliko derivacionih klasa koliko ima različitih flektivnih klasa njegovih izvedenica).

S druge strane, ako u obzir budu uzete sve kolone (izvedenice) ulazne datoteke, klasifikator će svakom toponimu pridružiti tačno jednu derivacionu klasu koja opisuje njegovu kompletnu (regularnu) derivacionu paradigmu. Tako dobijene klase će moći da se iskoriste u nekom budućem sistemu (koji ne mora biti nova implementacija sistema NooJ) za modelovanje regularnih derivacionih procesa toponima.

Leksički resursi srpskog jezika u sistemu NooJ koriste kao azbuku sledeće podskupove karaktera kodne sheme Unicode: 1) skup karaktera srpske latinice; 2) skup karaktera srpske ćirilice; 3) skup karaktera koda aurora. Za svaku od ovih azbuka postoji posebna verzija leksičkih resursa, pa je neophodna i posebna verzija morfografemskih pravila kojima se opisuju flektivne i derivacione paradigme. Automatski klasifikator je implementiran na takav način da se pre konstrukcije morfografemskih pravila može izabrati odgovarajuća azbuka, odnosno verzija leksičkih resursa, za koju se ta pravila konstruišu. Algoritam za konstrukciju pravila je nezavisan od kodne sheme, te je dovoljno da se pre njegove primene ulazni podaci prevedu u odgovarajuću kodnu shemu. Pošto su ulazni podaci kodirani shemom aurora, po potrebi se se mogu prevesti u odgovarajuće (ćirilične ili latinične) karaktere sheme Unicode jednostavnom zamenom karaktera, odnosno digrafa (Tabela 2 i Tabela 3 uvodne glave).

Konstruisanje morfografemskih pravila se bitno razlikuje za jednočlane i dvočlane toponime, te će za svaku grupu biti posebno razmotreno. Međutim, u oba slučaja se koristi ista funkcija **GetLCPCaseInsensitive**, pa će najpre biti objašnjena njena uloga.

Funkcija **GetLCPCaseInsensitive** za dati niz niski računa njihov **najduži zajednički prefiks** (engl. Longest Common Prefix, skr. LCP), tj. najduži zajednički levi faktor. Pri tom se pojam LCP može interpretirati na dva načina:

a) kao najduži zajednički levi faktor posmatran kao niska karaktera;

b) kao najduži zajednički levi faktor posmatran kao niska slova srpskog jezika[102].

Razliku između ova dva slučaja ilustruje Tabela 4.4. Funkcija **GetLCPCaseInsensitive** je implementirana na takav način da može da računa odgovarajući LCP za svaku od navedene dve interpretacije. U oba slučaja LCP se računa korišćenjem interpretacije a), a onda se konsultuje bulovska promenljiva **bUseDigraphs**. Ako je vrednost te promenljive **false**, računanje je završeno; u suprotnom, u skladu sa interpretacijom b), vrši se eventualna korekcija.

Funkcija **GetLCPCaseInsensitive** očekuje da sva slova ulazne niske (sem prvog) budu mala. Prvi karakter svake od niski može biti ili veliko ili malo slovo. Prilikom računanja LCP sva početna slova niski se tretiraju kao velika (ako je početno slovo prve niske u nizu

[102] Tj. digrafi *lj*, *nj*, *dž* (u latiničnom pismu) i *lx*, *nx*, *sx*, *zx*, *cx*, *cy*, *dx*, *dy* (u kodu aurora) se tretiraju kao jedan simbol azbuke.

veliko), odnosno kao mala (ako je početno slovo prve niske u nizu malo). Tabela 4.5 ilustruje rezultat funkcije **GetLCPCaseInsensitive** za dva tročlana niza niski, koristeći obe interpretacije pojma LCP.

Lema (1. niska)	**Izvedenica (2. niska)**	**LCP niski (slučaj a)**	**LCP niski (slučaj b)**
Pula	Puljanin	Pul	Pu
Atina	Atinxanin	Atin	Ati
Beograd	Beogradxanin	Beogradx	Beograd
Andaluzija	Andaluzxanin	Andaluz	Andalu

Tabela 4.4

1. niska	**2. niska**	**3. niska**	**LCP niski (slučaj a)**	**LCP niski (slučaj b)**
Atina	atinski	Atinxanin	Atin	Ati
Beograd	beogradski	Beogradxanin	Beograd	Beograd

Tabela 4.5

4.3.1 Klasifikacija derivacionih paradigmi jednočlanih toponima

Neka je *t* lema jednočlanog toponima i *d* proizvoljna izvedenica njegove regularne derivacione paradigme. Konstruisanje morfografemskog pravila koje opisuje generisanje izvedenice *d* na osnovu njene derivacione leme *t* sastoji se iz sledećih koraka:

1. za svaku izvedenicu *d* koja je tokom klasifikacije uzeta u obzir se korišćenjem funkcije **GetLCPCaseInsensitive** računa **najduži zajednički prefiks** niski *t* i *d* (u oznaci sLCP[103]);
2. U slučaju da su sLCP i lema *t* istovetne niske, a oblik nominativa jednine leme *t* ima nenulti flektivni nastavak[104], vrši se korekcija tako da korigovani sLCP ne sadrži taj nastavak. Tabela 4.6 to ilustruje na primeru toponima *Austrija*;
3. Na osnovu izvedenice *d* i sLCP određuje se niska *b*. Neka je $n = |t| - |\text{sLCP}|$. Ako je $n = 0$, tada je *b* prazna niska. Ako je $n \geq 1$, tada *b* postaje NooJ-regularni izraz <B*n*>[105]. Niska *b* opisuje broj karaktera *n* koji treba obrisati sa desne strane leme *t* kako bi se dobio sLCP.

[103] U pitanju je akronim od engl. **s**tring **L**ongest **C**ommon **P**refix (niska najduži zajednički prefiks)

[104] Npr, imenice ženskog roda u nominativu jednine mogu imati nenulti flektivni nastavak –*a* (*Austrij-a*)

[105] Umesto <B1> se češće koristi skraćeni oblik <B>.

4. Na osnovu izvedenice *d* i sLCP određuje se niska *s* kao desni faktor niske *d* dužine $m = |d| - |\text{sLCP}| \geq 0$. U slučaju da je *s* prazna niska, ona se zamenjuje odgovarajućim NooJ-regularnim izrazom <E>.
5. ako je početno slovo izvedenice malo, niska *l* se definiše kao <LW><R><C><RW>. U protivnom, *l* je prazna niska.
6. Ako je φ morfosintaksički opis pridružen izvedenici *d*, tada proizvod dopisivanja lbs / φ predstavlja morfografemsko pravilo koje opisuje generisanje izvedenice *d* na osnovu njene derivacione leme *t*. Tabela 4.7 ilustruje slučaj kada je *l* prazna niska.

Lema	**Izvedenica**	**sLCP**	**korigovan sLCP**
London	Londonac	London	London
Bukurešt	Bukureštanac	Bukurešt	Bukurešt
Austrija	Austrijanac	Austrija	Austrij

Tabela 4.6

Ostaje da se dokaže da je niska lbs / φ dobijena u koraku 6 zaista morfografemsko pravilo koje opisuje generisanje izvedenice *d* na osnovu njene derivacione leme *t*. U dokazu se umesto oznake sLCP koristi jednostavnija oznaka *p*. Takođe, sa x^C biće označena niska dobijena od proizvoljne niske *x* tako što je njeno početno veliko (malo) slovo zamenjeno odgovarajućim malim (velikim) slovom. Primetimo da je $|x^C| = |x|$.

Lema *t*	**Izvedenica** *d*	**Morfosint. opis izvedenice (φ)**	**korigovan sLCP**	$m = \|d\| - \|\text{sLCP}\|$	***s***	$n = \|t\| - \|\text{sLCP}\|$	***b***	**Generisano pravilo**
London	Londonac	N+Inh[106]	London	2 = 8 – 6	ac	0 = 6 - 6	<E>	ac/N+Inh
Bukurešt	Bukureštanac	N+Inh	Bukurešt	4 = 12 – 8	anac	0 = 8 - 8	<E>	anac/N+Inh
Austrija	Austrijanac	N+Inh	Austrij	4 = 11 - 7	anac	1 = 8 - 7	<B>	<B>anac/N+Inh

Tabela 4.7

Razmotrićemo slučaj kada nijedna od niski *l*, *b*, *s* nije prazna, a odatle neposredno slede ostali slučajevi. Pošto je početno slovo leme *t* uvek veliko, a *l* je neprazna niska, to znači da je početno slovo izvedenice *d* malo (korak 5), a početno slovo niske *p* – veliko (definicija funkcije **GetLCPCaseInsensitive**). Na osnovu toga i definicije niske *s* sledi da je $d = p^C s$.

[106] Morfosintaksički opis navedenih izvedenica može da uključi i druge sintaksičko-semantičke markere (npr. *+Hum*), ali je zbog nedostatka prostora naveden samo *+Inh.*

Takođe, na osnovu definicije niske $b = <\mathrm{B}n>$, jednakosti $n = |t| - |p| = |t| - |p^C|$, sledi da je $(<\mathrm{B}n>(t^C, |t|) = p^C$. Tada važi:

$$derive(t, <\mathrm{LW}><\mathrm{R}><\mathrm{C}><\mathrm{RW}><\mathrm{B}n>s) = \pi_1 \circ <\mathrm{LW}><\mathrm{R}><\mathrm{C}><\mathrm{RW}><\mathrm{B}n>\ s\ (t, |t|),$$

$$<\mathrm{LW}><\mathrm{R}><\mathrm{C}><\mathrm{RW}><\mathrm{B}n>\ s\ (t, |t|) =$$

$$<\mathrm{R}><\mathrm{C}><\mathrm{RW}><\mathrm{B}n>\ s\ (t, 0) =$$

$$<\mathrm{C}><\mathrm{RW}><\mathrm{B}n>\ s\ (t, 1) =$$

$$<\mathrm{RW}><\mathrm{B}n>\ s\ (t^C, 1) =$$

$$<\mathrm{B}n>\ s\ (t^C, |t|) =$$

$$s(p^C, |t| - n) =$$

$$(p^C s, |t| - n + |s|) = (d, |t| - n + |s|);$$

Odatle neposredno sledi

$$derive(t, <\mathrm{LW}><\mathrm{R}><\mathrm{C}><\mathrm{RW}><\mathrm{B}n>s) = \pi_1(d, |t| - n + |s|) = d$$

čime je dokaz završen. □

Smisao korekcije u 2. koraku je da se izbegnu suvišne klase sa neregularnim sufiksima. Npr, u slučaju izvedenice *Austrijanac*, derivaciona lema *Austrija* (tj. derivaciona osnova *Austrij-*) se kombinuje sa regularnim sufiksom *–anac* a ne *–nac* (Tabela 4.6).

Opisana konstrukcija morfografemskog pravila za generisanje izvedenice *d* na osnovu njene derivacione leme *t* implementirana je u funkciji **Gen1ReSimple**.

U slučaju da su tokom klasifikacije uzete u obzir dve ili više izvedenica derivacione paradigme, svako generisano pojedinačno morfografemsko pravilo se dopisuje na prethodno generisana pravila, uz korišćenje operatora alternacije (␣+␣) kao separatora. Tim postupkom dobija se rezultujuće morfografemsko pravilo koje opisuje deo ili pak celu derivacionu paradigmu (Tabela 4.8).

Izvedenica	**Morfografemsko pravilo**
Austrijanac	<B>anac/N
austrijski	<LW><R><C><RW><B>ski/A
Rezult. pravilo	<B>anac/N␣+␣<LW><R><C><RW><B>ski/A

Tabela 4.8

Međutim, u slučaju najučestalijih sufiksa za izvođenje etnika (*-anin* i *–ac*/*-ka*), umesto da se LCP računa za lemu i svaku izvedenicu posebno, dovoljno je izračunati jednu nisku koja predstavlja LCP leme, etnika i njihovih prisvojnih prideva. Prilikom računanja

morfografemskih pravila za generisanje etnika i njihovih prisvojnih prideva, niska *l* biće prazna, a niska *b* će za sve izvedenice biti ista, te se i ona može izračunati samo jednom. S druge strane, niska *s* mora da se računa posebno za svaku izvedenicu. Ako rezultate tog računanja označimo sa s_1, s_2, s_3, s_4, a odgovarajuće morfosintaksičke opise izvedenica sa $\varphi_1, \varphi_2, \varphi_3, \varphi_4$, tada će morfografemsko pravilo za generisanje etnika i njihovih prisvojnih prideva imati oblik $b(s_1 / \varphi_1 \sqcup + \sqcup s_2 / \varphi_2 \sqcup + \sqcup s_3 / \varphi_3 \sqcup + \sqcup s_4 / \varphi_4)$. Tabela 4.9 ilustruje opisanu konstrukciju na primeru derivacione paradigme toponima *Valjevo*. Funkcija **GenDemonymReSimple** implementira ovakav pristup konstrukciji pravila.

Izvedenica	***b***	***s***	
Valjevac	<B>	ac/N	s_1
Valjevčev		čev/A	s_2
Valjevka		ka/N	s_3
Valjevkin		kin/A	s_4
Rezult. pravilo	<B>(ac/N␣+␣čev/A␣+␣ka/N␣+␣kin/A)		

Tabela 4.9

Prema tome, za računanje morfografemskog pravila koje opisuje izvođenje dela ili cele derivacione paradigme jednočlanog toponima, koriste se funkcije **Gen1ReSimple** i **GenDemonymReSimple** na sledeći način:

- ukoliko se računa pravilo koje opisuje izvođenje samo jedne izvedenice, koristi se funkcija **Gen1ReSimple**;
- za morfografemsko pravilo koje opisuje izvođenje ktetika uvek se koristi funkcija **Gen1ReSimple**;
- u slučaju da su tokom klasifikacije uzeta u obzir oba etnika (sa ili bez prisvojnih prideva), njihovo izvođenje (uključujući i prisvojne prideve, ako su uzeti u obzir) se opisuje samo jednim pravilom koje računa funkcija **GenDemonymReSimple**;
- rezultujuće pravilo se dobija dopisivanjem pojedinačnih pravila, uz korišćenje operatora alternacije (␣+␣) kao separatora.

Npr, da bi se konstruisalo pravilo koje opisuje derivacionu paradigmu toponima *Pančevo*:

<LW><R><C><RW><B>ački/A␣+␣
<B>(ac/N␣+␣čev/A␣+␣ka/N␣+␣kin/A)[107]

funkcijom **Gen1ReSimple** je najpre izračunato pravilo za ktetik:

[107] Ovde je, radi jednostavnosti, naveden minimalan morfosintaksički opis izvedenica, tj. izostavljeni su sintaksičko-semantički markeri.

<LW><R><C><RW><B>ački/A,

zatim je funkcijom **GenDemonymReSimple** dobijeno pravilo za etnike i njihove prisvojne prideve

<B>(ac/N␣+␣čev/A␣+␣ka/N␣+␣kin/A),

a na kraju je dopisivanjem (uz korišćenje separatora ␣+␣) dobijeno rezultujuće pravilo.

4.3.2 Klasifikacija derivacionih paradigmi dvočlanih toponima

Opšti algoritam za računanje morfografemskog pravila koje opisuje izvođenje dela ili cele derivacione paradigme dvočlanog toponima je sličan odgovarajućem algoritmu za jednočlane toponime, s tom razlikom što se koriste funkcije **Gen1ReMWU** i **GenDemonymReMWU** umesto funkcija **Gen1ReSimple** i **GenDemonymReSimple**:

- ukoliko se računa pravilo koje opisuje izvođenje samo jedne izvedenice, koristi se funkcija **Gen1ReMWU**;
- za morfografemsko pravilo koje opisuje izvođenje ktetika uvek se koristi funkcija **Gen1ReMWU**;
- u slučaju da su tokom klasifikacije uzeta u obzir oba etnika (sa ili bez prisvojnih prideva), njihovo izvođenje (uključujući i prisvojne prideve, ako su uzeti u obzir) se opisuje samo jednim pravilom koje računa funkcija **GenDemonymReMWU**;
- rezultujuće pravilo se dobija dopisivanjem pojedinačnih pravila, uz korišćenje operatora alternacije (␣+␣) kao separatora.

Tabela 4.10 ilustruje morfografemsko pravilo koje opisuje kako se izvode etnici i ktetik od toponima *Crna␣Gora*. Pravilo za izvođenje ktetika računa funkcija **Gen1ReMWU**, a pravilo za izvođenje etnika računa funkcija **GenDemonymReMWU**. Dopisivanjem tih pravila (uz korišćenje separatora ␣+␣) dobija se rezultujuće pravilo.

Izvedenica	**Morfografemsko pravilo**	
crnogorski	<P><LW><R><C><RW><B>o<S><R><C><RW><B>ski/A	
Crnogorac	<P><B>o<S><R><C><RW><B>	ac/N
Crnogorka		ka/N
Rezult. pravilo	<P><LW><R><C><RW><B>o<S><R><C><RW><B>ski/A ␣+␣<P><B>o<S><R><C><RW><B>(ac/N␣+␣ ka/N)	

Tabela 4.10

Obe ove funkcije koriste pomoćnu funkciju **GetReMWU2SWU**. Njeni argumenti su dve niske, višečlana i jednočlana niska. Vrednost funkcije je NooJ-regularni izraz koji opisuje morfografemsku transformaciju zadate višečlane niske u zadatu jednočlanu nisku.

U praksi, prvi argument te funkcije (višečlana niska) će uvek predstavljati lemu toponima, dok će drugi argument (jednočlana niska) biti ili izvedenica ili najduži zajednički prefiks (LCP) izvedenica (etnika i njihovih prisvojnih prideva), zavisno od toga koja od funkcija **Gen1ReMWU** i **GenDemonymReMWU** poziva funkciju **GetReMWU2SWU**.

Izvedenice koje se izvode od dvočlanih toponima mogu biti jednog od sledeća tri tipa:

(tip 1) prvi član leme ne utiče na izvođenje;

(tip 2) drugi član leme ne utiče na izvođenje;

(tip 3) oba člana leme utiču na izvođenje. U ovom slučaju izvedenice predstavljaju složenice u kojima se, kao sastavni deo (pored članova leme ili njihovih delova), ponekad pojavljuje i spojni vokal 'o' ili 'e' ([Klajn 02]). Primere prikazuje Tabela 4.11.

Funkcija **GetReMWU2SWU** izračunava morfografemsku transformaciju prvog u drugi argument u dve faze:

1. Heurističkim metodom se određuju tip i podtip izvođenja. U slučaju tipa 3, definiše se vrednost promenljivih *v*, *r*, *p*, *s* i *c*;
2. Na osnovu određenog tipa i podtipa (a u slučaju tipa 3, i izračunatih vrednosti promenljivih) konstruiše se odgovarajući NooJ-regularni izraz.

Tip izvođenja	lema	izvedenica
(tip 1) Prvi član ne učestvuje u izvođenju	Herceg Novi	novlxanski
	Macyvanska Mitrovica	mitrovacyki
(tip 2) Drugi član ne učestvuje u izvođenju	Homolxske planine	Homolx
	Jonsko more	jonski
(tip 3) Oba člana učestvuju u izvođenju	Crna Gora	crnogorski
	Banxa Luka	banxalucyki
	Goli otok	golootocyki
	Nxu Orleans	nxuorleanski

Tabela 4.11

Podtipove izvođenja ilustruje Tabela 4.12, a njihove precizne definicije su date u opisu implementacije prve faze koja sledi.

Sa x^C biće označena niska dobijena od proizvoljne niske *x* tako što je njeno početno veliko (malo) slovo zamenjeno odgovarajućim malim (velikim) slovom. Neka je *t* lema dvočlanog toponima i *d* proizvoljna izvedenica ili LCP dve ili više izvedenica (regularne) derivacione paradigme toponima *t*. Tada postoje jedinstvene formalne reči t_1, t_2 (članovi toponima) takve da t_2 počinje malim slovom i važi ili $t = t_1 \sqcup t_2$ ili $t = t_1 \sqcup t_2^C$ [108]. Označimo sa p_j LCP niski t_j i *d*, $j \in \{1,2\}$. Neka je s_j desni faktor niske *d* takav da je $d = p_j s_j$, $j \in \{1,2\}$. Prva faza funkcije **GetReMWU2SWU** se sastoji iz sledećih koraka:

[108] Npr, za *t* = *Homoljske planine*, $t_1 = Homoljske$, $t_2 = planine$, dok je za *t* = *Banja luka*, $t_1 = Banja$, $t_2 =$ luka.

1. Najpre se izračunaju vrednosti niski p_j, $j \in \{1,2\}$, promenljiva *v* se inicijalizuje kao prazna niska i prelazi se na korak 2;
2. Ako je $p_1 = \varepsilon$, odnosno $p_1 \neq \varepsilon \wedge |p_1| \leq |p_2|$, u pitanju je tip 1 (podtip 1a, odnosno podtip 1b) i prva faza je završena; u protivnom, prelazi se na korak 3;
3. Na osnovu vrednosti niski p_j i *d*, računaju se vrednosti niski s_j, $j \in \{1,2\}$. Promenljiva *r* se inicijalizuje tako da ima istu vrednost kao i niska s_1. Prelazi se na korak 4;
4. Ako je $s_1 = \varepsilon$, tada je u pitanju tip 2 (podtip 2a) i prva faza je završena; u protivnom, prelazi se na korak 5;
5. upoređuju se prvi karakteri niski t_2 i s_1: ako su jednaki, prelazi se na korak 8; u protivnom prelazi se na korak 6;
6. ako niska s_1 ne počinje karakterom 'o' ili 'e', prelazi se na korak 7. U protivnom se radi o tipu 3 (podtipu 3a). Vrednost promenljive *v* postaje prvi karakter niske s_1 ('o' ili 'e'). Vrednost promenljive *r* se koriguje tako da važi $s_1 = vr$. Prelazi se na korak 11;
7. pošto prvi karakteri niski t_2 i s_1 nisu jednaki, a niska s_1 ne počinje karakterom 'o' ili 'e', radi se tipu 2 (podtipu 2b) i prva faza je završena;
8. ako je prvi karakter niske s_1 (a time i t_2) jednak 'o' ili 'e', prelazi se na korak 9. U protivnom je u pitanju tip 3 (podtip 3b) i prelazi se na korak 11;
9. pošto je prvi karakter niske s_1 jednak 'o' ili 'e', u pitanju je tip 3. Ostaje da se proveri da li je to spojni vokal ili ne (podtip 3c ili podtip 3d). Ako su prva dva karaktera niske s_1 međusobno jednaka (i jednaka 'o' ili 'e') prelazi se na korak 10. U protivnom, u pitanju je podtip 3d i prelazi se na korak 11;
10. U pitanju je podtip 3c. Vrednost promenljive *v* postaje prvi karakter niske s_1 ('o' ili 'e'). Vrednost promenljive *r* se koriguje tako da važi $s_1 = vr$. Prelazi se na korak 11;
11. Pošto je u pitanju tip 3, inicijalizuju se promenljive *p*, *s* i *c*. Vrednost promenljive *p* je LCP niski t_2 i *r*. U slučaju da su niske *p* i t_2 istovetne, a oblik nominativa jednine niske t_2 ima nenulti flektivni nastavak[109], vrši se korekcija promenljive *p* tako da korigovana vrednost ne sadrži taj nastavak. Promenljiva *s* dobija vrednost tako da zadovoljava jednakost $r = ps$. Što se tiče promenljive *c*, ako je $t = t_1 \llcorner t_2$ tada je njena vrednost prazna niska; ako je $t = t_1 \llcorner t_2^C$ tada je njena vrednost niska <R><C>. Prva faza je završena.

Navedena heuristika pokušava da prepozna strukturu izvedenice *d*, i da je predstavi kao:

- $d = p_2 s_2$ ili $d = p_2^C s_2$ (tip 1);

[109] Npr, drugi član toponima *Južna Koreja* je imenica ženskog roda koja u nominativu jednine ima nenulti flektivni nastavak *–a*. Stoga je vrednost promenljive *p* (Tabela 4.12) umesto LCP-vrednosti (*koreja*) korigovana u *korej*. Time se izbegavaju veštački sufiksi izvedenica i veštačke derivacione klase toponima.

- $d = p_1 s_1$ ili $d = p_1^C s_1$ (tip 2);
- $d = p_1 vps$ ili $d = p_1^C vps$.

Posebno, u slučaju tipa 3, navedena heuristika pokušava da prepozna strukturu složenice *d*, gde:

- p_1 predstavlja prvi član toponima ili njegov levi faktor,
- *v* predstavlja opcioni spojni vokal,
- *p* predstavlja drugi član toponima ili njegov levi faktor,
- *s* predstavlja sufiks.

Tabela 4.12 prikazuje navedene sastavne delove izvedenica masnim slovima (tip 3).

pod-tip	**lema *t***	***d***	t_2	p_1	s_1	***v***	***r***	p_2	s_2	***p***	***s***
1a	Herceg Novi	novljanski	novi	ε				**Nov**	**ljanski**		
1b	Macyvanska Mitrovica	mitrovački	mitrovica	M				**Mitrov**	**ački**		
2a	Homolxske planine	Homolx	planine	**Homolx**	**ε**			ε			
2b	Jonsko more	jonski	more	**Jonsk**	**i**			ε			
3a	Crna Gora	crnogorski	gora	**Crn**	ogorski	**o**	gorski	ε		**gor**	**ski**
3b	Banxa Luka	banxalucyki	luka	**Banxa**	lucyki	**ε**	lucyki	ε		**lu**	**cyki**
3c	Goli otok	golootocyki	otok	**Gol**	ootocyki	**o**	otocyki	ε		**oto**	**cyki**
3c	Srednja Evropa	srednje-evropski	evropa	**Srednj**	eevropski	**e**	evropski	ε		**evrop**	**ski**
3c	Južna Koreja	Južno-korejanac	koreja	**Južn**	okorejanac	**o**	korejanac	ε		**korej**	**anac**
3d	Nxu Orleans	nxu-orleanski	orleans	**Nxu**	orleanski	**ε**	orleanski	ε		**orleans**	**ki**

Tabela 4.12

Posle završetka prve faze, na osnovu tipa izvođenja i vrednosti izračunatih promenljivih, odgovarajući NooJ-regularni izraz se računa na način koji opisuje Tabela 4.13. U slučaju tipova 1 i 2 razmatrana konstrukcija se može neposredno svesti na odgovarajuću konstrukciju za jednočlane niske, pri čemu se mora voditi računa o eliminaciji člana koji ne učestvuje u regularnoj derivaciji. Najinteresantniji je tip 3, gde se konstrukcija za jednočlane niske primenjuje dva puta (za svaki član posebno), uz dodatne operacije koje omogućavaju pozicioniranje u okviru višečlane niske, eventualno umetanje spojnog vokala na odgovarajuće mesto, kao i eventualnu zamenu početnog velikog slova malim kod drugog člana toponima. Detaljan primer prikazuje Tabela 4.2 u odeljku 4.2.2.4.

Oznaka tipa izvođenja	*n*	*m*	NooJ-regularni izraz
1		$\lvert t_2\rvert - \lvert p_2\rvert$	<LW><BW><RW><B*m*> s_2
2	$\lvert t_1\rvert - \lvert p_1\rvert$		<P><SW><B*n*> s_1
3	$\lvert t_1\rvert - \lvert p_1\rvert$	$\lvert t_2\rvert - \lvert p\rvert$	<P><B*n*>*v*<S>*c*<RW><B*m*>*s* (tj. P><B*n*>*v*<S><RW><B*m*>*s* ili <P><B*n*>*v*<S><R><C><RW><B*m*>*s*)

Tabela 4.13

Funkcija **Gen1ReMWU** ima dva argumenta, lemu toponima *t* (dvočlana niska) i njenu regularnu izvedenicu *d* (jednočlana niska). Vrednost funkcije je morfografemsko pravilo koje opisuje izvođenje niske *d* od niske *t*. Funkcija **Gen1ReMWU** je implementirana na sledeći način:

- Pozivom funkcije **GetReMWU2SWU**(*t*, *d*) računa se NooJ-regulani izraz *r*;
- Ukoliko izvedenica *d* počinje malim slovom, vrši se korekcija regularnog izraza *r* u zavisnosti od tipa izvođenja:
  - ako *r* počinje operatorom <P> (tip 2 ili tip 3), tada se taj operator zamenjuje sa <P><LW><R><C><RW>;
  - u protivnom (tip 1), pošto je *r* oblika <LW><BW><RW><B*m*> s_2, <LW><BW><RW> se zamenjuje sa <LW><BW><R><C><RW>.
- Ako je φ morfosintaksički opis pridružen izvedenici *d*, tada proizvod dopisivanja r/φ predstavlja morfografemsko pravilo koje opisuje generisanje izvedenice *d* na osnovu njene derivacione leme *t*.

Funkcija **GenDemonymReMWU** ima jedan argument - niz niski d_j dužine k ($0 \le j < k$). Prvi element niza d_0 je lema toponima *t* (dvočlana niska), a ostali elementi su njegove regularne izvedenice (jednočlane niske), i to oba etnika (muški i ženski) sa ili bez prisvojnih prideva. Vrednost funkcije je morfografemsko pravilo koje opisuje izvođenje izvedenica d_j ($1 \le j < k$) od niske *t*. Funkcija **GenDemonymReMWU** je implementirana na sledeći način:

- Pozivom funkcije GetLCP[110] se računa LCP izvedenica d_j ($1 \le j < k$) i rezultat se smešta u promenljivu sLCP;
- Pozivom funkcije **GetReMWU2SWU**(*t*, sLCP) računa se NooJ-regulani izraz *r*;

[110] Pošto sve izvedenice (etnici i njihovi prisvojni pridevi) počinju velikim slovom, nema potrebe koristiti funkciju **GetLCPCaseInsensitive.**

- Za svako *j*, $(1 \le j < k)$ se na osnovu izvedenice d_j i niske sLCP izračunava niska s_j kao desni faktor niske d_j dužine $m_j = |d_j| - |\text{sLCP}| \ge 0$. U slučaju da je s_j prazna niska, ona se zamenjuje NooJ-regularnim izrazom <E>;
- Neka je φ_j morfosintaksički opis pridružen izvedenici d_j $(1 \le j < k)$. Tada $r(s_1 / \varphi_1 \prod_{2 \le j < k} \text{␣+␣} s_j / \varphi_j)$ predstavlja morfografemsko pravilo koje opisuje generisanje izvedenica d_j $(1 \le j < k)$ na osnovu njihove derivacione leme *t*.

4.3.3 Rezultati primene klasifikatora

Primenom algoritama opisanih u 4.3.1 i 4.3.2 za svaki jednočlani i dvočlani toponim iz ulazne datoteke je automatski konstruisano morfografemsko pravilo sistema NooJ. Tim pravilom se opisuju transformacije koje je potrebno izvršiti nad lemom toponima kako bi se generisala odgovarajuća N-derivaciona paradigma. Svi toponimi kojima je pridruženo isto morfografemsko pravilo sistema NooJ, smatraju se elementima jedne derivacione klase koja je jednoznačno određena tim pravilom.

Konstruisana pravila i derivacione klase zavise od izabrane azbuke (aurora, latinica, ćirilica), interpretacije pojma LCP (odeljak 4.3, Tabela 4.4) i izvedenica koje su uzete u obzir tokom klasifikacije (tj. izabranih kolona ulazne datoteke). S obzirom na obim tako dobijenih različitih rezultata, ovde su navedeni rezultati dobijeni pod sledećim pretpostavkama:

- azbuku čine karakteri srpske latinice u kodnoj shemi Unicode;
- LCP predstavlja najduži zajednički levi faktor posmatran kao niska slova srpskog jezika;
- kad je izbor izvedenica u pitanju, klasifikacija je izvršena za kompletne N-derivacione paradigme (etnici, ktetici i prisvojni pridevi etnika), i posebno za ktetike, muške etnike i ženske etnike.

Rezultate klasifikacije za kompletne N-derivacione paradigme (etnici, ktetici i prisvojni pridevi etnika) jednočlanih i dvočlanih toponima prikazuju Tabela 4.14 i Tabela 4.15 tim redom.

Kompletne N-derivacione paradigme (etnici, ktetici i prisvojni pridevi etnika) latinično pismo								
jednočlani toponimi	ukupan broj toponima	broj klasa	broj singltona	broj toponima u 5 najbrojnijih klasa				
etnik na *-ac*	293	46	31	93	79	27	21	7
etnik na *–in* (*-(j)anin*)	331	81	41	40	34	14	13	13
ostali etnici	34	26	18	2	2	2	2	2

Tabela 4.14

Kompletne N-derivacione paradigme (etnici, ktetici i prisvojni pridevi etnika)								
dvočlani toponimi	ukupan broj toponima	broj klasa	broj singltona	broj toponima u 5 najbrojnijih klasa				
etnik na *–ac*	104	25	11	24	12	12	10	7
etnik na *–in* (-(*j*)*anin*)	80	43	26	7	5	5	4	4
ostali etnici	2	2	2	1	1			

Tabela 4.15

Rezultate klasifikacije za pojedinačne izvedenice jednočlanih toponima prikazuju Tabela 4.16 (ktetik), Tabela 4.17 (muški etnik i odgovarajući prisvojni pridev), Tabela 4.18 (ženski etnik i odgovarajući prisvojni pridev).

ktetik								
jednočlani toponimi	ukupan broj toponima	broj klasa	broj singltona	broj toponima u 5 najbrojnijih klasa				
etnik na *-ac*	293	20	9	107	100	27	25	7
etnik na *–in* (-(*j*)*anin*)	331	34	16	63	63	55	34	20
ostali etnici	34	14	8	12	4	3	3	2

Tabela 4.16

muški etnik i odgovarajući prisvojni pridev								
jednočlani toponimi	ukupan broj toponima	broj klasa	broj singltona	broj toponima u 5 najbrojnijih klasa				
etnik na *-ac*	293	27	19	95	81	44	37	10
etnik na *–in* (-(*j*)*anin*)	331	46	22	96	30	26	17	15
ostali etnici	34	20	13	7	3	3	2	2

Tabela 4.17

ženski etnik i odgovarajući prisvojni pridev								
jednočlani toponimi	ukupan broj toponima	broj klasa	broj singltona	broj toponima u 5 najbrojnijih klasa				
etnik na *-ac*	293	29	20	94	82	44	32	10
etnik na *–in* (*-(j)anin*)	331	45	20	96	30	26	17	15
ostali etnici	34	16	8	5	5	4	3	3

Tabela 4.18

Rezultate klasifikacije za pojedinačne izvedenice dvočlanih toponima prikazuju Tabela 4.19 (ktetici), Tabela 4.20 (muški etnik i odgovarajući prisvojni pridev), Tabela 4.21 (ženski etnik i odgovarajući prisvojni pridev).

ktetici								
dvočlani toponimi	ukupan broj toponima	broj klasa	broj singltona	broj toponima u 5 najbrojnijih klasa				
etnik na *–ac*	104	20	7	37	13	10	8	7
etnik na *–in* (*-(j)anin*)	80	31	15	10	7	6	6	6
ostali etnici	2	2	2	1	1			

Tabela 4.19

muški etnik i odgovarajući prisvojni pridev								
dvočlani toponimi	ukupan broj toponima	broj klasa	broj singltona	broj toponima u 5 najbrojnijih klasa				
etnik na *–ac*	104	21	10	24	24	13	7	7
etnik na *–in* (*-(j)anin*)	80	37	22	16	5	4	4	4
ostali etnici	2	2	2	1	1			

Tabela 4.20

ženski etnik i odgovarajući prisvojni pridev								
dvočlani toponimi	ukupan broj toponima	broj klasa	broj singltona	broj toponima u 5 najbrojnijih klasa				
etnik na *–ac*	104	20	9	24	24	13	7	7
etnik na *–in* (-(*j*)*anin*)	80	37	22	16	5	4	4	4
ostali etnici	2	2	2	1	1			

Tabela 4.21

Zaključak

Ovaj rad je inspirisan višejezičnom bazom vlastitih imena Prolex. Njegova prvobitna motivacija je bila da u okviru baze Prolex pojednostavi kreiranje i održavanje resursa za srpski jezik. Stoga je najpre dat kratak pregled problema automatskog prepoznavanja i klasifikacije imenovanih entiteta, posebno vlastitih imena. Detaljno je opisana ontologija kojom su predstavljena vlastita imena u bazi Prolex i uvedeni pojmovi pivota (konceptualno vlastito ime) i prolekseme (kanonski oblik, tj. lema svih instanci vlastitog imena u konkretnom jeziku). Primećeno je da iako raspoloživi resursi imaju značajnu ulogu prilikom prepoznavanja i klasifikacije imenovanih entiteta, nije bitna samo količina informacija koju oni poseduju već i način na koji su te informacije organizovane i međusobno povezane raznim leksičkim i semantičkim relacijama.

Problem automatskog prepoznavanja i klasifikacije vlastitih imena je u direktnoj vezi sa njihovim predstavljanjem u morfološkom elektronskom rečniku. Automatska morfološka analiza teksta pomoću morfološkog elektronskog rečnika efikasno se implementira korišćenjem konačnih automata i transduktora. Stoga je iscrpno opisana primena konačnih automata i transduktora u obradi prirodnih jezika. Aciklični konačni automati se uvode kao struktura koja je podesna za efikasnu i kompaktnu reprezentaciju leksičkih informacija.

U nastavku rada razmatraju se relacije koje se uspostavljaju između prolekseme i njenih derivacionih oblika, kao i relacije između njenih aliasa i njihovih derivacionih oblika. Primeri ovih tipova relacija su imena muških i ženskih stanovnika toponima, prisvojni i relacioni pridevi izvedeni od toponima i stanovnika, itd. Posebno je razmotren fenomen regularne derivacije (značenje izvedene reči se može izvesti iz značenja motivne reči.). Ovaj fenomen je značajan zbog toga što prilikom automatske morfološke analize teksta pomoću morfološkog elektronskog rečnika posebno brojnu grupu nepoznatih reči (tekstuelne reči kojih nema u e-rečniku) predstavljaju rezultati regularne derivacije i imenovani entiteti. Razmotrena je uloga regularne derivacije u dopuni sadržaja elektronskog rečnika i uveden pojam super-leme. Super-lema obezbeđuje kompaktan i sistematičan opis strukture rečničke odrednice tako da pored njenih flektivnih osobina budu obuhvaćena i pojedina derivaciona svojstva. Jedna od osnovnih ideja u ovom radu jeste da se u morfološki elektronski rečnik unose isključivo superleme sa pridruženim pravilom regularne derivacije koje opisuje generisanje odgovarajućih izvedenica. Na taj način bi se aproksimiralo prisustvo lema izvedenih regularnom derivacijom i omogućilo njihovo prepoznavanje tokom morfološke analize.

Celovit prikaz problema opisa regularnih derivacionih svojstva najšire klase imenovanih entiteta ilustrovan je na primeru toponima. Izloženi su principi klasifikovanja navedenih svojstava toponima i uvedeni pojmovi N-regularnog izraza i N-regularnog derivacionog pravila. Pokazano je na koji način se fenomen regularne derivacije može primeniti na izgradnju baze vlastitih imena tipa Prolex na primeru toponima.

Korišćenjem programskog jezika C# implementiran je automatski klasifikator regularnih derivacionih paradigmi jednočlanih i dvočlanih toponima. Za opis morfoloških procesa korišćen je formalizam sistema NooJ. Prikazani su i nedostaci postojeće implementacije sistema NooJ. U odnosu na retke sisteme koji su razvijani za druge slovenske jezike, prilaz koji je primenjen omogućava precizno, iscrpno i sistematsko opisivanje fenomena regularne derivacije u razvoju rečnika vlastitih imena.

Krajnji cilj rada je da se započne opis klasa regularne derivacije u srpskom jeziku (pre svega za vlastita imena), nezavisno od baze Prolex, sistema NooJ ili bilo koje druge implementacije.

Bibliografija

[Aho 72] Aho, Ullman, **The Theory of Parsing**, **Translation and Compiling**, vol 1, Prentice-Hall, New Jersy, 1972.

[Antworth 90] Evan L. Antworth, **PC-KIMMO: A Two-level Processor for Morphological Analysys**, Summer Institute of Linguistics, Occasional Publications in Academic Computing, Number 16, Dallas, Texas, 1990.

[Bekavac 05] Božo Bekavac, **Strojno prepoznavanje naziva u suvremenim hrvatskim tekstovima**, Doktorska disertacija, Filozofski fakultet, Sveučilište u Zagrebu, Zagreb, 2005.

[Bugarski 95] Ranko Bugarski, **Uvod u opštu lingvistiku**, Zavod za udžbenike i nastavna sredstva, Beograd 1995.

[Chinchor 97] Chinchor Nancy, **MUC-7 Named Entity Task Definition**, http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings/ne_task.html, 1997.

[Chinchor 98] Chinchor, Nancy, Elaine Marsh, **MUC-7 Information Extraction Task Definition**, Technical report, verzija 5.1, in the Proceedings of MUC-7, Fairfax, Virginia, 1998.

[Chinchor 99] Nancy Chinchor, Erica Brown, Lisa Ferro and Patty Robinson, **Named Entity Recognition Task Definition**, MITRE, http://www.nist.gov/speech/tests/ie-er/er_99/doc/ne99_taskdef_v1_4.pdf, 1999.

[Coates 93] Coates-Stephens Sam, **The Analysis and Acquisition of Proper Names for the Understanding of Free Text**, Kluwer Academic Publishers, Hingham, MA, 1993.

[Courtois 90] Courtois B, Silberztein M, **Dictionnaires électroniques du française**, Langues française, n°87, 11-22, 1990.

[Cunningham 02] Cunningham, Hamish (2002), **GATE, a General Architecture for Text Engineering**, Computers and the Humanities, vol. 36, str. 223-254.

[Friburger 04] Friburger Nathalie, Maurel Denis, **Finite-state transducer cascades to extract named entities in texts**, Theoretical Computer Science, 313:94-104, 2004.

[Gross 88a] Maurice Gross, **The Use of Finite State Automata in the Lexical Representation of Natural Language**, in [Gross 89b], 1988.

[Gross 88b] Maurice Gross, **Methods and Tactics in the Construction of a Lexicon-Grammar**, SICOL-86, Linguistic Society of Korea, Soeul, 1988, pp. 177-197.

[Gross 89a] Maurice Gross, **La construction de dictionnaires électroniques**, Annales des télécommunications, 44 (1-2), 1989.

[Gross 89b] Gross M, Perrin D, **Electronic Dictionaries and Automata in Computational Linguistics**, Lecture Notes in Computer Science, no. 377, Springer-Verlag, Berlin, 1989.

[Gross 93] Gross, M. **Local Grammars and their Representation by Finite Automata**. In (Ed) Hoey, M. Data, Description, Discourse: Papers on the English Language in Honour of John McH Sinclair. HarperCollins Publishers. pp 26-38, 1993.

[Gruber 95] Gruber T. R, **Toward Principles for the Design of Onthologies Used for Knowledge Sharing**, International Journal of Human-Computer Strudies 43: 907-928, 1995.

[Gucul 06] Sandra Gucul, Vanja Radulović, Cvetana Krstev, **The Usage of NooJ Resources for Detection of Certain Classes of Events**, in Proceedings of the 9th NooJ Conference, Belgrade, 2006.

[Huffman 54] Huffman, D. A., **The synthesis of sequential switching circuits**, Journal of the Franklin Institute, 3, 161-191, Continued in Volume 4, 1954.

[Jurafsky 00] Jurafsky D, Martin S, James H, **Speech and Language Processing**, Prentice Hall, 2000.

[Karttunen 87] Karttunen, Koskenniemi, Kaplan, **A Compiler for Two-level Morphological Rules**, Tools for Morphological Analysis, Stanford University, 1987.

[Karttunen 01] Lauri Karttunen, Kenneth R. Beesley, **A Short History of Two-Level Morphology**, Presented at ESSLLI 2001 Special Event "Twenty Years of Two-Level Morphology" organized by Lauri Karttunen, Kimmo Koskenniemi and Gertjan van Noord. Helsinki. http://www.helsinki.fi/esslli/evening/20years/twol-history.pdf

[Klajn 02] Ivan Klajn, **Tvorba reči u savremenom srpskom jeziku**, **Deo 1**, **Slaganje i prefiksacija**, Prilozi gramatici srpskoga jezika I, Zavod za udžbenike i nastavna sredstva, Beograd, Matica srpska, Novi Sad, Institut za srpski jezik SANU, Beograd, 2002.

[Klajn 03] Ivan Klajn, **Tvorba reči u savremenom srpskom jeziku**, **Deo 2**, **Sufiksacija i konverzija**, Prilozi gramatici srpskoga jezika II, Zavod za udžbenike i nastavna sredstva, Beograd, Matica srpska, Novi Sad, Institut za srpski jezik SANU, Beograd, 2003.

[Korpus 06] **Korpus savremenog srpskog jezika**, Matematički fakultet, http://www.korpus.matf.bg.ac.yu, 2006.

[Koskenniemi 83] Kimmo Koskenniemi, **Two-level Morphology: A General Computational Model for Word-form Recognition and Production**, University of Helsinki, 1983.

[Kristal 96] Dejvid Kristal, **Kembrička enciklopedija jezika**, Nolit, Beograd, 1996.

[Krstev 97] Cvetana Krstev, **Jedan prilaz informatičkom modeliranju teksta i algoritmi njegove transformacije**, Doktorska disertacija, Matematički fakultet, Univerzitet u Beogradu, Beograd, 1997.

[Krstev 04a] Cvetana Krstev, Gordana Pavlović-Lažetić, Duško Vitas, Ivan Obradović, **Using Textual and Lexical Resources in Developing Serbian Wordnet**, in Romanian Journal of Information Science and Technology", vol. 7, No. 1-2, pp. 147-161, Romanian Academy, Publishing House of the Romanian Academy, 2004.

[Krstev 04b] Cvetana Krstev, Duško Vitas, **Restructuring Lemma in a Dictionary of Serbian**, in Zbornik 7. mednarodne multikonference "Informacijska druzba IS 2004" Jezikovne tehnologije 9-15 Oktober 2004, Ljubljana, Slovenija, eds. Tomaz Erjavec, Jerneja Zganec Gros, Institut "Jozef Stefan", Ljubljana, 2004.

[Krstev 05a] Cvetana Krstev, Duško Vitas, Denis Maurel, Mickaël Tran, **Multilingual ontology of proper names**, in Proceedings of 2nd Language & Technology Conference, April 21-23, 2005, Poznań, Poland, ed. Zygmunt Vetulani, pp. 116-119, Wydawnictwo Poznańskie Sp. z o.o., Poznań, 2005.

[Krstev 05b] Cvetana Krstev, Duško Vitas, Sandra Gucul, **Recognition of Personal Names in Serbian Texts**, in Proceedings of the International Conference Recent Advances in Natural Language Processing, 21-23 September 2005, Borovets, Bulgaria, eds. G. Angelova et als., pp. 288-292, 2005.

[Krstev 05c] Cvetana Krstev, Duško Vitas, **Corpus and Lexicon - Mutual Incompletness**, in Proceedings of the Corpus Linguistics Conference, 14-17 July 2005, Birmingham, eds. Pernilla Danielsson and Martijn Wagenmakers, ISSN 1747-9398, http://www.corpus.bham.ac.uk/PCLC/, 2005.

[Krstev 06] Krstev C, Stanković R, Vitas D, Obradović I, **A Workstation for Lexical Resources**, in Proceedings of the 5th International Conference on Language Resources and Evaluation, LREC 2006, pp. 1692-1697, Genoa, Italy, May 2006.

[Laporte 99] Eric Laporte, Anne Monceaux. **Elimination of lexical ambiguities by grammars**. **The ELAG system**, Lingvisticae Investigationes XXII, Amsterdam-Philadelphie : Benjamins, pp. 341-367, 1999.

[Maurel 87] Denis Maurel, **Grammaire des dates**, Mémoires du CERIL, vol. 1, CNAM et Université Paris 7, Paris, pp 218-240, 1987.

[Maurel 96] Maurel D, Belleil C, Eggert E, Piton O, **Le projet prolex**, **séminaire représentations et outils pour les bases lexicales**, In Morphologie

Robuste de l'action Lexique du GDR-PRC CHM, pages 164–175, 1996.

[Maurel 04] Denis Maurel, **Les mots inconnus sont-ils des noms propres?**, JADT 2004, Louvain-la-Neuve, Belgium, 776-784, 2004.

[Maurel 06a] Maurel D, Tran M, Vitas D, Grass T, Savary A, **Prolex : Implantation d'une ontologie multilingue des noms propres**, Rapport interne du Laboratoire d'Informatique de l'Université François-Rabelais de Tours, n°286, 47 p, 2006.

[Maurel 06b] Denis Maurel, Mickaël Tran, Cvetana Krstev, Duško Vitas, **Dictionary of Proper Names and NooJ**, Proceedings of the 9th NooJ Conference, Belgrade, 2006.

[Maurel 06c] Denis Maurel, Franz Guenthner, **Automata and Dictionaries**, King's College Publications, London, 2006.

[Maurel 07] Maurel D., Vitas D., Krstev S., Koeva S., **Prolex: a lexical model for translation of proper names. Application to French, Serbian and Bulgarian**, BULAG n°32, 2007.

[McDonald 96] McDonald David D, **Internal and external evidence in the identification and semantic categorization of proper names**. In Corpus Processing for Lexical Acquisition (ed. by Bran Boguraev and James Pustejovsky), chapter 2, pp. 21-39. The MIT Press, Cambridge, MA, 1996.

[Mealy 55] Mealy, G. H., **A method for synthesizing sequential circuits**, Bell System Technical Journal, 34(5), 1045-1079, 1955.

[Mikheev 99] Mikheev A, Moens M, Grover C, **Named entity Recognition without Gazetteers**, EACL'99 Bergen, Norway, pp. 1-8, 1999.

[Mohri 96] Mohri M, Pereira F. C. N, and Riley M. **Weighted Automata in Text and Speech Processing**, In Proceedings of the 12th biennial European Conference on Artificial Intelligence (ECAI-96), Workshop on Extended finite state models of language, Budapest, Hungary, John Wiley and Sons, Chichester, 1996.

[Mohri 97] Mehryar Mohri. **Finite-State Transducers in Language and Speech Processing**, Computational Linguistics, 23:2, 1997.

[Moore 56] Moore, E. F., **Gedanken-experiments on sequential machines**, In Shannon, C. and McCarthy, J. (Eds.), Automata Studies, pp. 129-153, Princeton University Press, Princeton, NJ, 1956.

[Paskeleva 02] E. Paskaleva, G. Angelova, M. Yankova, K. Bontcheva, H. Cunningham and Y. Wilks, **Slavonic named entities in GATE**. Technical Report CS-02-01, University of Sheffield, 2002.

[Paumier 06] Paumier, S. **Unitex user manual**. [http://www-igm.univ-mlv.fr/%7Eunitex/UnitexManual.pdf] , 2006.

[Pavlović 04] Gordana Pavlović-Lažetić, Duško Vitas, Cvetana Krstev, **Towards Full Lexical Recognition**, in Proceedings of the 7th International Conference TSD 2004 : Text, Speech and Dialogue, Brno, Czech Republic, September 8-11, 2004, eds. Petr Sojka, Ivan Kopček, Karel

Pala, serija “Lecture Notes in Artificial Intelligence” : Subseries of Lecture Notes in Computer Science, eds. J.G. Carbonell, J. Siekmann, pp. 179-186, Springer, Berlin, Heidelberg, 2004.

[Ritchie 92] Graeme D. Ritchie, Graham J. Russel, Alan W. Black, Stephen G. Pulman, **Computational Morphology: Practical Mechanisms for the English Lexicon**, ACL-MIT Press Series in Natural Language Processing, A Bradford Book, The MIT Press, Cambridge, Massachusetts, London, England, 1992.

[Roche 97] Emmanuel Roche, Yves Schabes, editors, **Finite-state Language Processing**, Cambridge, Mass./London, England: MIT Press, 1997.

[Sager 90] Sager J. C, **A Practical Course in Terminology Processing**, John Benjamins, Amsterdam, 1990.

[Savary 00] Agata Savary, **Recensement et description de mots composés – méthodes et applications**, PhD thesis, Université de Marne-la-Vallée, 2000.

[Savary 05] Agata Savary, **Towards a Formalism for the Computational Morphology of Multi-Word Units**, in Proceedings of the 2nd Language & Technology Conference, April 21-23, 2005, Poznań, Poland, ed. Zygmunt Vetulani, pp. 305-309, Wydawnictwo Poznańskie Sp. z o.o., Poznań, 2005.

[Silberztein 94] Max Silberztein, **INTEX: A Corpus Processing System**, Proceedings of COLING 94, ALC, Tokyo, 1994.

[Silberztein 03] Max Silberztein, **INTEX Manual**, **v. 4.33**, http://intex.univ-fcomte.fr/downloads/Manual.pdf, 2004.

[Silberztein 06] Max Silberztein, **NooJ Manual, v. 1.21**, Université de Franche-Comté, http://www.noj4nlp.net, 2006.

[Stanojčić 99] Živojin Stanojčić, Ljubomir Popović, **Gramatika srpskog jezika**, Šesto prerađeno izdanje, Zavod za udžbenike i nastavna sredstva, Beograd, Zavod za izdavanje udžbenika, Novi Sad, 1999.

[Sperberg 94] C. M. Sperberg-McQueen, Lou Burnard, **Guidelines for electronic text encoding and interchange (TEI P3)**, Chicago and Oxford, ACH-ALLC-ACL Text Encoding Initiative, 1994.

[Stanković 04] Stanković, R. Krstev C., Vitas D., Obradović I., **Integrisanje heterogenih leksičkih resursa**, INFOFEST 2004, Budva, pp. 308-316, 2004.

[Stanković 07] Ranka Stanković, **Modeli ekspanzije upita nad tekstuelnim resursima**, Doktorska disertacija, Matematički fakultet, Univerzitet u Beogradu, Beograd, 2007 (u pripremi).

[Tran 04] Tran M., Grass T., Maurel D., **An ontology for multilingual treatment of proper names**, Ontologies and Lexical Resources in Distributed Environments (OntoLex 2004), in Association with LREC2004 (Actes p. 75-78), Lisbonne, Portugal, 2004.

[Tran 05a] Mickaël Tran, Denis Maurel, Duško Vitas, Cvetana Krstev, **A French-Serbian Web Collaborative Work on a Multilingual**

Dictionary of Proper Names, Papillon 2005 workshop on Multilingual Lexical Databases, in Association with the Sixth Symposium on Natural Language Processing (SNLP 2005), Chiang Rai, Thailande, 12-14 décembre, 2005.

[Tran 05b] Tran M, Maurel D, Savary A, **Implantation d'un tri lexical respectant la particularité des noms propres**, Lingvisticae Investigationes, XXVIII-2, 2005.

[Utvić 06a] Miloš Utvić, **The Derivation from Multi-word Proper Names Using NooJ**, Proceedings of the 9th INTEX/NooJ Conference, Belgrade, 2006.

[Utvić 06b] Miloš Utvić, **Regular Derivation of Name Entities in Serbian**, Proceedings of the 5th FASSBL Conference, Sofia, 2006.

[Vitas 81] Duško Vitas, **Generisanje imeničkih oblika u srpskohrvatskom jeziku**, Informatica 3/81, pp. 49-55, Ljubljana 1981.

[Vitas 93a] Duško Vitas, Gordana Pavlović-Lažetić, Cvetana Krstev, **Electronic Dictionary and Text Processing in Serbo-Croatian**, Linguistische Arbeiten, 293, Max Neimeyer Verlag, Tübingen, 1993 (1), pp. 225-231.

[Vitas 93b] Duško Vitas, **Matematički model morfologije srpskohrvatskog jezika (imenska fleksija)**, Doktorska disertacija, Matematički fakultet, Univerzitet u Beogradu, Beograd, 1993.

[Vitas 01] Duško Vitas, Cvetana Krstev, Gordana Pavlović-Lažetić, **Flexible Dictionary Entry**, in Current Issues in Formal Slavic Linguistics, eds. Gerhild Zybatow, Uwe Junghanns, Grit Mehlhorn, Luka Szucsich, pp. 461-468, Peter Lang, Frankfurt amMain; Berlin; Bern; Bruxelles; New York; Oxford; Wien, 2001.

[Vitas 03] Duško Vitas, Cvetana Krstev, Ivan Obradović, Ljubomir Popović, Gordana Pavlović-Lažetić, **An Processing Serbian Written Texts**: **An Overview of Resources and Basic Tools**, in Workshop on Balkan Language Resources and Tools, 21 Novembar 2003, Thessaloniki, Greece, eds, S. Piperidis and V. Karkaletsis, pp. 97-104, 2003.

[Vitas 05a] Duško Vitas, Cvetana Krstev, **Structural derivation and meaning extraction: a comparative study on French-Serbo-Coratian parallel texts,** in Meaningful Texts: The Extraction of Semantic Information from Monolingual and Multilingual Corpora, eds. Geoff Barnbrook, Pernilla Danielsson, Michaela Mahlberg, pp. 166-178, The University of Birmingham Press, Birmingham, 2005.

[Vitas 05b] Duško Vitas, Cvetana Krstev, **Derivational Morphology in an E-Dictionary of Serbian**, in Proceedings of 2nd Language & Technology Conference, April 21-23, 2005, Poznań, Poland, ed. Zygmunt Vetulani, pp. 139-143, Wydawnictwo Poznańskie Sp. z o.o., Poznań, 2005.

[Vitas 05c] Duško Vitas, Gordana Pavlović-Lažetić, **Extraction of named entities in Serbian using Intex**, Proceedings of the 6th and 7th

INTEX/Nooj Workshop, eds. Svetla Koeva, Maurel Denis, Silberztein Max, Formaliser les langues avec l'ordinateur: De INTEX à NooJ, Presses Universitaires de Franche Compte, Paris, 2007.

[Vitas 05d] Duško Vitas, Cvetana Krstev, **Regular derivation and synonymy in an e-dictionary of Serbian**, **in Archives of Control Sciences**, Volume 15 (LI), No. 3, pp. 469-480, Committee of Automation and Robotics, Polish Academy of Sciences, 2005.

[Vitas 05e] Duško Vitas, Cvetana Krstev, **Extending Serbian E-dictionary by the Use of the Lexical Transducers**, Proceedings of the 6th and 7th INTEX/Nooj Workshop, eds. Svetla Koeva, Maurel Denis, Silberztein Max, Formaliser les langues avec l'ordinateur: De INTEX à NooJ, Presses Universitaires de Franche Compte, Paris, 2007.

[Vitas 05f] Duško Vitas, **O problemu nepoznate reči u srpskom**, Zbornik sa 35. međunarodnog naučnog sastanka slavista u Vukove dane, MSC, Beograd, 7-10. 09. 2005.

[Vitas 06] Duško M. Vitas, **Prevodioci i interpretatori: (Uvod u teoriju i metode kompilacije programskih jezika)**, Matematički fakultet, Beograd, 2006.

[Vitas 07a] Duško Vitas, **Lokalne gramatike srpskog jezika**, Zbornik Matice srpske za slavistiku, br. 71-72, Novi Sad, 2007.

[Vitas 07b] Vitas, Dusko; Krstev, Cvetana; Maurel, Denis, **A note on the semantic and morphological properties of proper names in the Prolex project**, In: Sekine, Satoshi and Elisabete Ranchhod (eds.), Named Entities: Recognition, classification and use: Linguisticae Investigationes, Volume 30, Number 1, pp. 115-133(19), John Benjamins Publishing Company, 2007.

[Vossen 98] Vossen P, **EuroWordNet: A Multilingual Database with Lexical Semantic Networks***,* Kluwer Academic Publishers, Dordrecht, 1998.