TARTU ÜLIKOOL

MATEMAATIKA-INFORMAATIKATEADUSKOND

ARVUTITEADUSE INSTITUUT

Jaak Pruulmann-Vengerfeldt

Praktiline lõplikel automaatidel põhinev eesti keele morfoloogiakirjeldus

Magistritöö

Juhendaja: Heli Uibo, MSc

Autor: ..... "..." 2010

Juhendaja: ..... "..." 2010

TARTU 2010

Sisukord

Sissejuhatus

Selle magistritöö motivaatoriks on peamiselt vabavaraliste (siin ja edaspidi on mõeldud avatud lähtetekstiga tarkvara) eesti keele keeletehnoloogiliste vahendite hetkeseis. Üldiselt tegeldakse eesti keele uurimise ja keeletehnoloogiliste vahendite loomisega üksikutes teadusasutustes (Tartu Ülikool, Eesti Keele Instituut, Tallinna Tehnikaülikooli Küberneetika Instituut) ning erafirmades (Filosoft, aga ka nt Google[6]). Vahel tegutsevad nii akadeemilises maailmas kui ettevõtluses samad inimesed. Tulemus on, et teemat valdavaid inimesi on vähe ning neil pole kas tahtmist või ajalist ressurssi vormistada loodud vahendeid vabalt levitataval või taaskasutamist võimaldaval kujul.

Arvatavasti on keeleressursside ja keeletehnoloogiliste vahendite suhteliselt vähese avatuse põhjuseks ka asjaolu, et sisulise töö tegemise hetkel kasutada olnud lähtematerjalid või eesti keelele sobivad tööriistad olid ja enamasti on jätkuvalt kaitstud kinniste litsentside ja autoriõigusega.

Vahendid inglise keele õigekirja kontrollimiseks, poolituseks, aga ka elementaarseks süntaksi- ja stiilikontrolliks olid olemas juba eelmise sajandi 70ndatel aastate lõpus [2]. Muude keelte jaoks vajalikud täiendused ilmusid aga alles 80ndate lõpus ning 90ndate keskel, siis kui laiemalt hakkasid levima Richard Stallmani GNU projektis loodud vahendid ning vaba litsentsiga BSD variandid. Paraku olid ka esimesed täiendused peamiselt teiste keelte jaoks vajalike märgistike lubamine, alternatiivsete sõnastike valimine ning vaid vähesel määral näiteks keerulisemad reeglid, mis on mõeldud liitsõnamoodustuse kirjeldamiseks. Rakenduste aluseks olevad algoritmid on enamasti samad, inglise keelele väljatöötatud sõnastikupõhised lahendused. Teistsuguseid formalisme kasutavad süsteemid on levima hakanud alles hiljuti ning pole veel suuremate vabavaraliste süsteemidega (OpenOffice.org, Mozilla jt) liidestatud.

Eesti keel erineb suurema kasutajaskonna ja sellest tuleneva parema keeletehnoloogilise toega keeltest (eriti inglise keel) oluliselt ning suurematele keeltele mõeldud keeletehnoloogilised vahendid pole eesti keele jaoks ilma lisatööta kasutatavad. Eriti tugevalt mõjutab vabavaraliste keeletehnoloogiliste rakenduste eesti keelele sobivust eesti keele suhteliselt keeruline sõnamoodustus nii lihtsõnade muutumise kui ka sõnatuletuse ja liitsõnamoodustuse osas. Praktiliselt vaba liitsõnamoodustus ja paljud produktiivsed tuletised tähendavad, et korrektseid sõnavorme on lihtsalt sõnastikus loetlemiseks liiga palju, sõna keelde kuulumise otsustamiseks ehk tavaliseks õigekirjakontrolliks on vältimatult vaja tunda mingit varianti sõnamoodustuse reegleist.

Eesti keele jaoks on olemas Enn Saare koostatud poolitusreeglid, mida saab

kasutada algselt TEX'i jaoks loodud poolitusalgoritmiga [20]. Kuna algoritm on hästi dokumenteeritud ning ajalooliselt on olemas ka poolitusreeglid paljudele keeltele, on algoritmi kasutatud ka teistes tekstitöötlussüsteemides. Sisuliselt näitavad poolitusreeglid mingi hulga tähekombinatsioonide kohta positsioone, millel poolitamine on lubatud või keelatud ning sellisena ei saa väga mugavalt arvestada reeglitega nagu "poolita liitsõna piiril". Rangelt võttes pole poolitusreeglid ka avatud lähtetekstiga, sest nende saamiseks kasutatud andmed ja programmid pole avalikult kättesaadavad, kuid pole päris välistatud, et need ongi "iseenda lähtetekst", st on käsitsi koostatud [28]. Reeglite esitus on algoritmi erinevate realisatsioonide piires muutunud kuid TEX'ile mõeldud reeglid on lihtsalt teisendatavad ka nt OpenOffice.org sees kasutatava pisut täiendatud, omapärasemate poolitusreeglitega (aga ka liitsõnadega) keeli toetava mootori [25] jaoks.

Eesti keele õigekirjakontrolliks on peamiselt olemas erinevad sõnastikud kümmekond aastat tagasi üsna laialt levinud olnud programmile *ispell*[1] ja selle analoogidele ja järglastele. Sõnastikest värskeim[2] valmis aastal 2003 [26]. Selle sõnastiku peamiseks probleemiks on liitsõnamoodustuse kirjelduse puudumine. Minimaalse kasutatavuse tagamiseks on vajalik rakendada lülitit "luba kõigi sõnastikus esinevate (liht)sõnade omavaheline liitmine". *Ispell* ja selle hilisemad järeltulijad võimaldavad afikspakkimise ja paisktabelite abil esitada suuri sõnastikke nii, et need kasutaksid mõistlikult vähe mälu, neist saaks kiiresti otsida ning sõna mitteleidmisel oleks võimalik mugavalt pakkuda ka asendusi. Uuemate õigekirjakontrolliprogrammide erinevused seisnevad pakutavates (programmiliselt kasutatavates) liidestes, pisut erinevas afiksitekirjelduses (peamiselt lubatud afiksikomplektide arvu ja afiksite tähistamise osas, *hunspell*[3] võimaldab ka afiksitele veel ühe kihi afikseid liita), liitsõnade moodustamise reeglite osas ning eelnevatest uuendustest tingitud failiformaadi erinevustest. Üldiselt on *ispelli* sõnastikud teisendatavad hilisemate süsteemide sõnastikeks. Eesti keele liitsõnamoodustuse kirjeldamiseks vajalikke vahendeid *ispell* ei paku, suhteliselt hea tulemuse võiks saada programmiga *hunspell*.

Osalt leevendavad keeletehnoloogiliste vahendite põuda ka Filosofti tasuta speller ja poolitaja OpenOffice.org'ile[4]. Paraku on need olemas vaid teatud platvormidele ja teatud OpenOffice.org'i versioonidele. Sellisena, nii tehnilistel kui ka kasutustingimustest tulenevatel põhjustel, pole need integreeritavad muu tarkvaraga ning pole ka sobivad vähem levinud platvormidel (mõeldud on eeskätt arvutiarhitektuure nagu 32bitised vs

[1] `http://www.lasr.cs.ucla.edu/geoff/ispell.html`, 23.07.2010
[2] `http://www.lasr.cs.ucla.edu/geoff/ispell-dictionaries.html`, 23.07.2010
[3] `http://hunspell.sourceforge.net`, 23.07.2010
[4] `http://www.filosoft.ee/freeware/`, 23.07.2010

64bitised süsteemid, x86 vs sparc vs ARM jne) kasutamiseks. Seesama platvormide ja versioonide rohkus on vähemalt osaliselt ka põhjuseks, miks alates umbes 2006. aastast pole Filosoft avaldanud OpenOffice.org keelevahenditest uut Linuxi-versiooni. Mõistlik testimine ja kõigi platvormide toetus oleks ehk mõeldav, kui see mootor oleks avatud lähtetekstiga – siis oleks vabavaraentusiastidel vähemalt võimalus ehitada vahendid ka oma lemmikplatvormi jaoks, kuid autoritel on õigus oma tööle litsents valida ning seni on Filosofti vahendid suletud.

Kokkuvõtvalt on praegu olemas vabalt kasutatavad väga esmase taseme vahendid eesti keele õigekirjakontrolliks ning poolituseks. Paraku pole need (vähemalt õigekirja kontrolli osas) eriti kvaliteetsed ning nende kasutamine keerulisemates keeletehnoloogilistes rakendustes, nagu grammatikakorrektor või sõnavormide süntees, pole võimalik. Keerulisemate rakenduste eelduseks on taaskasutatav morfoloogiakirjeldus või -mootor, mille põhjal saaks muuhulgas luua ka täpsemaid sõnastikke või isegi morfoanalüsaatoril põhineva spellerimootori.

Eesti keele keeletehnoloogiliste vahendite arendamist toetab ka riiklik programm "Eesti Keele Keeletehnoloogiline Tugi"[5]. Programmi kohta käivad tekstid soovitavad tulemite jaoks kasutada vabavaralisi litsentse [36], kuid sobiva litsentsiga morfoloogiamooduli puudumine teeb ka vaba litsentsiga, kuid morfoloogilisest analüsaatorist või süntesaatorist sõltuvate tulemite kasutamise vabavaralistes süsteemides keeruliseks.

Käesolev töö kirjeldabki samme praktiliseks kasutamiseks sobiva vabavaralise morfoloogiakirjelduse saamisel. Aluseks on võetud lõplikel automaatidel põhinev morfoloogiamudel, mille koostamisega tegeles oma magistritöö ja sellele järgnenud uurimistöö raames Heli Uibo. Sellise valiku põhjendused on:

- Lõplikel automaatidel põhinev morfoloogiamudel on üks edukamaid morfoloogiakirjeldamise viise alates esimestest sellealastest sammudest 1980ndate alguses [27].

- Keelekirjeldus on (suhteliselt lihtsatest) rakendusmoodulitest eraldatud, kirjeldust kasutavad programmid on üldiselt keelest sõltumatud. Tõenäoliselt on vajadusel võimalik kas teisendada keelekirjeldus mingile muule kujule (näiteks sõnastik programmile *hunspell*) või ehitada uute vahenditega rakendusmoodul, mis kasutaks loodud lõplikku automaati.

- Juba loodud ning kättesaadavad ressursid (Heli Uibo kahetasemelised reeglid ja vastavad sõnastikud ning EKI andmefailid) on omavahel vähemalt teoreetiliselt

[5] `http://www.keeletehnoloogia.ee/`, 01.08.2010. Töö autor oli seotud projektiga "Reeglipõhine keeletarkvara", mille täitmise käigus kogutud teadmised ja kogemused ka käesoleva töö valmimist kiirendasid.

ühilduvad ning arvestavad kasutatava märgenduse osas ka järgmise taseme keeletehnoloogiliste vahenditega nagu süntaksianalüsaator jms [24].

Töö koosneb kahest poolest. Esimene pool tutvustab arvutimorfoloogia olemust ja tähtsust võrreldes teiste võimalike keeletehnoloogiliste moodulitega ning seni olemasolevaid eesti keele arvutimorfoloogia süsteeme. Lähemalt on juttu ka lõplikel automaatidel põhinevate morfoloogiasüsteemide ajaloost ja ülesehitusest. Teine pool käsitleb töö käigus eesti keele lõplikel automaatidel põhinevale morfoloogiakirjeldusele tehtud uuendusi. Toodud on uuenenud süsteemi ülesehitus ja soovitused võimalikele järgmistele arendajatele.

Tööle on lisatud CD-plaat, millel on kogu töö käigus loodud tarkvara, valmisehitatud lõplik teisendaja ning käesolev tekst.

Peatükk 1

Eesti keele morfoloogia ja arvuti

Eesti keele keeletehnoloogilisest toest 2009. aasta seisuga saab ülevaate artiklist [22]. Eesti keele morfoloogiauuringute ajalugu ja sisu kirjeldab artikkel [10].

1.1 Keele erinevad tasandid

Keel on märgisüsteem, mida kasutatakse suhtlemiseks [3]. Inimeste vahel kasutatavate (suuliste) keelte väikseimateks ehituskivideks on häälikud. Eesti Keele Käsiraamatu põhjal koostatud lihtsustatud mudel keele koostisest võiks välja näha umbes järgnev:

Madalaim tase on häälikud. Häälikuid ei loeta veel otseselt keele osaks, sest sageli ei muuda hääliku varieerimine või asendamine sarnaselt kõlavaga (r kurgunibuga vs keeleotsaga) öeldud sõna tähendust kuulaja jaoks. Enamgi veel, kasutatavat häälikut mõjutavad sellele eelnevad ja järgnevad. Häälikute uurimisega tegeleb foneetika.

Häälikuhulki, mida keelekasutajad eristavad, nimetatakse foneemideks. Foneem on väikseim keele üksus, mis on võimeline eristama tähendusi. See tähendab, et kui sõnas mõni foneem asendada, siis tajub kuulaja seda teise sõnana. Foneemil endal tähendust pole. Sama abstraktse foneemi erinevaid esinemisi suulises kõnes nimetatakse allofoonideks. Eesti häälikkiri tähendab, et tavaliselt vastab kirjutatud keeles, sõltumata kasutatud foneemivariandist, igale kasutatavale foneemile üks täht. Foneemide kasutamist keeltes, näiteks mingi foneemi kuulumist või mittekuulumist keelde, keelele iseloomulikke foneemipiire ja foneemide võimalikku kombineerumist (fonotaktika) keeles uurib fonoloogia.

Foneemidest moodustatakse morfeemid. Morfeem on vähim iseseisev tähendust kandev üksus keeles. Morfeemid on nii sõnade tüved kui ka tüvede tähendust ja sõnaliiki muutvad tuletusliited ning grammatilist tähendust täpsustavad tunnused ja lõpud. Sarnaselt foneemidele on ka morfeemid abstraktsed üksused, millel on tegelikus

kasutuses mitmeid eri kujusid ehk allomorfe (nt kala-de-le vs õpiku-te-le).

Morfeemidest moodustatakse sõnad ja sõnavormid. Tavaliselt koosneb sõna vähemalt ühest tüvimorfeemist ning võib sisaldada ka liidemorfeeme. Keeles esinevaid morfeeme, nende variante ja kombineerumisvõimalusi (morfotaktika) uurib morfoloogia. Siit on näha, et traditsiooniliselt tegeleb morfoloogia ka (liit)sõnamoodustusega, kuid eesti keele morfoloogiakäsitlused on enamasti piirdunud vaid vormimoodustusega ehk tüvevariantide ja grammatilist tähendust väljendavate morfeemide ja nende omavahelise kombineerumise probleemidega. Sõnamoodustust ehk liitsõnade ja tuletiste moodustamist on uuritud eraldi ning kahjuks ka pisut vähem kui vormimoodustust.

Mõnikord räägitakse ka morfofonoloogiast, mis uurib nii fonoloogia kui morfoloogia aspekte ning eriti seda, kuidas need vastastikku üksteist mõjutavad.

Sõnavormidest moodustatakse fraasid, lauselühendid, osalaused ja laused. Sõnajärge lauses, kasutatavaid sõnavorme ja lausete moodustamise reegleid uurib süntaks ehk lauseõpetus. Lausete, aga ka sõnade moodustamise reegleid nimetatakse grammatikaks.

Sõnade ja lausete tähendust uurib semantika. Mingi tähendusega lausete kasutamist ja tähenduste muutumist (teiste lausete, aga ka kultuuri) kontekstis uurib pragmaatika.

Nimetatud tasemed kirjeldavad peamiselt verbaalset keelt, kuid lisaks sõnaliselt edastatud informatsioonile kannavad suulises loomulikus kõnes tähendust ka sõnade rütm, intonatsioon, rõhud jms, mida uurib prosoodia. Veel on keelekasutajate arsenalis pausid, näoilmed, suhtlushäälitsused (“ee”, “mhmh”), žestid, jms, mille analüüsimise ja kirjeldamisega tegelevad multimodaalse suhtluse uuringud.

Märkimist väärib ka asjaolu, et (ametlik) kirjakeel ja tegelikult kasutatav suuline kõne erinevad üksteisest nii kasutatavate sõnavormide (kakskend $<$ kakskümmend) kui lauseehituse (poolikud laused jms) poolest – mõnes mõttes on tegu kahe lähedase, kuid siiski erineva keelega.

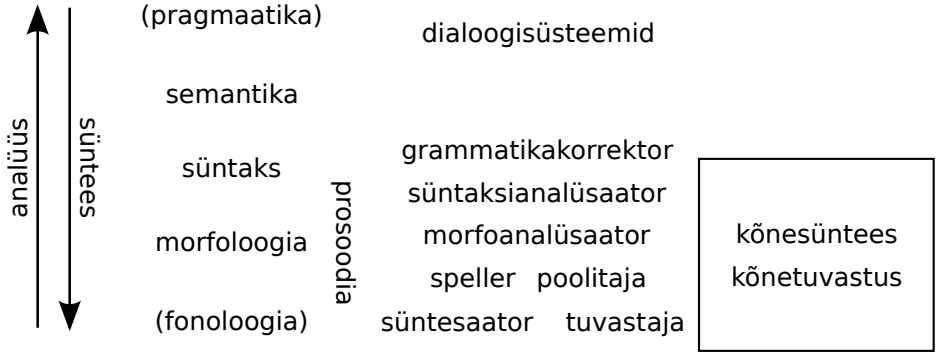
1.2 Keele tasandid ja arvuti

Keeletehnoloogilisi vahendeid kasutatakse üsna erinevatel eesmärkidel nii keele teaduslikuks uurimiseks kui ka komponentidena kõikvõimalikes tehnoloogilistes süsteemides. Näiteks võib tuua tehisintellekti ja kasutajakogemusega seotud rakendused, mille üheks osaks on loomulikus keeles esitatud teadmistest, soovidest, käskudest, kavatsustest jne arusaamine (loomuliku keele analüüs) ning sünteesitud teadmiste esitamine loomuliku keele vahenditega. Palju rakendusi on mõeldud inimese keelega seotud tegevuste toetamiseks, näiteks õigekirjakontrollijad, süntaksikontrollijad, aga ka tekste või kasutajaliideseid ettelugevad süntesaatorid ning suulist kõnet kirjalikuks tekstiks

teisendavad kõnetuvastajad.

Tavaliselt vajavad kõrgemal keele tasemel töötavad rakendused madalamate tuge. Näiteks (lause)grammatikakorrektor vajab tööks teadmist lauses esinevates sõnades sisalduva grammatilise informatsiooni kohta. Tarvis on morfoloogilise analüsaatori väljundit. Sageli on vaja ka teistpidist tuge. Madalama taseme mooduli väljundi kontrollimiseks tuleb veenduda ka kõrgema taseme reeglitega määratud struktuuri sobivuses. Nii töötab tavaline (sõnade) õigekirja kontrollija (speller) vaid morfoloogia tasemel ning just seetõttu jääb hätta lausesse mittesobivate sõnavormide või isegi sõnatüvede tuvastamisega. Enamgi veel, sageli ongi vaja keelekasutuse korrektsuse hindamiseks edastatavast mõttest aru saada. Näiteks lause "Ilm on jäälegi soe." võib olla kas täiesti korrektne või siis, tavalist keelekasutust arvestades, on sõna "jäälegi" asemel mõeldud sõna "jällegi".

Joonis 1.1 visualiseerib üksteise kohal olevate kihtide mõtte ning paigutab võrdluseks kõrvale mitmesugused keeletehnoloogilised moodulid. Kõnesüntees ja kõnetuvastus on näited rakendustest, mis vajavad oma töös päris mitme kihi abi ja pigem tegelevadki erinevate väiksemate moodulite koordineerimisega.



Joonis 1.1: Keele kihid ja neile vastavate rakenduste näiteid

Sissejuhatuses kirjeldatud vabavaraliste keeletehnoloogiliste vahendite üheks puuduseks ongi piirdumine vaid minimaalse vajaliku keelekirjeldusega. Nii on poolitusreeglid väljendatud vaid tähejadade ja neis esinevate poolitus(keelu)kohtadega, kuid ei sisalda tavaliselt infot morfoloogiliste nähtuste nagu sõnade liitumine kohta. Liitsõnade poolitamise juures on eriti põnevad liitsõnad, mida on võimalik analüüsida mitmeti (*kuu–setukas vs kuuse–tukas*). Sõnastikupõhine õigekirjakontroll on küll suhteliselt töötav, kuid ei anna piisavalt informatsiooni järgmise taseme kirjutamisabivahendite nagu grammatikakorrektori või sisukokkuvõtjate tegemisel.

1.3 Senised morfoloogiakirjeldused

Selle töö sisu on luua vabavaralisteks ettevõtmisteks kasutatav formaalne morfoloogiakirjeldus. Eesti keele jaoks on juba mõned morfoloogiakirjeldused olemas, kuid nende kasutamine vabavaralistes rakendustes on mitmetel põhjustel raskendatud.

Teoreetiliselt on eesti keele kohta olemas mitmeid morfoloogiakirjeldusi. Arvutiprogrammina esitamiseks, automaatseks morfoloogiliseks analüüsiks ja -sünteesiks sobib neist kõige paremini Ülle Viksi töödes kirjeldatu. Ülle Viksi Väike Vormisõnastik [34] (VVS), eesti keele esimene morfoloogiasõnastik, ning selles esitatud tüübisüsteem ongi suuremal või vähemal määral kõigi kolme suurema arvutimorfoloogiasüsteemi aluseks. Ülle Viksi tööd, sh Väike Vormisõnastik tegelevad vaid lihtsõnade ja üksikute tuletiste kirjeldamisega. Liitsõnamoodustuse formaalselt kirjeldamine on märksa keerulisem probleem, sest sõnade liitumine või mitteliitumine ning võimalikud liitumismallid sõltuvad vaadeldavate sõnade tähendusest. Eesti sõnamoodustust, nii tuletiste kui liitsõnade osas kirjeldavad näiteks [4] ja [3]. Mitmeid näiteid pealtnäha reeglipärasel liitsõnamoodustusel tekkivatest takistustest toob Reet Kasik [17].

1.3.1 EKI tarkvara

Eesti Keele Instituudis loodud tarkvara on Väikese Vormisõnastikuga väga tihedalt seotud, sest süsteemi autorite hulgas on ka Ülle Viks ise. Süsteemi juured ulatuvad 1970ndate lõppu ja 1980ndate algusesse, kui toonases Keele ja Kirjanduse Instituudis tegeldi automaatse morfoloogilise analüüsi ja sünteesiga. Alates 1993. aastast käivitus projektide jada, mille käigus loodud tarkvara on EKI veebilehelt[1] kättesaadav ning suhteliselt vabadel tingimustel kasutatav.

Süsteem koosneb hulgast mitmesuguse otstarbega moodulitest, mida saab omavahel konkreetsete rakenduste jaoks kombineerida. Näiteks on eraldi moodulid silbituse, tüvemuutuste, tüübituvastuse, morfoloogilise analüüsi ja sünteesi jaoks. Iga mooduli jaoks on olemas formaalne reeglikomplekt, mida moodul interpreteerib. Kahjuks on suhteliselt kehvas seisus liitsõnade analüüs ja süntees.

Analüüsialgoritm seisneb sisuliselt võimalike lõppude ja liidete eemaldamises ning vajadusel tüve teisendamises algvormi moodustamiseks vajalikuks variandiks. Soovi korral võib seejärel kontrollida, kas saadud algvorm ka tegelikult sõnastikus esineb. Sünteesiks kasutatakse samu reegleid teistpidi – alustades algvormist ja teadaolevast muuttüübist genereeritakse vajalik tüvevariant ning liidetakse sellele soovitud vormi formatiiv. Vajadusel saab muuttüübi oletada.

[1] `http://www.eki.ee/tarkvara/`, 01.08.2010

EKI süsteemide alussõnastikuks on VVS sõnastiku elektroonne esitus. VVS sõnastiku aluseks on 1976. aasta õigekeelsussõnaraamatu sõnastik, mille keelekasutust on uuendatud vahepeal muutunud normide osas ning kust on eemaldatud palju morfoloogiasõnastikule ebaolulist. See on arvatavasti põhjuseks näiteks pärisnimede (kohanimed, levinud eesti eesnimed) puudumisele aga suhteliselt paljude vanade või murdesõnade leidumisele sõnastikus. Hilisemad, EKI sisemiseks kasutamiseks mõeldud süsteemid kasutavad tööks morfoloogilist tüvebaasi ja sõnamoodustuse andmebaasi [10], kusjuures algne versioon tüvebaasist on tegelikult seesama VVS sisu. Sõnastiku sisu mõjutab ühe olulise aspektina morfoloogiasüsteemi kasutatavust konkreetsete keeletehnoloogiliste rakenduste jaoks. Kui näiteks morfoloogilise analüüsi jaoks saab tundmatute sõnade analüüsiks kasutada (ja EKI süsteemi korral kasutataksegi) oletamist, mis suudab analüüsida ka üsna kummalisi sõnu, siis õigekirjakontrolli juures on oluline, kas oletaja leitud tüvi ka tegelikult keelde kuulub.

Morfoloogiliseks analüüsiks ja sünteesiks kasutatavate märgendisüsteemide osas on EKI tarkvara ajalooliselt kõige tublim, sest omab sisseehitatud tuge erinevate märgendisüsteemidega töötamiseks. Süsteem kasutab sisemiselt, reegli- ja erandifailides, enda-spetsiifilisi vormikoode. Sisendis ja väljundis on kasutatavad sisemised vormikoodid, VVSis kasutatud vorminimedest tuletatud lühendid, formatiividest tuletatud vormikoodid ning "Filosofti koodid". Filosofti koodidega analüüsi korral on ka analüsaatori väljundi kuju lähedane ESTMORFi väljundile ning ühtlasi ühestaja ja süntaksianalüsaatori sisendile. Arvestades kasutatud tehnikat pole ka mõne muu märgendisüsteemi lisamine keeruline. Süsteem oskab tüveteisendusel ja vormide moodustamisel arvestada (ka VVSis leiduvate) andmetega kolmanda välte kohta.

EKI tarkvara praktilisel kasutamisel vabavaralistes süsteemides tekib kaks probleemi. Esiteks on süsteemi erinevad moodulid realiseeritud erinevate vahenditega (Pascal vs C vs C++) ning arvestades toona levinud MS-DOS'i platvormi. See raskendab pisut tarkvara kompileerimist tänapäevaseid vahendeid kasutades, kuid põhimõtteliselt pole selline kohandamine võimatu ning osaliselt on seda ka tehtud [26]. Teine, pisut keerulisema lahendusega probleem on suhteline aeglus võrreldes teiste süsteemidega. Aegluse põhjuseks on ühelt poolt modulaarne struktuur – moodulid kasutavad üksteise tulemusi ning iga sõltumatu moodul peab oma töö tegema "lõpuni", sest pole teada, milliseid võimalikke väljundeid kasutatakse, otseteid ja optimeerimisi on nii keerulisem teha. Aeglusse annab oma panuse ka reeglifailide interpreteerimine, mida annaks võib-olla parandada suurema vahetulemuste puhverdamisega. Tüvebaas, sisuliselt ettearvutatud tüvevariantide baas, ongi tõenäoliselt samm selles suunas.

1.3.2 ESTMORF

OÜ Filosoft (Heiki-Jaan Kaalep, Tarmo Vaino ja Rene Prillop) loodud morfoloogia-analüsaator ja -süntesaator on praegu arvatavasti parim praktiliselt kasutatav eesti keele morfoloogiakirjeldust sisaldav süsteem. ESTMORFil põhinevad ka paljud kommertstarkvarale mõeldud õigekirjakontrolli ja poolitusmoodulid, sh MS Office'ile mõeldu.

Süsteemi arendamisel võeti üheks eesmärgiks tegelike tekstide võimalikult veavaba analüüsimine ning selleks sobib ESTMORF kindlasti paremini kui EKI analüsaator [9]. Oluline sisuline erinevus võrreldes EKI süsteemiga on liitsõnade ja tuletiste osa. Liitsõnade võimaliku struktuuri, tõenäoliste komponentide jms analüüsiks on ESTMORFi jaoks tehtud täiendavat uurimistööd, mille tulemused muudes morfoloogiasüsteemides ei sisaldu. Samuti on üsna suures mahus muudetud kasutatavat sõnastikku. Mittesisuliseks eripäraks on süsteemi lähteteksti kinnisus – süsteemi ega selle komponente ei saa autorite aktiivse panustamiseta kasutada võimalike uute rakenduste jaoks kohandamiseks ega ka kasutada süsteemi platvormidel, mida autorid ei toeta.

Algoritmi poolest on ESTMORF üldjoontes sarnane EKI süsteemile, kasutatakse sõnatüvede ja liitumistingimustega varustatud liidete loendeid, analüüsitava sõna juppideks lammutamist ning juppide loenditega võrdlemist. Oluline kiirusevõit võrreldes EKI süsteemiga on saavutatud programmi mahuka testimisega praktiliselt kasutataval keelel, s.o. tekstikorpustel ning saadud tulemuste alusel algoritmi ja kasutatavate tugiandmete esituse parandamisega. Arvestatava kiirusevõidu annab tõenäoliselt ka tüvevariantide otse sõnastikus esitamine, erinevalt EKI süsteemist, mis neid pidevalt käigupealt tuletab. Liitsõnade analüüsiks on korpuste abil tuletatud algoritm, mis arvestab liitsõna struktuuri tõenäosust.

ESTMORFi sõnastiku aluseks on, nagu EKI süsteemilgi, Väikese Vormisõnastiku sõnastikuosa, millest on eemaldatud suur osa vähekasutatud (murdelisi või vananenud) sõnu ja tuletisi, mis on muudest sõnadest tuletusmehhanismi abil saadavad ning kuhu on lisatud uusi lihtsõnu, pärisnimesid, aga ka liitsõnu, mille moodustamine on ebatavaline (keeruline või pole reeglipärane). Sõnastiku sisu on optimeeritud arvestades tänapäevast keelekasutust, st reaalsetes tekstides esinevaid sõnu ning analüsaatori kasutamist õigekirja kontrollimiseks.

ESTMORFi morfoloogilise märgendisüsteemi kasutajale paistev osa tugineb VVSis kasutatud tähistusele, väljastatakse sõnaliik ja sõnaliigist sõltuv vormitunnus. Käändsõnade korral on tunnus lühend arvu ja käände ladinakeelsest nimetusest, nii nagu see on antud VVSis (*sg n, pl ad*), pöördsõnade korral on aga antud formatiiv.

Tüübinumbrit ESTMORF ei väljasta. ESTMORFi sisemine esitus, kui see peaks välisest erinema, kasutajale ei paista. ESTMORF oskab analüüsil väljastada ka infot (tüve?) palatalisatsiooni ja kolmanda välte kohta. Viimane on arvatavasti vajalik ESTMORFi mootori kasutamisel kõnesünteesirakendustes, kuid täpsemat dokumentatsiooni selle kohta ei õnnestunud leida.

ESTMORFi ja EKI analüsaatorite väljundite omavahelise võrdluse on koostanud Eveli Saue [29].

1.3.3 Lõplike automaatidega morfoloogiakirjeldus

Lõplikel automaatidel põhineva morfoloogiamudeli loomisega on peamiselt tegelenud Heli Uibo. Oma magistritöös veendub ta, et eesti keele morfoloogiat on põhimõtteliselt võimalik kirjeldada kasutades kahetasemelist morfoloogiamudelit. Tehniliselt on juba magistritöös tegemist mitte enam puhta kahetasemelise mudeliga vaid pigem selle üldistusega, kus kogu morfoloogiakirjeldus kompileeritakse üheks suureks lõplikuks teisendajaks, mis teisendab kahe regulaarse keele, süvaesituses sõnastiku ja pindesituses loomuliku keele sõnade vahel. Viimastes versioonides on asjaolu, et süsteem koosneb lõplikest teisendajatest, veel rohkem ära kasutatud ja seega puhtast kahetasemelisest mudelist veelgi eemaldutud [33].

Erinevalt kahest eelmisest süsteemist ei ole lõplikel automaatidel põhinev morfoloogiasüsteem niivõrd programm või algoritm kui abstraktne keelekirjeldus, mis on esitatud teatavat lihtsat ja suhteliselt universaalset formalismi kasutades. Siiski saame formalismi iseärasusi võrrelda kahe eelmise süsteemi omadustega. Üks olulisemaid erinevusi tuleb asjaolust, et sisuliselt on tegu regulaarse relatsiooniga – (võimalik, et lõpmatu) hulga vastetega sõnastikuesituse (tüvi + grammatiline info) ja pindesituse (tegelikus keeles esinevad sõnavormid) vahel. Morfoloogiline analüüs leiab kõik (sõnastikuesitus, pindesitus) paarid, kus pindesitus vastab etteantud sõnale. Leitakse kõik võimalikud analüüsid, eelistamata üht teisele. Näiteks sõna *aastaid* võimalikud analüüsid on teiste hulgas nii mitmuse osastav sõnast *aasta* kui ka liitsõna *aas+taid* (*aas* ainsuse nimetav + *tai* ainsuse osastav või mitmuse nimetav)[2]. EKI süsteem ja ESTMORF liitsõna ei tuvasta, sest lõpetavad analüüsi, kui sõna on võimalik analüüsida lihtsõnana. Lõplike automaatidega pole see aga triviaalselt tehtav, sarnase efekti võiks saada (lõpliku lihtsõnade hulga korral) ebasoovitavate kombineerumiste eelneva väljaarvutamise ja vastava keelufiltri lisamisega.

[2]Uibo sõnastikes ei leidu tegelikult sõnu *aas* ja *tai*, seega see konkreetne näide on konstrueeritud. Vajalik liitsõnamoodustuse reegel, mis lubab ainsuse nimetavale liitnimisõna saamiseks (suvalise) teise nimisõna liita, on aga olemas.

Teine oluline erinevus võrreldes eelmiste kirjeldatud süsteemidega on analüüsikiirus. Lõplik teisendaja vajab pindesitusest sõnastikuesituse saamiseks sama palju samme, kui on kahest sõnest (sõnastikuesitus ja pindesitus) pikemas sümboleid. Iga järgmise variandi leidmiseks kulub sõltuvalt sellest, kui suur osa kahel järjestikusel analüüsil ühine on, pisut vähem samme. Sõltuvalt automaadi esitusest seda kasutavas programmis saab lõplik teisendaja töötada väga kiiresti.

Heli Uibo loodud versioonide kõige nõrgem koht on sõnastik, täpsemalt selle väike maht. Kahetasemelises süsteemis kasutatakse (muu hulgas) tüvemuutuste kirjeldamiseks kahetasemelisi reegleid, mis eeldavad, et sõnastikus antud tüvedes on astmevahelduse tõttu kirjapildis muutuvad tähed tähistatud. Jätkusõnastikusüsteemiga liidetakse tüve süvakujule liited ja lõpud koos tunnusega, mille alusel reeglid väljastavad kas tugeva või nõrga tüvevariandi jms. Kuna sobival kujul sõnastikku olemas ei olnud, koosnevad senised sõnastikud peamiselt väikesest hulgast käsitsi lisatud tüvedest. Tüvede vähesus teeb selle süsteemi aga tegelike tekstide analüüsimisel mittekasutatavaks.

Morfoloogiliste märgendite osas on süsteemi arengu jooksul toimunud muutus – algne, magistritöös kirjeldatud VVSist inspireeritud tähistus on hiljem asendatud T. Puolakaineni morfoloogilises ühestajas kasutatava tähistuse variandiga, mis võimaldab teisendada analüsaatori väljundi lihtsate vahenditega ühestaja sisendiks. Paralleelselt märgendisüsteemi vahetusega on ka jätkuleksikonidena esitatud morfotaktikareeglistik lähenenud Väikese Vormisõnastiku struktuurile, pärast mõnede tüvelõpumuutuste käsitlemist jagatakse sõnad Viksi muuttüüpide alusel koostatud jätkusõnastikesse, olemas on Viksi analoogiarühmi esindavad jätkuklassid jne.

Tehniliselt peaks lõplikel automaatidel põhinev morfoloogiakirjeldus olema vabavaralisteks tegemisteks suhteliselt sobiv. Ühest küljest on lõplikel automaatidel põhinev morfoloogia suhteliselt populaarne ning on olemas näited lõpliku teisendaja kasutamisest süntaksianalüsaatori sisendis [31]. Teisalt on lõplik teisendaja arvatavasti pisut mugavam abstraktsioon kui üldotstarbelises programmeerimiskeeles esitatud algoritm ning keelekirjeldus on sellisena loodetavasti vajadusel teisendatav ka teistsugustesse esitustesse, näiteks afikspakkimise reegliteks. Olemasoleva süsteemi litsentsitingimused ja kasutatavus õiguslikus mõttes on aga segane ja vajab täpsustamist.

1.4 Lõplikel automaatidel põhinevad morfoloogiakirjeldused

1.4.1 Lõplikud automaadid ja lõplikud teisendajad

Lõplik automaat on viisik $(\Sigma, S, s_0, \delta, F)$, kus

- Σ on lubatud sisendsümbolite hulk (sisendtähestik)
- S on lõplik olekute hulk
- $s_0 \in S$ on algolek
- $\delta : S \times \Sigma \to S$ on olekuteisenduse reeglid
- $F \subset S$ on lõppolekute (aktsepteerivate olekute) hulk.

Automaat alustab tööd algolekus ning loeb igal sammul ühe sisendsümboli. Sisendsümboli ja kehtinud oleku alusel siirdub automaat uude olekusse. Kui sisendsõne lõppemisel on automaat mõnes lõppolekus, öeldakse, et automaat aktsepteeris sisendi (sisendsümbolitest moodustatud sisendsõne). Kõigi ühe automaadi poolt aktsepteeritavate sõnede hulka ($L \subset \Sigma^\star$) nimetatakse selle automaadi poolt aktsepteeritavaks keeleks. Lõplike automaatide poolt aktsepteeritavad keeled moodustavad regulaarsete keelte klassi.

Lõplik teisendaja on kuuik $(\Sigma, \Gamma, S, s_0, \delta, F)$, kus

- Σ on sisendtähestik
- Γ on võimalike väljundsümbolite hulk (väljundtähestik)
- S on lõplik olekute hulk
- $s_0 \in S$ on algolek
- $\delta : S \times \{\Sigma \cup \epsilon\} \times \{\Gamma \cup \epsilon\} \to S$ on olekuteisenduse reeglid
- $F \subset S$ on lõppolekute (aktsepteerivate olekute) hulk.

Lõplik teisendaja alustab tööd algolekus. Igal sammul võib teisendaja lugeda sümboli sisendist või jätta sisendi puutumata (seda tähistab olekuteisenduse relatsioonis sisendsümboli asemel esinev epsilon) ning võib väljastada sümboli väljundtähestikust või jätta sümboli väljastamata (tähistatud jälle epsiloniga). Lõplik teisendaja aktsepteerib

regulaarset keelt $L_1 \subset \Sigma^\star$ ning väljastab töö käigus sõnesid regulaarsest keelest $L_2 \subset \Gamma^\star$. Sisuliselt defineerib lõplik teisendaja regulaarse relatsiooni keelte L_1 ning L_2 vahel. Lõplik teisendaja on lihtsalt pööratav, st sisendi ja väljundi saab vahetada.

Kompaktselt, aga siiski üsna põhjalikult kirjeldab lõplikke automaate ning regulaarseid keeli Markus Forsberg oma magistritöö esimeses peatükis [5].

1.4.2 Ajalugu

Lõplikel automaatidel põhinevate morfoloogiakirjelduste juured ulatuvad aastasse 1968, kui Chomsky ja Halle rakendasid generatiivse lingvistika ideid inglise keele fonoloogia kirjeldamiseks [15]. Fonoloogilised grammatikad koosnesid järjestatud ümberkirjutusreeglitest *(rewrite rules)*, mis teisendasid abstraktseid fonoloogilisi süvaesitusi läbi vaheetappide tegelikus keeles esinevateks pindesitusteks. Üldine reeglite kuju oli $\alpha \rightarrow \beta / \gamma \_\_ \delta$, kus α, β, γ ja δ on kuitahes keerulised sümbolijadad või fonoloogiliste tunnuste hulgad. Formaalsete keelte teooria mõttes on need kontekstitundlikud reeglid.

1972. aastal näitas Douglas Johnson, et kuigi tähistuselt on reeglid kontekstitundlikud, kasutasid lingvistid tegelikes keelekirjeldustes oluliselt vähem võimsaid reegleid [7]. Nimelt võib kontekstitundlikke reegleid rakendada ka juba teisendatud osale kuid lingvistid rakendasid neid alati viisil, mis seda ei vajanud. Johnson näitas, et selline piirang tähendab, et fonoloogilised ümberkirjutusreeglid on modelleeritavad lõplike teisendajatega. Johnsoni tulemustest mitteteadlikena taasavastasid selle 1980nda aasta paiku Ronald M. Kaplan ja Martin Kay, kes näitasid, et ümberkirjutusreeglid esitavad tegelikult regulaarseid relatsioone, mis on (definitsiooni järgi) esitatavad lõplike teisendajatena.

Juba Johnson oli teadlik Schützenbergeri 1961. aastal saadud tulemusest: iga kahe järjestikku rakendatud lõplike teisendajate paar on asendatav kolmanda lõpliku teisendusega, mis realiseerib sama teisenduse, mille realiseeris esimese kahe teisendaja kompositsioon [30]. Seega on järjestikku rakendatavad teisendusreeglid esitatavad ühe lõpliku teisendajana, mis teisendab sõnu otse abstraktsest süvakujust pindesitusse.

Need teoreetilised teadmised ei viinud veel aga kiirete praktiliste tulemusteni. Ilmnes, et ümberkirjutusreeglite kompileerimine lõplikuks automaadiks on üsna keeruline ettevõtmine, alustuseks tuli defineerida lõplike automaatide hulgal töötavad elementaaroperatsioonid nagu automaatide ühend, vahe, täiend, kompositsioon jne. Ka lõplike automaatide töötlemine ning kõigi vajalike tehete realiseerimine toonastel arvutitel oli väljakutse. Samuti polnud päris selge, kas lõplikel teisendajail põhinev lahendus on kasutatav ka (morfoloogiliseks) analüüsiks. Lihtne on luua reeglikomplekt,

mis genereerib ühest süvaesitusest parajasti ühe pindesituse, kuid mitmed pindesitused on saadavad erinevatest süvaesitustest ning seega tekib tõenäoliselt mitmene analüüs, halvemal juhul on võimalikud isegi lõpmatult paljud süvaesitused.

Lühidalt oli 1980ndate alguseks teoreetiliselt lahendatud süvaesituse efektiivse pindesituseks teisendamise probleem, kuid polnud selge, kuidas vabaneda liiasest analüüsist. Kümmekond aastat hiljem (1992) taibati, et kui ka süvaesitusi sisaldav sõnastik esitada lõpliku teisendajana, siis sõnastiku-teisendaja ja reeglite-teisendaja täiendavad teineteist parajasti nii, et liigsed analüüsid ei ole võimalikud ning vajalikud kitsendused rakenduvad juba liit-teisendaja kompileerimise ajal. Avastuse tegemise hetkeks oli aga juba laiemalt levinud pisut erinev lõplikel automaatidel põhinev formalism.

1.4.3 Kahetasemeline mudel

Kahetasemelist morfoloogiamudelit kirjeldab Kimmo Koskenniemi oma doktoritöös [18]. Koskenniemi oli suhelnud Kay ja Kaplaniga ning teadlik nende tulemustest. Ka tema polnud veendunud, et lõplike teisendajate liiase analüüsi probleem on lahendatav ning seega otsis ta alternatiivset viisi lõplike automaatide kasutamiseks. Oma doktoritöös pakubki ta välja uue viisi fonoloogiliste muutuste kirjeldamiseks. Erinevalt järjestikku rakendatavate ning vaheesitustega ümberkirjutusreeglitest, määravad (lubavad, keelavad või nõuavad) Koskenniemi reeglid otse mõne sümboli teisenemise süvaesitusest pindesitusse. Kahetasemelised reeglid saavad kasutada teadmisi nii sõna süva- kui ka pindesituse kohta. Reegleid rakendatakse paralleelselt. Koskenniemi realiseeris oma mudeli viisil, mis ei vajanud keerulisi tehteid automaatidega, mis pidurdas Kay ja Kaplani tegemisi. Olulised kahetasemelist morfoloogiat *(two-level morphology)* defineerivad omadused on:

- Iga reegel kirjeldab parajasti ühte pind- ja süvasümboli vastavust kontekstis (või kontekstides).
- Kõik reeglid töötavad paralleelselt ja peavad andma positiivse tulemuse.
- Reeglid saavad kasutada kas süva- või pindesituse konteksti või mõlemat korraga.
- Sõnastiku-otsingut ja reeglite rakendamist tehakse paralleelselt.

Kahetasemelise mudeli sõnastik on jätkuklasside abil seotud sõnehulgad, mis on annoteeritud morfoloogilise infoga. Analüüsi käigus tuvastatakse kõik pindesituse vaadeldavale sümbolile vastavad (reeglite poolt lubatud) süvaesituse sümbolid ning

jätkatakse vaid nende analüüsiharudega, millele vastav süvaesituse sõne ka sõnastikus tegelikult olemas on. Reeglid on esitatud lõplike automaatidena, mida jooksutatakse paralleelselt – kui kasvõi üks automaat jõuab lubamatusse olekusse või ei jõua sisendi lõpuks lõppolekusse, tähendab see, et analüüs jõudis tupikusse ja uurida tuleb teisi variante. Analüüs väljastab leitud süvaesitused ning süvaesituse komponentidega seotud morfoloogilise info.

Kahetaseline mudel oli esimene praktiline, kuid keelest sõltumatu mudel, mis võimaldas morfoloogiliselt keeruliste keelte analüüsi.

Koskenniemi doktoritöös esitatud automaadid on saadud reeglite käsitsi kompileerimisel. Reeglite automaatseks kompileerimiseks vajaliku algoritmi töötasid Kaplan ja Koskenniemi välja 1985. aastal, kui Koskenniemi Stanfordi külastas [19]. Ilmnes, et kuigi ümberkirjutusreeglid ja kahetasemelised reeglid on formaalselt üsna erinevad, on mitmed ümberkirjutusreeglite kompileerimiseks kasutatud tehnikad kasutatavad ka kahetasemeliste reeglite juures. Esimene, InterLispis kirjutatud versioon kompilaatorist kirjutati aastatel 1985-1987, hilisem C-versioon aastatel 1991-1992 [14]. Praktiliselt kasutatava kompilaatoriga läks hoolimata teoreetilise algoritmi suhteliselt kiirest loomisest palju aega, sest see sisaldab ka keerulisi tehnikaid reeglikonfliktide leidmiseks ja lahendamiseks. Ilma sellise silumistoeta on suuri reeglikomplekte raske või võimatu ehitada – reeglikomplektide konfliktivabaks saamine on tülikas, isegi kui kompilaator täpse konfliktikoha näitab.

1.4.4 Lõplikel automaatidel põhinev mudel

Kaplani ja Kay lõplike automaatidega tegelevad algoritmid vormistati valmis tööriistadeks Xeroxi Palo Alto uurimiskeskuses. Sealsamas loodi ka kahetasemeliste reeglite kompilaatori C-versioon *twolc*. 1988. aastal näitas Kaplan, et kuigi üldiselt ei ole regulaarsete relatsioonide hulk ühisosa võtmise suhtes kinnine, on seda regulaarsed relatsioonid, mille lähte- ja sihtsõned on sama pikad ning seega ka kahetasemeliste reeglitega väljendatud regulaarsed relatsioonid ([11], avaldamiseni jõudis see tulemus alles 1994. aastal). See tähendas, et on võimalik leida Koskenniemi kahetasemeliste reeglite koguga ekvivalentne lõplik teisendaja, mis tehniliselt on üksikuid reegleid esitavate automaatide ühisosa.

1992. aastal kirjeldasid Karttunen, Kaplan ja Annie Zaenen mitmeid Koskenniemi süsteemi, eriti selle sõnastikuosa puudusi ning pakkusid välja ka lahendusi [16]. Kaks suuremat probleemi olid sõnastikuesituse suhteline meelevaldsus, tavaliselt püüti süvaesitusi sisaldav sõnastik koostada võimalikult lähedaselt pindesitusele, et minimeerida sõnastikukuju teisendavate reeglite keerukust ning asjaolu, et morfoloogiline info oli

sõnastikule lisatud annotatsioonidena. Nendest asjaoludest tulenevalt oli sõnavormide genereerimine üsna tülikas ning sõnastiku pakkimine raskendatud. Pindesitusele sarnanev kuju tähendab kummalisi lisareegleid ka ebareeglipäraselt muutuvate sõnade jaoks (*go* vs *went* jms).

Pakutud lahendus seadis eesmärgiks ühe sõna sõnastikukuju normaliseerimise ning morfoloogilisi kategooriaid tähistava info (arv, isik, kääne jms) kodeerimise sõnastiku osana. Kuna sellised eesmärgid teevad sõnastikukuju pindesitusega võrreldes suhteliselt erinevaks (ja seega nende vahel teisendavad reeglid keeruliseks), pakkusid autorid välja kahetasemeliste reeglite mitmetasemelise rakendamise. Näiteks: esimene tase teisendab erisümbolitega kodeeritud morfoloogilise info tegelikeks formatiivideks ning sellega koos sõnastikuesituse kahetasemelistele reeglitele tavalisemaks süvaesituseks, teisel tasemel töötavad "tavalised" kahetasemelised reeglid. Kuna reeglikogude ühisosad ning ka lõpliku automaadina esitatud sõnastik on komponeeritavad, siis on võimalik sõnastik ja reeglikihid komponeerida üheks suureks, kuid kiireks lõplikuks teisendajaks, mis teisendab otse mugavalt sõnastikukujult pindesituseks või vastupidi. Omapärase kõrvalefektina avastasid Xeroxi uurimiskeskuse teadlased, et kuigi teoreetiliselt võiks keeruliste reeglite ja suure sõnastiku kompositsioon anda väga suure automaadi, siis tegelikult piirab (piisavalt suur) sõnastik üldise, kõigile sümbolijadadele rakendatava reegliteteisendaja vaid tegelikult keeles esinevate sõnade ja sisenditega ning seetõttu ei ole sõnastiku ja reeglite kompositsioon tavaliselt märkimisväärselt suurem sõnastikku esitavast automaadist.

Üldistatult seisnebki tänapäevane lõplikel automaatidel põhinev morfoloogiakirjeldus keerulise regulaarse relatsiooni (sõnastikuesitused-pindesitused vastavuse – leksikaalse teisendaja) dekomponeerimises ning saadud komponentide kirjeldamises mitmete pisemate lõplike automaatidega. Sealjuures võivad komponent-automaadid olla esitatud kasutades erinevaid formalisme nagu regulaaravaldised, jätkuklassidega sõnastikusüsteemid, ümberkirjutusreeglid, kahetasemelised reeglid, Xeroxis loodud asendusreeglid *(replace rules)* [12] jms. Komponendid ühendatakse tervikuks, kasutades kompositsiooni, ühisosa, eelistusega ühendid, leebet kompositsiooni *(lenient composition)* jm elementaarsemaid või keerulisemaid lõplikel teisendajatel või lõplikel automaatidel defineeritud tehteid. Morfoloogiakirjelduste tavaline ja tähtis komponent on lõpliku automaadina või (sagedamini) lõpliku teisendajana esitatud sõnastik.

Omapärase dekomponeerimisvõttena soovitavad Karttunen ja Beesley kasutada ülegenereerimist ning filtreid [1]. See on kasulik keerulisemate konkatenatiivsete nähtuste korral, kus tavaliselt kasutatava jätkuklassidega sõnastiku võimalustest väheks jääb. Sellisel juhul kirjeldatakse sõnastikes paiknevad (vms automaadiga antud) reeglid

teadaolevalt ülegenereerivalt, kuid komponeeritakse selline automaat teisega, mis kirjeldab piiravaid reegleid. Näiteks võiks nii kirjeldada võrdlusastmete moodustamise eesti keeles. Komparatiiv saadakse lisades omadussõna ainuse omastavale liite *-m*. Jätkuklassidega on mugav kirjeldada sellise reegli kehtimist kõigi käänduvate sõnade korral, reegli kehtimist vaid omadussõnadele saab piirata eraldiseisva filtriga.

On näidatud [13], et ka uuem fonoloogiakirjeldamise paradigma, optimaalsusteooria, on realiseeritav lõplike teisendajate abil.

Lõplike teisendajate süsteemidena saab kirjeldada ka mitmeid morfoloogilise analüsaatori tööd toetavaid või sellel põhinevaid vahendeid, alustades teksti sõnadeks jagajast ja suurtähtede-väiketähtede teisendajast kuni spelleri jms lihtsamate rakendusteni välja.

Peatükk 2

Uuendatud, lõplikel automaatidel põhinev morfoloogiakirjeldus

Arvestades lõplikel automaatidel põhinevate morfoloogiakirjelduste suhtelist edukust ning kasutatavust keeletehnoloogiliste rakenduste loomisel tundub, et just kahetasemelise mudeli praktilise kasutatavuseni edasiarendamine on mõistlik viis vabavaraliseks kasutamiseks mõeldud morfoloogiakirjelduse esitamiseks. Käesoleva töö peamiseks sisuks ongi Heli Uibo tööle tehtud ja tegemist vajavate täienduste kirjeldus.

2.1 Algseis

Töö lähtekohaks on Heli Uibolt saadud lõplike automaatidega morfoloogiakirjeldus ning selle kohta avaldatud artiklid. Värskeim artikkel on pärit aastast 2006 [33]. Võrreldes magistritöös [32] kirjeldatuga on põhjalikult muutunud sõnastikuosa, kuid kahetasemelised reeglid on jäänud suures osas samaks. Oluline täiendus on ka verbituletiste sõnastikuesituse saamine rakendades pööratud astmevahelduse reegleid sõnastik-teisendaja "ülemisele poolele" (sisendile). Algseisu kirjelduse juures on oluline, et tõenäoliselt ei sisalda minuni jõudnud failid päris kõige viimast seisu, teadaolevalt on puudu eranditeloetelud jms.

2.1.1 Sõnastiku struktuur

Sõnastik on jätkuklasside hierarhia. Esimesel tasemel toimub hargnemine sõnaliikide järgi. See on vajalik liitsõnadega seotud reeglite jaoks – teatud sõnavormide juures on viidad sõnaliikidele, mis sellele vormile liituda võivad. Minuni jõudnud sõnastikus on kirjeldatud vaid nimisõnad, omadussõnad ning tegusõnad, muude sõnaliikide sõnastikke

pole, kuid need ongi enamasti muutumatud sõnad, mille lisamine pole seega keeruline.

Sõnaliigisõnastikud jagunevad eeskätt Ülle Viksi Väikeses Vormisõnastikus esitatud tüübisüsteemi alusel, kusjuures osa tüüpe on kahetasemeliste reeglitega keeruliselt kirjeldatava lõpumuutuse (ne-se) või lõpuvokaali (palga-palgi) alusel jagatud väiksemateks alltüüpideks.

Tüübisõnastikud lisavad vajalikud lõpuhäälikud (-märgid) ning vajadusel ka nõrga tüve tunnuse, pikeneva konsonandi tunnuse jms. Tüübisõnastikud viitavad tüvevariantide sõnastikele, mis (üldjuhul tüve muutmata) viitavad omakorda lõputunnuseid lisavate põhi- ja analoogiavormide sõnastikele. Kui mõnel tunnusel on mitu varianti (*puu-DE-le* vs *las-TE-le*), siis on vastav morfotaktiline valik kodeeritud lõputunnuse sõnastikku, mis valib õige tunnusega sisaldava põhivormi – see tuleb otseselt VVSi tüübisüsteemist ning tüübikirjeldustest.

Verbituletiste genereerimiseks on verbitüved esitatud viisil, mis lubab sõnastikuesitusse genereerida vajalikus astmes tüve. Tuletiste tegelikuks genereerimiseks viitavad jätkusõnastikud vajalikes kohtades sobivat tüüpi nimi- ja omadussõnadega seotud jätkusõnastikele.

Kõigile sõnastikus leiduvatele sõnadele võib lisanduda liide *-gi,* kusjuures valiku *-gi* ja *-ki* vahel teevad kahetasemelised reeglid.

Sõnastikus on 26 omadussõna, 116 nimisõna ning 93 tegusõnatüve.

2.1.2 Sõnatuletus

Tuletistest on esindatud eelkõige juba mainitud verbituletised (-nud, -tud (-dud), -nu, -tu (-du), -v, -tav (-dav), -mine, -ja). Omadussõna võrdlusastmete kohta käivad märgendid on küll sõnastikus defineeritud, kuid see versioon ei sisalda reegleid võrdlusastmete tuletamiseks.

2.1.3 Liitsõnamoodustus

Selles sõnastikus on lubatud vaid omadussõnade ning nimisõnade liitumine teatavatele käändsõnade vormidele. Sellised vormid on ainsuse nimetav, ainuse omastav ja mõned (aga mitte kõik) mitmuse omastava variandid. Sõnastikus on jälgi ka teistsuguste liitsõnareeglite kohta, kuid need ei ole täielikud (vajalikud jätkusõnastikud on puudu) ning neid ei kasutata. Liitsõnade ülegenereerimisega see kirjeldus ei tegele, pigem on probleem liiga väheste liitumiste lubamises. Vaid liitsõnades esinevaid komponente (viker- jms) sõnastikus pole.

2.1.4 Süsteemi üldine ülesehitus

Süsteem koosneb reeglifailist ning sõnastikufailist. Artikkel [33] kirjeldab komponentide omavahelist suhet. Teisendaja koosneb kolmest osast: sõnastikust, kahetasemeliste reeglitega esitatud teisendajast ning vaid astmevahelduse reegleid sisaldavast teisendajast. Viimase pööratud varianti, mis teisendab sõnad sõnastikukujust süvakujule, kasutatakse sõnastiku ees verbituletistele sobivate tüvede saamiseks. Kogu süsteem on esitatav valemiga

$$lemma + morf.info \Leftarrow reeglid_{AV}^{-1} \circ sõnastik \circ reeglid \Rightarrow pindesitus,$$

kus $reeglid_{AV}^{-1}$ on reeglite astmevahelduse (tegelikult tüveteisendustega) tegelev alamhulk tagurpidi rakendatud teisendajana ning $\circ$ on tavaline teisendajate kompositsioon.

Süsteemi moodustavad (teksti)failid kasutavad oma loomise ajal Eestis levinud ISO-8859-1 kodeeringut, ž ja š on reeglites kodeeritud mitmemärgiliste sümbolitega sh ja zh, kuid sõnastikus neid ei esine.

Failid on mõeldud kasutamiseks Xeroxi lõplike automaatide tööriistadega (*xfst*, *lexc*, *twolc*). Minuni ei jõudnud skripte, mis oleks sisaldanud täpset eeskirja leksikaalse teisendaja loomiseks.

2.2 EKI andmefailid

Tähtis grupp lähteandmeid on ka EKI morfoloogiasüsteemi andmefailid[1], millest seni olulisim oli vormieranditefail, mis loetleb 1287 erandlikku või paralleelvormi. Edaspidi on arvatavasti kasu ka liitsõnakomponentide loenditest.

Kõige olulisem EKIst pärit andmekogu on EKI tüvebaas, mis on sisuliselt loetelu kõigist EKI morfoloogiasüsteemile tuntud tüvedest koos muuttüübi ja võimalike tüvevariantidega. Tüvebaas pole avalikult kättesaadav, aga kasutada olnud variant on saadaolevaist andmefailidest suhteliselt lihtsalt genereeritav.

2.3 Täpsustatud eesmärk

Morfoloogiakirjeldust on võimalik kasutada erinevates keeletehnoloogilistes rakendustes, aga ka teadustööks. Süsteemile esitatavad nõuded sõltuvad konkreetsest kasutajast, kasutusviisist ning kasutuseesmärgist. Nii võiks näiteks tundmatute tekstide analüüsile orienteeritud süsteem püüda ära tunda ka kummalisi ja potentsiaalselt vigaseid

[1] `http://www.eki.ee/tarkvara/est_morpho_data.zip`, 01.08.2010

sõnavorme, samas kui õigekirjakontrollimiseks on vaja rangemat kirjeldust, mis vigaseid vorme pigem ei luba. Ideaalne oleks luua võimalikult universaalne keelekirjeldus, mille koosseisus olevad reeglid, komponendid ja sõnastikukirjed võiks olla märgendatud nii, et samast algkirjeldusest saaks kompileerida erinevatele rakendustele sobivaid alamhulki. Lihtsaim näide võiks olla sõnastikus paiknevate tüvede tähistamine kasutussageduse või mingi kitsa valdkonna terminoloogiasse, murdesse vms kuulumise alusel. Sellegipoolest on esimese sammu tegemise huvides valitud järgmised eesmärgid ja põhimõtted:

- Kirjelduse esimene praktiline kasutus on tõenäoliselt õigekirjakontrollija andmefailide loomine. Teine, kaugemasse tulevikku jääv rakendus, millega siiski arvestada tuleks, on grammatikakontrollija. Muuhulgas peaks õigekirjakontrollimise tulemus olema võrreldav või parem kui praegune *ispelli* sõnastikuga lahendus, mille suurimaks puuduseks on liitsõnamoodustuse kirjelduse puudumine.

- Sõnatüved, reeglid jms peaks kirjas olema vaid kord. Kui samu andmeid on vaja mitmes kohas kasutada, siis tuleks need sobival kujul eraldada nii, et neid saaks taaskasutada kas otse või eelnevalt sobivale kujule teisendades. See lihtsustab süsteemi muutmist edaspidi.

- Süsteemi arenduse käigus peaks olema võimalik mugavalt kontrollida tulemuse "headust", näiteks mõõta analüüsitud sõnavormide arvu, õigete analüüside arvu vms. Analüsaatori väljundit võiks saada võrrelda ESTMORFi või EKI analüsaatorite väljunditega.

- Süsteem peaks olema automaatselt "ehitatav". See tähendab, et peab leiduma mehhanism, mis oskab arvesse võtta muudatusi süsteemi komponentides ning ehitada uue leksikaalse teisendaja ning vajadusel viia läbi uued testid jms. Selline ehitussüsteem võimaldab süsteemi arendust võrreldes uuendamiseks vajalike käsuridade käsitsi sisestamisega oluliselt kiirendada. Sama põhimõtte laiendusena – sõnastikud, mis baseeruvad elektroonilistel andmekogudel, tuleb vajalikule kujule viia võimalikult automaatselt, et algandmete uuenemisel oleks protsess mugavalt korratav.

- Leksikaalse teisendaja saamiseks peaks kasutama võimalikult vabasid vahendeid. Xeroxi tööriistade kasutuslitsents on suhteliselt piirav ning sisuliselt tohib neid kasutada vaid akadeemilisteks tegevusteks, kõigeks muuks on vaja hankida eraldi litsents. Võimalik alternatiiv on HFST[2].

[2] `http://www.ling.helsinki.fi/kieliteknologia/tutkimus/hfst/`, 01.08.2010

2.4 Uuendamistegevused

Võrreldes algse süsteemiga on tehtud mitmeid täiendusi ja muudatusi. Üldine skeem oli sõnastiku järkjärguline täiendamine lisades sõnu muuttüüpide ja nende alamtüüpide kaupa. Pärast iga alamtüübi lisamist toimus genereerimise kontroll, mille käigus genereeriti mõne alamtüüpi esindava sõna muutevormid. Kui kõik tüübid olid lisatud, järgnes analüüsi kontroll, mille käigus analüüsimata sõnu uuriti ning analüüsi ebaõnnestumise põhjus tuvastati. Tavaliselt oli tegu kas veaga alamtüübi süvaesituse leidmise algoritmis või tüübiga seotud jätkusõnastikes. Harvem põhjustasid vigu kahetasemelised reeglid. Peamine automaatselt leitav mõõdik oli analüüsitavate sõnavormide arv testkorpuses.

2.4.1 Tehnilised täiendused

Sõnastiku- ja reeglifailid on teisendatud UTF–8 kodeeringusse. Muuhulgas võimaldab see mugavamini ja korrektsemalt (ilma mitmemärgiliste erisümboliteta) kirjeldada ž ja š käitumist. Samuti on UTF–8 kodeering üha laiemalt kasutusel keeletehnoloogilistes vahendites, mis üsna sageli peavad suutma korraga töödelda väga erinevate keelte kirjeldusi või nendes keeltes esitatud tekste ja seega on selle kodeeringu kasutamine tõenäolisemalt jätkusuutlik.

Põhisõnastik on jagatud mitmeks failiks. Selle tingis eelkõige vajadus asendada osa alamsõnastikke (eeskätt tüvedesõnastikud), kuid see võimaldab ka näiteks morfoloogiliseks märgendamiseks kasutatavate mitmemärgiliste sümbolite loendi jagamist mitme komponendi vahel. Selline jagamine suurendab veelgi automaatse ehitussüsteemi vajadust – kuna sõnastiku failiformaat ei võimalda otse teiste failide kaasamist, siis on ehitussüsteemi üheks ülesandeks tükkidest vajalike sõnastikufailide koostamine.

Hilisemates faasides lisandusid erandite sõnastik, derivatsioonifiltrid, liitsõnareeglid jms ning ehitussüsteemi täiendused lisandunud komponentide sidumiseks üheks leksikaalseks teisendajaks.

Ehitussüsteemi aluseks on UNIXi-maailmast pärit *make.* Keerulisemad lõplike automaatide kombineerimised on kirjeldatud *xfst* skriptina. Erinevat päritolu algandmetega opereerimine on mahukate tegevuste automatiseerimise ja korratavuse huvides vormistatud skriptidena keeltes *perl, shell*, *awk* ja *sed.*

2.4.2 Täiendused keelekirjeldusele

Üks tähtsamaid täiendusi on sõnastiku laiendamine. Olulised aspektid sõnastiku laiendamisel olid vajadus saavutada suur maht ning vajadus iga sõna tüvemuutused sõnastikku kodeerida. Sobiva tulemuse andis *ispelli* sõnastiku loomiseks kasutatud laiendatud tüvebaasis (lisaks EKI tüvebaasile veel hulk matemaatikasõnastikust pärit sõnu ning VVSist puuduvaid kuid uuemal ajal kasutatavaid sõnu [26]) antud tüvevariantide alusel tüvede süvakujude sünteesimine. Sünteesireeglid sõltuvad tüve muuttüübist, vahel peab arvestama (või tuvastama) ka alamtüüpi. Iga sõnastikku lisatud tüvi on varustatud ka tüvevariante arvestava jätkuleksikoni viitega. EKI tüvebaasi teeb eriti sobivaks lähtematerjaliks asjaolu, et tüved on juba (ka jätkusõnastikesüsteemile aluseks olevate!) VVSi tüüpidega tähistatud ning (põhi)tüübituvastusega pole eraldi vaja tegeleda. Esialgu jagas tüvebaasi teisendaja tüved sõnaliigiti eraldi alamsõnastikeks, jätkates algsetes failides olnud skeemi. See oli tarvilik liitsõnamoodustuse kirjeldamiseks. Edaspidi, pärast liitsõnamoodustuse põhisõnastikust eraldamist, kadus selline vajadus ning tegelikult pole sõnaliigipõhised alamsõnastikud enam vajalikud.

Sõnastiku laiendamise alla käib ka tuletiste, sh võrdlusastmete moodustamise reeglite lisamine. Esialgu on lisatud *-ke(ne)* liide nimisõnadele, *-us* liide omadussõnadele ja määrsõnu tuletav liide *-lt*. Tuletusreeglite kirjeldamine jätkuleksikonides tähendab, et reeglites pole võimalik kasutada teadmist sõna liigi kohta (kõik käändsõnad kasutavad samu jätkuleksikone, sõltumata sõnaliigist) ega ka selle kohta, kas üks või teine liide on juba rakendunud (võivad tekkida nt komparatiivi topeltrakendamised nagu **suuremam*). Sellised teadlikud ülegenereerimised tasakaalustatakse eraldi filtriga, mis lubab vaid teatud sõnaliigi ja liite kombinatsiooni või lubab mõne liite liitumist vaid kord. Sama filter teeb vajalikud teisendused nii, et liigseks muutunud sõnaliigitähistus sõnastikuesitusse ei jõuaks.

Sõnastiku massiline laiendamine tõi välja asjaolu, et algses sõnastikus polnud olemas kõiki vajalikke alamtüüpe esitavaid jätkusõnastikke, eriti tüüpidel 02, 05, 10, 22. Analoogseid, sõnastiku suurenenud mahu ja ulatusliku testimise abil välja tulnud pisemaid parandusvajadusi oli veel. Muuhulgas mõned tüübisõnastikud, mis seni esindatud sõnade osas kokku langesid ning mis olidki kirjeldatud jagatult – peaaegu iga VVSist erinevalt jagatud jätkusõnastiku jaoks leidus näide, mis jagamise võimalikkuse ümber lükkas.

Pärast regulaarsete tüübikirjelduste jõudmist suhteliselt heale tasemele hakkas mitteanalüüsitavate sõnade seas ilmnema üha rohkem erandlikke sõnu. Erandite kirjeldus oli EKI andmefailides mugaval kujul olemas, peamiselt tuli korraldada EKI morfoloogilise info tähistuse teisendamine vajalikule kujule ning valida skeem

asendavate (vs paralleelvormidena esinevate) erandite süsteemi lisamiseks. Ilmnes, et kõige mugavam oli (asendavad) erandid kirjeldada eraldi teisendajana ning kombineerida erandite teisendaja regulaarsete sõnade teisendajaga eelistusega ühendi *(priority union)* tehet. Sisuliselt ühendatakse eelistatud teisendaja teise teisendaja sellise variandiga, kust on eemaldatud esimese teisendaja lähtekeeles leiduvad sõned. Paralleelvormid kirjeldatakse põhisõnastikus alternatiivsete harudena.

Selline, sõnastikku mitme automaadi vahel jagav kirjeldus põhjustas olukorra, kus senine viis liitsõnu kirjeldada ei ole kasutatav. Jätkuklassidega saab viidata vaid ühe automaadi sees, kuid nüüd oli tarvis viidata ka teise, täiesti sõltumatu automaadi sisse. Lahenduseks oli liitsõnamoodustuse viimine tüvesõnastikust välja. Samal tasemel, kus rakendatakse eelistusega ühendit, konkateneeritakse saadud ühendit (lihtsõnu teisendavat automaati) iseendale suvaline arv kordi – sisuliselt on sellega lubatud suvaliste lihtsõnavormide omavaheline liitmine. Ülegenereeritud liitsõnade hulka piirab võimalikke liitumismustreid kirjeldav liitsõnafilter. Esialgu tundus, et liitsõnafilter on otstarbekas jagada kaheks, sõnaliikide ja grammatiliste vormide tasemel töötav üldreeglistik ning konkreetseid sõnu ning keeletava arvestav eranditeloetelu.

Testimise käigus ilmnes, et kasutatavas testkorpuses on ka numbritega väljendatud arvsõnu, mille analüüsimisega nt ESTMORF hakkama saab. Selliste arvsõnade ning neile vahetult liituda võivate käändelõppude analüüsimiseks on loodud eraldi automaat, mis on tavalise ühendi abil ühendatud lihtsõnu teisendava automaadiga. See tähendab, et kui liitsõnareeglid seda lubavad, siis võiks süsteem suuta analüüsida sõnu nagu *100-kordne* jms.

Täiendused olid tarvilikud ka kahetasemeliste reeglite osas. Sagedasemad muudatuste põhjused olid reeglites markeeritud vokaali kadumise lubamine tüvemitmuse toimimise võimaldamiseks, reeglite kontekstides esinevate kaashäälikute hulga täpsustamine, tüvemitmuse reeglite täpsustamine. Täiendusi ja parandusi said ka reeglites kasutatavad häälikuklassid ja eritähistuseks kasutatav sümbolite hulk, näiteks lisandus vahevokaalide (konsonantide vahelt kaduda võivate vokaalide) hulka *i*, mis tuleb mittekaduvatest *i*-dest eristamiseks süvakujul suurtähega märkida. Mõned reeglid osutusid ka liiaseks, nt *ks+t -> st* , mis esialgsel kujul töötas ka sõnadel *raksti, fokstrott, vakstu* (millel on kst juba tüves olemas), aga ka sõnadel nagu *maksti*, kus *kst* tekib lõpuformatiivi liitmisel.

2.4.3 Testimine

Peamine testimisvahend oli Tartu Ülikooli Arvutuslingvistika Uurimisrühma kodulehel leiduv morfoloogiliselt ühestatud korpus[3]. Esialgu piisas vaid G. Orwelli "1984" tekstist, hiljem kasutasime kogu ühestatud korpust (üle poole miljoni sõna). Analüüsi testimisel pole analüsaatori sisendis morfoloogilist infot vaja, seega on testfailist see eemaldatud. Testimise käigus ilmnes kiiresti, et praegune kirjeldus ei saa eristada suuri ja väikeseid tähti ning eeldab, et sisend on läbivalt väikeste tähtedega (mõnedel suurtähtedel on kahetasemelistes reeglites eritähendus). Selle asjaoluga arvestamiseks on testsisend teisendatud läbivalt väiketäheliseks. "1984" korpusefailist saadud testfailis on praegu 16808 erinevat sõna.

Esimene ja peamine mõõdik oli tundmatute (mitteanalüüsitavate) vormide arv. Selline vormide arv arvutatakse nii ESTMORFi (290), EKI analüsaatori (1423) kui ka arendatava leksikaalse teisendaja jaoks (703). See ei anna päris täpset ülevaadet tegelikust tekstist õigesti äratuntavate sõnade osakaalu (saagise)[4] kohta, kuid tundmatute sõnade loend on olnud mugav vahend analüsaatori puudujääkide uurimiseks.

Üsna hiljuti lisandus testimisvahendite hulka ka analüüsi korrektsuse kontroll – asjaolu, et mingi sõna analüsaatori poolt "ära tuntakse" ja sellele oletatav analüüs leitakse, ei tähenda veel, et see analüüs õige on. Kogemus *ispelli* sõnastikuga näitas väga selgelt, et lõtvade sõnamoodustusreeglite korral on võimalik leida liitsõna-analüüse nii korrektsetele kui mitteeksisteerivatele sõnavormidele. Analüüsi korrektsuse kontrolliks võrreldakse analüsaatori väljundit (tegelikult ainult morfoloogilist märgendust, leitud tüve ja korpuses märgitud tüve ignoreeritakse) kõigi korpuses esinevate võimalike analüüsidega. Probleemiks loetakse korpuses leiduva analüüsi puudumist analüsaatori väljundist.

Mõlemad nimetatud testiviisid testivad peamiselt positiivset varianti, kus analüüs peaks leiduma. Kasulik, eriti õigekirjakontrolli-rakendust silmas pidades, oleks testida ka negatiivsete sisendandmetega, sõnedega mis ei kuulu eesti keelde. Kõige kasulikum võiks olla tüüpiliste kirjavigade korpus. Paraku sellist korpust avalikult eesti keele jaoks olemas pole. Teadaolevalt koostati selline korpus ESTMORFi loomisel [8]. Selle loomiseks kasutatud lähtematerjalid pole kahjuks avalikult saadaval ning

[3] `http://www.cl.ut.ee/korpused/morfkorpus/`, 01.08.2010

[4] saagis (*recall*) ja täpsus (*precision*) on sageli kasutatavad statistilised näitajad. Morfoloogilise analüüsi kontekstis on saagis edukalt (muuhulgas õige analüüsi saanud) analüüsitud sõnade arvu suhe kõigi tekstis leiduvate sõnade arvu. Täpsus on (muuhulgas) korrektse analüüsi saanud sõnade arvu suhe kõigi analüüsi saanud sõnade arvu. Arvude interpreteerimisel tuleb tähele panna, kas mõeldakse unikaalseid sõnu (nagu meie siin praegu) või vaba teksti, kus sagedasemate sõnade õige äratundmine nii saagist kui täpsust kiirelt tõstab.

korpuse enda kättesaadavuse ja kasutatavuse uurimine on kindlasti üks suundi tulevikuks. Paralleelselt on arvatavasti võimalik käivitada uudisteportaalidest värskete ja toimetamata ning hiljem pisut toimetatud tekstide kogumine, kuid ESTMORFi loomise ajal kogutud materjalide eeliseks on teadmine, et toonased tekstide sisestajad ei saanud kindlasti kasutada eesti keele õigekirjakontrollijat. Võimaliku õigekirjavigade allikana on pakutud ka lausevigade korpust, mille vigade hulgas on ka õigekirjaveaga sõnu. See mõte vajab täpsemat uurimist ja analüüsi sealsete vigade tegeliku hulga osas.

2.5 Praegune süsteem

Käesoleva töö raames loodud vahendeid on otstarbekas kirjeldada kahes osas. Alustuseks morfoloogilist analüsaatorit ning süntesaatorit esitav leksikaalne teisendaja oma komponentidega ning seejärel selle ehitamiseks kasutatavad tööriistad.

Peaaegu kogu süsteem on avalikult kättesaadav GitHub hoidlast[5], esialgu on erandiks EKI tüvebaas, mille levitamisõigused pole selged – vajadusel saab tüvebaasi kas EKIst või töö autorilt. Täielik süsteem on olemas tööle lisatud CD-l. Süsteemi ehitamiseks ja kasutamiseks vajalikud nõuded on kirjeldatud jaotises 2.5.2.

2.5.1 Leksikaalne teisendaja

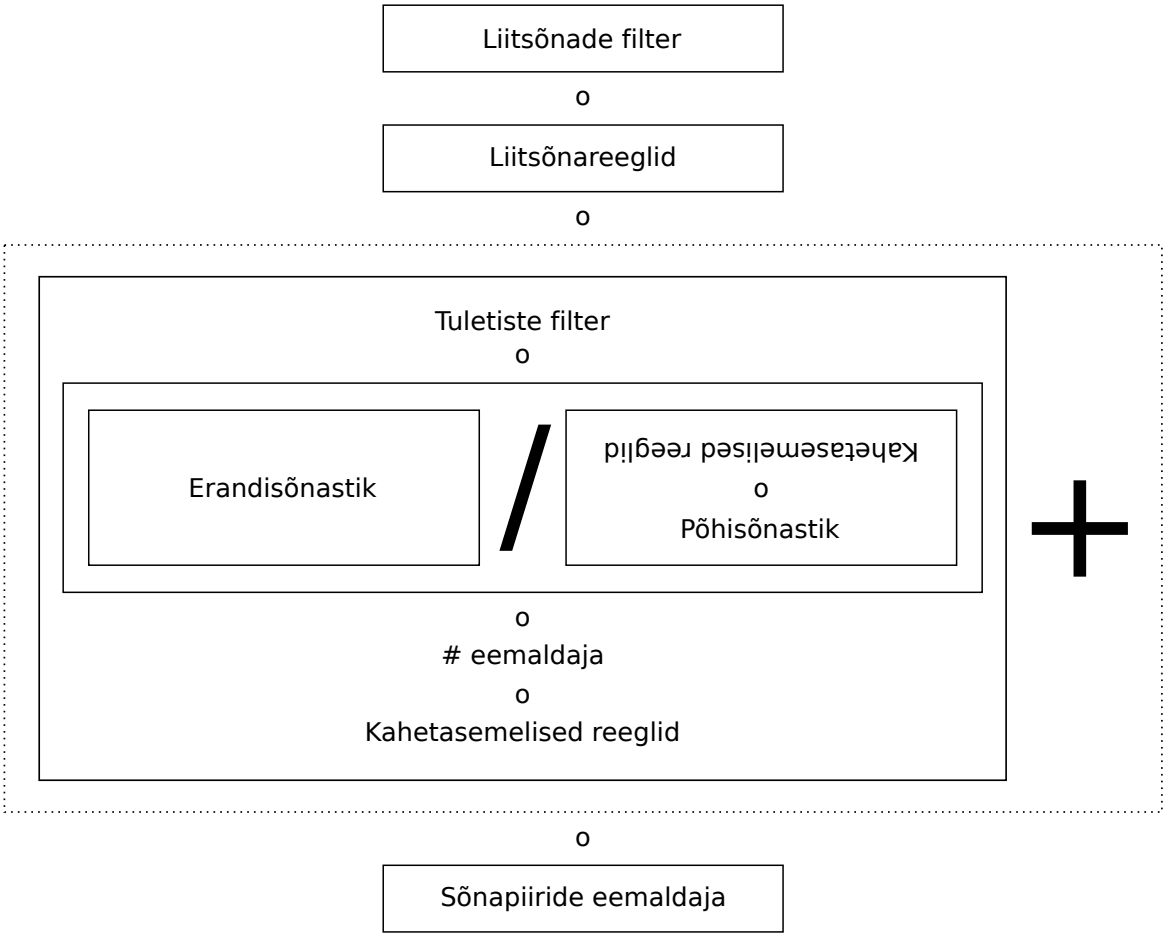
Tähtsaks uuenduseks võrreldes seniste lõplikel teisendajatel põhinevate eesti keele morfoloogiakirjeldustega, on teisendaja senisest modulaarsem ülesehitus. Sõltumatute lõplike teisendajatena kirjeldatakse reeglipäraste lihtsõnade sõnastik, vormimoodustus- ja tuletusreeglid, ebareeglipäraste lihtsõnade sõnastik, sõnavorme süvaesitusest pind- ja sõnastikuesitusse teisendav kahetasemeliste reeglite komplekt, numbritega esitatud arvsõnade sõnastik, liitsõnamoodustuse reeglid. Täiendavalt on sisse toodud kaks filtreerivat teisendajat – tuletiste filter ja liitsõnade filter. Kõik see oli ühelt poolt vajalik mitmete tarvilike omaduste (nt liitsõnade moodustamine ka erandlikest sõnavormidest) saavutamiseks, kuid peaks teisalt mõnevõrra lihtsustama morfoloogiakirjelduse edasist arendamist.

Võimalik edasiarendus oleks näiteks lihtsõnade süvaesitus-pindesitus teisenduste realiseerimine kahetasemeliste reeglite asemel mõnd muud lõplike automaatide kirjeldamise formalismi kasutades. Pähe tuleb näiteks variant kirjeldada Evelin Kuusiku ja EKI tüvemuutused (nt [35]) kahetasemeliste asendusreeglitega *(replace rules)*, sedasi saaks nii mõnedki praegu lihtsõnade sõnastikus mitme kirjega kirjeldatud erandlike

[5] `http://github.com/jjpp/plamk`, 01.08.2010. Töö kirjeldab sildiga (*tag*) *plaadiversioon* tähistatud seisu.

tüvevormidega sõnad asendada ühe tüveerandiga ning kokkuvõttes võiks saada väiksema eranditehulgaga keelekirjelduse. Oluline on seejuures, et lihtsõnamoodustuse osas saaks asju ümber korraldada liitsõnamoodustuse reegleid kuidagigi muutmata.

Joonisel 2.1 on esitatud praeguse teisendaja komponentide skeem. Rõngas (**o**)



Joonis 2.1: Leksikaalse teisendaja struktuur

tähistab kompositsiooni, kaldkriips (/) eelistusega ühendit (sõnastikuesitustele, millel on vaste nii erandite kui põhisõnastikus, võetakse vaste eranditesõnastikust), pluss (+) on Kleene'i pluss ja tähendab sellel joonisel, et lihtsõnade teisendaja võidakse iseendaga konkateneerida. Tagurpidi kirjutatud kahetasemelised reeglid põhisõnastiku kohal tähistavad alumiste kahetasemeliste reeglitega täpselt sama reeglikomplekti rakendamist tavalisega võrreldes tagurpidi (vastava teisendaja ülemine ja alumine keel on vahetatud). # eemaldaja ja sõnapiiride eemaldaja on lihtsad asendusreeglid, mis kõigist ülemise keele sõnedest eemaldavad vastavalt # ja & ning seetõttu neist allpool juttu ei tule, teised komponendid on kirjeldatud "seest välja" järjekorras.

Lihtsõnade sõnastik

Lihtsõnade sõnastik on morfoloogiakirjelduse üks mahukamaid osi. Seesmiselt jaguneb see ühelt poolt käsitsi kirjeldatud jätkusõnastikeks ja erandi- ja paralleelvormide-sõnastikuks ning teiselt poolt EKI tüvebaasi alusel genereeritud tüvedesõnastikuks.

Põhiosas koosneb lihtsõnade sõnastiku käsitsikirjutatud osa Heli Uibolt saadud sõnastikufailist pärit jätkusõnastikest, mida on sõnastiku kasvades ja süsteemi testides täiendatud ja parandatud. Lihtsõnade sõnastikus on kirjeldatud ka tuletusreeglite produktiivne, "lubav pool". Sõnastik kirjeldab tuletised liidetega *-us, -lt, -ke(ne), -ini* (liitub komparatiivile, annab määrsõna), *-m, -im* (so võrdlusastmed), *-(e)v, -nu(d), -tu(d), -tav, -du(d), -dav, -ja* (verbist tegijanimeks), *-mine*. Neist viimased 7 olid osaliselt ka varem olemas. Varem olemas olnud verbi kesksõnadele on pärast analüüsiväljundite kontrolli lisandunud paralleelvormidena nii substantiivi kui adjektiivivariant, kusjuures *-nu* ja *-tu*-lõppudega sõnad muutuvad nagu tüübi 01 sõnad ning sõnad lõppudega *-nud* ja *-tud* on muutumatud omadussõnad.

Tüvedesõnastik genereeritakse suures osas automaatselt EKI tüvebaasist (tüvebaasi formaadis andmefailidest). Lisaks on mõnede ebareeglipäraste tüvevormidega või ka erandlike paralleelvormidega sõnade jaoks eraldi, käsitsi hallatav sõnastik ning eranditetabelist automaatselt genereeritav paralleelvormide sõnastik. Üldjuhul genereerib kompilaator iga tüve kohta ühe kirje, mis sisaldab tüve süvaesitust, sõna muuttüüpi ja alltüüpi (jätkusõnastikku) ning käändsõnade korral ka sõnaliiki (verbidel järeldub see muuttüübist). Mitmesse sõnaliiki või mitmesse muuttüüpi kuuluvatele sõnadele tekib iga tüübi ja liigi kombinatsiooni kohta eraldi kirje. Käändsõnadel on sõnaliiki sisaldav ülemine esitus ja ilma selleta alumine esitus alati erinevad, verbidel samad.

Praegu on sõna muuttüüp kodeeritud vaid jätkusõnastiku nimesse. Selle probleemiks on, et see informatsioon on nähtav vaid sõnastikufaili kompilaatorile ning seda ei kanta kuidagi edasi tekkivasse lõplikku automaati. See tähendab, et analüüs ei ütle millisesse muuttüüpi sõna kuulus ning ka sünteesil ei ole võimalik määrata, millise tüübi järgi sõna genereerimist me soovime.

Kahjuks tuleb tunnistada, et tuletistele tüve süvaesitusest mõistlikus vormis sõnastikuesituse genereerimine, põhjus miks Uibo sõnastikes sõnastiku kohal olevat teisenduskihti üldse vaja läks, ei tööta praeguse sõnastikuga nii nagu peaks. Käändsõnade juures on probleemiks asjaolu, et sõnaliigi marker tuleb lisada juba tüvedeleksikonis ning see teeb edasised teisendused keerulisemaks (markerid segaksid näiteks kahetasemeliste reeglite tööd). Võimalik lahendus oleks loobuda praeguse sõnastiku käändsõnade "suunatusest". See tähendaks, et liigseid markereid eemaldav

või teisaldav, praeguse tuletiste filtriga analoogne filter tuleks lisada mõlemale poole sõnastikku, kahetasemeliste reeglite ja lihtsõnade sõnastiku vahele. Sõnastik oleks lõpliku teisendaja asemel tavaline, üht keelt kirjeldav lõplik automaat ning selle poolt ära tuntava keele sõnades peaks sisalduma vajalik informatsioon nii sõnastikuesituse kui pindesituste genereerimiseks. Arvestades asjaolu, et eesti keeles võib ühel tüvel olla kuni viis varianti [35], võibki selline lahendus otstarbekaks osutuda.

Praeguse lahenduse korral kirjeldatakse erinevad tüvevariandid pigem jätkusõnastike ja neis sõnastikutüvele lisatavate tähtede ja markerite kaudu. Verbituletiste juures kohati esineva tuletusaluseks oleva tüve genereerimise probleemid saabki tõenäoliselt lahendada jätkusõnastikke veelgi täpsemaks tehes ning jätkude süsteemi veidi keerulisemaks tehes. Käändsõnade jaoks tähendaks see halvemal juhul nimisõnadele ja omadussõnadele paralleelsete, sõnaliigipõhiste jätkusõnastike tegemist, mis oleks vastuolus eesmärgiga kirjeldusi võimalikult lihtsatena hoida.

Radikaalne, aga võibolla vähem huvitav lahendus võiks olla ka, et tüvede muutumist ei kirjeldatagi lõpliku automaadina, lõplike automaatidena oleks (lihtsõnade tasemel) kirjeldatud vaid lihtsamad konkatenatiivsed nähtused. Andmeid tüvede omaduste, muutumise jms kohta hoitaks mingis teises esituses (nagu see ju tegelikult ka praegu on) ning sellest esitusest genereeritaks otse vajalikud tüvevormid. Sellise lahenduse puuduseks on arvatavasti produktiivsete tuletusmallide kirjeldamise tülikus (või mahukus).

Probleeme leidub ka sõnastiku sisu poolel. VVSi sõnastiku kasutatavust tänapäevaste tekstide analüüsimisel kirjeldab muuhulgas Heiki-Jaan Kaalepi doktoritöö [9]. ESTMORFi jaoks on sõnastikku oluliselt parendatud. Ka käesoleva töö raames tehtud katsetused ja testid näitavad, et tegelikult on tarvis ESTMORFi alussõnastikuga tehtule analoogset sõnastikukorrastustööd, so harva kasutatavate sõnade märgendamist ning uute, sõnastikust puuduvate sõnade lisamist. Erinevalt ESTMORFist ei pea harvakasutatavaid sõnu sõnastikust kustutama, need võib hoopis märgendada mittekasutamise põhjusega (vananenud, murre, erialakeel vms) ning esialgu teisendaja kompileerimisel lisafiltritega eemaldada. Hiljem võimaldab see luua näiteks spetsiifilistele allkeeltele sobivaid teisendajaid või kasutada seda informatsiooni näiteks kaaludega lõplike teisendajatega[6] analüsaatori loomisel.

Lisaks kasutussagedust ja -kohti tähistava märgenduse lisamisele on potentsiaalselt kasulik ka semantilise märgenduse lisamine. Sõltuvalt semantilise informatsiooni täpsest sisust võiks sellest abi olla nii kõrgema taseme keeletehnoloogilistele moodulitele

[6] *Weighted finite state transducer*, selline lõplik teisendaja, kus erinevatele olekusiiretele vastavad lisaks sisend- ja väljundsümbolitele ka kaalud. Sel viisil saab teisendajasse kodeerida statistilist informatsiooni keele kohta.

(analüüs palk+S+sg+gen ütleb vähem kui teadmine, kas juttu on palgast või palgist) kui ka analüsaatori enda sees, liitsõnamoodustuse kirjeldamisel. Väikese osa sellisest märgendusvajadusest kaotaks sõnade (all)muuttüüpide esitamine märgenduse osana (palgi – 02_I, palga – 02_A).

Veel üks mõeldav lisamärgenduse suund on täiendavad morfofonoloogilised teadmised tüve kohta nagu rõhk, välde, palatalisatsioon jms. Sellised teadmised võiksid lihtsustada tüvemuutuse reeglite kirjeldamist ning võivad olla kasulikud kõnesünteesi ning -tuvastuse jaoks. Kirjaliku keelega tegelevate süsteemide jaoks saab vastava märgenduse sobival tasemel filtritega lihtsalt eemaldada.

Tõenäoliselt on jätkuvalt mõistlik esitada tüvede sõnastik või pigem tüvede ja nende omaduste andmebaas mingil välisel kujul ning sellest vajadusel lõpliku automaadina esitatamiseks sobilik kuju kompileerida. Kas tüvede andmebaasiks sobib ka edaspidi juba olemasolev EKI tüvebaas või on vaja midagi muud, selgub arvatavasti ülalkirjeldatud sõnastiku-probleemide lahendamise ja võimalike märgendamislaienduste otstarbekuse uurimise käigus.

Eraldi tähelepanu vajab asjaolu, et praegune tüvede esitus on efektiivselt tõstutundetu. Kuna suurtähtedel on tüvedes eritähendus, siis tohib tegelike sõnade esitamiseks kasutada vaid väiketähti ja nii ei saa sõnastikus eristada näiteks suure ja väikese algustähe võrra erinevate sõnade liiki.

Erandisõnastik

Erandisõnastik peab põhisõnastikust eraldi olema peamiselt tehnoloogilistel põhjustel. Nimelt ei pea sellise lahenduse korral tegelema eraldi erandlikele vormidele vastavate regulaarsete vormide põhisõnastikus mittegenereerimise probleemiga. Erandisõnastik ja põhisõnastik kombineeritakse, kasutades prioriteediga ühendi *(priority union)* tehet. Sisuliselt on tegu kahe teisendaja ühendiga, kusjuures teise sisendist on välja filtreeritud kõik esimese sisendis esineda võivad sõned.

Erandisõnastiku põhisisu tuleb EKI morfoloogiasüsteemi vormierandite failist, mida on minimaalselt täiendatud asesõnade *mis* ja *kes* mitmuse käänete variantide täiendamise ja sõna *olema* mõnede negatiivi-variantide erandlikkuse määra osas. Lisaks on olemas käsitsi hallatav eranditeloetelu, mis praegu sisaldab peamiselt mõnede omadussõna-tüvede liitsõnades esinevaid lühivorme.

Iga EKI vormierandite faili kirje sisaldab sõna muuttüüpi, algvormi, muutevormi koodi, sõna kuju selles muutevormis ning teadmist selle kohta, kas tegemist on erandliku paralleelvormiga või esinebki kirjeldatav sõna antud vormis vaid antud kujul. Sellistest kirjetest genereeritakse kaks sõnastikku, "päris" erandite ja paralleelvormide oma.

Viimane on tegelikult lihtsõnade sõnastiku osa. Oluline nüanss on, et muuttüüpi pole eranditesõnastikus võimalik esitada ning põhimõtteliselt võib eranditefail varjutada homograafse tüvega aga teisest muuttüübist sõnu. Nii juhtus näiteks sõna *kohus* (*kohus-kohtu*, tüüp 05 ja *kohus-kohuse* tüüp 09) ainsuse osastavaga, mille ainsaks lubatavaks vormiks (tüübis 05) on eranditefailis *kohut*. Lahendus oli teise tüübi jaoks vajalik vorm samuti erandina kirjeldada.

Mõnel juhul on erandi sisu mingi vormi puudumine. Sellistele vormidele vastab süvaesituses sümbol #, mis eesti keeles üheski päris sõnas esineda ei saa ja mida on selles mõttes ohutu kasutada. Põhisõnastiku ja erandisõnastiku ühendist filtreeritakse (keelatakse) kõik sõned, kus sümbol # esineb.

Erandisõnastiku praegu teadaolevaks suurimaks probleemiks on tuletuses osalevate erandite käitumine. Reeglipõhiselt on mineviku kesksõna ühtlasi ka tuletatud nimisõna- ja omadussõna tüveks. Selline tuletusreegel on kirjeldatud põhisõnastikus. Tehnoloogiliselt asendab erandisõnastik konkreetse vormi ning tavaliselt on eranditeloetelus toodud ka teised, sõltuvad vormid sama sõnaliigi piires, kuid tuletisi erandid ei muuda. Nii on näiteks tüvest *näge(ma)* tuletatud vorm *nähtud* ("reeglipäraselt" **näetud*), mis on erandite failis kirjas, kuid mis ei mõjuta vigase omadussõna **ettenäetud* tekkimist. Analoogselt on vigane ka selle tüve tuletiste sõnastikukuju. Ka siin võiks lahendus olla praeguste kahetasemeliste reeglite ja jätkusõnastiku süsteemi asendamine mingi eraldiseisva, paindlikumalt erandeid lubava tüvevariantide genereerimise mehhanismiga.

Arvsõnade sõnastik

Arvsõnade sõnastik on inspireeritud asjaolust, et analüsaator ESTMORF suudab numbritega väljendatud sõnedele mingil tasemel analüüsi anda ning teadmisest, et numbritega esitatud arvsõnade korrektne analüüs on ka süntaksianalüsaatorile oluline. Sõnadega kirjutatud arvsõnad sisalduvad lihtsõnade sõnastikus, selle sõnastiku osaks on just numbritega kirjeldatud arvsõnade kirjeldamine.

Sõnastik defineerib regulaaravaldised põhiarvude ja järgarvude kirjeldamiseks ning hulga võimalikke käändelõppe. Aluseks on võetud Eesti Keele Käsiraamatus kirjeldatud arvsõnade kirjutamise ortograafiareeglid [3]. Võrdlusest ESTMORFiga lisandus luba kasutada arvujadas punkti, miinust ja koma ning vahetult põhiarvu järele kirjutatult protsendimärki ja ühte või kahte apostroofi (vastavalt minuti ja sekundi tähenduses). Järgarvu järel peab olema punkt või liide *-nda* mingis käändes. Tähtedega kirjutatud käändelõppudele võib liituda ka *-gi* liide.

Sõnastikus ei ole ajapuudusel veel defineeritud rooma numbreid.

Arvsõnade sõnastiku kaudu defineeritud sõned osalevad liitsõnamoodustuses ana-

loogselt lihtsõnade sõnastiku sõnadega.

Kahetasemelised reeglid

Kahetasemeliste reeglite ülesanne on kirjeldatavas süsteemis sama, mis Heli Uibo teisendajateski. Eeskätt moodustavad kahetasemelised reeglid tüve süvakujust ja jätkusõnastike abil lisatud kontekstist lähtuvalt erinevaid tüvevariante. Samuti valivad kahetasemelised reeglid tüvevokaali, tegelevad vokaalimuutuste ja konsonantühenditega. Võrreldes algse reeglitefailiga on parandused peamiselt kosmeetilised, reeglite rakendumiskontekste täpsustavad. Üks reegel (oo+i, öö+i -> õi) on jagatud kaheks eraldi reegliks (mugavamaks kontekstitäpsustamiseks) ning lisandunud on reegel, mis liite *-im* eest vokaali kustutab.

Teadaolevalt on reeglitefailis hetkel ka mitmesuguseid kontekstikonflikte, mille ilmutatud lahendamisega oleks mõistlik tegeleda, kuid mille automaatsed lahendused või ka ignoreerimine seni reeglikomplekti toimivust takistanud pole. Kontekstitäpsustamise käigus lisandunud konfliktid, mis tööd siiski segasid, on praeguseks lahendatud.

Filtrid

Filtreid kasutab praegune süsteem mitmes etapis. Keerulisemad on tuletistefilter, liitsõnareeglid ja liitsõnafilter. Tuletistefiltri sisu on piirata peamiselt käändsõnatuletiste moodustusvõimalusi ning puhastada sõnastikuesitusi liiasest sõnaliigi-infost.

Tüve juurde kirjutatud sõnaliigi infot ei saa tavaliselt mitu jätkuleksikoni kihti hiljem mugavalt kasutada – selleks peaks looma iga sõnaliigi jaoks eraldi, paralleelse jätkuleksikonide süsteemi. Seega on leksikonide süsteemis lubatud kõiksugused käändsõnade tuletused, sõltumata tuletusaluse sõnaliigist. Nii tuletusaluse kui tuletise sõnaliigid on esialgu tuletise sõnastikuesituses kirjas ning just seda asjaolu kasutabki ära tuletistefilter. Tuletised tähistatakse sõnastikuesituses vastava erisümboliga (näiteks *+_lt, +_ke, +_kene* ja *+_us*) ning tavaliselt moodustub sõnastikuesitusse kolmesümboliline alamsõne, mis sisaldab aluse sõnaliiki, tuletise tähist ja tuletise sõnaliiki. Filter laseb läbi vaid sellised sõned, kus alus ja tuletis sobivad ning keelab kõik muud tuletise tähise esinemised.

Praeguses lahenduses tegeleb tuletistefilter ka liigsete, tuletuse käigus irrelevantseks muutunud märgendite eemaldamisega – ühtlasi üritatakse tuletis ka sõnastikuesituses tüvele liita, kuid kuna seal on pigem tuletusaluse sõnastikutüvi ja mitte (tavaliselt erinev) tuletusaluseks olnud tüvi, siis annab see hetkel kummalisi analüüsitulemusi.

Kõik sellised tüvele liidetud tuletised sisaldavad võrdusmärki ja kummaliste vormide päritolu peaks sedasi tuvastav olema.

Osaliselt tegelevad tuletiste filtreerimisega ka liitsõnareeglid, mis on ka sisuliselt filter. Liitsõnareeglid on kirjeldatud muuhulgas lubatud sõnaliike kirjeldavate regulaaravaldistega, kus saaks olla (ja algselt olidki) kirjas kõikvõimalikud mingi sõnaliigi esindajate moodustumise võimalused. Kuna valetuletiste võimalusi kirjas polnud, siis ei saanud need ka liitsõnamoodustuses ega liitsõna erijuhu – ühekomponendilise liitsõna – sees esineda. Muud liitsõnareeglite ülesanded on kirjeldatud järgmises jaotises.

Liitsõnafilter on mõeldud täiendama liitsõnareegleid spetsiifiliste sõnade ja reeglierandite osas. Esialgu on see peamiselt kasulik paljusid valeliiteid andvate sõnade liitumise keelamiseks.

Liitsõnareeglid

Süsteemi kõige keerulisema filterkihi tekkimise põhjuseks oli lihtsõna-erandite kirjeldamiseks kasutatud tehnika valik. Lihtsõnade jagamine mitme sõnastiku vahel tähendas muuhulgas näiteks, et jätkusõnastike süsteemis kirjeldatud liitumised ei saanud viidata erandlikele vormidele ning eranditesõnastikus esinevad sõnad ei saanud ise ühelegi sõnaliigisõnastikule viidata.

Lahendus on lubada kõigi lihtsõnade liitumisi, kusjuures "kõik lihtsõnad" on defineeritud üheks suureks teisendajaks liidetud erandi-, põhi- ja arvudesõnastikuga, millele on juba vajalikul viisil rakendatud kahetasemelisi reegleid ja tuletistefiltreid. Selline suur teisendaja on vaadeldav elementaarse regulaaravaldisena ning sellisena on suvaliste sõnakombinatsioonide keele (relatsiooni) kirjeldamine suhteliselt mugav.

Kõigi kombinatsioonide relatsioonis eraldab lihtsõnu kas tavaline ampersand (&) või kahe ampersandi vahel olev sidekriips (&-&), mis vastab ka pindesituses sidekriipsule. Lisaks on lubatud (aga mitte kohustuslik) muust sõnast ampersandiga eraldatud sidekriipsu liitumine kogusõna ette või taha, see vastab liitsõnalühenditele nagu *nii- ja naasugune, tööandja ja -võtja* jms.

Tegeliku pindesituse saamiseks eemaldatakse selle relatsiooni alumise keele sõnadest ampersand.

Liitsõnareeglite kihi ülesanne on kasutades relatsiooni ülemises keeles olevat informatsiooni – sõnade sõnastikuvorme ja morfoloogilist märgendust – piirata võimalike moodustuvate sõnade hulka. Siit on näha ka, miks liitsõnareeglite tööks on oluline vaid stabiilne sõnastikuesitus. Mil viisil sõnastiku-esitusest pindesitus saadakse või kuidas sõnastikuesituse aluseks olevad tüved vastavates teisendajates tegelikult kodeeritud on, pole oluline.

Tehniliselt on liitsõnareeglid esitatud regulaaravaldistest koosneva jätkusõnastike süsteemina Xeroxi *lexc* formaadis. Selle eeliseks on eelkõige mugav viis komponent-regulaaravaldisi defineerida ning võimalus Uibo süsteemile sarnaselt sõnaliigipõhiste jätkuklasside kaudu liitsõnamustreid kirjeldada. Formaat on ka jätkusuutlik, ka HFST jaoks on *lexc* keele kompilaator [21]. Sõnastikufaili alguses on kirjeldatud hulk pisikesi regulaaravaldisi, mille kaudu hiljem saab mugavalt sõnaliigipõhiseid liitumismalle kirjeldada. Sõnastikuosa on jagatud kõigepealt käänduvateks, pöörduvateks ja muutumatuteks sõnadeks ning käänduvad ja muutumatud sõnad veel omakorda alaliikideks. Kuigi *lexc* võimaldab kirjeldada ka lõplikke teisendajaid, kasutavad liitsõnafiltrid vaid tavalisi regulaaravaldisi. Iga sõnaliigi sõnastik sisaldab kohustuslikus korras ka vastava sõnaliigi lihtsõnu lubavat reeglit.

Liitsõnareeglite sisu peamine inspiratsioon oli Eesti Keele Käsiraamatu sõnamoodustuse peatükk [3], kus üsna lihtsal ja arusaadaval moel on esitatud üldise taseme liitumismallid ja võimalikud mallide kasutuskohad ja tähendused. Põhilised reeglid lühidalt:

- Nimisõna ja omadussõna ainsuse nimetavale ning määrsõnale võib järgneda tegusõna.
- Nimisõna ainsuse nimetavale, mõlemale omastavale ja lühendatud tüvele võib järgneda nimisõna, omadussõna, määrsõna või sidekriips ja sõna lõpp.
- Omadussõna ainuse nimetavale või lühendatud tüvele võib järgneda nimisõna, omadussõna, määrsõna või sidekriips ja sõna lõpp.
- Määrsõnale võib järgneda nimisõna, omadussõna või määrsõna.

Lisaks on veel arvuliselt suur, kuid sisult väike hulk reegleid arvsõnade omavahelise ja nimisõnadega liitumise kohta, mis kirjeldavad väikest hulka liitsõnu, kus ka esikomponent käändub ning veel paar väiksemat reeglit. Reeglid on jätkuvalt liiga lubavad ja ilma piiravate reegliteta võiks need lubada näiteks määrsõna *ei* ja nimisõna *la* liitumist, mis õigekirjakontrollimise seisukohast on müra ja mida pigem lubama ei peaks. Samuti on praeguste reeglite järgi keelatud nimisõna + verbituletis (nimisõna) liited, kus täiendkomponent ei ole nimetavas või omastavas käändes (näiteks: *südantpuistav*). Puudu on ka reeglid asesõnade ja arvsõnade liitumise kohta.

Tõenäoliselt saaks natuke parema tulemuse, analüüsides tegelikke tekstides esinevaid liitumismalle, kirjutades sageliesinevad reeglid välja ning harvemate kohta lubaks erandlikke esinemisi vaid ükshaaval, nagu seda on tehtud ESTMORFis [9].

Tulemuse oluliseks parandamiseks tuleks vähekasutatavad kuid lühikesed sõnad tüvebaasis kas märgistada või eemaldada. Samuti võib tulemust märgatavalt parandada verbituletiste sõnastikuesituses markeerimine ja võimaluse korral ka rektsiooni-infoga varustamine või muul moel võimalike ühendtegusõnade tuletistega arvestamine. Praegu pole liitsõnareeglites võimalik verbituletisi kuidagi eristada.

2.5.2 Ehitussüsteem

Ehitussüsteemi ülesanne on, kasutades mitmeid originaalseid ja teiste loodud komponente, ehitada leksikaalne teisendaja valmis. Lisaks on ehitussüsteemi kaudu mugav automatiseerida testimist jms arenduse käigus ette tulevaid rutiinseid tegevusi. Kirjeldatava süsteemi ehitamiseks kasutatakse UNIXites levinud tööriista *make.* Ehitussüsteemi südameks on konfiguratsioonifail *(Makefile)*, mis kirjeldab võimalikud ehitussihid ja nende saavutamiseks vajalikud tegevused.

Nõuded süsteemile

Süsteemi arendamine toimus Linux-keskkonnas, kasutades Debian GNU/Linuxi arendusharu. Sellest tulenevalt on süsteemi ehitamiseks vajalikud mitmesugused UNIXilistes tavalised tekstitöötluse programmid, kuid mitte ainult. Konkreetne loetelu on:

- tavalised UNIXi-vahendid: *cat, sed, (rm, awk, grep, bash, wc, sort, uniq)*
- *perl*, vähemalt versioon 5.10
- *iconv* – EKI andmefailide ja analüsaatorite Windowsi-versioonide väljundite teisendamiseks.
- Xeroxi lõplike automaatide tööriistad: *xfst, twolc, (lookup)* vajalikud on UTF–8 toega versioonid.
- (realõputeisendajad: *fromdos, todos*)
- (analüsaator ESTMORF)
- (EKI analüsaator)
- (*wine* – analüsaatorite Windowsi-versioonide jooksutamiseks)
- (*wget* – korpusefailide automaatseks allalaadimiseks)

Sulgudes olevad programmid on vajalikud vaid testimiseks, leksikaalse teisendaja saab ehitada ka ilma nendeta.

Spetsiifilisi nõudeid arvuti mälule ja protsessorile pole, põhilisel arendamiseks kasutatud süsteemil on 2GHz Core2Duo protsessor ja 3GB mälu. Sellise süsteemiga võtab teisendaja ehitamine aega pisut alla 10s.

Selle töö käigus on kasutatud ESTMORFi versiooni, mis on saadaval EKI FTP-serveris[7]. EKI analüsaatori ja andmefailid leiab EKI veebist[8]. Mõlemad viidatud aadressid sisaldavad programmide Windowsi-versioone. Nende jooksutamiseks Linuxi-keskkonnas sobis emulaator *wine*.

Samuti on praeguses faasis vajalikud *Xerox*i tööriistad, mille sobiliku versiooni mittekommertskasutuseks mõeldud litsentsi alusel saab nende kasutamisõpiku [1] kodulehelt[9].

Vajalikud lähteandmed

Leksikaalse teisendaja ehitamiseks on vaja mitmesuguseid lähteandmeid, mis on loodud kas spetsiaalselt morfoloogiakirjelduse jaoks või on loodud mujal ning mida kasutatakse otse või kohandatud kujul. Vajalikud andmed asuvad failides:

- lex_main.txt: lihtsõnade sõnastiku reegleid ja käsitsi hallatavaid erandeid sisaldav osa, baseerub Heli Uibo sõnastikul.
- tyvebaas.txt: EKI tüvebaas, saadud kasutamiseks eesti spelleri loomisel, pärit aastast 2003. Levitamistingimused esialgu segased.
- tyvebaas-lisa.txt: ise koostatud täiendused tüvebaasile, suurem osa pärit aastast 2003.
- lex_multichar.txt: sõnastikes kasutatavad mitmemärgilised sümbolid, sh morfoloogiliseks märgenduseks ja sisemisteks vajadusteks kasutatavad. Kasutamiseks sõnastikufailide alguses.
- lex_gi.txt: vaid jätkuleksikoni GI sisaldav sõnastikufail, kombineerimiseks teiste failidega.
- deriv_filter.txt: tuletistefilter *xfst* regulaaravaldisena.
- lex_override.txt: erandisõnastiku käsitsihallatavad erandid.

[7] `ftp://ftp.eki.ee/pub/keeletehnoloogia/estmorf/`, 01.08.2010
[8] `http://www.eki.ee/tarkvara/analyys/`, 01.08.2010
[9] `http://www.fsmbook.com/`, 01.08.2010

- rul.txt: kahetasemelised reeglid, *twolc* formaadis, suures osas Heli Uibo fail.
- fcodes.ini: modifitseeritud EKI andmefail[10], võimalike vormikoodide tabel. Lisatud on meie kasutatavad märgendid, kus see võimalik oli.
- form.exc: modifitseeritud EKI andmefail, vormierandid. Lisatud mõnede asesõnade vorme ja muudetud sõna *ole(ma)* negatiivseid vorme.
- arvud.txt: arvsõnade sõnastiku põhiosa.
- liitsona_def.txt: liitsõnareeglites ja liitsõnafiltris kasutatavad definitsioonid *lexc* sõnastiku definitsioonideosa formaadis.
- liitsona.txt: liitsõnareeglid
- liitsona_filter.txt: liitsõnafilter

Lisaks on olemas hulk testimisega seotud faile:

- 1984.txt: lausemärgenduseta ja kirjavahemärkideta versioon G. Orwelli "1984" morfoloogiliselt ühestatud variandist[11].
- 1984_words.txt: eelmise variant, kust märgendus on eemaldatud.
- 1984_words_u.txt: eelmise variant, kus kõik sõnad on teisendatud väiketähelisteks ning alles on vaid unikaalsed sõnad. Ühtlasi on fail sorteeritud. See on vaikimisi testsisend.
- morftrtabel.txt: modifitseeritud konfiguratsioonifail ESTMORFi morfoloogilise analüüsi väljundi võrdlemiseks sobivale kujule teisendamiseks.

Süsteemi osaks on ka hulk skripte, mis koos otstarvetega on loetletud järgnevalt:

- eki2lex.pl: tüvebaasi teisendaja, täpsemalt kirjeldatud jaotises 2.5.4
- exc2lex.pl: eranditefaili teisendaja, täpsemalt kirjeldatud jaotises 2.5.4
- eki_out2out.pl: EKI analüsaatori väljundi "normaliseerija". Eemaldab oletuslikud tulemused, et paremini "ebaõnnestunud analüüse" kokku lugeda.
- gen-nouns.sh: käändsõna vormide generaator, täpsemalt kirjeldatud jaotises 2.6.1

[10] Allikas: `http://www.eki.ee/tarkvara/est_morpho_data.zip`, 01.08.2010

[11] Algallikas: `http://test.cl.ut.ee/korpused/morfkorpus/myh01/1984.kym`, 01.08.2010

- gen-verb.sh: verbivormide generaator, täpsemalt kirjeldatud jaotises 2.6.1
- check_analyze.pl: analüsaatori väljundi kontrollija, võrdleb korpuse põhjal koostatud kontrollfailiga ning arvutab mitmesugust statistikat.
- tolkija.pl: teisendab ESTMORFi väljundis kasutatavaid morfoloogilisi märgendeid vastavalt etteantud reeglitele. See ja vastav teisendusfail pärinevad süntaksianalüsaatori EstCG teisendusskriptide[12] seast.
- korpus/to_lc_words.sh: teisendab TÜ Arvutuslingvistika Uurimisrühma morfoloogiliselt ühestatud korpuse faile analüsaatori sisendiks sobivale kujule (eemaldab märgenduse, vaikimisi teeb väiketäheliseks ja unikaalseks).
- korpus/to_lc_testfile.sh: teisendab morfkorpuse faile check_analyze.pl kontrollfailideks.
- Makefile: reeglid teisendaja ehitamiseks, testiskriptide väljakutsumiseks jms. Siin on kirjeldatud ka Xeroxi programmide, ESTMORFi ja EKI analüsaatorite asukohad.
- korpus/Makefile: reeglid korpusefailide hankimiseks ja teisendamiseks.
- xfst.script: reeglid mitmete pisemate teisendajate kompileerimiseks ning suureks leksikaalseks teisendajaks komponeerimiseks.

2.5.3 Ehitussüsteemi kasutamine

Praeguses variandis on mõistlik tekitada süsteemist enda kohalikule kettale koopia. Githubi kasutamisel juhtub see automaatselt, CD kasutamisel tuleb seda eraldi teha. Kopeerimine on põhjendatud vajadusega täpsustada vajalike tööriistade asukohti. Kasulik on kopeerida kogu failipuu või pakkida lahti vastav arhiivifail. Parameetrid on kirjas peamise *Makefile*'i alguses, see tuleb avada tekstiredaktoriga ning teha vajalikud muudatused. Kohaliku keskkonna eripärasid kirjeldavad muutujad on:

- **XEROX:** *Xerox*'i tööriistu sisaldav kataloog. Kasulik sätestada, kui *xfst* ja *twolc* ei asu vaikimisi otsinguteel.
- **XFST:** *Xerox*'i lõplike automaatide tööriist – käsk, vajadusel täispika teega, mis *xfst* käima paneb.

[12] `http://math.ut.ee/~kaili/grammatika/estmorfcg.tar.gz`, 01.08.2010

- **TWOLC:** *Xerox*'i kahetasemeliste reeglite kompilaator – käsk, vajadusel täispika teega, mis *twolc* käima paneb.
- **ICONV:** märgistikuteisendaja *iconv* – käsk, vajadusel täispika teega, mis *iconv* käima paneb.
- **ESTMORF:** analüsaator ESTMORF – käsk, vajadusel täispika teega, mis ESTMORFi käima paneb.
- **EKI_DATA:** viide kataloogile, kuhu on lahti pakitud EKI andmefailid.
- **EKI_ANA:** EKI analüsaator – käsk, vajadusel täispika teega, mis EKI analüsaatori käima paneb.

Teisendaja ehitamiseks piisab muutujate **XFST, TWOLC** ja **ICONV** sobivast väärtustamisest. Muutujate **ESTMORF**, **EKI_DATA** ja **EKI_ANA** väärtustamine on vajalik, kui soovitakse ka nende analüsaatoritega testida.

Kui vajalike väliste programmide asukohad määratud on, käivitab teisendaja ehitamise käsk **`make.`** Valmis teisendaja kirjutatakse faili eesti.fst.

Interaktiivse *xfst*-liidese käivitab käsk **`make xfsti`**, mis vajadusel ka eelnevalt teisendaja või selle osad ehitab.

Genereeritud failid kustutab käsk **`make clean`**.

2.5.4 EKI andmefailide teisendajad

Käesoleva magistritöö käigus loodud originaalsetest programmidest kõige tähtsamad on teisendajad, mis EKI andmefaile *lexc* muus raamistikus kasutatavate sõnastikena esitavad.

eki2lex.pl

Kahest olulisest teisendajast töömahukam on tüvebaasi teisendaja. EKI tüvebaas on esitatud tekstifailina, mille igal real on ühe sõna kirjeldus. Esimesel positsioonil on võimalike paralleelvormide marker, sellele järgneb vahetult sõna algvorm ning tühiku järel muuttüübi ja sõnaliigi kood. Kui sõna kuulub muutuvasse tüüpi, siis järgneb koodile tühikutega eraldatud püstkriips ning koma ja tühiku paaridega eraldatud tüvevariandid kujul variandikood, koolon, tühik, tüvevariant. Kui variant puudub, on selle asemel trellid (#) või null (0). Tüvebaasi algvormides ja tüvevariantides on märgitud ka kolmas välde ja ebaregulaarne rõhk, kuid seda infot teisendaja praegu ei kasuta.

Programmi algoritm on lihtsustatult järgmine:

- loe sisse järgmine mittetühi rida, puhasta see ebavajalikust (paralleelvormide marker, vältemärgid), valmista ette tüvevariantide tabel.
- sõltuvalt muuttüübist tuvasta tähtsamate käänete (omastava, mõnikord ka osastava) aluseks olevad tüvevariandid ning võrdle neid algvormiga ja omavahel muuttüübist sõltuvate mustrite alusel. Sõltuvalt sobinud mustrist vali sõnale alltüüpi esindav jätkusõnastikuviide ning vajadusel markeeri tüve süvakujus muutuvad märgid.
- teisenda sõnaliigikoodis olnud EKI tähistused meie sõnaliigimärgenditeks. Lisa iga leitud sõnaliigi kohta sõnastikku üks kirje.
- alusta algusest, kuni sisendis rohkem ridu pole.
- sorteeri ning kirjuta saadud sõnastik faili.

Peamiselt võrreldakse tüvevariantide vastavust mitmetele regulaaravaldistele ning uuritakse nende teisendatavust ühest teiseks. Mustrikomplektid on iga muuttüübi jaoks erinevad, kuigi arenesid töö käigus mitmes osas tüüpide võrdluses suhteliselt sarnaseks – astmevaheldus jt protsessid, mis tüvevariantide taga on, on ju kõigile üldised ja süvakujust pindesitusi toodab kõigist tüüpidest sama reeglikomplekt.

Suhteliselt lihtsa teisendaja arendamise tegi ajamahukamaks vajadus paralleelselt ka sõnastikesüsteemi seni kirjeldamata alltüüpide osas laiendada ja vigu parandada.

exc2lex.pl

Erandifaili teisendaja lähteandmeteks on EKI vormikoodide tabel ja erandifail. Vormikoodide tabelis on igal real komaga eraldatult loetelu erinevaid vormikoode. Algses failis on veerud alates esimesest vormi lühend, formatiivikood, EKI sisekood ning Filosofti (ESTMORFi) kood. Parandatud failis on viiendas veerus teisendaja jaoks kasutatavad märgendid. Failis on vormikoodide tabel jaotatud gruppideks, kuid selle infoga pole erandifaili teisendamise kontekstis midagi kasulikku teha.

Erandifail koosneb erandiridadest, millel on komaga eraldatud muuttüübi ja sõnaliikide koondkood, sõna algvorm, muutevorm, kus formatiiv on kantsulgudega markeeritud, EKI vormikood, lisatingimuse tüvekood või miinus, lisatingimuse tüvekoodile vastava tüve kuju või miinus ning paralleelvormi tunnus pluss või tärn. Pluss tähistab paralleelvormi, tärn erandlikku vormi. Nii algvormis kui muutevormis on märgitud välde.

Erandifailide teisendaja on tüvebaasi teisendajast veel lihtsam. Sisuliselt tehakse iga reaga:

- leia EKI sisekoodi järgi leksikaalse teisendaja vormimärgendid
- leia (eelmise teisendajaga analoogselt) võimalikud sõnaliigi märgendid
- puhasta muutevorm kantsulgudest ning muutevorm ja algvorm vältemärkidest.
- lisa sobivasse (paralleelvormide või erandite) sõnastikku iga sõnaliigi märgendi kohta *lexc* formaadis vastavus algvormi, sõnaliigi märgendi ja vormimärgendite konkatenatsiooni ja muutevormi vastavus.

Lõpuks kirjutatakse saadud sõnastikud eraldi failidesse.

2.6 Testimine ja tulemused

Teisendaja arendamise oluline tugi oli tulemuse pidev testimine. Selleks tekkis töö käigus üsna mitmesuguseid vahendeid. Järgnev kirjeldabki loodud testimisvahendeid ning nende kasutamist.

2.6.1 Testimisvahendid

Vormigeneraatorid gen-nouns.sh ja gen-verb.sh

Vormigeneraatorid olid kasulikud tüvebaasi teisendaja arendamisel ja testimisel ning jätkusõnastike süsteemi parandamisel. Vormigeneraatoritega oli mugav genereerida vajalikke süvaesitusi kahetasemeliste reeglite testimiseks.

Mõlema skripti ülesehitus ja tööpõhimõte on sama. Vormigeneraator valmistab *xfst* abil ette pööratud teisendaja, vajadusel üritades seda *make* abil tekitada. Seejärel genereeritakse hulk võimalikke morfoloogiliste tunnuste kombinatsioone ning lõpuks palutakse need programmi *lookup* abil sünteesida (*lookup* on tegelikult mõeldud analüüsiks – sellepärast ka pööratud sõnastik). Lõpuks eemaldatakse väljundist tühjad read ning joondatakse see mugavama loetavuse huvides.

Ainus erinevus on sisendparameetriks antava sõna kuju. Käändsõnadel (gen-nouns.sh jaoks) tuleb anda sõnastikuju koos sõnaliigi märgendiga, näiteks:

```
./gen-nouns.sh maja+S
```

Verbigeneraatorile sobib verbi sõnastiku-algvorm:

```
./gen-verb.sh ujuma
```

Lisavõimalusena saab juhtida, millist teisendajat vormide genereerimiseks kasutatakse. Vaikimisi kasutatakse täielikku leksikaalset teisendajat failis *eesti.fst*. Kasulik alternatiiv on aga näiteks selline põhisõnastik, mille ülemisele poolele (sõnastikukujule) on teisendusreegleid juba rakendatud kuid alumisele mitte. See genereeritakse süsteemi ehitamise vaheetapina faili *lex-av.fst*. Sedasi on võimalik genereerida konkreetse sõna vormide süvaesitused. Alternatiivne teisendaja antakse ette keskkonnamuutuja **FST** kaudu:

```
FST=lex-av.fst ./gen-nouns.sh susi+S
```

Vormigeneraatorite tekstides on kirjas *xfst* ja *lookup*'i asukohad, need tuleks vajadusel enne kasutamist sobivaks parandada.

Testkorpuste ettevalmistamine

Korratavate tulemuste saamiseks on tähtis, et lisaks programmikoodile oleks ka kasutatavad testandmed ühesugused. Testandmete allikana oleme kasutanud TÜ Arvutuslingvistika Uurimisrühma morfoloogiliselt ühestatud korpust[13]. Alamkataloogis *korpus* on vahendid failide allalaadimiseks viidatud lehelt ning teisendamiseks analüsaatori sisendkujule (sõnavorm per rida) ning kontrollformaadiks (iga sisendkorpuse unikaalse reakohta rida, mille alguses on selliste ridade korpuses esinemise arv). Korpusefailidest eemaldatakse lausete ja lõikude märgendus, kirjavahemärgid ja teisendatakse olulisemad SGML-olemid UTF–8 märkideks.

Kiireim viis testfailid genereerida on anda käsk

```
make; make
```

Kaks korda, sest korpusefailid, millest järgmine samm sõltub, tekivad uude alamkataloogi ning make ei suutnud sealt kohe faile leida.

Vaikimisi genereeritakse sisendfailist ja analüüsikontrollija (vt 2.6.1) kontrollfailist koosnevad paarid 1984 tekstist ning kõigist korpuse tekstidest kokku. 1984 teksti põhjal genereeritud sisendfail (1984.uwords) peaks olema identne põhikataloogis oleva vaikesisendiga (1984_words_u.txt).

Konkreetsetest korpusefailidest sisendi või kontrollfaili ehitamiseks saab kasutada make'i, sihtmärgiks peaks olema korpusefaili nimi, kust on eemaldatud sufiks .kym ning lisatud vastavalt .uwords või .check. Olemas on ka reeglid tõstutundlike sisendi ja kontrollfaili tekitamiseks, siis on uued sufiksid .csuwords ja .cscheck:

```
make hor_2002_2_12.uwords hor_2002_2_12.check
```

[13] `http://www.cl.ut.ee/korpused/morfkorpus/`, 01.08.2010

```
make hor_2002_2_12.csuwords hor_2002_2_12.cscheck
```

Failigruppide teisendamise näide on näha Makefile's.

Teisendamise sisulise töö korraldamisega tegelevad samas kataloogis asuvad *shelli*-skriptid to_lc_testfile.sh ja to_lc_words.sh. Mõlemad eeldavad korpuse-formaadis teksti standardsisendis ning annavad tulemuse standardväljundisse.

Analüüsi käivitamine testfailil

Üks ehitussüsteemis kirjeldatud rutiintegevusi on analüsaatorite rakendamine etteantud testfailile. Kõige mugavam on selleks kasutada sihti *test:*

```
make TESTFILE=korpus/full.uwords test
```

Kui parameeter *TESTFILE* väärtustamata jätta, kasutatakse vaikimisi testfaili *1984_u_words.txt*. Sel viisil käivitatult eeldab ehitussüsteem, et olemas on kõik kolm analüsaatorit ning genereerib kõigi kolme väljundfailid *xfst.out, estmorph.out* ja *eki.out* ning loeb kokku tundmatud sõnad igas neist ning siendfaili ridade arvu. Vajadusel saab kasutada ka vaid ühte või kahte analüsaatorit, selleks tuleb make käsureal test asemel nimetada vastavate analüsaatorite väljundfail või -failid.

Muud kasulikud ehitussüsteemi sihid on:

- testx – "ehitab" vajadusel xfst.out ja loeb kokku tundmatud sõnad,
- tundmatud – "ehitab" vajadusel xfst.out ja kuvab tundmatud sõnad,
- estmorf_check.out – (vajadusel genereerib ja) teisendab estmorf'i väljundi analüüsikontrollijale sobivale kujule.

Analüüsi õigsuse kontrollimiseks on veel ehitussüsteemi integreerimata programm check_analyze.pl, mis ootab esimese argumendina kontrollfaili nime. Kui keskkonnamuutujal MAPPER on väärtus FS, kasutatakse ESTMORFi režiimi, muidu leksikaalse teisendaja režiimi. Analüüsiväljundit ootab programm standardsisendisse:

```
./check_analyze korpus/full.check < xfst.out | less

MAPPER=FS ./check_analyze korpus/full.check < estmorf_check.out | less
```

Analüüsikontrollija väljastab morfoanalüsaatori väljundis puudunud kuid korpuses esinenud analüüsid koos vastava sõna kohta analüsaatorilt saadud analüüsidega. Kui kogu analüsaatori väljund on töödeldud, väljastab kontrollija statistika unikaalsete sõnade ja kõigi sõnade kohta (nagu oleks korpust järjest analüüsitud ning iga sõna juures kontrollitud, kas korpuse-variant analüüsist leidus analüsaatori väljundis või mitte). Väljastatavad indikaatorid on:

- unikaalsete sõnade / kokku sõnade arv korpuses,
- eelmistest mingigi analüüsi saanud sõnade arv,
- ja analüüsita jäänud sõnade arv (protsendina kõigist),
- sõnade arv, millele korpuses (kontrollfailis) analüüsi ei ole, kuid analüsaatori väljundis on. Üldiselt viitab nullist erinev arv valele kontrollfailile, probleemidele sisendi kodeeringuga vms,
- sõnade arv, mille kõik võimalikud korpuses esinenud analüüsid analüsaatori väljundis esinesid ning korpuse jooksvas tekstis esinenud sõnavormide arv, mille ühestatud analüüs ka analüsaatori väljundis leidus
- eelmise täiend
- selliste sõnade arv, millele leidus analüüs peale võrdlusastme eemaldamist. Osa komparatiive ja superlatiive (parem, parim) on EKI baasis otse omadussõnana toodud ja teisendaja peab neid tavalisteks omadussõnadeks.
- analüüsita sõnad, mille õige analüüsi sõnaliik on pärisnimi
- analüüsita sõnad, mille õige analüüsi sõnaliik on määrsõna
- analüüsita sõnad, mille õige analüüsi sõnaliik on kaassõna ja analüsaator pakkus määrsõna
- analüüsita sõnad, mille õige analüüsi sõnaliik on määrsõna ja analüsaator pakkus kaassõna

Lisaks väljastatakse valik korpuses sagedamini esinevaid sõna+analüüs paare, mis analüsaatori väljundis ei esinenud.

2.6.2 Analüüsi testimine

Leksikaalse teisendaja testimisel oli analüüsi suund märksa põhjalikuma vaatluse all kui sünteesi suund. Peamised kaks testimeetodit olid analüüsita jäänud sõnade arvu jälgimine (ning nende sõnade uurimine eesmärgiga analüsaatorit parandada ning vigu avastada) ning leitud analüüside võrdlemine ühestatud korpuses väljapakututega.

Mitteanalüüsitavate sõnade loendamine toimus peamiselt "1984"-põhise korpuse peal, kuhu kuulus 16808 unikaalset sõna. Praegune tulemus on, et leksikaalne teisendaja ei suuda analüüsida 703 sõnavormi. Need on erinevad pärisnimed, tehiskeele sõnad,

tundmatud (sõnastikust puuduvad) tüved ning erinevad liitsõnad. Tundmatuid liitsõnu on 368, enamus neist sisaldab kas ase- või arvsõna (mille kohta liitsõnareegleid veel eriti ei ole). Oluline on märkida, et selline kontroll ei uuri kuidagi saadud analüüsi õigsust ning saadud suuruste interpreteerimine saagisena ei ole korrektne.

Tundmatute sõnade loendamise ja uurimise põhjal saaks arvatavasti sõnastiku ja reeglite täiustamist jätkata.

Analüüsi kontrollimise eeliseks on "negatiivne tagasiside". Selle kaudu on võimalik tuvastada juhud, kus vigaste reeglite vms tõttu tekivad valeanalüüsid. Praktiliselt kasutatava analüsaatori saamiseks peaks analüsaator väga suure osa sõnade jaoks leidma kõik sellised analüüsid, mis ka korpuses leiduvad. Samas ei saa see osa olla 100%, sest korpuses leidub "kahtlaselt" analüüsitud sõnu. Näiteks on "1984" tekstis sõna *aaras*, mis kõigile analüsaatoritele on sõna *aara* ainsuse seesütleva käände vorm. Tekstis on see aga mugandus sõnast *haaras (< haarama)* ja sellisena hoopis verbivorm.

Lõplikel automaatidel töötava analüsaatori testimise kontekstis on oluline ka analüsaatori tõstutundetus ja sellest tulenev võimetus pärisnimesid kergemini eristada. Ühtlasi tähendab see, et mingite tavaliste väiketäheliste sõnade analüüsidele lisanduvad pärisnime-variandid, kuigi neid tegelikult olema ei peaks. Kui sama, väiketäheliseks teisendatud teksti teiste, suurtähti eeldavate ja kasutada oskavate analüsaatoritega (ESTMORFiga) analüüsida, siis põhjustab see massiliselt vigu, mida analüsaator muidu ei teeks.

Analüüsi kontrolli algoritm peab lahendama kaks põhilist probleemi. Esiteks pole kummagi analüsaatori väljund ei formaadi ega kasutatava märgenduse poolest samane ühestatud korpuse failidega. Probleem algab asjaolust, et korpuses on põhimõtteliselt vaid üks õige analüüs, samas kui analüsaator peab tõenäoliselt pakkuma mitmeid. Märgenduse osas on leksikaalne teisendaja korpusele lähedal, kuid ei vasta sellele täpselt, sest osa korpusemärgenduses olevat infot on üheselt määratav alles kogu lause konteksti arvestades, st ühestamisel. Selline osa korpuse märgendusest, mida analüsaatorid ei väljasta, on eemaldatud analüüsikontrollija testfaili ettevalmistamisel (täpsemad asendused on loetavad failist korpus/to_lc_testfile.sh). Võrdlemiseks teisendatakse ka analüsaatorite väljundid samasugusele kujule.

Teine suurem ülesanne on tegelik võrdlus. Loodud võrdleja ehitab kontrollfaili alusel paisktabelite struktuuri, mille abil on kiiresti võimalik tuvastada sõna korpuses esinenud analüüsid[14] ning nende esinemiskordade arvud. Iga analüsaatori väljundis analüüse omava sõna korral kontrollitakse kõigi korpuses esinenud analüüside kohta, kas need

[14]NB! Praegune analüüsikontrollija uurib vaid morfoloogilist märgendust ning ei vaata leitud tüve. See põhjustab kindlasti mitmete tegelikult vigaste analüüside õigeks lugemist.

leidusid ka analüsaatori väljundis. Iga sõna loetakse edukalt analüüsituks unikaalsete sõnade arvestuses, kui leidusid kõik korpuses esindatud erinevad analüüsivõimalused. Kõigi sõnade arvestuses loetakse iga leidunud analüüsivariandi kohta nii palju sõnu, kui oli sellise analüüsivariandi esinemiste arv. Sisuliselt peaks see viimane aitama vastata küsimusele "kui suur osa korpusega esindatud tekstitüübis esinevatest sõnadest tegelikult õige analüüsi saab" ehk mis on analüsaatori täpsus tavalise teksti (vs unikaalsete sõnade) analüüsimisel. Kui sagedamini esinevad sõnad õigesti analüüsitakse, siis on lootust, et korduvate sõnadega teksti analüüsitäpsus on suurem kui unikaalsete sõnade analüüsitäpsus.

Meetodina on selline analüüside kontrollimine kindlasti esimesena kirjeldatud sõnaloendusest efektiivsem ja annab täpsemat infot analüsaatori (või vahel ka korpuse) probleemide kohta. Tabelis 2.1 on võrreldud leksikaalse teisendaja ja ESTMORFi tulemusi kõigist morfoloogiliselt ühestatud korpuse failidest genereeritud testsisendil. Paarisveergudes on paremal pool protsent kõigist sõnavormidest, analüüsiga sõnavormidest või vigadest.

	LT				ESTMORF			
	unikaalsed		kõik		unikaalsed		kõik	
Sõnavormid	88662		514767		88662		514767	
Analüüsiga	78252	88.3	485820	94.4	80918	91.3	494112	96
Analüüsita	10410	11.7	28947	5.62	7744	8.73	20655	4.01
Edukaid anal.	74924	95.7	475198	97.8	78052	96.5	483007	97.8
Vigaseid anal.	3328	4.25	10622	2.19	2866	3.54	11105	2.25
Pärisnimed	2518	75.7	6911	65.1	2609	91	9433	84.9

(teisendaja saagis ~92.3% ja täpsus 97.8%, ESTMORFil vastavalt 93.8% ja 97.8%)

Tabel 2.1: Teisendaja ja ESTMORFi analüüsikontroll, väiketähtedega sisend

Siinkohal tuleb uuesti rõhutada, et see test on ESTMORFi suhtes pisut ebaaus, sest tegelikult suurtähtedega arvestavalt analüsaatorilt "oodatakse" tulemusi, mida ta ei peagi andma. Tabelis 2.2 on võrdluseks toodud ka tulemused väiketähestamata testfaililt, mis käivad leksikaalsel teisendajal teadaolevalt üle jõu, kuid vastavad paremini päris elule.

Esmapilgul suhteliselt täpse leksikaalse teisendaja saladus peitub tõenäoliselt "võimes" keerulisi sõnu mitte analüüsida ja seeläbi pääseda ka nende vigasest analüüsimisest. Viimase tabeli praktiliselt olematu pärisnimedega eksimine tuleb

	LT				ESTMORF			
	unikaalsed		kõik		unikaalsed		kõik	
Sõnavormid	96584		514767		96584		514767	
Analüüsiga	73465	76.1	438004	85.1	90431	93.6	501933	97.5
Analüüsita	23119	23.9	76763	14.9	6153	6.37	12834	2.49
Edukaid anal.	72524	98.7	434840	99.3	88130	97.5	496132	98.8
Vigaseid anal.	941	1.28	3164	0.72	2301	2.54	5801	1.16
Pärisnimed	5	0.53	7	0.22	1833	79.7	4152	71.6

(teisendaja saagis 84.5% ja täpsus 99.3%, ESTMORFil vastavalt 96.4% ja 98.8%)

Tabel 2.2: Teisendaja ja ESTMORFi analüüsikontroll, tähti väikeseks tegemata

kindlasti sellest, et pärisnimed on tavaliselt suure algustähega ning suurtähelised sõnavormid jäid analüüsita. ESTMORFi sagedasemini valesti analüüsitud pärisnimed on paralleelselt ka tavalised sõnad (*Vene, Liidu, Horisont, Poom, Mõis, Lääne*, *Malka* jm) ja arvatavasti pakkus analüsaator nende analüüsiks ainult tavasõna-variante.

Analüüsikontrolli veelgi täpsemaks muutmiseks oleks vaja lisada ka tüve kontroll, ehk võrdlus, kas korpuse-analüüs ja analüsaatori leitu käivad ikka sama tüve kohta. Näiteks on sõna *meetrilaiust* analüüside seas nii variant *meeter+S+sg+gen&-lai+A+sg+nom&-uks+S+sg+part* kui ka *meeter-+S+sg+gen&laius+S+sg+part.* Siit on näha, et asjaolude kokkulangemisel võib analüsaator leida "sobiva" vastuse, kuigi tegelikult korrektset analüüsi ei leitud. Sellise kontrolli lisamine pole päris triviaalne, sest analüsaator peaks vajaliku tüve ja lisanduva muutevormi sünteesima. Nagu jaotises 2.5.1 mainitud, ei tööta praegu tuletiste algvormide genereerimine korralikult. Lisaks tüvele on korpuse-esituses märgitud tüvele liitunud lõpp, mida praeguses analüüsiformaadis üldse pole. Tõsi, ka ilma lõppudeta tüvede/algvormide järgi kontrollimine annaks kontrollitäpsust juurde. Ülaltoodud, ilma tüvekontrollita saadud tulemustesse tuleks seega suhtuda üsna ettevaatlikult, tõenäoliselt on nii tegelik saagis kui täpsus pisut halvemad.

Teistpidi on teadmine, et juba toimiv analüüs ei jää ka niimoodi, pisut auklikult defineeritud vigadehulga (või pigem -vähesuse) poolest ESTMORFile eriti alla, üsna julgustav. Analüüsikontrolli tulemusest on selge ka see, et püstitatud eesmärk – praktiliselt kasutatav morfoloogiakirjeldus – pole veel saavutatud. Samas on vigu analüüsides võimalik sihile kiiremini jõuda. Kõige tähtsamad arendussuunad on arvatavasti:

- tõstutundetuse probleemi lahendamine,
- sõnastiku täiendamine,

- liitsõnareeglite täiendamine.

Täiesti omaette testitav kvaliteet on analüüsi kiirus võrreldes teiste kasutatavate analüsaatoritega, Eesti tegelikkust arvestades, ESTMORFiga. Korrektse testi tegemiseks tuleks katse tingimused korralikult defineerida ning anda ka ESTMORFi autoritele võimalus oma programmi tingimustele vastavalt kohandada. Jämeda hinnangu saamiseks saab aga taaskasutada vahendeid, mille abil mõõtsime analüüsi kvaliteeti. Tekitasime testimiseks suure sisendfaili, mille analüüs kestaks piisavalt kaua, et muuta tühiseks mõõteviga, emulaatori *wine* kasutamisest tingitud lisaajakulu jms. Testifail koosneb morfoloogiliselt ühestatud korpuse sõnadest, mida on võetud 10 korda järjest. Testfailis on 5147670 sõna. ESTMORFi testimiseks on testfail eelnevalt teisendatud vajalikule kujule, teisendamise aega ei mõõdeta. Ajahinnangu saamiseks käivitasime analüsaatoreid sisendil kolm korda järjest, mõõtes kasutatud protsessoriaega.

Teisendaja jooksutamine kolmel katsel kestis 140.3, 141.2 ja 150.9 sekundit, so keskmiselt 144.1 sekundit. See tähendab umbes 35715 sõnavormi analüüsimist sekundis.

ESTMORFi jooksutamine kestis 319.3, 353.7 ja 351.4 sekundit, so keskmiselt 341.5 sekundit ja umbes 15075 analüüsimist sekundis.

Praktiliselt ainsa järeldusena saame tõdeda, et nendes katsetes oli teisendaja-põhine analüsaator üle kahe korra kiirem. Arvestades algoritmide erinevust oli teisendaja edu mõnevõrra oodatav.

2.6.3 Sünteesi testimine

Süstemaatilise ja massilise sünteesi testimisega pole seni tegeletud. Kardetavasti ei sobigi praegune versioon teisendajast väga hästi üldotstarbeliseks sõnageneraatoriks, sest kuidagi pole võimalik eristada homograafseid algvorme. Veidi paremaks võiks olukord muutuda, kui sõnastikuesituses oleks märgitud kas muuttüvi või mingid muud markerid, millega homograafe eristada.

Ad hoc sünteesiteste tegime tüvebaasi teisendaja ja jätkusõnastike süsteemi arendamise käigus ning pisteliselt ka hiljem, analüüsitestide käigus, tavaliselt eesmärgiga vigade põhjustest aru saada. Selleks kasutatud vahendeid kirjeldas jaotis 2.6.1.

2.7 Rakendused

Praegune versioon teisendajast pole praktiliselt rakendatav. Mõnesid praktilise rakendatavuse saavutamiseks vajalikke tegevusi loetles jaotis 2.6.2. Oletame aga hetkeks,

et kirjeldatud probleemid on lahendatud ning meil on enamvähem mõistlikult töötav lõplik teisendaja.

2.7.1 Õigekirjakontroll

Plaanitud esmane rakendus on vabavaraline õigekirjakontrollija. Sealjuures on huvitavad sellised lahendused, mis võimaldavad loodud kontrollijat vähese vaevaga kasutada tarkvarapakketidega, mille abil tavaline vabavarakasutaja kirjutab suure osa oma tekstidest – OpenOffice.org, Mozilla, LyX jms. Lähtudes lõpliku teisendajana esitatud morfoloogiakirjeldusest on hetkel näha kaks põhimõttelist võimalust õigekirjakontrollija loomiseks.

Esimene võimalus on leida või luua mõni spellimismootor, mis suudaks otse lõplikku teisendajat kasutada. Erinevalt aastatetagusest seisust ei tundugi see võimalus päris lootusetu. Mitmetele keeltele on vabavaraliselt leksikaalsed teisendajad olemas ning soomlaste projekt *Voikko* on deklareerinud eesmärgi kunagi toetada *HFST*-formaadis teisendajaid[15]. Praegu on see eksperimentaalne ning ka vajalike tugiteekide (sh *HFST*) kasutamine on raskendatud. *Voikko* eelis on ka juba olemasolevad moodulid levinumatele rakendustele.

Teine, mõneti huvitavam võimalus seisneb lõpliku teisendaja (või näiteks selle defineeritud regulaarse keele) pool- või täisautomaatses teisendamises mõnda teise esitusse, näiteks *hunspelli* sõnastikuks ja afiksifailiks. Arvestades *hunspelli* afiksifaili värskemaid täiendusi, võiks lõpliku automaadi esitamine *hunspelli* raamistikus võimalik olla[16]. Ühilduvusprobleeme sellisel juhul poleks – hunspell on viimasel ajal üks populaarsemaid mootoreid.

2.7.2 Lemmatiseerija

Mitmetes infootsimise rakendustes on kasulik, kui leidub moodul sõna algvormide kiireks leidmiseks. Kui lahendada tuletusaluste tüvede moodustamine, siis võiks teisendajat praktiliselt kohe lemmade leidmiseks kasutada. Probleemiks võib olla liitsõnade esikomponentide analüüsimine, kui kasutaja tahaks lihtsalt käändelõppudest vabaneda vms. Samuti oleks süsteem lemmatiseerijana kasulikum, kui leiduks oletamisrežiim. Oletamisrežiimi ehitamise võimalikkusele kasutades sõna fonoloogilist struktuuri kirjeldavaid regulaaravaldisi viitab ka Heli Uibo [33]. Lõplikel automaatidel põhinev

[15] `http://sourceforge.net/apps/trac/voikko/wiki/libvoikko/SupportedLanguages`, 01.08.2010

[16] `http://sourceforge.net/projects/hunspell/files/Hunspell/Documentation/hunspell4.pdf/download`, vt COMPOUNDRULE. 01.08.2010

lemmatiseerija suudab olla ka väga kiire ja sedasi sobib hästi suurte andmemahtude indekseerimiseks.

2.7.3 Iseseisev morfoloogiline analüsaator

Selleks vajalikud tükid on juba praegu olemas ning *Xeroxi* tööriistu kasutades saab analüsaatorit kasutada. Just nii käib ka keelekirjelduse kvaliteedi kontroll. Skriptid, mille abil teisendaja väljund näiteks ESTMORFi väljundi sarnaseks muuta, saab lihtsalt luua juba olemasoleva analüüsikontrollija põhjal. See tähendab, et soovi korral (ja morfoanalüsaatori kvaliteedi sobivuse korral) peaks saama kasutada teisendajat kohtades, kus praegu kasutatakse ESTMORFi. Teisendaja eeliseks peaks olema ka teoreetiliselt pisut suurem kiirus.

2.7.4 Süntaksianalüüs

Teine tähtis eesmärk analüsaatori kasutatavus keerulisemate keeletehnoloogiliste moodulite sisendina. Analüsaatori sobitamine süntaksianalüsaatori [23] eeltöötlusahelasse[17] pole tegelikult keeruline ning tegelikult on suur osa vajalikust tööst juba ka analüüsikontrollija loomise raames tehtud. Siinkohal kerkib huvitav küsimus, kas või kui palju erinevalt käituvad reeglipõhine morfoloogiline ühestaja ja süntaksianalüsaator, kui morfoanalüsaatori väljundis on rohkem võimalikke vorme kui ESTMORFil, või üldisemalt, kui kasutatakse mingit muud analüsaatorit peale ESTMORFi?

Huvitav süntaksianalüüsil põhinev rakendus on grammatikakorrektor (millest ka käesoleva töö kirjutamise käigus korduvalt puudust tundsime). Eesti keele grammatikakorrektori loomisega tegeldakse vastava raamprogrammi toel[18]. Tõenäoliselt kasutab loodav korrektor seesmiselt EstCG süntaksianalüsaatorit ja seetõttu võiks kehtida eelmises lõigus viidatud asendamislihtsus.

Pealiskaudne uurimine näitab, et lõpliku teisendaja kohandamine ka OpenOffice.org vabavaralise grammatikakorrektori LanguageTool[19] lauseosade märgendajaks *(POS tagger)* pole keeruline. Tõsi, sellele programmile mõeldud grammatikareegleid poleks valmiskujul veel kuskilt võtta ning sellisena oleks see pigem töötavuse kontroll või väljakutse süntaksikirjeldustega rohkem tegelenud inimestele.

[17] `http://math.ut.ee/~kaili/grammatika/`, 01.08.2010

[18] `http://www.keeletehnoloogia.ee/projektid/syntaktiline-keeletarkvara/grammatikakorrektor`, 01.08.2010

[19] `http://www.languagetool.org/`, 01.08.2010

2.7.5 Kõne süntees ja tuvastus

Teadaolevalt ei sobi praegune sõnastik pigem kõne sünteesi rakendustesse, sest andmed ei sisalda vajalikku, vaid kõnes väljenduvat lisainfot. Osa sellest infost on EKI tüvebaasis olemas ning mõeldav oleks selle sealt kasutusele võtmine. See tähendaks, et pisut tuleb üle mõelda fonoloogia aspektide kodeerimine sõnade süvakujudes ning ideaalne oleks, kui sellise töö tegemise ajal oleks silmapiiril ka konkreetne rakendus.

Kõnetuvastuse jaoks võiks analüsaator olla kasutatav õigekirjakontrollijana ning süntaksianalüsaatori osana, kui neid kasutatakse kõnetuvastuse väljundite hindamiseks või kontrollimiseks.

Peatükk 3

Edasised tegevused

Praegune morfoloogiakirjeldus ei ole veel päris valmis. See tähendab, et on terve hulk tegevusi, mis tuleks kasutatava tulemuse saamiseks veel teha. Samuti on terve hulk tegevusi, mis tõstaksid morfoloogiakirjelduse väärtust ja kasutatavust, kuid mis pole praktilise, kirjaliku teksti õigekirjakontrolliks ja süntaksianalüüsi all töötava morfoloogilise analüsaatori jaoks hädatarvilikud.

3.1 Vältimatud tegevused

3.1.1 Sõnastik

Jaotis 2.5.1 loetles mitmed praeguse tüvedesõnastiku puudused. Üks olulisemaid on baassõnastiku suhteline vanus ning vajadus sõnastikku uuendada. Tõenäoliselt ei ole tüvebaasi esitus sõnastikule kõige optimaalsem, kuid näiteks tüvevariantideta sõnastikukirje tähendaks variantide generaatori ehitamist jne. Kuigi mittekasutatavaid sõnu saab eemalda neid lihtsalt tüvebaasist kustutades, oleks mõistlikum need pigem märgendada koos mittekasutamise selgitusega. See võimaldaks sellised sõnad vajadusel (näiteks analüsaatori kohandamine vanade tekstide analüüsile?) taastada.

Lahendamist vajab tuletusaluste sõnastikuesituse tuletamine. Peamine sisuline lahendus seisneb sõnastikuesituse täielikult genereeritavaks muutmises. Seda saab aga teha mitmel viisil ning milline neist kõige otstarbekam on, pole selge ning vajab täiendavat analüüsi.

Sõnastikuga, eriti kahetasemeliste reeglite kodeerimisega selles, on seotud ka tõstutundetuse probleem. Kõige lihtsam lahendus oleks praeguste eritähendusega suurtähtede asemel võtta kasutusele näiteks mitmemärgilised erisümbolid. Sümbolikomplekti vahetamine tähendaks praktiliselt vaid tüvebaasi teisendaja ja väheste

käsitsi kirjutatud erandite täiendamist. Pisut keerulisem probleem on kahetasemelistes reeglites ka suurtähtede lubamine ning selle tagamine, et valikukohtades pindsümbol alati õiges suuruses oleks.

Lahendamist vajab ka erandite ja tuletamise vahel ilmnenud konflikt.

3.1.2 Liitsõnareeglid

Suhteliselt suur osa mitteanalüüsitavatest sõnadest on liitsõnad. Liitsõnareeglistik tuleb veelkord üle vaadata ning võibolla praegune jätkuleksikonidega kirjeldus ebaõnnestunuks kuulutada. Liitsõnamoodustuse praegu käsitlemata aspekt on sõnaliigi muutus. Praeguste reeglitega on raske kirjeldada olukorda, kus sõnaliigid A ja B annavad liitudes C (*omapead* – käändsõna + käändsõna = määrsõna jms).

3.1.3 Testimisvahendid

Efektiivsemaks arenduseks on kasulik ka analüüsikontrollija ehitussüsteemi integreerida. Samuti tuleks täpsema kvaliteedihinnangu saamiseks realiseerida viidatud tüvesamasuse kontroll.

3.1.4 Ehitamissüsteem

Praegune süsteem kasutab XFST mittekommerts-versiooni. Sellel versioonil on mitmeid ebamugavaid piiranguid ning sellisel kujul ei saa teisendajat praktiliselt kuskil kasutada. Korrektseks vabavaraliseks kasutamiseks tuleb süsteem üle viia HFST vms vabavaralise tööriistakomplekti kasutamisele. Sellise üleviimise nõrk koht on HFST enda segane seis, see pole veel suuremate Linuxi-distributsioonide osa (kasutaja peab selle endale ise ehitama) jms.

3.1.5 Esimene rakendus

Morfoloogiakirjelduse kohta saab öelda, et see on tõeliselt praktiline, kui selle põhjal on valminud esimene rakendus ning kasutajad on selle heaks kiitnud. Huvitaval kombel võib juhtuda, et lõpliku teisendaja kasutamine kõrgema taseme keelerakendustes vajab vähem tööd kui kõige lihtsamates – õigekirjakontrollijas, lemmatiseerijas või poolitajas. Seda peamiselt sellepärast, et kõrgema taseme rakendused kas suudavad kasutada abstraktset teisendajat (automaat tuleks lihtsalt teisendada vastava rakenduse esitusse) või eeldavad, et morfoanalüsaator on eraldiseisva programmine realiseeritud moodul (selle tegemine oleks lihtne).

Nii ongi kõige optimaalsem otsida esimest rakendust just morfoloogilise analüsaatori asendamisest mõnes olemasolevas süsteemis, kus teisendajapõhine lahendus oleks kiirem, sobivama litsentsiga või mugavamini integreeritav. Eesmärgiks võetud õigekirjakontrolli osas on tõenäoliselt otstarbekas ära oodata projekti Voikko tulemused ning olla selleks ajaks, kui vajalik tugi tekib, valmis pakkuma eesti keelele sobivat teisendajat.

3.2 Kaugema-tuleviku-arendused

3.2.1 Tüvemoodustuse eraldamine

Usume, et tüvemoodustuse reeglite eraldamine eraldi moodulisse võimaldaks süsteemi teisi mooduleid, eeskätt tuletamist, lihtsamalt ning kompaktsemalt kirjeldada. Praktiliselt võiks see tähendada, et "süvaesitus" koosneb algvormi sõnastikuesitusest ja tüvevarianti valivatest märgenditest ning tüvemoodustuse moodulis saaks mugavamalt valida konkreetsete pindesituste tekitamise meetodi. Praegu on sõnastikku sissekirjutatud kahetasemeliste reeglite eeldatav süvaesitus ning ka jätkusõnastikud sisaldavad konkreetseid, kahetasemelistes reeglites kasutatavaid markereid nagu nõrga astme tunnus ($) ja konsonandi pikenemise tunnus (2). Selline eraldamine võimaldaks morfotaktikat kirjeldavatest jätkusõnastikest eemaldada ka seal praegu kahetasemelise reeglite eripäradest tulevad tüveteisendused nagu ne-se jms – see oleks tüveteisenduse mooduli probleem.

Eraldatus võimaldaks kasvõi eksperimendina hüpoteetilise tüveteisenduse mooduli kahetasemelised reeglid asendada muudsorti teisendusreeglitega (nt Karttuneni/Xeroxi asendusreeglid [12]).

3.2.2 Avatud muutüübid, oletamine

Mitmetes rakendustes on vajalik tundmatute sõnade oletamine. Tõenäoliselt kasutatav lahendus on fonoloogiliste mustrite kirjeldamine regulaaravaldistena. Oletaja kvaliteedi hindamiseks võiks võtta eesmärgiks süsteemi sõnastikus olevate sõnade minimeerimise nii, et seniste tekstide analüüsi kvaliteet ei halveneks. Päris ilma sõnastikuta pole täpne morfoloogiline analüüs paraku võimalik, sest just sagedamini kasutatavad sõnad kipuvad muganduma ebareeglipäraseks või siis vastupidi, hoidma visalt kinni ammu mitteproduktiivseks muutunud muutemallidest, mida pole enam otstarbekas avatud süsteemi reeglitena kirjeldada, sest mallile vastavaid sõnu keelde juure ei tule.

3.2.3 Lisamärgendus, spetsialiseeritud teisendajad

Sõnastiku minimeerimise vastu räägib mõte kodeerida sellesse täiendavat, konkreetsete tüvedega seotud infot, mida saaks kasutada liitsõnareeglite kirjeldamisel või süntaksianalüsaatorile väljastamiseks.

Võimalik lisainfo võib olla seotud tüve semantiliste omadustega, kasutussagedusega, käitumisega tuletistes ja liitsõnades jms. Samuti on tüvesid võimalik märkida kasutusvaldkonna või -allkeele järgi. Selliste märgendite järgi kasutatavate tüvede allhulka valides oleks mugav genereerida spetsiifilisteks otstarveteks mõeldud analüsaatoreid. Kas spetsialiseeritud analüsaatoreid ka tegelikult vaja on, pole teada, aga see vääriks eraldi uurimist.

3.2.4 Kõnesünteesiks vajalik info

Kõnesünteesi, eriti olemasolevat kirjalikku teksti ettelugevate süsteemide jaoks oleks kasulik, kui teksti analüüsivad süsteemid väljastaks võimalikult palju fonoloogilist informatsiooni. Sõna häälduslseärasused, välte- ja rõhumuutuse mallid on seotud konkreetsete tüvedega ning mõnes mõttes olekski loogiline vastav informatsioon tüvede juurde kodeerida. Enamgi veel – tegelikult on osa sellest ju ka EKI tüvebaasis olemas ja peamine raskus on lisada juba olemasoleva märgenduse tugi ka hüpoteetilisele tüvemuutusemoodulile või praegustele kahetasemelistele reeglitele. Kõrvalefektina võivad tüvemuutuse reeglid isegi lihtsamaks minna, sest üks või teine morfonoloogilisi muutusi tingiv asjaolu tuleb tüves eraldi välja kirjutada (süvaesitus läheneb "tõelisele" süvaesitusele).

3.2.5 Lemmatiseerija, poolitaja jt

Morfoloogiakirjelduse põhjal saab luua hulga pisirakendusi, mis otseselt ei vaja tuge kõrgema taseme moodulitelt. Huvitav arendusharu oleks nende rakenduste ehitamine lõplike teisendajatena. Selliseks pisirakenduseks kvalifitseeruvad ka mitmesugused lõplike teisendajatena realiseeritud filtrid, mida saaks kasutada näiteks kirjavigadega teksti analüüsiks (filter, mis tüüpilised vead enne analüüsi parandada üritab) jms.

3.2.6 Kaalutud lõplikud automaadid

Kaalutud lõplikud automaadid erinevad kaaludeta automaatidest sellepoolest, et olekusiirded on varustatud ka kaaluga. See võimaldab automaati kodeerida mitmesugust statistilist informatsiooni, mis võib olla kasulik leitud analüüside tõenäose järgi

järjestamiseks, liitsõnareeglite kirjeldamiseks, tuvastamisreeglite jaoks ja paljuks muuks. Seni pole kaalutud lõplike automaatide rakendatavust eesti morfoloogia kirjeldamiseks uuritud.

3.2.7 Veebiliides

Tuleviku-arendus võiks olla ka mitmesugused veebiliidesed. Esimesena tuleb pähe teisendajapõhise morfoloogilise analüüsi ja -sünteesi veebidemo. Samas võib veebiliidest kasutada ka hoopis teistsugustel eesmärkidel, näiteks kogukonnapõhiseks sõnastikuhalduseks. Viimase huvitav näide on soomlaste Joukahainen[1], mida kasutataksegi viimasel ajal vabavaraliste soome keele keeltehnoloogia-rakenduste alussõnastikuna. Kas Eesti tingimustes on otstarbekas sellist süsteemi eraldi üleval pidada või õnnestuks midagi sellist luua nt koostöös EKIga, vääriks edasist uurimist.

[1] `http://joukahainen.puimula.org/`, 01.08.2010

Peatükk 4

Kokkuvõte

Käesolev magistritöö kirjeldab lõplikel automaatidel põhineva eesti keele morfoloogiakirjelduse hetkeseisu ning selle praktiliseks kasutamiseks sobivaks arendamiseks tehtud tegevusi. Töö aluseks on Heli Uibo selleteemalised tööd kuni aastani 2005. Käesoleva töö lähtepunkt oli vajadus märgatavalt laiendada olemasoleva leksikaalse teisendaja (lõplikel teisendajatel põhineva morfoloogia-analüsaatori ja -süntesaatori) sõnastikku, kuid töö käigus muutus ka süsteemi üldine ülesehitus modulaarsemaks, täienesid tuletiste ja liitsõnamoodustamise reeglid. Teisendajate süsteemi arendamisega paralleelselt tekkisid vahendid teisendaja töö hindamiseks ja automaatseks testimiseks.

Kahjuks ei saa öelda, et lõplikel automaatidel põhinev morfoloogiakirjeldus olekski nüüd valmis. Neli kõige olulisemat lahendamist vajavat probleemi on sõnastiku jätkuv parandamine, tuletatud sõnade analüüsiväljundisse korrektse tüve arvutamine, liitsõnareeglite jätkuv testimine ja parandamine ning kirjelduse laiendamine vaid väiketähelistelt sõnadelt ka suurtähti sisaldavatele. Loodetavasti aitab töö materjalide avatus ja töö tekst morfoloogiakirjelduse arendamise kiirusele kaasa.

Töö originaalsetest osadest väärivadki nimetamist:

- vahendid EKI tüvebaasi teisendamiseks Heli Uibo kahetasemeliste reeglitega ja vastava jätkusõnastike süsteemiga sobivale kujule.
- täiendused teisendajate süsteemile ning välja pakutud struktuur, kus liitsõnareeglid on lihtsõnade sõnastikust eraldatud.
- ehitussüsteem leksikaalse teisendaja automaatseks konstrueerimiseks lähteandmetest ning vahendid morfoanalüsaatori väljundi kvaliteedi hindamiseks.

Practical Finite-State Morphology of Estonian

Master Thesis

Jaak Pruulmann-Vengerfeldt

Summary

This thesis describes the work that has been done to refresh, improve and extend the finite-state description of Estonian morphology. The need for such a description is motivated by the fact that there is no morphological analyzer for Estonian that could be easily used in open source projects.

The basis of this work is a system built by Heli Uibo for her master's thesis in 1999 and improved later by her. The main problem seemed to be a really small stem lexicon but during the extension of lexicon also the overall structure of lexical transducer was modularized and various rules for derivation and compounding were added. Several tools for measuring the quality of morphological analysis, for automatic updating of transducers and for automatic testing were built.

However, the lexical transducer that was built for this thesis is not complete either. Four most important unsolved problems are continuing improvement of lexicon, generation of correct lexical stems for derived words, need to improve the compounding rules and the technical inability to analyze words that contain uppercase letters. The underlying open-source principles of this work allow for easier contribution by other parties.

The most important original contributions in this work are:

- tools for transforming the stem database by the Institute of the Estonian Language to the format that is compatible with the two-level rules by Heli Uibo,
- improvements in the derivation rules and a new structure for the lexical transducer that completely separates the compounding rules from main lexicon,
- automated build system and tools for testing and assessing the quality of the analyzer.

Kirjandus

[1] Kenneth R. Beesley ja Lauri Karttunen. *Finite State Morphology.* CSLI Studies in Computational Linguistics. CSLI Publishing, 2003.

[2] Geoffrey Collyer ja Ian F. Darwin. A History of UNIX before Berkeley: UNIX Evolution: 1975-1984. *MicroSystems*, 1984.

[3] Mati Erelt, Tiiu Erelt ja Kirstiina Ross. *Eesti Keele Käsiraamat.* Eesti Keele Sihtasutus, 2000.

[4] Mati Erelt, Reet Kasik, Helle Metslang, Henno Rajandi, Kristina Ross, Henn Saari, Kaja Tael ja Silvi Vare. *Eesti Keele Grammatika I. Morfoloogia, sõnamoodustus.* Teaduste Akadeemia Eesti Keele Instituut, 1995.

[5] Markus Forsberg. Finite State Transducers in Haskell. Magistritöö, Chalmers University of Technology, august 2001.

[6] Paul Haahr ja Steve Baker. Making search better in Catalonia, Estonia, and everywhere else, http://googleblog.blogspot.com/2008/03/making-search-better-in-catalonia.html, 2008. (viimati kontrollitud 1.08.2010).

[7] C. Douglas Johnson. *Formal Aspects of Phonological Description.* Mouton, The Hague, 1972.

[8] Heiki-Jaan Kaalep. An Estonian Morphological Analyser and the Impact of a Corpus on Its Development. *Computers and the Humanities*, 31(2), lk. 115, 1997.

[9] Heiki-Jaan Kaalep. *Eesti keele ressursside loomine ja kasutamine keeletehnoloogilises arendustöös.* Doktoritöö, Tartu Ülikool, 1999.

[10] Heiki-Jaan Kaalep, Ülle Viks, Martin Ehala ja Annika Kilgi. Tänapäeva eesti kirjakeele uurimine. Morfoloogia. *Emakeele Seltsi aastaraamat*, 48, lk. 36–48, 2002.

[11] Ronald M. Kaplan ja Martin Kay. Regular models of phonological rule systems. *Comput. Linguist.*, 20(3), lk. 331–378, 1994.

[12] Lauri Karttunen. The replace operator. *In 33th Annual Meeting of the Association for Computational Linguistics*, lk. 16–23, Morristown, NJ, USA, 1995. Association for Computational Linguistics.

[13] Lauri Karttunen. The proper treatment of optimality in computational phonology: plenary talk. *FSMNLP '09: Proceedings of the International Workshop on Finite State Methods in Natural Language Processing*, lk. 1–12, Morristown, NJ, USA, 1998. Association for Computational Linguistics.

[14] Lauri Karttunen ja Kenneth Beesley. Two-level rule compiler. Technical Report ISTL-92-2. Tehniline aruanne, Xerox Palo Alto Research Center, Palu Alto, CA, 1992.

[15] Lauri Karttunen ja Kenneth R. Beesley. Twenty-Five Years of Finite-State Morphology. *Inquiries into Words, Constraints and Contexts: Festschrift for Kimmo Koskenniemi on his 60th Birthday*, lk. 71–83. CSLI Publications, 2005.

[16] Lauri Karttunen, Ronald M. Kaplan ja Annie Zaenen. Two-level morphology with composition. *Proceedings of the 14th conference on Computational linguistics* -, lk. 141–148, Morristown, NJ, USA, 1992. Association for Computational Linguistics.

[17] Reet Kasik. Tähelepanekuid soome ja eesti keele liitnimisõnadest. *Lähivõrdlusi. Lähivertailuja*, 19, lk. 9, 2010.

[18] Kimmo Koskenniemi. *Two-Level Morphology: A General Computation Model for Word-Form Recognition and Production.* Doktoritöö, University of Helsinki, Department of General Linguistics, 1983.

[19] Kimmo Koskenniemi. Compilation of automata from morphological two-level rules. Fred Karlsson, toimetaja, *Papers from the Fifth Scandinavian Conference of Computational Linguistics, Helsinki, December 11–12, 1985*, lk. 143–149. University of Helsinki, Department of General Linguistics., 1986.

[20] Franklin Mark Liang. *Word hy-phen-a-tion by com-put-er (hyphenation, computer).* Doktoritöö, Stanford, CA, USA, 1983.

[21] Krister Lindén, Miikka Silfverberg ja Tommi Pirinen. *State of the Art in Computational Morphology*, köide 41 / *Communications in Computer and*

Information Science, ptk. HFST Tools for Morphology – An Efficient Open-Source Package for Construction of Morphological Analyzers, lk. 28. Springer, 2009.

[22] Einar Meister, Tiit Roosmaa ja Jaak Vilo. Estonian language technology Anno 2009. Rickard Domeij, Kimmo Koskenniemi, Steven Krauwer, Bente Maegaard, Eiríkur Rögnvaldsson ja Koenraad deŠmedt, toimetajad, *Proceedings of the NODALIDA 2009 workshop Nordic Perspectives on the CLARIN Infrastructure of Language Resources*, lk. 21–26. Northern European Association for Language Technology (NEALT), 2009.

[23] Kaili Müürisep. *Eesti keele arvutigrammatika: süntaks.* Doktoritöö, Tartu Ülikooli Arvutiteaduse Instituut, 2000.

[24] Kaili Müürisep. Eesti keele süntaktiliselt märgendatud korpuse märgendusest, jaanuar 2008.

[25] László Németh. Automatic non-standard hyphenation in OpenOffice.org. *TUGboat*, 27(2), lk. 750–755, 2006.

[26] Jaak Pruulmann. Vabavaraline eesti keele speller. Bakalaureusetöö, 2003.

[27] Brian Roark ja Richard Sproat. *Computational Approaches to Morphology and Syntax*. Oxford Surveys in Syntax and Morphology. Oxford University Press, 2007.

[28] Enn Saar. Re: TeX-i poolitusfaili ajaloost. e-kiri Ain Vagulale, 9. aprill 2004.

[29] Eveli Saue. Morfoloogiliste analüsaatorite etmrf ja EKI võrdlus, 2007. http://lepo.it.da.ut.ee/h̃kaalep/arvutimorf_09/EKI_FS_vordlus_Saue.pdf.

[30] Marcel Paul Schützenberger. A Remark on Finite Transducers. *Information and Control*, 4(2-3), lk. 185–196, 1961.

[31] Trond Trosterud. A constraint grammar for Faroese. Eckhard Bick, Kristin Hagen, Kaili Müürisep ja Trond Trosterud, toimetajad, *Proceedings of the NODALIDA 2009 workshop Constraint Grammar and robust parsing*, köide 8, lk. 1–7, Tartu, Estonia, 2009. Tartu University Library.

[32] Heli Uibo. Eesti keele sõnavormide arvutianalüüs ja -süntees kahetasemelist morfoloogiamudelit rakendades. Magistritöö, Tartu Ülikooli Arvutiteaduse Instituut, 1999.

[33] Heli Uibo. Eesti keele morfoloogia modelleerimisest lõplike muundurite abil. M. Koit, R. Pajusalu ja H. Õim, toimetajad, *Keel ja arvuti*, lk. 13–35. Tartu Ülikooli Kirjastus, 2006.

[34] Ülle Viks. *Väike Vormisõnastik*. Keele ja Kirjanduse Instituut, Tallinn, 1992.

[35] Ülle Viks. Eesti keele avatud morfoloogiamudel. Tiit Hennoste, toimetaja, *Arvutuslingvistikalt inimesele. Tartu Ülikooli üldkeeleteaduse õppetooli toimetised 1*, lk. 9–36, Tartu, 2000. Tartu Ülikooli kirjastus.

[36] Jaak Vilo. EKKTT Ülevaade. Ettekanne riikliku programmi "Eesti Keele Keeletehnoloogiline Tugi"konverentsil , 11 2007.