TARTU ÜLIKOOL

MATEMAATIKA-INFORMAATIKATEADUSKOND

Arvutiteaduse instituut

Informaatika õppekava

**Annelii Halling**

Eesti keele keeleressursse kasutav mäng käänete õppimiseks.

**Bakalaureusetöö (9 EAP)**

Juhendaja: Sven Aller

Tartu 2016

**Lühikokkuvõte**

**Võtmesõnad:** Python, PHP, XML, morfoloogia, korpus, õppemäng, eesti keel

**Sissejuhatus**

**Tarkvara**

Python 3.4

Filosofti morfoloogiatarkvara Pythoni liides (https://github.com/estnltk/pyvabamorf)

<https://github.com/tpetmanson/pyvabamorf/blob/master/dist/pyvabamorf-1.6.win-amd64-py3.4.msi>

**Eesti keele käänded ja teiste keelte käänded**

Käänata saab kõiki käändsõnu: nimi-, omadus-, arv- ja asesõnu. Eesti keeles on 14 käänet nii ainususes kui mitmuses. Peakäänded on: ainsuse nimetav, ainsuse ja mitmuse omastav, ainususe ja mitmuse osastav ning ainsuse ja mitmuse sisseütlev. („Õigekeelsuse käsiraamat ja sõnastik“, 2006)

**Noomenikategooriate lühendid** (http://www.filosoft.ee/html\_morf\_et/morfoutinfo.html)

|  |  |  |
| --- | --- | --- |
| ab | abessiiv | ilmaütlev |
| abl | ablatiiv | alaltütlev |
| ad | adessiiv | alalütlev |
| adt | aditiiv | suunduv (lühike sisseütlev), nt. "majja" |
| all | allatiiv | alaleütlev |
| el | elatiiv | seestütlev |
| es | essiiv | olev |
| g | genitiiv | omastav |
| ill | illatiiv | sisseütlev |
| in | inessiiv | seesütlev |
| kom | komitatiiv | kaasaütlev |
| n | nominatiiv | nimetav |
| p | partitiiv | osastav |
| pl | pluural | mitmus |
| sg | ainsus | ainsus |
| ter | terminatiiv | rajav |
| tr | translatiiv | saav |

**Analoogilised programmid**

Eesti keelsed:

<https://sahver.wikispaces.com/K%C3%A4%C3%A4nded.+5.+klass.+Testid.>

Laused ei muutu. Programm ei kasuta morfoloogia analüsaatorit vaid on kindlad laused ja vastused. Mäng pole dünaamiline.

Muukeelsed:

TODO

**Morfoloogia analüsaator**

Programm kasutab morfoloogilise analüsaatori Pythoni liidest. Morfoloogilise analüsaatori vastest kasutab programm infot: „form“- sõna kääne ning kas sõna on ainsuses või mitmuses, „partofspeech“- sõnaliik on vajalik, lause struktuurilise ülesehituse kontrollimiseks ja „root\_tokens“, mis on vajalik sõna algvormi saamiseks.

(https://github.com/tpetmanson/pyvabamorf)

**Korpus**

Lauseid sorteeritakse välja ilukirjanudkorpusest. Selles allkorpuses on Eesti algupärased ilukirjandustekstid, kus on kokku ca 5,8 miljonit sõna. Valdav enamus tekstidest on avaldatud pärast 1990. aastat. Kasutatud on alla laetud XML TEI P5 (UTF-8) kuju. Failid on märgendatud TEI põhimõtete järgi (<http://www.tei-c.org/Guidelines/>).

<http://www.cl.ut.ee/korpused/segakorpus/eesti_ilukirjandus_1990/index.php?lang=et>

**Korpuse töötlemine**

Programm valib korpusest eeltöötlusega sobivad laused välja ning esitab juhuslikult valides neid kasutajale, sealjuures arvestades raskustasemeid.

Lausete sorteerimine toimub Pythonis, kus valitakse välja laused, mis on sobiva pikkusega. Lause sõnade hulk jääb 3-9 vahele (*pikkus esialgu 9, kuna Elve Voltein pakkus välja, otseselt kuskilt kirjandusest sobivat pikkus ei ole veel leidnud*). Lause kõik sõnad peavad olema üheselt määratud.

**Sõna on üheselt määratud** kui morfoloogiline analüsaator annab vasteks ainult ühe võimaluse. Näiteks sõna „puudesse“ vasteks on:

[{'analysis': [{'clitic': '',

'ending': 'desse',

'form': 'pl ill',

'lemma': 'puu',

'partofspeech': 'S',

'root': 'p<uu',

'root\_tokens': ['puu']},

{'clitic': '',

'ending': 'sse',

'form': 'sg ill',

'lemma': 'puue',

'partofspeech': 'S',

'root': 'puue',

'root\_tokens': ['puue']}],

'text': 'puudesse'}]

Sõna „puudesse“ võib tuleneda nii sõnast „puue“ kui ka sõnast „puu“, seega sisu arvestamata pole võimalik kindlaks määrata, kas sõna on ainusese sisseütlevas või mitmuse sisseütlevas käändes.

Seega sõnad, mis pole üheselt määratud pole võimalik kasutada otsitava käändsõnana.

Teiseks näiteks on sõna „sai“, mille morfoloogilise analüsaatori tulemus on :

[{'analysis': [{'clitic': '',

'ending': 'i',

'form': 's',

'lemma': 'saama',

'partofspeech': 'V',

'root': 'saa',

'root\_tokens': ['saa']},

{'clitic': '',

'ending': '0',

'form': 'sg n',

'lemma': 'sai',

'partofspeech': 'S',

'root': 'sai',

'root\_tokens': ['sai']}],

'text': 'sai'}]

Sõna „sai“ võib olla nii tegusõna kui ka nimisõna. Kuna esimeses raskustasemes kontrollitakse lause struktuuri, siis selleks on vajalik sõna üheselt määratavus.

**Algoritm**

**PYTHON**

* Programm läbib kõik korpuses olevad laused
  + Kontroll, et lause ei sisaldaks sobimatuid sõnu. TODO
  + Leian lause pikkuse (kui liiga pikk või lühike( 2-10 sõna) siis ei sobi)
  + Kontroll, et lause oleks üheselt määratud.
  + Kontroll, et lause sobiks esimesse või teisest raskusastmesse.
    - Esimene raskustase:
      * Koosneb kolme- ja neljasõnalistest lausetest.
      * Lause peab vastama lubatud struktuurile
        + Struktuuri leidmine:

Kõige populaarsemad lausestruktuurid ilukirjanduskorpuses.

* + - Teine raskustase:
      * Lause koosneb viiest kuni üheksast sõnast (esialgu).
      * Lause peab sisaldama vähemalt ühe tegusõna ning ühte nimisõna.
  + Leian lauses olevate sõnade morfoloogilise analüüsi vastused.
    - Kontroll, kas sõna on liitsõna ( TODO- kui sõna koosneb rohkem kui kahest sõnast)
    - Kontroll kas sõna on, käändes.
    - Kontroll, kas sõna on lühike sisseütlev, mis on ainsana teistsuguse ülesehitusega. Kui kõikidel teistel on määratud ära ainus või mitmus ja kääne (näiteks 'form': 'sg abl'), siis lühike sisseütlev on määratud ära  'form': 'adt'.
    - Kontroll, kas lause kuulub esimesse või teise raskustasemesse.
    - Määrata lausele id.
    - Kontroll, kas sõnal leidub rohkem kui üks vastus antud käändes.
      * Leian morfoloogilise süntesaatoriga, kõik sobivad vastused
    - Salvestan info (lause teksti, nimisõna, käände info, vastuse) xml faili. Iga ilukirjandus korpusest läbitud failile luuakse, eraldi töödeldud käänete XML fail.
      * XML faili ülesehitus:
        + <info id =“0“>
        + <s>Ärkasin %%% üles ja läksin sööma .</s>
        + <nr>ainsuse</nr>
        + <case>alalütlev</case>
        + <n>hommik</n>
        + <answer>hommikul</answer>
        + </info>
  + Valitakse järgmine lause, kuni korpus on läbitud.

**PHP**

* Programm kuvab juhusliku lause vastavalt valitud raskustasemele.
* Lause kohale kuvatakse sõna nimetavas käändes, kääne ja ainsus või mitmus.
* TODO

**Struktuuripuu loomine:**

*Üheks võimaluseks on luua sõnastik - {'P':{'S':{'V':{'S':True}},'D':{'A':{'S':True}}}}*

*Sellisel viisil on lihtne sooritada kontrolli, kui sõnastik[’P’][’S’][’V’][’S’] päringu vastuseks on „True“, siis lause sobib.* *TODO- TULEB VEEL UURIDA!*

**Pythoni koodi kommentaar:**

**Sõnastik case\_dict** koosneb noomenkategooriate lühenditest, mille võtmeks on lühend ja väärtuseks on lühendi tähendus.

**List inappropriateWords** koosneb sõnadest, mis ei tohi lauses esineda. Nendest sõnadest on aru saada, et lause sisu on ebasobilik kasutajale.

**Funktsioon xml\_formatting** formaadib loodava XML faili elemendid, et oleks võimalik paremini lugeda antud XML faili.

**Funktsioon structureCompatibilityLevel1** määrab ära 1. raskustaseme.

Esimesse raskustasemesse kuuluvad 3-4 sõnalised laused, mis sisaldab tegusõna. Samuti toimub kontroll, kas lause struktuur on sobilik.

**Funktsioon structureCompatibilityLevel2** määrab ära 2. raskustaseme, mis kontrollib, kas lause sisaldab tegusõna ja nimisõna.

**Funktsioon getPartOfSpeech** kontrollib, et kõik lauses olevad sõnad oleksid üheselt määratud, ning tagastab listi lause struktuurist. Antud funktsiooni abil on võimalik kontrollida kas lause struktuur on sobilik ning kas lause sisaldab vajalikke lauseliikmeid.

**Funktsioon listtostring** teeb listist sõne.

**Funktsioon getBestPOSCombination** käib läbi kogu korpuse salvestades kõikide 4 (esialgu) sõnaliste lausete struktuuri sõnena sõnastikku lisades väärtuseks struktuuri esinemise arvu korpuses.

**Funktsioon combintotree** võtab funktsiooni getBestPOSCombination tagastatud sõnastiku, ning moodustab sellest puustruktuuri, et oleks võimalik efektiivsemalt kontrollida, kas lause sobib esimesse raskustasemesse. TODO

**Funktsioon runCaseAnalys** käib läbi kogu korpuse ning kui lause kuulub esimesse või teisse raskustasemesse, siis toimub lause kõigisõnade läbivaatus. Kui leitakse sobiv käändsõna (sõna ei ole nimetavas käändes), siis kontrollitakse kas sõna on liitsõna.

Edasi kontrollitaks, kas sõna on lühikeses sisseütlevas käändes, kuna morfoloogia analüsaator annab lühikese sisseütleva käände vastuseks „adt“ ilma ainsust „sg“ määramata. Seega kui sõna on lühikeses sisseütlevas käändes, siis määratakse ainsus eraldi ära.

Järgmisena moodustatakse ära muutujad, mis lisatakse uude loodavasse korpuse XML faili. Sõnes **sen\_x** asendatakse otsitav sõna kolme protsendi märgiga( „%%%“). **Sõne casename** on käände nimetus. **Sõne word** on otsitav sõna, mis on mängus vastuseks, mida võib mitu olla. **Sõne nominatiive** on sõna nimetavas käändes.

**Kasutajalood**

Mängu õige sõna sisestamine:

* Kasutaja vajutab nuppu “Alusta”.
* Kasutajale kuvatakse lause, kus on ühe käändes oleva sõna asemel lahter ning lause kohal on antud puuduv käändsõna nimetavas käändes ning õige käände nimi.
* Kasutaja sisestab õige sõna õiges käändes ning vajutab edasi noolt.
* Õigete vastuste lugeja lisab ühe õige vastuse ja kasutajale tuleb uus lause ette.

Mängu vale sõna sisestamine:

* Kasutaja vajutab nuppu “Alusta”.
* Kasutajale kuvatakse lause, kus on ühe käändes oleva sõna asemel lahter ning lause kohal on antud puuduv käändsõna nimetavas käändes ning õige käände nimi. Kasutaja sisestab õige sõna vales käändes ning vajutab edasi noolt.
* Õigete vastuste lugeja ei lisa vastuse eest puntki ning kasutajale antakse teada, et tema esitatud vastus oli vale ning näidatakse õiget vastust
* Seejärel kuvatakse kasutajale uus lause.

Mängu lõppemine:

* Pärast viimasele lausele vastamist antakse kasutajale teada mitu protsenti oli õigeid vastuseid.

**Probleemid**

* “Mina” omastav: mina, minu , mu

Süntesaatoriga kontrollides tuleb vastuseks mina, kuid lause „Tere tulemast minu korterisse“. Sobib vasteks vaid mu või minu.

**Edasiarendamise võimalused**

* Populaarsemad sõnad lihtsamasse raskusastmesse.
* Lisada raskusastmeid, kuna praegu on teine raskustase liiga üldine.
* Morfoloogilise analüsaatori vastuse põhjalikum analüüs.

**Kokkuvõte**

**Kirjandus**

<https://keeleressursid.ee/et/keeleressursid>

<http://www.tei-c.org/Guidelines/>

<http://www.cl.ut.ee/korpused/segakorpus/eesti_ilukirjandus_1990/index.php?lang=et>

“Eesti keele struktuud”

“Õigekeele käsiraamat ja sõnastik”

“Sõnavorm ja kõne”

“Grammar from the Humane Perspective”

*MUU:*

*Lehelt:* [*http://www.cs.ut.ee/sites/default/files/2014/l6put88d/Soovitusi-2014.pdf*](http://www.cs.ut.ee/sites/default/files/2014/l6put88d/Soovitusi-2014.pdf)

*3.2 Rakendusliku ülesande lahendus*

*Valminud töö on tarkvaralahendus püstitatud eesmärkide saavutamiseks. Sellise töö puhul on oluliseks komponendiks tulemuse analüüs, mis peab andma muuhulgas vastused järgmises loetelus toodud küsimustele.*

*• Millised on teised sarnased realisatsioonid?*

*• Mille poolest on autori oma olemasolevatest parem?*

*• Kas tarkvara töötas? Mis meetoditega kvaliteet tagati?*

*• Kui hästi see töötas? Millised on mõõdetavad jõudlustulemused?*

*Insener-tehnilised valikud peavad olema dokumenteeritud. Tööst peab olema selge, miks kasutati just valitud tehnoloogiaid, millised olid alternatiivid ja kui kulukaks tehtud valikud läksid. Töös peab olema kirjeldatud ka lahenduse arhitektuur. Programmeerimisalaste tööde puhul on oluline järgida veel tarkvara dokumenteerimisnõudeid. Oluline on lisada ka valminud tarkvara nõuded arvuti riist- ja tarkvarale, installeerimis- ja kasutamisjuhend.*