

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond
Tarkvarateaduse instituut

Siim Düüna 155619IAPB

**NÕRGAL UNIFITSEERIMISEL JA
SEMANTILISEL VÕRGUL PÕHINEV
INIMENE-MASIN DIALOOGISÜSTEEM**

Bakalaureusetöö

Juhendaja: Jüri Vain
PhD

Tallinn 2018

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autor: Siim Düüna

21.05.2018

Annotatsioon

Töö eesmärgiks on luua sotsiaalse käitumisega suhtlusrobot, mis suudaks teksti vahendusel kasutajaga vestlust pidada. Töö tulemus võiks tulevikus kasulik olla inimestele üksinduse ja igavuse peletamiseks.

Bakalaureusetöö käigus luuakse Prologi keskkonnas dialoogisüsteem, mis suudab teadmusbaasist leida nõrga unifitseerimise ja semantilise võrgu alusel kasutaja sisestatud lausele sobiliku vastusfraasi. Loodav süsteem suhtleb kasutajatega eesti keeles ja on võimeline ka vestluse käigus õppima uusi teadmisi.

Tulemuseks olev juturobot suudab vastata inimese küsimustele ja leiab vastusfraasi ka teistele lauseliikidele. Süsteem on robustne, mis tähendab, et see suudab aru saada sünonüümidest, erinevatest sõnavormidest ning oskab ignoreerida mittevajalikke täitesõnu.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 25 leheküljel, 5 peatükki, 9 joonist, 0 tabelit.

Abstract

Human-machine dialogue system based on weak unification and semantic network

The purpose of this thesis is to implement a dialogue system that could communicate with user in writing. System takes in a human written phrase, analyses it and forms a suitable response. This chatbot can be useful for people who are lonely and do not have anyone else available to speak to.

Dialogue system is implemented in Prolog and it works with phrases written in Estonian. System uses weak unification to match human input with its knowledge base. Semantic network is necessary to find how words are related to each other and it helps to reduce the size of the knowledge base. To find an answer to human input, program uses weak unification and forward and backward chaining.

Chatbot is also capable of learning from conversations. It can learn unfamiliar words and new sentences that are saved in knowledge base. Learning is important because it makes teaching the system easier and much faster. Word forms are generated using Python script and EstNLTK library.

The result of this thesis is working dialogue system that can conduct a social conversation with a human. The system can answer questions and reply to other types of sentences. Chatbot is designed to be robust: it can understand different word forms, synonyms and it can ignore fillers.

The thesis is in Estonian and contains 25 pages of text, 5 chapters, 9 figures, 0 tables.

Lühendite ja mõistete sõnastik

ASR	<i>Automatic Speech Recognizer</i> , automaatne kõnetuvastus
NLU	<i>Natural Language understanding Unit</i> , loomuliku keele analüsaator
TSE	<i>Text-to-speech engine</i> , kõnesüntesaator
AIML	<i>Artificial Intelligence Markup Language</i> , tehisintellekti märgistuskeel
XML	<i>Extensible Markup Language</i> , üldotstarbeline märgistuskeel info jagamiseks

Sisukord

1 Sissejuhatus	9
1.1 Motivatsioon.....	9
1.2 Sarnased lahendused.....	9
1.3 Metoodika.....	11
1.4 Töö eesmärk	11
2 Teoreetilised alused	12
2.1 Dialoogisüsteemi komponendid	12
2.2 Teadmusbass	13
2.3 Loogiline tuletamine.....	14
2.4 Kontekstid.....	15
3 Dialoogisüsteemi loomine	17
3.1 Teadmusbass	17
3.2 Semantiline võrk.....	17
3.3 Eesti keele grammatika.....	18
3.4 Vastuse genereerimine.....	20
3.4.1 Edaspidine ja tagurpidine tuletamine	20
3.4.2 Semantiline võrk.....	22
3.4.3 Küsimustele vastamine	22
3.4.4 Tagavaralused	23
3.5 Uute sõnade lisamine.....	23
3.5.1 Sõnavormide leidmine.....	24
3.5.2 Sõna tundma õppimine	24
3.6 Peapredikaat.....	25
4 Tulemuse valideerimine	27
4.1 Turingi test.....	27
4.2 Näidisdialogi hindamine.....	28
4.3 Dialoogisüsteemi võrdlus sarnaste lahendustega	29
5 Kokkuvõte	32
5.1 Edasine töö	32

Kasutatud kirjandus 34

Jooniste loetelu

Joonis 1. Inimene-masin dialoogisüsteemi arhitektuur.	12
Joonis 2. Näide semantilise võrgu kohta.	18
Joonis 3. Programmikood tagurpidiseks vastuse tuletamiseks.....	21
Joonis 4. Tagavaralause te faktid.....	23
Joonis 5. Programmikood tagavaralause valimiseks.	23
Joonis 6. Dialoogisüsteemi peapredikaat.....	26
Joonis 7. Näidisdialoog.	28
Joonis 8. Tele2 juturobot Juta.....	30
Joonis 9. Vestlus robotiga Cleverbot.....	31

1 Sissejuhatus

1.1 Motivatsioon

Inimesed on loodud sotsiaalseteks olenditeks, seetõttu on suhtlemine elu lahutamatu osa. Kahjuks ei ole aga alati võimalus leida endale vestluskaaslast. See probleem on eriti suur just vanemate inimeste seas, kes on jäänud üksikuks. Kuigi tänapäeval on olemas telefonid ja ka internet, mille abil iga vahemaa tagant suhelda, ei pruugi lähedastel suhtlemiseks aega leiduda või puuduvad sõbrad sootuks. Sellises olukorras oleks suhtlusrobot väga heaks lahenduseks, sest robotiga saaks vestelda igal ajal ja kasvõi ööpäev läbi.

Aga peale eelneva leidub suhtlusrobotil veel teisigi kasutusi. Suhtlusroboteid saaks kasutada ka lastele sõnavara õpetamiseks. Vanematel endal pole kõigeaega ja tihtipeale ei ole ka peresisene sõnavara piisavalt lai. Sellisel juhul oleks samuti kasulik robot, kes räägiks lapsega ning õpetaks talle vestluse käigus uusi sõnu. Samal ajal oleks see robot ka mõnes mõttes mängukaaslase eest ja hoiaks last tegevuses.

Juturoboteid võib leida ka juba internetilehekülgedelt, kus need asendavad päris klienditeenindajaid. Kuna paljudel inimestel tekivad tõenäoliselt samad küsimused, siis on lihtsam teha robot, mis tüüpiküsimustele ise vastab ja ainult keerulisemad neist inimesele edastab. Kokkuvõttes saab infoteenust pakkuva firma nii raha kokku hoida, kuna on vaja vähem töötajad. Ka kliendil on võimalik lihtsatele küsimustele roboti käest saada vastus palju kiiremini, kui oodata, et inimene vastaks.

1.2 Sarnased lahendused

Kuna vajadus on olemas, siis on praeguseks loodud juba päris palju erinevaid eesmärged täitvaid suhtlusroboteid. Kahjuks või õnneks ei ole veel üksi loodud robot piisavalt hea, et suudaks inimest täielikult asendada, aga areng selles vallas on kiire.

Enamik praegu kasutatavatest robotitest on mõeldud just konkreetse valdkonna jaoks. Need on kasutusel näiteks mõne ettevõtte veebileheküljel. Kui mõnel kliendil tekib mingi küsimus, siis on võimalik esitada see robotile ja nii saab vastuse kohe kätte. Kui robot ei oska küsimusele vastata, edastatakse küsimus antud teema valdkonna spetsialistile. Nii aitab robot pakkuda inimestele igal ajal kliendituge, kuid samas vähendab ka ettevõtte kulusid, kuna pole vaja enam nii palju töötajaid. Näitena saab tuua eestikeelse juturoboti Juta, mis on loodud firma Tele2 poolt ja vastab Facebooki vahendusel klientide küsimustele [1]. Kasutades klientidelt saadud tagasisidet, arendatakse seda robotit pidevalt edasi, võimaldades robotil vastata üha keerulisematele küsimustele. Juta on üks vähestest eestikeelsetest juturobotitest, millega saab igäüks suhelda. Klienditeenidusalaseid roboteid on katsetanud ka teised firmad, nagu SEB [2] ja Telia [3].

Olemas on ka üldisemaid juturoboteid, mis püüavad inimese suhtlust jäljendada. Sellised robotid ei suhtle aga veel eesti keeles, vaid enamasti inglise keeles. Pikaajaliseks eesmärgiks on luua robot, mis oleks vestluses inimesest eristamatu. Selliste robotite vahel on juba palju aastaid korraldatud ka erinevaid võistlusi, millest kuulsam on Loebneri auhinnavõistlus [4]. Aastatel 2016 ja 2017 võitis selle robot Mitsuku [5], millega saab igäüks ka ise suhelda näiteks Facebooki, Skype'i või veebilehe vahendusel [6]. See robot on kirjutatud keeles AIML (*Artificial Intelligence Markup Language*) [7], mis on XML-i dialekt ja mis arendati Dr. Richard Wallace'i poolt ning ongi mõeldud juturobotitele teadmiste sisestamiseks [8].

Tuntud on ka internetis leiduv juturobot Cleverbot, mille programmeeris Rollo Carpenter [9]. See robot on pidevas arengus, kuna kasutab kasutajate öeldut õppimiseks [10]. Lihtsustatuna öeldes vastab robot seda, mida inimesed on varem samale küsimusele või lausele vastanud. Sellise õppimise tõttu oskab Cleverbot ka paljusid erinevaid keeli ja suudab suhelda ka eesti keeles. Kuna aga väiksemate keelte puhul on ka vähem kõnelejaid ja seega ka vähem robotiga suhtlejaid, siis on roboti suhtlus eesti keeles algeline ja tihti on roboti vastus seotetu inimese poolt öelduga. Aga üldiselt peaks õppimise idee kohaselt ajapikku ja suure hulga vestluste abil kujunema väga hea juturobot eeldusel, et inimeste sisestatud vastused on otstarbekad. Kahjuks on seda internetis aga keeruline kontrollida. Näitena saab tuua roboti Tay, mis oli loodud Microsofti poolt ja mis suhtles Twitteri vahendusel ning õppis samuti enda kasutajatelt [11]. Robot pidi jäljendama tavalist 19-aastast naist, aga vähem kui 24 tunni jooksul hakkas Tay propageerima natsiideoloogiat

ja teisi Twitteri kasutajaid ahistama. See toob välja probleemi, et robot ei saa tegelikult ise aru, millest ta räägib ja pahatahtlikud inimesed suudavad robotile õpetada ka rumalusi.

1.3 Metoodika

Lõputöö käigus luuakse dialoogisüsteem, mis on võimeline tekstipõhiselt inimesega vestlema. Suhtlemine toimub teadmusbasis olevate teadmiste alusel ning programm on võimeline ka kasutajalt õppima lauseid ja uusi sõnu. Luuakse otsustusmootor eesti keele fraaside interpreteerimiseks ja vastusfraaside genereerimiseks. Fraaside interpreteerimine toimub sisendfraaside unifitseerimisel teadmusbasis olevate fraaside ja semantilise võrgu baasil. Vastusfraasi genereerimiseks tuleb konstrueerida reeglid, mis tagastavad samal semantilisel võrgul defineeritud fraasi, mis on semantiliselt lähim eelnevalt sisestatud fraasi suhtes. Nii peaks suutma masin genereerida vastuse, mis oleks seotud jutu konteksti ja inimese öelduga. Süsteemi loomiseks kasutatakse programmeerimiskeelt Prolog ja keskkonnana on kasutusel SWI-Prolog [12].

1.4 Töö eesmärk

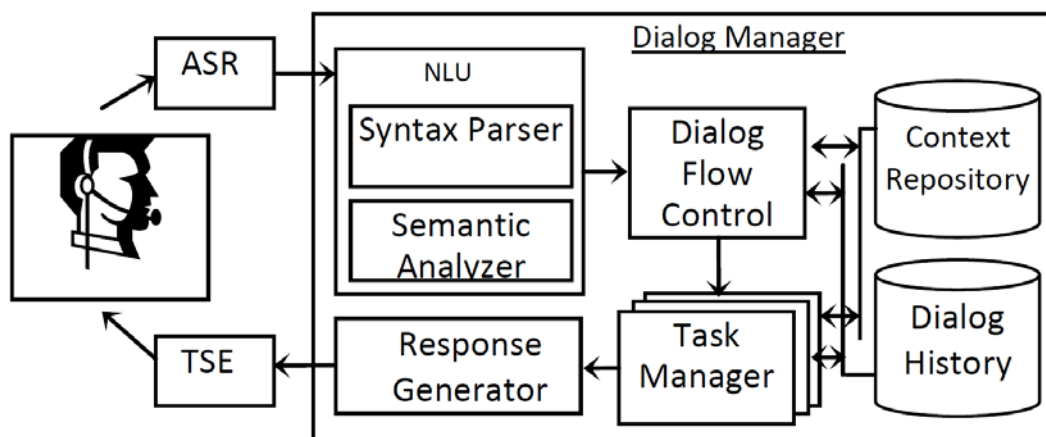
Käesoleva lõputöö eesmärgiks on luua suhtlusroboti tarkvara, mis suhtleks kasutajaga eesti keeles. Valmima peab sotsiaalse käitumisega robot, mis räägiks inimesega eelnevalt õpitud teadmusbasisi põhjal. Robot peab oskama vastata küsimustele ja jätkata juttu ka kasutaja sisestatud jutustavate lausete alusel. Lisaks peab robot ka kasutaja öeldud lauseid ja uusi sõnu meelde jätma ning nende abil oma teadmusbasisi täiendama.

2 Teoreetilised alused

Lõputöös loodava dialoogisüsteemi struktuur põhineb Jüri Vainu ja Helena Sarapuu artiklil „Towards context-sensitive dialogue with robot companion“ [13]. Selles peatükis kirjeldatakse loodava dialoogisüsteemi üldisi põhimõtteid mainitud artikli põhjal.

2.1 Dialoogisüsteemi komponendid

Dialoogisüsteemi põhikomponendid on välja toodud joonisel 1. Süsteemi tööpõhimõtte põhietapid on järgnevad:



Joonis 1. Inimene-masin dialoogisüsteemi arhitektuur.

1. Inimese kõne teisendamine tekstiks, et programm oleks võimeline seda analüüsima. Selleks saab kasutada kõnetuvastust (ASR - *Automated Speech Recognizer*). Samas on võimalik ka sisend saada käekirjatuvastuse abil või siis sisestada tekst muul viisil arvutisse.
2. Teksti analüüsimine loomuliku keele analüsaatori poolt (NLU - *Natural Language understanding Unit*), mis sisaldab nii süntaktilist (*Syntax Parser*) kui ka semantilist analüüsimist (*Semantic Analyser*).

3. Dialoogi vookontroll (*Dialog Flow Control*) ehk osa, mis tegeleb üldiselt dialoogi käiguga. Sinna hulka kuulub vestluse ajaloo hoidmine (*Dialog History*) ja siseolekute muutmine.
4. Tegumihaldur (*Task Manager*), mis tegeleb vastavalt kontekstile (*Context Repository*) õigete reeglikogumite haldamisega (uue reeglikogu laadimine konteksti vahetuse korral ja aegunud konteksti reeglite eemaldamine töömälust).
5. Väljundi ehk vastuse genereerimine kasutades vastuse moodustajat (*Response Generator*). Valitud reeglite abil moodustatakse juturoboti vastus.
6. Tekstilise vastuse teisendamine heliks kõnesüntesaatori abil (TSE - *text-to-speech engine*).

Dialoogisüsteemile on esitatud mõningad eeldused. Kõigepealt on olemas üks robot ja üks inimene, kes vestlevad kordamööda. Kumbki ei tohi üksteist rääkimise ajal segada. Eeldame ka, et inimene ütleb ainult ühe lause korraga, kuid robot võib vastusena öelda ka mitu lauset.

2.2 Teadmusbass

Dialoogisüsteem on ehitatud loogilise programmeerimise põhimõtteid kasutades. Kasutusel on faktid, mida robot teab, ning reeglid, mille alusel robot järeldusi teeb. Reeglid on Horni lausete kujul olevad väited $A \leftarrow B_1, B_2, \dots, B_n$, kus $n > 0$. See tähendab, et A on järeldus, mida tehakse, kui eeldused B_1 kuni B_n on kõik õiged. Erijuhul, kui $n=0$ ehk eeldused puuduvad, nimetatakse Horni lauset faktiks. See tähendab, et väide A on igal juhul tõene. Kui lausel on olemas ainult eeldused, siis on tegemist päringuga. Päring on seega Horni lause, millele otsitakse eelduste põhjal järeldust.

Kuna loomuliku keele fraaside interpreteerimiseks ja teadmiste tuletamiseks oleks vaja väga palju konkreetseid reegleid, siis tuletusmehhanismi üldistamiseks on kasutusel teadmusbassi jaoks üldised *knowledge* („/“ tähistab argumentide arvu) laused, et kõiki teadmisi ühel kujul hoida. Lausete kuju on järgmine:

$knowledge(A_n, CL_n, A_{inx}, CL_{inx}, A, CL, C_{txtL})$.

kus

A_n - Horni lause järelalus algvormis;

CL_n - Horni lause eeldused algvormis;

A_{inx} - järelaluse grammatilise vormi numbrid;

CL_{inx} - eelduste grammatiliste vormide numbrid;

A - esialgsel kujul järelalus;

CL - esialgsel kujul eeldused;

C_{txtL} - kontekstinumbrid, kus lause on sobilik;

Näitena:

```
knowledge([võima,nägema,vikerkaar],[[vihm,sadama],[päike,paistma]],[5,2,3],[[2,4],[1,5]],[võib,näha,vikerkaart],[[vihma,sajab],[päike,paistab]],[0]).
```

Sellele faktile vastab loomuliku keele fraas „Kui vihma sajab ja päike paistab, siis võib näha vikerkaart“.

2.3 Loogiline tuletamine

Loogikareeglite rakendamise suuna järgi eristatakse kahte tüüpi tuletusmehhanisme: edaspidine ja tagurpidine tuletamine (*forward* ja *backward chaining*). Edaspidine tuletamine püüab teadaolevaid fakte kokku viia reeglite eeldustega ja nii tuletada uusi fakte. Teisisõnu öeldes, et kui eeldused on tõesed, siis on ka tuletusreegli põhjal saadud järelalus tõene. Kui vihma sajab ja päike paistab, siis saame tuletada, et võib näha vikerkaart. Tagurpidine tuletamine on vastupidine: teades, et järelalus on tõene, saab tuletada, et järelikult peavad kehtima ka eeldused. Kui teame, et õues võib näha vikerkaart, siis saame teada, et vihma sajab ja päike paistab.

Dialoogisüsteemis on kasutusel mõlemat tüüpi tuletusreeglid. Edaspidise tuletuse abil saab vastused küsimustele „Mis saab kui ...?“ ja „Mis on tagajärg, kui ...?“, kus „...“ tähistab eeldusi, mis on tõesed. Näiteks: „Mis saab kui vihma sajab ja päike paistab?“.

Tagurpidine tuletamine annab vastused küsimustele „Miks ...?“ ja „Mis põhjusel ...?“. Näiteks: „Miks võib näha vikerkaart?“.

Küsimusele vastamiseks tuleb rakendada eelnevalt mainitud tuletusreegleid. Kasutaja sisestatud lauset saame võrrelda teadmusbasis leiduvate *knowledge* faktidega. Kui mõnel faktil on täpselt samasugune järeldus või samasugune eeldus, siis vastav fakt unifitseerub ehk sobib kasutaja öeldud lausega. Kahjuks on aga tavalise unifitseerimise kasutamine ebaotstarbekas, kuna sellisel juhul läheks vaja teadmusbasis väga palju lauseid, sest eesti keeles on võimalik kasutada lausetes erinevat sõnade järjekorda. Samuti pika lause korral võib mõni vähemtähtsam sõna puududa või lisaks olla, kuid üldiselt oleks lausel ikka sama tähendus. Näiteks pole üldjoontes vahet, kas kasutaja ütleb, et „Õues on külm ilm“ või „Õues on väga külm ilm“. Oleks mõeldamatu neid kõiki erinevaid võimalusi teadmusbasisi lisada.

Probleemi lahenduseks on dialoogisüsteemis kasutusel nõrk unifitseerimine. See lisab tavalisele unifitseerimisele juurde semantilise kauguse mõiste - identsete lausete puhul 1 ja täiesti erinevate lausete puhul 0. Nii saab programmis määrata ära sarnasuse läviväärtuse, millest alates kaks lauset omavahel unifitseeruvad. Selle jaoks on sarnasuse funktsioon, mida saab defineerida mitmeti. Näiteks lihtsamal juhul leiab see kahe lausel puhul samade sõnade arvu ja jagab selle kasutaja sisestatud lause pikkusega. Eelneva näite korral, kui teadmusbasis on lause „Õues on külm ilm“, aga kasutaja sisestab „Õues on väga külm ilm“, siis nendest neli sõna on samad ja kasutaja sisestatud lause pikkus on 5. Sarnasus oleks seega $4/5 = 0,8$. Kui nüüd sarnasuse piiriks on seatud näiteks 0,7, siis need laused omavahel unifitseeruksid.

2.4 Kontekstid

Inimeste omavahelises vestluses saab eristada teemasid. Vestlus võib toimuda näiteks ilmast, spordist või koduloomadest. Emakeelt kõnelevatel inimestel on enamasti väga lihtne vahet teha, millises teemas parasjagu jutt toimub. Selleks, et dialoogisüsteem saaks rakendada teemale vastavaid reegleid ja fakte, ongi *knowledge/7* fakti viimase parameetrina kirjas kontekstide numbrid, millesse salvestatud fraas sobib. Samuti hoiab süsteem ka jooksvalt meeles, millises kontekstis parasjagu jutt toimub. Nii on võimalik otsingut kiirendada, kuna ei pea läbi vaatama kõiki teadmisi, vaid ainult neid, mis on sobivas kontekstis. Kui juhtub, et sobivas kontekstis ei ole ühtegi unifitseeruvat lauset,

siis tuleb dialoogisüsteemil konteksti muuta ehk otsida sobivat teadmist ka teiste teemade hulgast või vastata üldise fraasiga, mis ei nõua konkreetset konteksti. Viimast kasutatakse sageli ingliskeelsetes vestlusrobotites.

3 Dialoogisüsteemi loomine

Käesoleva lõputöö raames luuakse dialoogisüsteem, mis põhineb nõrgal unifitseerimisel ja semantilisel võrgul. Süsteemi ehitamise põhimõtted on kirjeldatud eelnevas peatükis. Töös luuakse reeglid vastuse genereerimiseks, teadmusbasis info salvestamiseks ja võimalus uute sõnade lisamiseks. Valminud dialoogisüsteemiga saab suhelda teksti vahendusel ehk eelmises peatükis mainitud osadest on liidestamata kõnetuvastus ja kõnesüntesaator.

3.1 Teadmusbasis

Dialoogisüsteemi teadmusbasis koosneb põhiliselt *knowledge/7* faktidest. Nende kuju on juba eelnevas osas lahti seletatud. *knowledge/7* faktid määravad ära dialoogisüsteemi teadmised. Suurem osa neist on terviklikul kujul: olemas on nii eeldus kui ka järeldus. Nende järgi saab robot teha kasutaja sisestatud lausete põhjal tuletusi. Mõnedel *knowledge/7* faktidel võivad aga eeldused puududa. Sellisel juhul on tegemist roboti jaoks tingimusteta teadmise, mis on alati tõene. Järgnevalt on toodud ka üks näide faktist, et õues sajab vihma:

```
knowledge([õu,sadama,vihm],[],[5,4,2],[],[õues,sajab,vihma],[],[0]).
```

Dialoogisüsteemi toimimiseks on vajalik mingi hulk neid fakte robotile eelnevalt sisestada. Mida rohkem on robotil erinevaid teadmisi, seda paremini ja usutavamalt peaks juturobot kasutajale vastama. Lisaks on dialoogisüsteemil ka võimekus kasutaja lausete abil õppida. Sellest on lähemalt räägitud hilisemas peatükis.

3.2 Semantiline võrk

Lisaks *knowledge/7* faktidele on olulisel kohal ka *is_a/3* faktid. Need faktid näitavad ülem- ja alamklassi omavahelist suhet. Esimeseks argumentiks on sõna algvormis ja teiseks argumentiks on selle sõna ülemklass, mis on samuti algvormis. Kolmandaks argumentiks on kontekstide list, milles vastav suhe kehtib. Näiteks fakt *is_a(paduvihm,vihm,[0])* tähendab, et vihma alamliigiks on paduvihm ja samas ka, et paduvihma ülemklass on lihtsalt vihm. See suhe kehtib ainult kontekstis numbriga null.

Niimoodi moodustub *is_a*/3 faktide abil sõnadest semantiline võrk, mis aitab *knowledge*/7 fakte kokku hoida, kuna nõrgal unifitseerimisel kasutab dialoogisüsteem ka objektide esivanemklasse. Selleks otsib tuletusmootor *knowledge*/7 fakte, millel on mõni sõna asendatud selle sõna alam- või ülemklassiga. Näiteks, kui kasutaja ütleb: „Õues sajab uduvihma“, siis peaks see lause ikkagi unifitseeruma lausega, et „Õues sajab vihma“. Üks osa semantilise võrgu *is_a* suhetest on toodud joonisel 2.

```
is_a(vihm, sade, [0]).
is_a(lumi, sade, [0]).
is_a(rahe, sade, [0]).
is_a(paduvihm, vihm, [0]).
is_a(seenevihm, vihm, [0]).
is_a(uduvihm, vihm, [0]).
is_a(vihmariietus, riietus, [0]).
is_a(vihmamantel, vihmariietus, [0]).
is_a(jope, vihmariietus, [0]).
is_a(kummikud, vihmariietus, [0]).
is_a(talveriietus, riietus, [0]).
is_a(kasukas, talveriietus, [0]).
is_a(tutimüts, talveriietus, [0]).
is_a(sall, talveriietus, [0]).
```

Joonis 2. Näide semantilise võrgu kohta.

3.3 Eesti keele grammatika

Kasutaja sisestatud lausete analüüsimiseks ja vastuste genereerimiseks peab dialoogisüsteem teadma ka teatud osa eesti keele grammatikast. Ühel sõnal on palju erinevaid vorme ja programm peab nendest kõikidest aru saama. Sõna vormide salvestamiseks on kasutusel erinevad faktid vastavalt sõna tüübile. Käändsõnade jaoks on fakt käänded/3. Sellel predikaadil on kolm argumenti:

1. Sõna algvorm.
2. Lisainfo sõna kohta, kus on märgitud kolm omadust: sõna liik, ainsus/mitmus ja elus/eluta. Praeguses töös ei ole need kasutusel, aga tulevikus võib neid vaja minna.
3. Nimekiri sõna erinevatest käänetest. Siin on kokku 28 vormi ehk siis kõik 14 käänet nii ainsuses kui ka mitmuses.

Näide:

käänded(vihm, [ns, ains, elus], [vihm, vihma, vihma, vihmasse, vihmas, vihmast, vihmale, vihmalt, vihmalt, vihmaks, vihmani, vihmana, vihmata, vihmaga, vihmad, vihmade, vihmude, vihmadesse, vihmades, vihmadest, vihmadele, vihmadel, vihmadel, vihmadeks, vihmadeni, vihmadena, vihmadeta, vihmadega]).

Pöörd sõnade jaoks on kasutusel predikaat vormid/3, mis on üsna sarnane käänete faktile. Järgnevalt ka selle predikaadi argumentide tähendused:

1. Sõna algvorm (ma-tegevusnimi).
2. Sõnaliik. Kuna pöörd sõnad on kõik tegusõnad, siis on siin alati lihtsalt „tegusõna“.
3. Nimekiri sõna vormidest. Siin on kokku 23 vormi: ma-tegevusnimi, ta-tegevusnimi, oleviku pöörded (6), mineviku pöörded (6), -nud, -tud, käskiv ainsus, käskiv mitmus, käskiv kaudne, tinglik, tinglik minevik, kaudne olevik, kaudne minevik.

Näide:

vormid(sadama, tegusõna, [sadama, sadada, sajan, sajad, sajab, sajame, sajate, sajavad, sadasin, sadasid, sadas, sadasime, sadasite, sadasid, sadanud, sajatud, saja, sadage, sadagu, sajak, sadanuks, sajatakse, sajati]).

Kolmandaks on kasutusel muutumatu_sõna/1 fakt, mis on siis kõikide ülejäänud sõnaliikide jaoks. Sellel predikaadil on ainult üks argument, milleks on sõna ise, kuna nendel sõnadel teisi vorme ei olegi.

Näide:

muutumatu_sõna(ära).

Lisaks nendele kolmele faktile on grammatikabaasis veel mõned predikaadid. Tähtis fakt on ka *synonyms*/3, mille abil saab määrata ära sõnade sünonüüme. Sellel predikaadil on kolm argumenti:

1. Põhisõna.

2. Nimekiri sõna sünonüümidest. Siia hulka kuulub ka põhisõna ise.

3. Nimekiri kontekstidest, milles need sünonüümid on sobivad.

Näiteks:

`synonyms(ilus, [ilus, kena, kaunis], [0]).`

Sünonüümid on kasulikud, et vähendada vajalikke *knowledge/7* fakte. Nõrk unifitseerimine kasutab analüüsimiseks ka sünonüüme ja vahetab kasutaja sisestatud lauses kõik sõnad, mille kohta on *synonyms/3* fakt, põhisõna vastu välja.

Inimese tavakõnes on tihti kuulda ka erinevaid täitesõnu ehk sisutühjasid sõnu, mis tegelikult ei lisa lause tähendusele midagi juurde. Nendeks on näiteks: „nagu“, „üldiselt“, „mingi“. Selleks, et dialoogisüsteem neid nõrga unifitseerimise käigus ei arvestaks, on loodud faktid täitesõna/1, mille ainukene argument ongi sõna algvormis. Semantilise kauguse arvutamisel täitesõnu ei arvestata.

3.4 Vastuse genereerimine

Kasutajale vastamiseks kasutatakse teadmusbaasis olevaid *knowledge/7* fakte ja tuletusmehhanisme, mida on kirjeldatud peatükis 2. Lisaks on abi ka *is_a/3* faktidest, mille abil leitakse sõnade ülem- ja alamklasse. Kasutaja jutustavale lausele vastamiseks on loodud erinevad võimalused, mida järgnevalt kirjeldatakse.

3.4.1 Edaspidine ja tagurpidine tuletamine

Edaspidise tuletamise abil saab vastata kasutajale järeldusega. Kasutades nõrka unifitseerimist püütakse kasutaja sisestatud lausele leida vastavuseks *knowledge/7* fakt, mille eeldused unifitseerisid kasutaja lausega. Kui selline fakt leitakse, siis kasutajale vastatakse leitud fakti järeldusega. Et vastust inimkõnele sarnasemaks muuta, lisatakse vastuse ette juhuslikult mõni ettemääratud fraas, näiteks „siis tavaliselt“, „siis võib juhtuda, et“.

Järgnevalt on toodud näide edaspidise tuletamise kaudu vastuse genereerimisest. Teadmusbaasina on kasutatud juba ka eelnevalt mainitud *knowledge/7* fakti:

```
knowledge([võima,nägema,vikerkaar],[[vihm,sadama],[päike,paistma]],[5,2,3],[[2,4],[1,5]],[võib,näha,vikerkaart],[[vihma,sajab],[päike,paistab]],[0]).
```

Kui kasutaja sisestab lause „vihma sajab ja ka päike paistab“, siis nõrga unifitseerimise abil leitakse üles eelnev *knowledge/7* fakt. Vastusena väljastab programm siis fakti järelduse ja lisab selle ette suvalise fraasi ettemääratud hulgast. Seega võib vastus tulla näiteks: „siis võib juhtuda, et võib näha vikerkaart“.

Teise võimalusena kasutab dialoogisüsteem tagurpidist tuletamist. See toimub juhul, kui kasutaja sisestab lause, mis unifitseerub mõne *knowledge/7* fakti järeldusega. Programm vastab siis fakti eeldustega ja lisab ühe juhuslikult valitud eelnevalt määratud fraasidest: „siis järelikult“, „siis ilmselt“, „siis tõenäoliselt“. Näiteks kui kasutaja sisestab lause „väljas võib näha vikerkaart“, siis programm võib vastata „siis järelikult vihma sajab ja päike paistab“. Joonisel 3 on välja toodud ka koodijupp, mis realiseerib tagurpidise tuletamise. Edaspidine tuletamine on sellele üsna sarnane.

```
1. interpret(Phrase, _,CC, Answer):-
2.     weak_unifications(bw,[Phrase],Matches, CC),
3.     flatten(Matches,MatchesF),
4.     random_member(Match, MatchesF),
5.     Match =.. MatchTerm,
6.     MatchTerm = [_,_,_,_,_,_,AnswerList,_],
7.     joinWith(AnswerList, AnswerTail, ning),
8.     AnswerHeadList = [[siis, järelikult], [siis, ilmselt]],
9.     random_member(AnswerHead, AnswerHeadList),
10.    append(AnswerHead, AnswerTail, Answer1),
11.    additional_sentence_bw(Phrase, AnswerTail, Answer2),
12.    append(Answer1, Answer2, Answer),
13.    !.
```

Joonis 3. Programmikood tagurpidiseks vastuse tuletamiseks.

Eelnevalt toodud tagurpidise tuletamise koodile (joonis 3) on lisatud parema arusaamise jaoks reanumbrid. Atribuut *Phrase* on listiga väärtustatav muutuja, mis koosneb kasutaja kirjutatud lause sõnadest. *CC* on jooksva konteksti muutuja ja *Answer* saab enda väärtuseks lõpuks roboti vastuse. Real nr 2 kasutatakse nõrka unifitseerimist ja leitakse selle abil kõik sobivad vastused (*Matches*). Reegli *flatten/2* (rida 3) abil eemaldatakse sobivatest vastustest üleliigsed sulud. Järgmisena valitakse vastuste hulgast välja juhuslikult üks suvaline (rida 4) ja *knowledge/7* faktist võetakse vastuseks õige osa (read 5, 6). Kui faktil on mitu eeldust, siis need konkateneeritakse kasutades sõna „ning“ (rida 7). Vastuse loomulikumaks tegemiseks lisatakse vastuse ette veel lisafraas, mis võetakse

samuti juhuslikult (read 8-10). Lõpuks proovitakse leida esimese lause põhjal vastusesse veel teinegi lause lisaks, samuti nõrga unifitseerimise abil (read 11-12). Viimasel real olev *cut*-operaator keelab tagasivõtu ehk lõpetab täitmise, kui vastuslause on leitud.

3.4.2 Vastuse tuletamine semantiline võrgu abil

Lisaks eelkirjeldatud kahele võimalusele kasutatakse vastuse leidmisel ka *is_a/3* fakte. Selle alusel on võimalik leida mõistete üldistusi ja konkretiseeringuid (ülem- ja alamklasse). See on vajalik, kuna keeles saab paljusid sõnu jagada hierarhilistesse suhetesse. Näiteks kui kasutaja ütleb: „õues sajab uduvihma“, siis peaks see lause ikkagi unifitseeruma lausega „õues sajab vihma“, kuna ka uduvihm on ikkagi vihm. Teisisõnu öeldes on uduvihm vihma alamliik. Selliste olukordade jaoks ongi olemas *is_a/3* faktid. Antud näite korral oleks selleks *is_a(uduvihm, vihm, [0])*, kus viimane [0] näitab, et antud suhe kehtib kontekstis, mis on indekseeritud numbriga null.

3.4.3 Küsimustele vastamine

Dialoogisüsteem peab oskama vastata ka kasutaja küsimustele. Selleks on kasutusel samuti kas edaspidine või tagurpidine tuletamine vastavalt küsimuse liigile. Kui küsimus on kujul „mis saab kui ...?“, „mille pärast ...?“ või „mille jaoks ...?“, siis tuleks vastus leida edaspidise tuletamise abil. Teisisõnu eemaldatakse kasutaja öeldud lausest küsisõnad, siis otsitakse nõrga unifitseerimise abil sobilik *knowledge/7* fakt, millest saadakse vastus. Täislausega vastamiseks lisatakse vastuse ette ka kasutaja küsimus ilma küsisõnadeta. Toon näite:

```
knowledge([tulema,kaasa,võtma,vihmavari],[[õu,olema,vihmane,ilm]],[20,1,2,1],  
[[5,5,1,1]],[tuleks,kaasa,võtta,vihmavari],[[õues,on,vihmane,ilm]],[0]).
```

Kui kasutaja esitab küsimuse: „Mis saab, kui õues on vihmane ilm?“, siis unifitseerub see *knowledge/7* faktiga ja programm leiab vastuseks, et „tuleks kaasa võtta vihmavari“. Täislausega vastamiseks eemaldatakse küsimusest küsisõnad „Mis saab kui“ ning dialoogisüsteem vastab: „Kui õues on vihmane ilm, siis tuleks kaasa võtta vihmavari“.

Võimalik on ka küsida teist liiki küsimusi, nagu „miks ...?“ või „millest tulenevalt ...?“. Nendele küsimustele vastamine käib väga sarnaselt, kuid edaspidise tuletamise asemel kasutatakse tagurpidist tuletamist. Seega küsimusele „Miks tuleks kaasa võtta vihmavari?“ vastab juturobot: „Tuleks kaasa võtta vihmavari, sest õues on vihmane ilm.“

3.4.4 Tagavaralauseid

Alati ei pruugi dialoogisüsteem kasutaja sisestatud lausele sobivat vastust leida. Selleks puhuks on robotil olemas teatud hulk üldkonstanteerivaid tagavaralauseid. Need on üldised mõtted või ütlemised, mida sobib vastata uue teema alustamiseks. Tagavaralauseite jaoks on loodud fakt *extra_sentence/3*, mille teiseks argumendiks ongi lause ja kolmandaks argumendiks on kontekst, milles lauset kasutada võib. Esimese argumendina on kirja pandud märksõnad, mis seda lauset iseloomustavad. Kui dialoogisüsteem sobivat vastust ei leia, siis valitakse tagavaralause, millel on kõige rohkem märksõnu, mis vastavad kasutaja öeldud lause sõnadele. Mõned näited *extra_sentence/3* faktide kohta on toodud joonisel 4.

1. `extra_sentence([olema, halb, ilm, olema, vale, riietus], [ei, ole, halba, ilma, on, vale, riietus], [0]).`
2. `extra_sentence([halb, ilm, järel, tulema, jälle, hea, ilm], [halva, ilma, järel, tuleb, jälle, hea, ilm], [0]).`
3. `extra_sentence([ilm, olema, keeruline, lugu], [ilmaga, on, keerulised, lood], [0]).`

Joonis 4. Tagavaralauseite faktid.

Tagavaralause valimise kood on toodud joonisel 5. Kõigepealt normaliseeritakse kasutaja öeldud lause ehk viiakse kõik sõnad algvormi (rida 2). Seejärel leitakse iga tagavaralause puhul, mitu märksõna vastavad kasutaja öeldud lausele (rida 3). Viimaks leitakse maksimaalse vastavuste arvuga tagavaralause, mis saabki dialoogisüsteemi vastuseks.

1. `choose_best_extra_sentence(Phrase, CC, Answer):-`
2. `normalize(Phrase, PhraseN, _, CC, 0),`
3. `findall(L, (extra_sentence(Words, Answer, CC),`
`intersection(PhraseN, Words, SameElements), length(SameElements, L)),`
`List),`
4. `max_list(List, M),`
5. `nth1(Index, List, M),`
6. `findall(R, extra_sentence(_, R, CC), SentenceList),`
7. `nth1(Index, SentenceList, Answer).`

Joonis 5. Programmikood tagavaralause valimiseks.

3.5 Uute sõnade lisamine

Suur osa teemakohaseid sõnu on juba teadmusbasi eelnevalt sisestatud. Selleks on kasutatud käänded/3 ja vormid/3 fakte. Kuna aga eesti keeles on väga palju sõnu, siis on

ikkagi võimalus, et inimene ütleb mõne sõna, mis teadmusbasisist puudub. Sellisel juhul tuleb leida kõigepealt sõna algvorm ja seejärel moodustada erinevad käänded või pöörded, mis teadmusbasisi salvestatakse.

3.5.1 Sõnavormide leidmine

Algvormide leidmiseks ja käänete moodustamiseks on kasutatud EstNLTK paketti [14]. See on avatud lähtekoodiga kogumik teke, mis on mõeldud eestikeelsete tekstide töötlemiseks. Selle projekti autoriteks on Siim Orasmaa, Timo Peterson, Alexander Tkachenko, Sven Laur ja Heiki-Jaan Kaalep. EstNLTK on peamiselt kirjutatud programmeerimiskeeles Python.

Uue sõna vormide genereerimiseks on kirjutatud Pythoni programm, millele saab argumendina kaasa anda sõna suvalises vormis. Seejärel leitakse EstNLTK paketi abil sõna algvorm ja kõik teised vajalikud vormid. Vastavalt sõnaliigile leitakse käänded või pöörded. Kolmas võimalus on, et uus sõna on muutumatu sõna. Programmi väljundiks on alati kolm rida: sõna algvorm, vormi number ja viimasena sõne kujul Prologi predikaat, mille abil uus sõna kohe teadmusbasisi lisada. Teisisõnu on viimane rida kas käänded/3, pöörded/3 või muutumatu_sona/1 fakt.

Uue sõna lisamine käivitub siis, kui kasutaja sisestatud lauses leidub mõni sõna, mida grammatikabaasis ei ole. Sellisel juhul luuakse Prologis protsess, mille abil pannakse tööle eelnevalt kirjeldatud Pythoni programm, millele antakse kaasa tundmatu sõna. Protsessi väljundina saadakse predikaat, mis lisatakse teadmusbasisi ja samuti lisatakse see predikaat ka grammatikabaasi faili, et teadmus säiliks ka peale Prologi programmi taaskäivitamist.

3.5.2 Sõna tundma õppimine

Dialoogisüsteem on ehitatud „õpihimuliseks“. See tähendab, et peale uue sõna vormide teadmusbasisi lisamist tahab juturobot rohkem sõna kohta teada. Uue sõna kohta küsitakse kõigepealt üldistust ehk ülemklassi. Näiteks kui uus sõna on „vihm“, siis ülemklass oleks „sademed“. Selle teadmise abil moodustatakse *is_a/3* fakt. Antud näite korral siis „*is_a(vihm, sade, [0])*“, mis lisatakse kohe teadmusbasisi ja salvestatakse ka grammatikafaili. Enne *is_a/3* fakti tegemist leitakse mõlemale sõnale algvorm. Samuti lubab see fakt ka polümorfismi ehk siis ühel sõnal võib olla mitu alternatiivset üldistust (esivanemklassi). Üks esivanem võib kehtida ühes kontekstis ja teine teises, kuid ka ühes

ja samas kontekstis võib olla mitu esivanemat. Näiteks peale sadet võib vihma esivanem olla veel ka teatud juhtudel vesi.

Järgmiseks küsitakse mõiste alamliike. Kasutaja võib vastata, et ta ei tea või kirjutada kas üks või mitu alamliiki. Nendest moodustatakse samuti *is_a/3* faktid. Näiteks vihma alamliigiks võib olla uduvihm ja sellisel juhul oleks faktiks „*is_a*(uduvihm, vihm, [0])“. Viimane argument ehk kontekst oleks siis parasjagu see, millest viimati süsteemis juttu oli.

Kolmandaks küsitakse kasutaja käest lauseid, milles oleks uue mõiste kasutusnäiteid. Kasutaja saab robotile vastata ühe või mitu sobivat lauset. Peale igat sisestatud lauset teeb robot sellest *knowledge/7* fakti, mis teadmusbbaasi salvestatakse. Kui kasutaja sisestab lause, milles puudub uus sõna, siis on see roboti jaoks märk, et kasutaja võttis uue teema ja robot enam seda lauset kohe meelde ei jäta.

3.6 Peapredikaat

Programmi käivitamiseks on loodud predikaat *räägime/0*, mis on välja toodud joonisel 6. Programmi alguses loetakse mällu teadmusbbaas (rida 2). Sinna hulka kuuluvad nii teadmised (*knowledge/7* faktid), semantiline võrk (*is_a/3* faktid) kui ka kõik eesti keele grammatikaga seotud predikaadid. Järgmiseks valitakse suvaliselt üks tervituslause (read 4-6). Seejärel salvestatakse, et viimane öeldud lause oli tervitus (read 7-8) ja samuti määratakse ajahetkeks 0 (rida 9). Leitakse parameetrina määratud sarnasuse kriteerium ja esialgne kontekst (read 10-11). Peale seda tehakse algust programmi põhitsükliga (read 13-22), mille käigus küsitakse kasutaja sisendit ja leitakse selle alusel vastus. Vastuse genereerimiseks on predikaat *interpret/4*, mis kasutaja sisendlause, sarnasuse kriteeriumi ja praeguse konteksti abil leiab sobiliku väljundlause. Neid fakte on mitu erinevat alternatiivi: üks edaspidise ja teine tagurpidise tuletamise (toodud joonisel 3) jaoks. Samuti ka küsimuste ja tagavaralause jaoks. Vastus kirjutatakse kasutajale ekraanile ja seejärel uuendatakse mälus olevat viimase lause fakti (read 18-19) ning salvestatakse mällu ka kasutaja öeldud lause (read 20-21). Süsteem tekitab ka vahepeal ajutise mälu, mis on vajalik nõrga unifitseerimise jaoks (read 14, 22). Viimastel ridadel on lõpetamistingimus (read 23-26). Kui robot leidis vastuseks „Tore!“, siis see tähendab, et inimene soovis vestlust lõpetada (*interpret/4* predikaadi lõpetamise alternatiiv tagastab

vastusena „Tore!“). Edasi valitakse suvaline vestluse lõpetamise lause ja väljastatakse see ekraanile (read 25-26), misjärel programm oma töö lõpetab.

```
1.   räägime:-
2.     load_knowledge,
3.     nl,nl,
4.     phrase_(alustamine,Welcomes),
5.     choose_random(Welcomes>Welcome),
6.     write>Welcome),
7.     retractall(last_phraseR(_)),
8.     assert(last_phraseR>Welcome)),
9.     assert(old_stamp(0)),
10.    similarity_treshold(Treshold),
11.    context(CC),
12.    repeat,
13.    sisesta_text(Text_in),
14.    create_temporary_memory,
15.    interpret(Text_in, Treshold,CC,Text_out1),
16.    delete(Text_out1,[],Text_out2),flatten(Text_out2,Text_out),
17.    write_answer_list(Text_out),
18.    retractall(last_phraseR(_)),
19.    assert(last_phraseR(Text_out)),
20.    retractall(dialogStack(Dialog)),
21.    assert(dialogStack([Text_in|Dialog])),
22.    clear_temporary_memory,
23.    Text_out = ['Tore! '],nl,
24.    phrase_(lõpetamine, Farewells),
25.    choose_random(Farewells, Farewell),
26.    write(Farewell), !.
```

Joonis 6. Dialoogisüsteemi peapredikaat.

4 Tulemuse valideerimine

Juturobotite valideerimine on üsna keeruline ja subjektiivne tegevus, kuna puudub üldine meetodika kuidas robotipoolsete fraaside sobivust dialoogis kvantitatiivselt hinnata. Siiski on olemas mõned moodused, mida järgnevalt kirjeldatakse ja mille abil valminud dialoogisüsteemi hinnatakse.

4.1 Turingi test

Kõige kuulsam juturobotite hindamise meetod on kindlasti Turingi test. Testi eesmärk on lasta inimesel ja robotil omavahel rääkida. Samal ajal on teises ruumis ka hindaja, kes teab, et üks vestluspartneritest on robot, aga ei tea kumb. Vestlus peaks toimuma kirjalikul teel ja arvuti vahendusel, et hindajal ei oleks võimalus kõne järgi robotit kindlaks määrata. Kui hindaja ei suuda piisavalt kindlalt öelda, kumb osapool on robot, siis loetaksegi test sooritatuks. Seega ka Turingi test on ikkagi subjektiivne ja sõltub hindajast. [15]

Üks tuntuimaid juturobotite võistlusi, milles Turingi testi kasutatakse, on iga-aastaselt toimuv Loebneri auhinnavõistlus. Sellega tehti algust juba 1991. aastal Hugh Loebneri poolt. Võistlus koosneb neljast voorust, milles igas ühes neli kohtunikku saavad suhelda arvuti vahendusel kahe isikuga, millest üks on robot ja teine inimene. Pärast 25 minutit küsitlemist peab kohtunik otsustama kumb on kumb. Juhul kui vähemalt pooled kohtunikud arvavad, et juturobot on inimene, loetakse test läbituks ja auhinnaks antakse hõbemedal ja 25 000\$. Kui ükski dialoogisüsteem ei suuda seda, siis antakse kohtunike skoori alusel parimale pronksmedal. Kulmedal ja saja tuhande dollari suurune auhind on mõeldud juturobotile, mis kasutab ka visuaalset ja helilist poolt. Praeguse hetkeni pole ükski robot suutnud ei kuld- ega hõbemedalit võita. [4]

Kuna Turingi testi ei ole siiani üksi juturobot veel läbinud, siis ei kasutata seda siin töös valminud dialoogisüsteemi testimiseks. Samuti vajaks Turingi test suurt teadmusbasi ja teadmisi igas erinevas valdkonnas ning nõuaks seega väga palju tööd.

4.2 Näidisdialoogi hindamine

Dialoogisüsteemi hindamiseks on koostatud näidisdialoog, mis on välja toodud joonisel 7. Inimese öeldud laused on tähistatud tähega „I“ ja roboti vastused tähega „R“. Dialoogi teemaks on põhiliselt ilm ja riietus ning nende omavaheline seos.

- I: täna on ilus ilm
R: jah, on tõesti, päike paistab ning on tuulevaikne
I: samuti on õues väga palav
R: siis tuleks selga panna suveriided, näiteks plätud ja t-särk
I: taevas pole isegi ühtegi pilve
R: järelikult ei ole ka vihma oodata
I: homseks lubab aga juba tumedamaid pilvi
R: siis võib tulla vihma või isegi äikest ja tuleks kaasa võtta vihmariietus, näiteks vihmakeep ja kummikud
I: ma loodan, et vihma siiski ei tule
R: õnneks robotina pole mul vahet, mis ilm on
I: see suvi pidi õnneks palju soojasid ilmasid olema
R: siis saab veeta õues palju aega ning saab käia tihti ujumas
I: loodan, et ei tule nii palju vihma kui eelmisel suvel
R: halva ilma järel tuleb jälle hea ilm
I: seda küll, aga äikest ma ikka ei taha
R: soojade ilmadega kaasneb paraku tihti ka äike
I: head aega
R: nägemiseni

Joonis 7. Näidisdialoog.

Dialoogi lugedes tundub vestlus olema igati loogiline ja inimesesarnane. Samas on selle dialoogi jaoks teadmusbaasi pandud just sobivad *knowledge/7* faktid. Suvalise teksti puhul ei pruugi dialoogisüsteem nii head vastust leida. Seega sõltub vastuse headus suuresti teadmusbaasi suurusest. Näidisdialoog aga näitab, et valminud juturobot suudab sobiva teadmusbaasi korral inimkõnele sarnast vestlust pidada küll.

Oluline omadus dialoogisüsteemi korral on ka robustsus ehk võime mõista lause tähendust erinevate lausete struktuuri korral. Näiteks lause „samuti on õues väga palav“ asemel võib öelda, et „õues on ikka tõesti palav“. Laused näevad küll üsna erinevad välja, aga tähendus on enam-vähem sama. Proovitud sai veel teisi sarnase tähendusega lauseid ja leiti, et dialoogisüsteem on võimeline aru saama kõigist järgnevatest lausetest:

- samuti on õues väga palav
- õues on ikka tõesti palav
- õues on palav
- palav on õues
- väljas on soe
- samuti on õues ka ikka tõesti nagu väga väga palav

Samuti saab dialoogisüsteem aru erinevatest sõnavormidest. Järgnevatest lausetest saab juturobot samuti ühtemoodi aru:

- homseks lubab aga juba tumedamaid pilvi
- homme lubatakse juba tumedamat pilve
- homseks on lubatud koledaid pilvesid

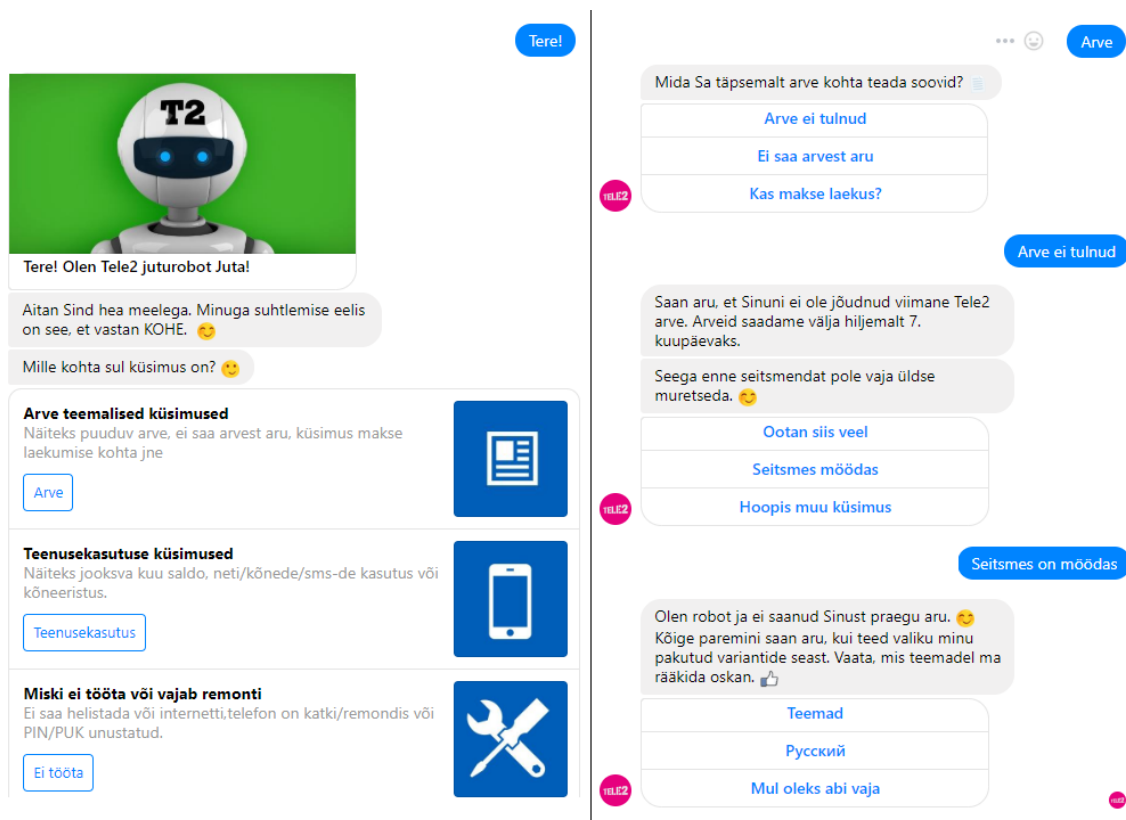
Seega valminud dialoogisüsteem on üsna robustne, kuna on kasutatud nõrka unifitseerimist ja kirja on pandud suur osa eesti keele grammatikat. Süsteem suudab unifitseerida ka sünonüüme ja ignoreerida lauses sisalduvaid üleliigseid täitesõnu.

4.3 Dialoogisüsteemi võrdlus sarnaste lahendustega

Alternatiivseks testimise võimaluseks on võrrelda valminud dialoogisüsteemi juba olemasolevate eestikeelsete lahendustega. Leiti kaks avalikult kasutatavat sobilikku juturobotit. Üks nendest on Tele2 juturobot Juta ja teine Cleverbot.

Tele2-e robot Juta on mõeldud pigem etteantud valikutele vastama: iga öeldud lausega annab robot ka mõned vastusevariandid kaasa. Kuna aga tegemist on klienditeenindusrobotiga, siis on selline lahendus täiesti sobilik korduvatele küsimustele vastamiseks. Näidisvestlus Tele2-e robotiga on toodud joonisel 8. Inimese tekst on sinisel taustal (paremal) ja roboti vastus hallil (vasakul). Juta poolt etteantud vastusvariandid on sinise kirjaga (roboti vastuste all). Nende peale vajutades kirjutatakse kasutaja tekst automaatselt, kuid on võimalik ka ise vastus kirjutada. Et Juta kasutaja öeldust aru saaks,

tuleks kirjutada üpris sarnaselt etteantud valikule. Seega peaks inimese tekst olema enamasti suhteliselt lakooniline: üks, kaks sõna. Leidsin, et robotil Juta on puudu suures osas robustsusest, mida lõputöö raames valminud dialoogisüsteemil on püütud saavutada. Samuti on Tele2-e robot üsna tundlik sõnade järjekorra ning sõnavormide suhtes. Tihipeal ei mõista robot ka teisel kujul öeldud samatähenduslikku lauset.



Joonis 8. Tele2 juturobot Juta.

Robot Cleverbot on võrdlemiseks parem, kuna see on sarnaselt töö käigus valminud dialoogisüsteemile samuti üldotstarbeline suhtlusrobot. Cleverboti eelisteks on universaalsus ja lihtne õppimismehhanism. See robot suudab vestelda paljudes erinevates keeltes ja õppida igalt inimeselt, kes temaga suhtleb. Lõputöö käigus valminud dialoogisüsteem on mõeldud aga ainult eesti keeles suhtlemiseks ja ka õppimine on palju piiratum, kuna selleks on eraldi režiim. Teisest küljest võib kontrollitud viisil roboti õpetamine olla isegi kasulik, sest nii on tagatud kvaliteet. Kuna Cleverbot õpib igalt inimeselt, siis sellel robotil puudub tihtipeale järjepidevus. Testimise käigus selgus, et kord on roboti nimeks Karl, aga teine kord Liis. Juhtus ka, et vahepeal oli robot 12-aastane ja natukese aja pärast juba üheaastane. Samuti ei pruugi inimesed robotile sobilikke ja

häid vastuseid anda, eriti kuna interneti vahendusel saavad juturoboti kasutajad anonüümseks jääda. Prooviti ka eelmise peatüki näitedialoogi lauseid Cleverboti peal. Vestlus on toodud joonisel 9. Juba esimese lause puhul ei ole vastus päris korrektne, aga vähemalt on õiges teemas. Järgnevate lausete puhul on vastus juba üsna suvaline. Üks vastus on üldse prantsuse keeles. Mõni roboti vastus on isegi päris ülbe („Ei pidanud:D taunik“, „Ok mine ära ma ei tahagi suga rääkida“). Teatud mõttes teeb selline käitumine robotit isegi usutavamaks, aga arvatavasti tahaksid enamus inimesi pigem sõbraliku robotiga vestelda. Cleverboti kvaliteet on tõenäoliselt tingitud vähestest inimestest, kellelt õppida. Eestlasi on kokku isegi üpris vähe ja vaevalt, et ka Cleverbotil palju eestikeelseid kasutajaid on.

User: Täna on ilus ilm.

Cleverbot: On jah vihma sajab ja tuul puhub.

User: Samuti on õues väga palav.

Cleverbot: Ei taha röstsaia.

User: Taevas pole isegi ühtegi pilve.

Cleverbot: Quoi quand quon fait l amour.

User: Homseks lubab aga juba tumedamaid pilvi.

Cleverbot: Ma ei saa olen praegu poisiga.

User: Ma loodan et vihma siiski ei tule.

Cleverbot: Ma ka, seega kohting jääb vist ära :(.

User: See suvi pidi õnneks palju soojasid ilmasid olema.

Cleverbot: Ei pidanud:D taunik.

User: Loodan et ei tule nii palju vihma kui eelmisel suvel.

Cleverbot: No ütle.

User: Seda küll aga äikest ma ikka ei taha.

Cleverbot: Kas ma olen nii kole vä.

User: Head aega.

Cleverbot: Ok mine ära ma ei tahagi suga rääkida.

Joonis 9. Vestlus robotiga Cleverbot.

5 Kokkuvõte

Lõputöö tulemusena valmis eesti keeles töötav dialoogisüsteem, millega on võimalik teksti vahendusel suhelda. Sisendiks saab kasutaja kirjutada teksti, mida robot analüüsib ja leiab sellele nõrga unifitseerimise ja sobivate reeglite alusel vastuslause. Juturobotile valmistati ka õppimise võimalus, mille abil jätab robot meelde uusi teadmisi ja genereerib ka uute sõnade jaoks vorme.

Loodi reeglid vastuse genereerimiseks nii edaspidise kui ka tagurpidise tuletamise abil. Samuti loodi semantiline võrk, mille abil siduda omavahel hierarhilises suhtes olevaid sõnu. Uute sõnade õppimiseks loodi Pythoni kood, mis kasutab EstNLTK teeki. Lisaks sellele oskab robot automaatselt täiendada teadmusbaasis olevaid lauseid ja ka semantilist võrku.

Teadmusbaasi sai töö käigus sisestatud vajalikud teadmised, et vestelda ilma ja riietuse teemal. Tulemusena leiti, et dialoogisüsteem on üpris robustne ja suudab ära tunda sünonüüme, erinevaid sõnavorme ning erinevat sõnade järjekorda. Piisavalt suure teadmusbaasi korral suureneb sobivate vastuslausete leidmise tõenäosus ning roboti vastused muutuvad inimvestlusele sarnasemaks.

5.1 Edasine töö

Tulevikus on võimalik tehtud tööd ka mitut moodi edasi arendada. Kuna päris aeganõudev ja isegi võimatu on kõiki erinevaid teadmisi süsteemi sisestada, siis oleks võimalus teadmusbaasi koguda veebist. Vastavalt inimese sisestatud lausele on võimalik otsida internetist vastusfraasi. Hästi võiks see töötada kasutaja küsimustele vastamiseks, kuna küsimuse sisestamisel näiteks Google'i otsingumootorisse saab enamasti alati vastuse kätte. Niimoodi saaks teha väga targa roboti. Samas jutustavatele lausetele vastamiseks ei pruugi internetist otsimine väga head tulemust anda, kui kontekst ei ole selgelt määratletud.

Üldiselt olenevad nõuded dialoogisüsteemile suures osas dialoogisüsteemi eesmärgist. Lõputöö käigus loodi üldotstarbeline suhtlusrobot. Kui aga tahta kohandada robot lastele sõnavara õpetama, siis tuleks teadmusbasis kasutada ainult lihtsamapoolset sõnavara. Samuti peaks robot olema targem kui laps ehk lapsega rääkides ei tohiks robot olla õppimisrežiimis, vaid dialoogisüsteem peaks ise suutma last õpetada. Kindlasti peaks väikeste laste puhul robot tuvastama lapse kõnet kõnetuvastuse abil ja ka ise teisendama oma vastuse heliks kõnesüntesaatorit kasutades. Lastele oleks kindlasti ka huvitavam, kui kogu süsteem oleks tööle pandud päris robotile.

Kasutatud kirjandus

- [1] Tulevik on käes: robot hakkab klienditeeninduses läbimurret tegema, *ITuudised.ee*. [Online] <http://www.ituudised.ee/uudised/2017/03/30/tulevik-on-kaes-robot-hakkab-klienditeeninduses-labimurret-tegema> (22.04.2018).
- [2] SEB käivitas Facebooki juturoboti | SEB. [Online] <https://www.seb.ee/uudised/2017-10-10/seb-kaivitas-facebooki-juturoboti> (13.05.2018).
- [3] Telia uues peakontoris tervitab kliente humanoidrobot Pepper, *Hansab.ee*, 12-mai-2017. [Online] <http://www.hansab.ee/et/telia-uues-peakontoris-tervitaab-kliente-humanoidrobot-pepper> (13.05.2018).
- [4] AISB - The Society for the Study of Artificial Intelligence and Simulation of Behaviour - Loebner Prize. [WWW] <http://www.aisb.org.uk/events/loebner-prize> (24.04.2018).
- [5] Mitsuku Chatbot, *Mitsuku*. [WWW] <http://mitsuku.com/> (25.04.2018).
- [6] Mitsuku Chatbot. [WWW] <http://www.square-bear.co.uk/mitsuku/nfchat.htm> (24.04.2018).
- [7] Mitsuku Chatbot - AIML. [WWW] <http://www.square-bear.co.uk/aiml/> (25.04.2018).
- [8] AIML: An Introduction · Pandorabots Documentation. [Online] <https://www.pandorabots.com/docs/aiml/aiml-basics.html> (25.04.2018).
- [9] Cleverbot, *Cleverbot*. [WWW] <http://www.cleverbot.com/> (25.04.2018).
- [10] A. Saenz, Cleverbot Chat Engine Is Learning From The Internet To Talk Like A Human, *Singularity Hub*, 13-jaan-2010. [Online] <https://singularityhub.com/2010/01/13/cleverbot-chat-engine-is-learning-from-the-internet-to-talk-like-a-human/> (25.04.2018).
- [11] P. Bright, Tay, the neo-Nazi millennial chatbot, gets autopsied, *Ars Technica*, 26-märts-2016. [Online] <https://arstechnica.com/information-technology/2016/03/tay-the-neo-nazi-millennial-chatbot-gets-autopsied/> (24.04.2018).
- [12] SWI-Prolog. [WWW] <http://www.swi-prolog.org/> (22.04.2018).
- [13] J. Vain ja H. Sarapuu, Towards context-sensitive dialogue with robot companion, *2014 14th Biennial Baltic Electronic Conference (BEC)*, 2014, lk 205–208.
- [14] *estnltk: Open source tools for Estonian natural language processing*. estnltk, 2018.

[15] Turing test, *Wikipedia*. [WWW] https://en.wikipedia.org/wiki/Turing_test (15.05.2018).