T A R T U Ü L I K O O L
MATEMAATIKA - INFORMAATIKATEADUSKOND
Arvutiteaduse instituut
Tarkvarasüsteemide õppetool
Informaatika eriala

Anton Golovko

Õpiprogramm Bethi tabelite konstrueerimiseks

Magistritöö

Juhendaja: dotsent Rein Prank

TARTU 2004

Sisukord

Sissejuhatus

Käesoleva magistritöö aluseks on Evert Willem Bethi artikkel (vt. [2]), mis kirjeldab meetodi sekventside tõestamiseks või ümberlükkamiseks matemaatilises loogikas. Tavaliselt selle meetodi abil konstrueeritav tõestuse puu kasvab liiga suureks ja keeruliseks ja selle käsitlemine paberil näiteks saab tohutult raskeks. Tartu Ülikoolis on Bethi semantiliste tabelite meetod üks osa ainest „Matemaatiline loogika ja algoritmiteooria“ ja tudengitele viiakse läbi praktikume ja kontrolltööd selles osas. Antud magistritöö ülesandeks on Bethi tabelite konstrueerimise õpiprogrammi loomine. Programm on ette nähtud iseseisvateks praktikumideks kui ka tudengite teadmiste kontrolliks. Programmis on realiseeritud semantiliste tabelite meetodi kasutamiseks vajalik funktsionaalsus lausearvutuses ja predikaatarvutuses ja sellega seotud vigade diagnoosid.

Magistritöö tulemuseks on Java rakendus. Programm on lisatud tööle CD-l. Peale programmi on loodud käesolev dokument, mis kirjeldab ülesande püstitust, Bethi semantiliste tabelite meetodit ja selle realiseerimist programmis. Lisaks sellele on töös ka kokkuvõte, resümee inglise keeles ja programmi kasutajajuhend. Magistritöö tekstiga kaasa läheb ülespool mainitud CD, mis sisaldab programmi ennast, lähtekoode, näidisülesandeid ja kasutajajuhendit.

1. Töö eesmärk

Magistritöö põhieesmärgiks on luua programm, mis oleks sekventside tõestamise (või ümberlükkamise) keskkonnaks Bethi semantiliste tabelite meetodi abil. Tõestamine peab toimuma samm-sammulise reeglite rakendamisega valemitele. Semantiliste tabelite meetodiga lahendatav sekvents on kas tõestatav (samaselt tõene) või ümberlükatav. Kui sekvents lükatakse ümber, siis leitakse kontramudel.

Programm peab olema varustatud kõikide vajalikkude lahendamise õigsuse kontrollidega ja hindamise võimalustega. Selleks peavad olema defineeritud vigade klassid. Vea tegemisel peab kasutaja saama tagasisidet tehtud vea kohta ja vastav viga arvestatakse programmis selleks, et saaks hinnata ülesande lahendamise tulemust.

Ülesandeid hoitakse ülesannete kogudes. Kogus määratakse trahvipunktid iga veaklassi jaoks, mida arvestatakse töö hindamisel.

Programmis peab õppejõu jaoks olema ka ülesannete koostamise ja lahendamiste tulemuste vaatamise võimalus. Lahendada peab saama nii praktikumi režiimis kui ka kontrolltöö režiimis. Kontrolltöö režiimis peab olema tagatud teatud turvalisuse tase, mis kaitseks tulemuste võltsimise eest tudengite poolt.

2. Semantiliste tabelite meetod

Selles peatükis vaadeldakse magistritöö aluseks olevat meetodit. Semantiliste tabelite meetod on algoritm, mille abil saab tõestada või ümber lükata lausearvutuse ja predikaatloogika valemeid.

Meetod oli pakutud hollandi teadlase ja filosoofi Evert Willem Bethi poolt. Ta kirjeldas selle meetodi oma raamatus [1]. Meile on kättesaadav artikli venekeelne tõlge kogumikus [2].

2.1. Meetodi idee

Meetodi põhiidee järgi saab valemi V samaselt tõesust kindlaks teha järgmiselt. Tehakse katse konstrueerida kontranäidet sellele, et V on samaselt tõene. Kui kontranäide on leitud, siis V ei ole samaselt tõene. Kui kõik kontranäite konstrueerimise variandid viivad vastuolule, siis V on samaselt tõene.

Bethi meetod on puhtalt formaalne. See vaatleb lausearvutust ja predikaatarvutust ainult nagu süntaksi ja ei tegele loogiliste mõistete tähendusega. Meetod baseerub Gentzeni alamvalemite printsiibil, kus valem jagatakse peatehte alusel alamvalemiteks.

Evert Beth esitab oma meetodi sekventside näitel. Piirdume siin kirjelduses allpool üksiku valemi näitel.

Meetodi kasutamiseks luuakse tabel, mis koosneb kahest veerust: tõeste ja väärade valemite veerud. Kui valem paigutatakse tõesesse veergu, siis me katsume konstrueerida kontranäidet, kus see valem on tõene ja vastupidi. Nagu ülespool oli mainitud, valemi tõestamine käib ümberlükkamise katse kaudu. Sel põhjusel Bethi tabeli konstrueerimise alguses me paigutame tõestava valemi teise veergu oletades, et antud valem ei ole samaselt tõene.

Bethi meetod defineerib teatud hulk reegleid lausearvutuse ja predikaatarvutuse tehte jaoks. Vaatleme siin ka predikaatloogika kvantoreid tehtena. Reegli rakendamine valemile tekitab selle valemi järeldusi, mis peavad olema lisatud Bethi tabeli puule teatud viisil vastavalt reegli kirjeldusele. Reegel võib nõuda ka tabeli alamtabeliteks jagamist.

Eeldusvalemile rakendatakse sobiv reegel vastavalt peatehtele. Pärast seda rakendatakse reegleid tema järeldustele seni kaua, kui igas harus leitakse vastuolu või leitakse mingi haru, milles vastuolu ei saa olla. Vastuolu olemasolu tähendab, et mingi valem ilmub antud harus nii tõeste valemite veerus kui ka väärade valemite veerus. Harus vastuolu ja reegli rakendamise võimaluse puudumine tähendab siis, et on leitud interpretatsioon, mille puhul meie eeldus on õige. Sellest järeldub, et valemi samaselt tõesuse tõestamine lõppes kontramudeli leidmisega.

Kui Bethi tabeli igas harus leidub vastuolu, siis see tähendab, et otsitavat interpretatsiooni ei saa leida ehk valemi samaselt tõesus on tõestatud.

2.2. Reeglite kirjeldus

Meetodi konkretiseerimiseks defineeris Beth oma raamatus hulk reegleid lausearvutuse ja predikaatarvutuse jaoks. Vaatleme neid:

(i) Kui seesama valem ilmub antud tabeli (alamtabeli) mõlemates veergudes, siis see tabel (alamtabel) suletakse. Kui antud tabeli mõlemad alamtabelid on suletud, siis see tabel on suletud.

(ii^a) Kui $\neg U$ ilmub vasakus veerus, siis U lisatakse antud tabeli vastavasse paremasse veergu.

(ii^b) Kui $\neg U$ ilmub paremas veerus, siis U tuuakse antud tabeli vastavasse vasakusse veergu.

(iii^a) Kui $U \,\&\, V$ ilmub tabeli vasakus veerus, siis U kui ka V tuuakse selle tabeli sellesse samasse veergu.

(iii^b) Kui $U \,\&\, V$ ilmub tabeli paremas veerus, siis antud tabel jagatakse kaheks alamtabeliks, mille parematesse veergudesse tuuakse vastavalt U ja V. Kui antud tabelil on alamtabelid juba olemas, siis antud tabeli alampuu iga mitte suletud leht-tabel jagatakse kaheks ning U ja V viiakse sisse iga jagatud tabeli alamtabelite parematesse veergudesse.

(iv^a) Kui $U \vee V$ ilmub tabeli vasakus veerus, siis antud tabel jagatakse kaheks alamtabeliks, mille vasakutesse veergudesse tuuakse vastavalt U ja V. Kui antud tabelil on juba alamtabeleid, siis jagatakse selle tabeli alampuu leht-tabelid ning U ja V paigutatakse nende jagatud tabelite alamtabelite vasakutesse veergudesse.

(iv^b) Kui $U \veebar V$ ilmub tabeli paremas veerus, siis U kui ka V tuuakse selle tabeli samasse veergu.

(v^a) Kui $U \supset V$ ilmub tabeli vasakus veerus, siis antud tabel jagatakse kaheks alamtabeliks ning ühe alamtabeli paremasse veergu paigutatakse U ja teise alamtabeli vasakusse veergu tuuakse V. Kui antud tabelil on juba alamtabelid, siis jagatakse selle tabeli alampuu leht-tabelid ning U ja V paigutatakse iga selle jagatud tabeli ühe alamtabeli paremasse veergu ja teise alamtabeli vasakusse veergu vastavalt.

(v^b) Kui $U \supset V$ ilmub tabeli paremas veerus, siis U paigutatakse selle tabeli vasakusse veergu ja V paigutatakse paremasse veergu.

(vi^a) Kui $\forall xU(x)$ ilmub tabeli vasakus veerus, siis antud tabeli alampuu vasakusse veergu paigutatakse $U(p)$ iga sissetoodud konstantse parameetri jaoks. See tähendab, et iga kord, kui tuuakse sisse uus konstantne parameeter p mingisse tabelisse (mis aga kuulub samasse harusse, mida $\forall xU(x)$ sisaldav tabel moodustab), siis selle reegli järgi tabelisse, kus see parameeter oli sisse toodud, tuuakse vasakusse veergu sisse $U(p)$. Kui aga konstandi sissetoomisega tabel on antud tabeli eellane, siis selle sissetoomisele vastav $U(p)$ paigutatakse antud tabeli vasakusse veergu.

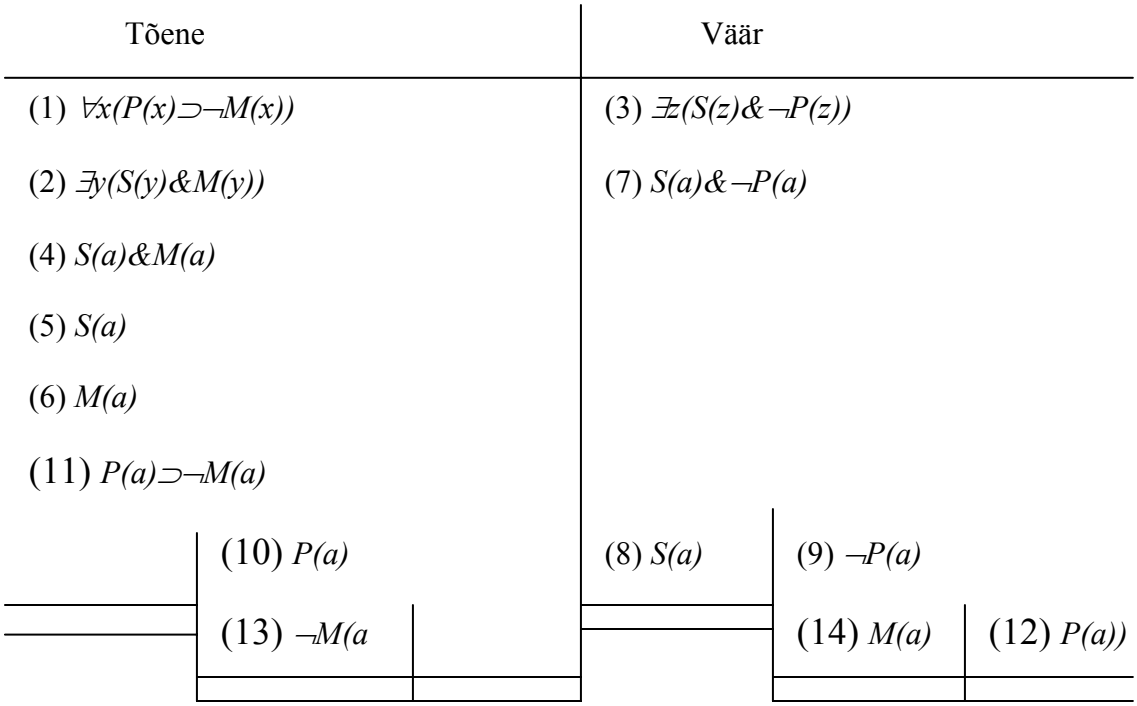
(vi^b) Kui $\forall xU(x)$ ilmub tabeli paremas veerus, siis tuuakse sisse uus parameeter p ja $U(p)$ paigutatakse sellesama tabeli paremasse veergu.

(vii^a) Kui $\exists xU(x)$ ilmub tabeli vasakus veerus, siis tuuakse sisse uus parameeter p ja $U(p)$ paigutatakse tabeli vasakusse veergu.

(vii^b) Kui $\exists xU(x)$ ilmub tabeli paremas veerus, siis antud tabeli alampuus oleva vastava tabeli paremasse veergu paigutatakse $U(p)$ iga sissetoodud konstantse parameetri jaoks. See tehakse samal moel nagu reegli (vi^a) kirjelduses.

Konstrueerimise alguses saavad tabeli vasakusse ja paremasse veergudesse olla paigutatud suvalised valemid. Sel põhjusel võib nii juhtuda, et on võimatu tuua sisse esimest indiviidparameetrit kasutades reegleid (vi^{b}) või (vii^{a}). Sellel juhul tuuakse sisse esimene konstantne parameeter, et saaks pärast kasutada reegleid (vi^{a}) või (vii^{b}). Käesolevas magistritöös nimetatakse seda võimalust „esimese konstandi sissetoomise lisareegliks“ ehk lihtsalt „lisareegliks“.

Vaatleme järgmist Bethi artiklist võetud näidet:



Siin autor üritab tõestada, et valemitest $\forall x(P(x)\supset\neg M(x))$ ja $\exists y(S(y)\&M(y))$ järeldub valem $\exists z(S(z)\&\neg P(z))$.

Konstrueerimine käib siin järgmiste sammude kaudu:

- Sekventsi valemid (1) ja (2) paigutatakse vasakusse veergu ning valem (3) paremasse veergu.
- Valemile (2) reeglit (vii^{a}) rakendades saame valemi (4). Siin tuuakse sisse uus konstant *a*.

- Valemile (4) reeglit (iii^a) rakendades saame valemid (5) ja (6).
- Valemile (3) rakendatakse reeglit (vii^b), kus indiviidmuutuja *z* asendatakse konstandiga *a*. Saame valemi (7).
- Rakendades valemile (7) reegli (iii^b), jagatakse tabel kaheks tabeliks ja nende paremasse veergu paigutatakse valemeid (8) ja (9).
- Valemi (8) tabel sulgub, sest valemid (8) ja (5) moodustavad vastuolu.
- Valemile (9) reeglit (ii^b) rakendades saame valemi (10).
- Valemile (1) reeglit (vi^a) rakendades ja kasutades konstanti *a* saame valemi (11).
- Valemile (11) reeglit (v^a) rakendades, teeme veel ühe hargnemise ja lisame Bethi tabelile valemid (12) ja (13).
- Valemite (10) ja (12) tõttu vastav tabel sulgub.
- Valemile (13) reeglit (ii^a) rakendades saame valemi (14).
- Tänu valemile (6) ja (14) sulgub ka valemi (14) tabel.

Konstrueerimine viib sellele, et kõik harud on suletud, seega sekventsi samaselt tõesus on tõestatud.

2.3. Kontramudel

Bethi tabelis reeglite rakendamine viib meetodi järgi sellele, et kas terve tabel sulgub või leidub selline haru, mis ei ole suletud. Kui terve tabel on suletud, siis kontramudeli otsimine viis vastuolule. Seega eeldus on vale ja valemi samaselt tõesus on tõestatud.

Kuna antud meetod on rakendatav nii lausearvutusele kui predikaatloogikale, siis üldistamise huvides loeme edaspidi lausearvutuse atomaarseid valemeid (lausemuutujaid) siin töös 0-argumendilisteks predikaatideks.

Kui leidub selline haru, mis ei ole suletud, ja reeglite rakendamine selles harus ei ole võimalik (st. kõik valemid on jagatud atomaarseteks), siis kontramudeli leidmine õnnestus.

Kontramudel koostakse siis seda haru kasutades. Selleks vaadeldakse kõiki antud haru atomaarseid valemeid. Olgu harus konstrueerimine viinud sellele, et antud haru vasakus veerus on atomaarsed valemid $P_1(a_{11}, ..., a_{1s}), ..., P_m(a_{m1}, ..., a_{mt})$ ja paremas veerus on $R_1(a_{11}, ..., a_{1u}), ..., R_n(a_{n1}, ..., a_{nv})$. Kontramudeliks on siis selline hulk X, et

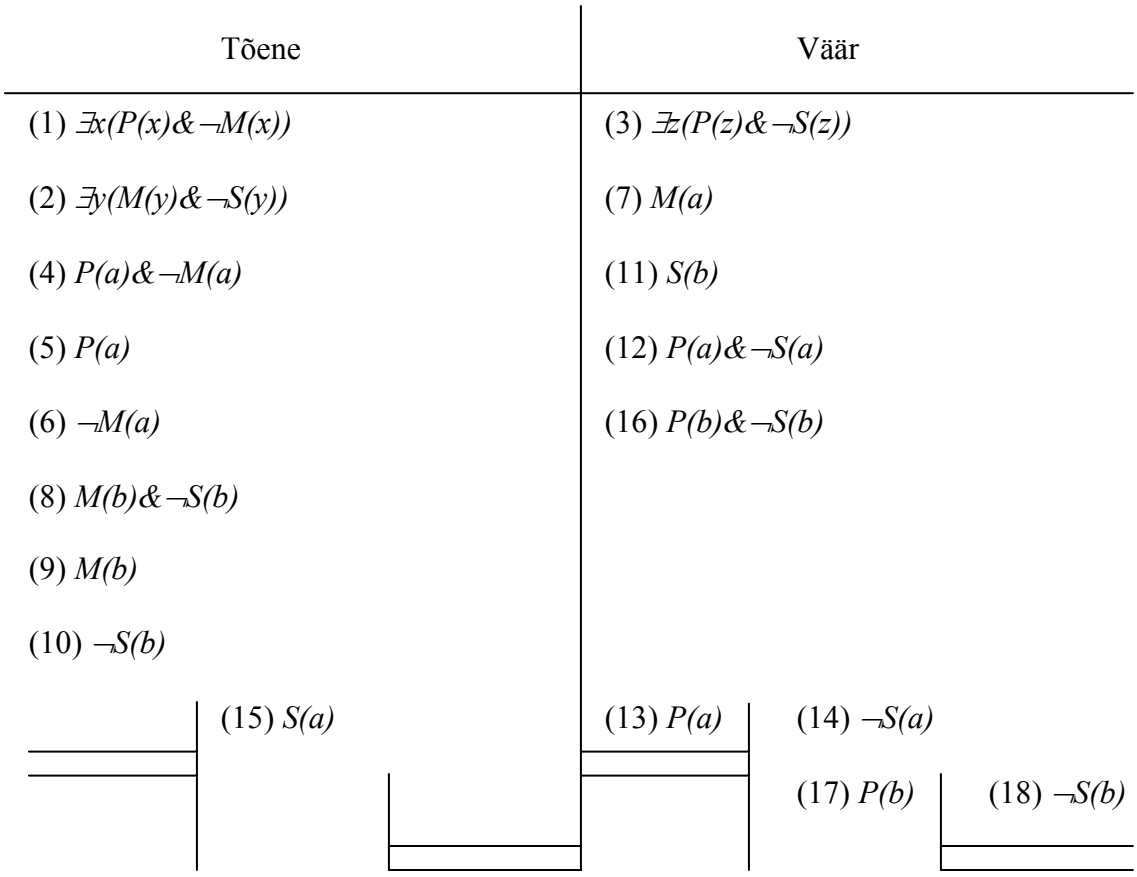
$a_{11}, ..., a_{1s}, ..., a_{m1}, ..., a_{mt} \in X$ ja $a_{11}, ..., a_{1u}, ..., a_{n1}, ..., a_{nv} \in X$,

mille puhul kehtib järgmine:

- $P_1(a_{11}, ..., a_{1s})$ on tõene
- ...
- $P_m(a_{m1}, ..., a_{mt})$ on tõene
- $R_1(a_{11}, ..., a_{1u})$ on väär
- ...
- $R_n(a_{n1}, ..., a_{nv})$ on väär

Kontramudeli olemasolu lükkab ümber vaadeldava valemi samaselt tõesuse.

Vaatleme näidet, kus katsume tõestada, et valemist $\exists x(P(x)\&\neg M(x))$ ja $\exists y(M(y)\&\neg S(y))$ järeldub valem $\exists z(P(z)\&\neg S(z))$. Kasutades Bethi meetodit, saame siis järgmise Bethi tabeli:



Nagu on näha, üks haru ei ole suletud ja reeglite rakendamine selles harus ei ole enam võimalik. Selle haru põhjal koostatakse siis kontramudel. Kontramudeli kandjaks on hulk, mis koosneb harus kasutatud konstantidest *a* ja *b* ning *P(a), M(b), S(a)* on tõesed ja *M(a), S(b), P(b)* on väärad.

Kahjuks selle meetodi abil ei ole võimalik tõestada või ümber lükata suvalise predikaatloogika valemi samaselt tõesust. See fakt järeldub Alonzo Churchi teoreemist, mis väidab, et „esimest järku loogika on mittelahenduv“. See tähendab, et predikaatarvutus ei ole lahenduv ehk pole võimalik kontrollida, kas suvaliselt valitud predikaatarvutuse valem on samaselt tõene või ei ole.

See mittelahendatavus avaldub Bethi tabeli konstrueerimises nii, et lahendus läheb lõpmatuks. Tüüpiliseks näiteks sellele on järgmine valem: $\forall x \exists y P(x,y) \supset \exists x \forall y P(x,y)$. See valem ei ole samaselt tõene ja kontramudeliks võib olla näiteks mudel, mille kandjaks on hulk $X=\{a,b\}$ ja $P(a,a)$ ja $P(b,a)$ on tõesed ning $P(b,b)$ ja $P(a,b)$ on väärad. Katse leida

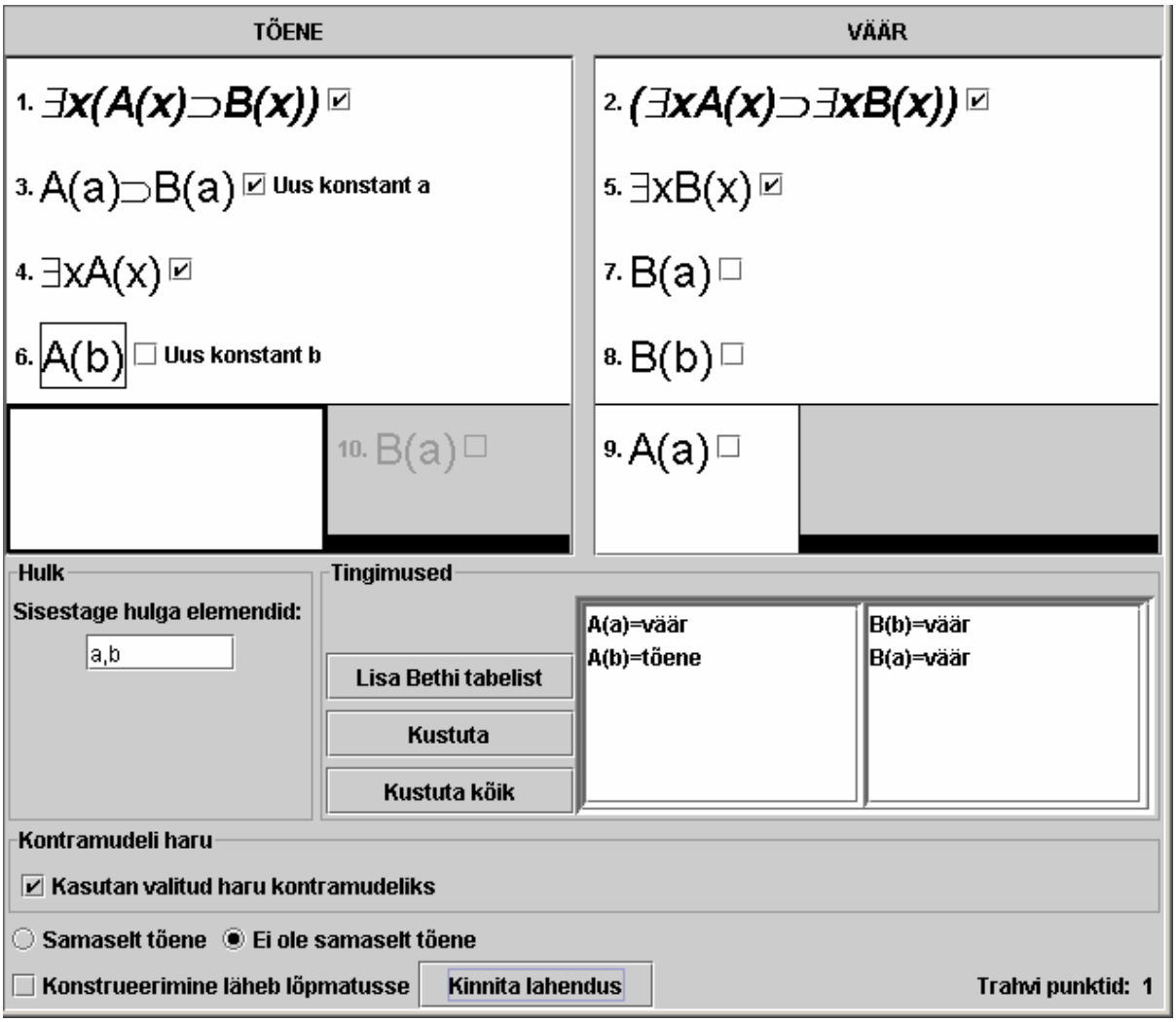
kontramudelit ebaõnnestub sellepärast, et iga kord, kui toome reegli (vii[a]) abil sisse uue konstandi, ilmub võimalus valemile $\forall x \exists y P(x,y)$ reeglit (vi[a]) rakendada, mille tulemuseks on jällegi valem, millele saab rakendada reeglit (vii[a]). Bethi tabeli ehitamine jätkub lõpmatult.

Vaatlemata sellele on meetodil kõrge praktiline väärtus. On tõestatud, et meetodiga on lahendatav suvaline valem, kus predikaatide aarsus on < 2.

Meetodi puuduseks on ka see, et paljud suhteliselt lihtsad lausearvutuse valemid viivad keerulistele Bethi tabelite struktuuridele, kus on palju hargnemisi.

3. Programmi kirjeldus

Programmi põhiosa on Bethi tabelite konstrueerimise keskkond. Keskkond kujutab endast Bethi tabeli visuaalset kujundit, mille kaudu kasutaja saab suhelda selliste elementidega nagu tabelid, veerud, valemid, kontramudel.



Joonis 1. Lahendamise keskkond

Programmi menüüdes on meetodiga seotud funktsionaalsus ehk reeglid, mida kasutaja saab rakendada valitud valemitele. Kasutades reegleid, toimub Bethi tabeli konstrueerimine. Konstrueerimist lõpetades annab kasutaja vastuse, mis peab kirjeldama, millele lahendus viis.

Programm kontrollib iga tehtavat sammu ja ei luba kasutajal teha midagi valesti, andes koheselt tagasisidet veateatega.

Käesolev peatükk selgitab programmis realiseeritud ülesannete lahendamise käiku, kirjeldab reegleid ja nende rakendamise õigsuse kontrollimise viise ning annab ülevaate programmi poolt diagnoositavatest vigadest.

3.1. Valem

Valem on põhiobjekt, millega programm tegeleb. Bethi tabelid sisaldavad valemeid, nendele rakendatakse reegleid vastavalt nende struktuurile. Valem on see, mis seostab programmi matemaatilise loogikaga. Selles punktis vaatleme valemi mõistet ja valemiga seotud kitsendusi.

3.1.1. Valemi formaalne definitsioon

Valemi saab kirjeldada allpool esitatud rekursiivse definitsiooniga.

- **Indiviid** on väike täht ladina tähestikust.
- **Atomaarne valem** on suur ladina täht, mille võivad järgneda sulud, mille sees on komadega eraldatud indiviidid, mille arv on suurem kui 0.
- Atomaarne valem on valem.
- Valemi U eitus $\neg U$ on valem.
- Valemite U ja V konjunktsioon $U\&V$ on valem.
- Valemite U ja V disjunktsioon $U\vee V$ on valem.
- Valemite U ja V implikatsioon $U\supset V$ on valem.
- Valemi U üldistus indiviidi x alusel $\forall xU$ on valem.
- Valemi U olemasolu indiviidi x alusel $\exists xU$ on valem.
- Valem U sulgudes (U) on valem.

Valemite juures programmis kehtib mitu kitsendust. Nende kitsenduste defineerimiseks on tähtis alamvalemi mõiste. Alamvalemi mõistel baseerub ka Bethi semantiliste tabelite meetod. Defineerime **vahetud alamvalemid** ja **peatehte** mõisted järgmise rekursiivse definitsiooni abil. Järgnevate väidete järjekord on tähtis, sest see defineerib tehte järjekorra.

- Valem *(U)* on valemi *U* **sulgudega valem**
- Valemi *U* **sulgudeta valem** on selline valem *V*, et kas *V* on *U* või leiduvad sellised valemid U_1, .., U_n, $n \geq 0$, mille puhul on õige, et valemi *V* sulgudega valem on U_1, valemi U_1 sulgudega valem on U_2, ..., valemi U_n sulgudega valem on *U;* ja ei leidu sellise valemi *W* nii, et *V* on valemi *W* sulgudega valem.
- Valemi $U \supset V$ peatehe on $\supset$ ja vahetuteks alamvalemiteks on *S* ja *T*, kui *U* ja *V* on valemid ning *S* ja *T* on vastavalt valemite *U* ja *V* sulgudeta valemid.
- Valemi $U \vee V$ peatehe on $\vee$ ja vahetuteks alamvalemiteks on *S* ja *T*, kui *U* ja *V* on valemid ning *S* ja *T* on vastavalt valemite *U* ja *V* sulgudeta valemid.
- Valemi $U \& V$ peatehe on & ja vahetuteks alamvalemiteks on *S* ja *T*, kui *U* ja *V* on valemid ning *S* ja *T* on vastavalt valemite *U* ja *V* sulgudeta valemid.
- Valemi $\neg U$ peatehe on $\neg$ ja vahetuks alamvalemiks on *S*, kui *U* on valem ja *S* valemi *U* sulgudeta valem.
- Valemi $\forall x U$ peatehe on $\forall$ indiviidi *x* alusel ja vahetuks alamvalemiks on *S*, kui *U* on valem ja *S* valemi *U* sulgudeta valem ja *x* on indiviid.
- Valemi $\exists x U$ peatehe on $\exists$ indiviidi *x* alusel ja vahetuks alamvalemiks on *S*, kui *U* on valem ja *S* valemi *U* sulgudeta valem ja *x* on indiviid.
- Valemi *(U)* vahetud alamvalemid ja peatehe on vastavalt valemi *U* vahetud alamvalemid ja peatehe.

Beth ei vaatle oma artiklis sulge kui tehet, tal ei ole sellele vastavat reeglit. Ülespool esitatud definitsiooni abil me saame jätta sulud vaatluse alt välja. Tänu sellele me saame näiteks nimetada *A&B* ja *C&D* valemi $((A\&B) \vee (C\&D))$ vahetuteks alamvalemiteks.

Nimetame valemi U **alamvalemiks** valemi V, kui

- leiduvad sellised valemid $U_1, .., U_n, n \geq 0$, mille puhul on õige, et valem V on valemi U_1 vahetu alamvalem, U_1 on valemi U_2 vahetu alamvalem, ..., U_n on valemi U vahetu alamvalem.
- või U on V

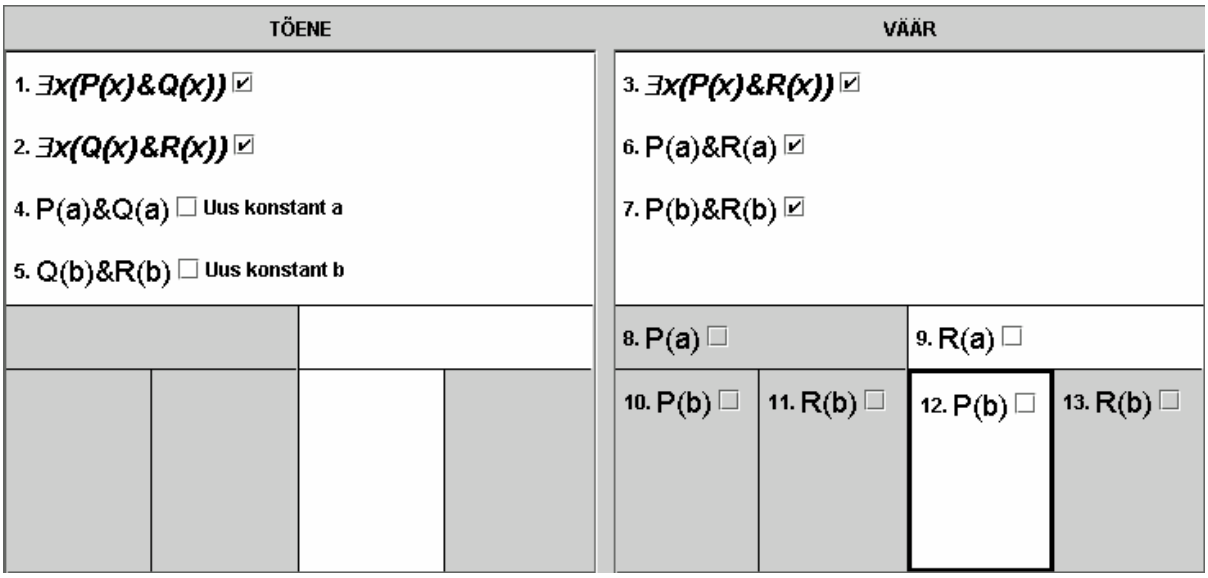
Valemitele programmis rakenduvad kitsendused. **Kitsendustega valemeid** on need valemid, mille puhul kehtivad järgnevad väited:

- Kui U on valemi alamvalem ja selle peatehe on kvantor indiviidi x alusel, siis ei leidu sellist valemit V, mis oleks U alamvalem ja mille peatehteks oleks kvantor indiviidi x alusel.
- Kui U on valemi alamvalem ja selle peatehe on kvantor indiviidi x alusel, siis leidub selline predikaat $P(x_1, ..., x_n)$, mis on valemi U alamvalem ja leidub i selline, et $x_i = x$, kus $1 \leq i \leq n$ ja $n \geq 1$.

Edasi vaatlemisel oletame, et iga valemil kehtivad antud kitsendused.

3.2. Bethi tabel

Bethi tabel koosneb kahest piirkonnast: tõesuse piirkond ja vääruse piirkond. Bethi tabel on tabelite puu, milles on vähemalt juurtabel olemas. Iga **tabel** koosneb kahest veerust: vasakust ja paremast. Vasakud veerud kuuluvad tõesuse piirkonda ja paremad veerud kuuluvad vääruse piirkonda. Iga tabel saab olla jagatud kaheks alamtabeliks ehk omada kahte alamtabelit. Tabelil on kas 0 või 2 alamtabelit. Tabelit, mille alamtabelite arv on 0, nimetame Bethi tabeli **leht-tabeliks**.



Joonis 2 Bethi tabel

Tabeli teeks nimetatakse kõik selle tabeli eellased (puu mõttes) koos tabeliga endaga.

Tabeli **alampuuks** nimetame kõiki selle tabeli järglasi (puu mõttes) koos tabeli endaga.

Tabeli **harusse** kuuluvad kõik selle tabeli eellased, tabel ise ja kõik antud tabeli järglased. **Haru veeruks** nimetatakse kõiki sellesse harusse kuuluvate tabelite vastavaid veerge.

Iga tabeli veerg võib sisaldada suvalist arvu valemeid.

Bethi tabeli konstrueerimise alguses valemeid pannakse juurtabeli veergudesse. Eelduse moodustavaid valemeid nimetame **eeldusvalemiteks**.

3.3. Ülesande lahendamine

Ülesande lahendamiseks paigutatakse sekventsi vasakpoolsed valemid Bethi tabeli vasakusse veergu ja sekventsi parempoolsed valemid paremasse veergu.

Konstrueerimise protsess toimub valemitele reeglite rakendamise kaudu. Reegli rakendamiseks valib kasutaja hiirega valemi. Valemi valimine teeb selle aktiivseks. Kui valem on valitud, avatakse võimalus valida reeglit. Programmis nimetatakse reegliteks kõiki

Bethil kirjeldatud reegleid peale (i). Reegel (i), mille järgi suletakse harusid, on programmis eraldi. Sel põhjusel rääkides reeglitest mõtleme reeglitest (ii) - (vii^b) ja lisareeglist.

Reegli rakendamist alustatakse valemi ja vastava reegli valimisega. Reegli rakendamisel paigutab kasutaja valemi vahetud alamvalemid Bethi tabelisse, vajadusel jagades tabeleid alamtabeliteks ja asendades indiviidmuutujaid konstantidega.

Kui reeglite rakendamine viib sellele, et mingis harus vastuolu olemas, siis kasutaja peab deklareerima selle haru suletuks. Kui kõik harud on suletud, siis ülesanne on lahendatud ja sekventsi samaselt tõesus on tõestatud.

Kui reeglite rakendamine viib aga sellele, et mingis harus reeglite rakendamine ei ole enam võimalik, aga vastuolu ei ilmunud, siis selle haru järgi peab kasutaja moodustama kontramudeli. Sel juhul ülesande lahendamine lõpeb sekventsi samaselt tõesuse ümberlükkamisega.

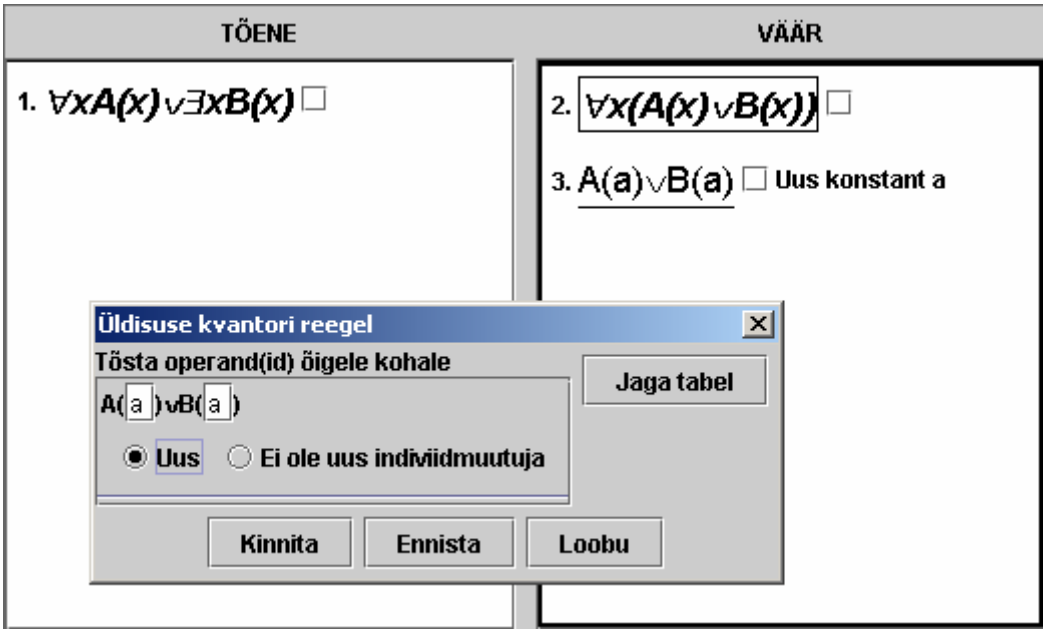
3.4. Reeglid

Nagu on selge reeglite nimetustest, on iga reegel assotsieerutud tehtega (lisareeglile vastab kvantori tehe). Valemile saab rakendada vaid selle valemi peatehtele vastavat reeglit. Kui kasutaja valib mittesobiva reegli, siis programm teatab veast.

Sobiva reegli valimise juhul avatakse reegli dialoogi aken. Reegli dialoogis näitab programm valemi vahetuid alamvalemeid. Vahetud alamvalemid leiab programm ise, sest nende saamine on triviaalne. Kasutaja ülesandeks on antud vahetute alamvalemite paigutamine Bethi tabelisse. Seda tuleb teha hiirega tõstmise abil. Reegli abil valemist tuletatud valemeid nimetame **järeldusvalemiteks**.

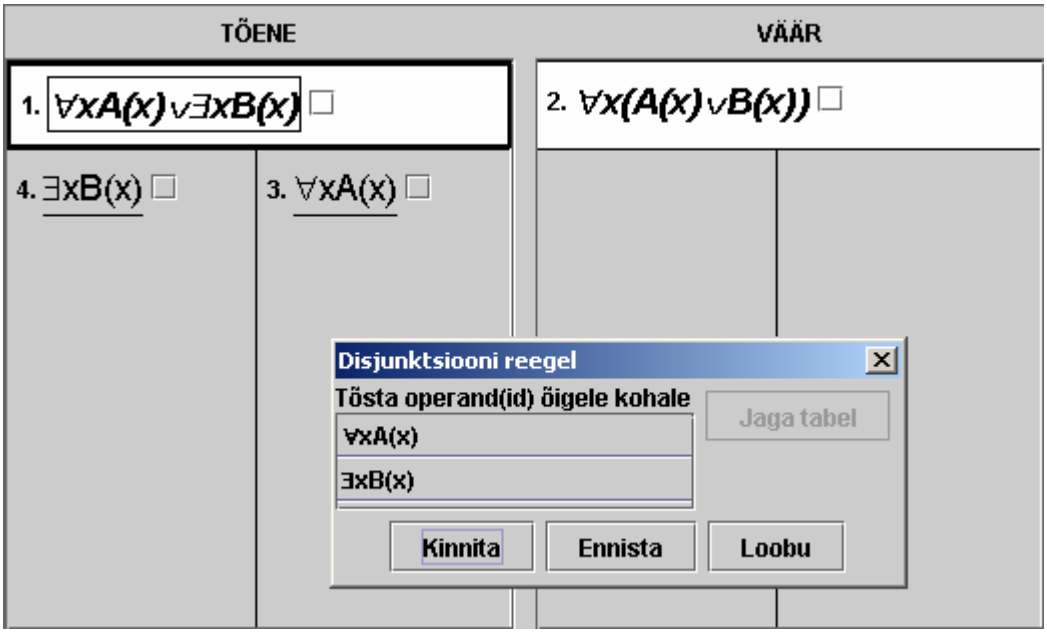
Reegli rakendamise kontekstis nimetame valemit, millele rakendatakse reeglit, selle sammu **algvalemiks**.

Kvantorite reeglite juhul pakub programm määrata seotud muutuja konstandiga asendamist. Programm küsib samuti, kas valitud konstant on uus selles harus, kuhu kasutaja hakkab vahetut alamvalemit paigutama.



Joonis 3 Kvantori reegli dialoog

Dialoogi aknas on võimalus jagada tabeli, kuhu kuulub sammu algvalem. Selle valimisel jagab programm algvalemi tabeli alampuu mittesuletud leht-tabeleid kaheks haruks.



Joonis 4 Tabeli jagamine disjunktsiooni reegli rakendamisel

Üldiselt reegli rakendamine koosneb järgmistest tegevustest (kusjuures samm 4 ja 5 võivad olla sooritatud mitu korda):

1. Analüüsitava valemi valik.
2. Reegli valik.
3. Tabeli jagamine kaheks haruks (mõnede reeglite korral).
4. Indiviidmuutuja asendamine konstandiga alamvalemis (kvantori reeglitel).
5. Alamvalemi paigutamine Bethi tabelisse.

Kasutajale lubatakse teha suvalisi tõstmisi ja jagada tabeleid isegi juhul, kui reegli järgi pole vaja midagi jagada. Reegli rakendamise õigsust kontrollitakse ainult peale nupu Kinnita vajutamist. Enne reegli rakenduse kinnitamist on kõikide uute valemite ja tabelite (jagamise puhul) seis **kinnitamata**. Kinnitamata seis tähendab siis, et nende elementide paigutus on veel kinnitamata.

Reegli rakendamise kinnitamisel teeb programm õigsuse kontrolli. Kui reegel oli vigaselt rakendatud, annab programm sellest teada ja kasutajal on võimalus oma vigu parandada. Kui rakendamine oli korrektne, siis kinnitamata Bethi tabeli elemendid muudavad oma staatuse kinnitatuks ja reegli rakendamine lõpeb seega edukalt.

Vaatleme kõiki programmis olemasolevaid reegleid.

3.4.1. Eituse reegel

Eituse reeglit lubab programm rakendada sellisele valemile, kus peatehe on eitus ehk $\neg$. Sellele reeglile vastab reegel (ii) Bethi meetodi kirjeldusest. Reegli kinnitamisel programm kontrollib, et algvalemil on täpselt üks järeldusvalem, mis vastab selle valemi vahetule alamvalemile peatehe $\neg$ alusel ja paikneb selle valemi tabeli vastasveerus.

3.4.2. Konjunktsiooni reegel

Konjunktsiooni reeglit lubatakse rakendada valemitele, mille peatehe on &. See reegel vastab reeglitele (iii^a) ja (iii^b) Bethi meetodi kirjeldusest. Reegli väljakutsel näitab programm reegli dialoogis kahte vahetut alamvalemit. Peale reegli rakendamise kinnitamist kontrollib programm järgmist.

1. Kui algvalem paikneb vasakus veerus, siis algvalemi veerus on kaks järeldusvalemit, mis vastavad algvalemi vahetutele alamvalemitele.
2. Kui algvalem asub paremas veerus, siis algvalemi tabeli alampuu mitte suletud leht-tabelid on jagatud ja nende jagatud tabelite alamtabelite paremates veergudes asuvad vastavalt esimesele ja teisele algvalemi vahetutele alamvalemitele vastavad järeldusvalemid.

3.4.3. Disjunktsiooni reegel

Disjunktsiooni reeglit saab rakendada valemile peatehtega $\vee$. Disjunktsiooni reegel vastab reeglitele (iv^a) ja (iv^b) Bethi meetodi kirjeldusest. Reegli valimisel näidatakse reegli dialoogis kahte vahetut alamvalemit. Reegli kinnitamisel kontrollitakse:

1. Kui algvalem on vasakust veerust, siis iga mitte suletud tabel, mis enne reegli väljakutset oli algvalemi tabeli alampuu leht-tabeliks, on jagatud kaheks tabeliks, mille vasakutes veergudes on algvalemi mõlemale vahetule alamvalemile vastavad järeldusvalemid. Samuti kontrollitakse, et Bethi tabelis kuskil ei ilmunud muid järeldusvalemeid.
2. Kui algvalem on paremast veerust pärit, siis Bethi tabelis on täpselt kaks tema järeldusvalemit, mis asuvad algvalemiga samas veerus.

3.4.4. Implikatsiooni reegel

Implikatsiooni reeglit lubatakse rakendada valemitele, mille peatehe on $\supset$. See reegel vastab reeglitele (v^a) ja (v^b) Bethi meetodi kirjeldusest. Reegli väljakutsel näitab programm reegli dialoogis kahte vahetut alamvalemit. Peale reegli rakendamise kinnitamist kontrollib programm järgmist.

1. Kui algvalem on vasakust veerust, siis iga mitte suletud tabel, mis enne reegli väljakutset oli algvalemi tabeli alampuu leht-tabeliks, on jagatud kaheks tabeliks ja ühel nendest paremas veerus on algvalemi esimesele vahetule alamvalemile vastav järeldusvalem ja teisel vasakus veerus on algvalemi

teisele vahetule alamvalemile vastav järeldusvalem. Samuti kontrollitakse, et Bethi tabelis kuskil ei ilmunud muid järeldusvalemeid.

2. Kui algvalem on paremast veerust pärit, siis Bethi tabelis on täpselt kaks tema järeldusvalemit, millest üks vastab esimesele algvalemi vahetule alamvalemile ja paikneb algvalemi tabeli vasakus veerus ja teisele vahetule alamvalemile vastav järeldusvalem siis paremas.

3.4.5. Üldisuse ja olemasolu reegel

Üldisuse reegel saab olla rakendatud valemile, mille peatehe on ∀. Üldisuse reegel vastab reeglitele (vi^a) ja (vi^b) Bethi meetodi kirjeldusest.

Olemasolu reeglit rakendatakse valemile, mille peatehe on ∃. Olemasolu reegel vastab reeglitele (vii^a) ja (vii^b) Bethi meetodi kirjeldusest.

Üldisuse reegel, olemasolu reegel ja lisareegel on kvantori reeglid. Nende väljakutsel näitab reeglidialoog vahetut alamvalemit, milles kasutajale pakutakse asendada indiviidmuutujat. Indiviidkonstandi valimisel kasutaja peab ka määrama, kas see konstant on uus või ei ole selles harus, mille moodustatavasse tabelisse kasutaja üritab selle asendamisega vahetut alamvalemit paigutada. Peale reegli kirjeldusele vastavuse kontrollib programm nende reeglite kinnitamisel, kas valitud konstant on tõesti uus või ei ole. Kui kasutaja valik oli vigane, siis programm annab sellest teada.

Konstant on uus antud tabelis, kui selle tabeli haru tabelites ei leidu sellist valemit, milles oleks predikaadil sama nimega argument.

Peale seda kontrollitakse iga kvantori reegli puhul, et kõik järeldusvalemid asuksid algvalemi tabeli alampuu vastavas veerus (olenevalt sellest, milles veerus algvalem ise asub).

Olemasolu ja üldisuse reegli kinnitamisel kontrollib programm järgmist.

1. Kui algvalem on peatehtega ∀ ja asub vasakus veerus või algvalem on peatehtega ∃ ja asub paremas veerus, siis programm kontrollib, et iga järeldusvalemi jaoks (mis oli moodustatud seotud muutuja konstandiga *c* asendamisel) leidub selles samas tabelis mingi valem, mis tõi selle konstandi *c*

sisse (sinna võib kuuluda ka eeldusvalem, milles konstant *c* on vaba muutuja). Erijuhuks on see olukord, kui mingi konstant oli sissetoodud algvalemi tabeli mingi ülemtabelis. Sel juhul vastav järeldusvalem peab paiknema algvalemi tabelis.

2. Kui algvalem on peatehtega ∀ ja asub paremas veerus või algvalem on peatehtega ∃ ja asub vasakus veerus, siis programm kontrollib, et algvalemi veerus asub järeldusvalem, mis toob uue konstandi sisse.

3.4.6. Uue konstandi sissetoomise lisareegel

Lisareegel saab olla rakendatud valemile, mille peatehe on kvantor. Lisareegli dialoog on sama nagu üldisuse või olemasolu reegli dialoog. Reegli kinnitamisel programm teeb kõik kvantori reeglitele vajalikud kontrollid (vt. eelmise punkti). Peale selle lisareegli kinnitamisel programm kontrollib, et algvalemi peatehe on ∀ ja see asub vasakus veerus või algvalem on peatehtega ∃ ja asub paremas veerus. Vastasel juhul antakse veateade.

Esimese konstandi sissetoomisel kontrollitakse samuti, et iga kinnitamata järeldusvalem toob sisse uue indiviidkonstandi ainult sellesse tabelisse, mis vastab järgmisele kirjeldusele:

1. Tabeli harus puudub võimalus rakendada mingi teist reeglit ja ei ole ühtki valemit, mis sisaldaks vaba indiviidmuutujat.
2. Tabeli ülemtabelitel puudub punktis 1 kirjeldatud omadus.

3.5. Tabeli sulgemine

Tabeli haru sulgemiseks kasutaja peab valima tabeli haru, milles tema arvates on vastuolu olemas, ja deklareerima selle tabeli suletuks. Programm lubab sulgeda ka tabeli, mille kõigis alamtabelites on vastuolu olemas (võimalik, et erineva valemi baasil).

Tabeli suletuks deklareerimisel kontrollib programm, et iga antud tabeli alampuu leht-tabeli tees on vastuolu. Tees vastuolu olemasolu tähendab, et tee vasakus veerus on selline valem *U* ja tee paremas veerus on selline valem *V*, mille sulgudeta valemid on identsed.

Kui kontroll annab positiivse tulemuse, siis tabeli staatus muutub suletuks. Tabeli haru staatus on suletud ka siis, kui tabeli mõlemate alamtabelite haru staatus on suletud. Suletud tabel ei osale edasi Bethi tabeli konstrueerimisel. Suletud tabeli valemitele ei saa rakendada reegleid.

3.6. Järeldus konstrueerimisest

Kasutajal on alati võimalus anda vastust, mis järeldub Bethi tabeli konstruktsioonist. Aga selle peab tegema ainult sel juhul, kui Bethi tabeli konstrueerimine viidud lõpuni ehk kui kõik tabelid on suletud või leidus selline mitte suletud haru, milles ei ole enam võimalik rakendada reegleid.

Konstrueerimise lõpetades kasutaja peab valima, kas tõestatav sekvents on samaselt tõene või ei ole, ja peale seda vajutama nupule Kinnita.

3.6.1. Samaselt tõesuse väitmine

Samaselt tõesuse kontrollimiseks programm vaatab, kas juurtabeli staatus on suletud või ei ole. Kui staatus on suletud, siis ülesanne loetakse lahendatuks, vastasel juhul antakse veateade.

3.6.2. Samaselt tõesuse ümberlükkamine

Kui kasutaja arvab, et tõestatav sekvents ei ole samaselt tõene, peab ta esitama kontramudeli. Kontramudel Bethi meetodi järgi ehitatakse mittesuletud haru alusel. Selleks kasutaja peab valima leht-tabeli. Kontramudel koosneb hulgast ja selle hulga elementidele vastavatest tingimustest (ehk atomaarsetest valemitest nende tõeväärtustega).

Kontramudeli määramiseks kasutaja peab tegema kindlaks, et valitud harus konstrueerimine on lõpuni viidud ja lisama tingimuste kasti kõik antud haru atomaarsed valemid. Hulgale lisatakse kõik nende atomaarsete valemite erinevad indiviidmuutujad.

Kontramudeli kontrollimisel programm teeb kindlaks järgmist:

1. Valitud harus ei ole võimalik rakendada reegleid.
2. Valitud harus ei ole vastuolusid.
3. Iga atomaarsete valemite predikaatide argumendiks olev indiviid on hulgas olemas.
4. Hulgas iga elemendi jaoks on tingimuses selline atomaarne valem, et see element oleks selle valemi predikaadi argumendiks.
5. Iga tingimustes olev atomaarne valem on valitud harus olemas ja see asub õiges veerus vastavalt oma tõeväärtusele.
6. Iga harus olev atomaarne valem on esitatud tingimuste hulgas oma õige tõeväärtusega.

3.6.3. Järelduse väitmine mittelahendatavuse juhul

Nagu oli juba ülespool mainitud, ei saa kõiki valemeid tõestada või ümber lükata Bethi semantiliste tabelite abil. Sellel juhul leidub niisugune haru, mille konstrueerimisel ei teki momenti, kus ei saa enam ühtki reeglit rakendada.

Programmis on aga see juhtum ettenähtud. Ja kasutaja võib anda vastuse ka selles olukorras, teatades programmile, et konstrueerimine jätkub lõpmatult. On selge, et valemite samaselt tõesust ei ole võimalik kontrollida predikaatarvutuse mittelahenduvuse tõttu. Võimalik on aga kontrollida kontramudeli sobivust.

Kontramudeli kontroll toimub sellel juhul teisiti, erinevalt juhust, kui konstrueerimine võib olla lõpuni viidud.

Kontramudeli defineerimisel kasutaja määrab hulga ja tingimused otse sisendist ja programm kontrollib neid, kasutades loogiliste tehete tähendusi. Olgu kontramudeliks näiteks

hulk $X=\{a_1, ..., a_n\}$ ja tingimused olgu järgmised: $P_1(a_{11}, ..., a_{1s}), ..., P_m(a_{m1}, ..., a_{mt})$ on tõesed ja $R_1(a_{11}, ..., a_{1u}), ..., R_n(a_{n1}, ..., a_{nv})$ on väärad.

Siis eeldusvalemite tõeväärtus peab vastama nende paigutusele (tõesed vasakus veerus ja väärad paremas veerus).

Valemi tõeväärtuse arvutamine toimub järgmise rekursiivse algoritmi abil.

1. Atomaarne valem on
   - tõene, kui ta leidub tõeste atomaarsete valemite hulgas mudeli tingimustes
   - väär, kui ta leidub väärade atomaarsete valemite hulgas mudeli tingimustes
   - defineerimata teistel juhtudel
2. $\neg U$ alamvalem on
   - tõene, kui U on väär
   - väär, kui U on tõene
   - defineerimata, kui U on defineerimata
3. $U\&V$ alamvalem on
   - tõene, kui U ja V on tõesed
   - väär, kui kas U või V on väär
   - defineerimata teistel juhtudel
4. $U \curlyvee V$ alamvalem on
   - tõene, kui U või V on tõene
   - väär, kui U ja V on väärad
   - defineerimata teistel juhtudel
5. $U \supset V$ alamvalem on
   - tõene, kui U on väär või V on tõene

- väär, kui U on tõene ja V on väär
- defineerimata teistel juhtudel

6. $\forall xU$ alamvalem (kus x on suvaline indiviid) on
   - tõene, kui iga elemendiga mudeli hulgast indiviidi x asendamine valemis U annab valemi, mis on tõene
   - väär, kui leidub selline element mudeli hulgast, millega indiviidi x asendamine valemis U annab valemi, mis on väär
   - defineerimata teistel juhtudel
7. $\exists xU$ alamvalem (kus x on suvaline indiviid) on
   - tõene, kui leidub selline element mudeli hulgast, millega indiviidi x asendamine valemis U annab valemi, mis on tõene
   - väär, kui iga elemendiga mudeli hulgast indiviidi x asendamine valemis U annab valemi, mis on väär
   - defineerimata teistel juhtudel
8. (U) alamvalem on
   - tõene, kui U on tõene
   - väär, kui U on väär
   - defineerimata teistel juhtudel

Kui iga vasakust veerust eeldusvalemi tõeväärtus on tõene ja iga paremast veerust eeldusvalemi tõeväärtus on väär, siis eeldus on korrektne ja kontramudel on korrektne. Vastasel juhul teatatakse kasutajale veast.

Programmis on samuti võimalik väita valemite samaselt tõesust, aga sellel juhul programm ei kontrolli õigsust, jättes selle õppejõule.

3.7. Lahendamisel diagnoositavad vead

Lahenduse protsessi hindamise huvides on programmis defineeritud lahenduse vigade tüübid. Lahenduse viga on idee järgi see viga, mida kasutaja saab teha mittepiisava teadmiste taseme tõttu. Lahendamise vigu loetakse lahendamise käigus ja arvestatakse ülesande hindamisel. Iga lahendamise vea tüübi jaoks määrab õppejõud ülesannete koostamisel trahvipunktid. Vea tegemisel näidatakse ekraanil veateadet ja näidatakse, kui palju punkte selle vea eest võetakse maha. Vea tekkimisel tehtav operatsioon katkestatakse.

Allpool on esitatud kõik programmi poolt diagnoositavad lahenduse vigade tüübid.

3.7.1. Ilma vasturääkivusteta tabeli sulgemine

See viga tekib juhul, kui kasutaja üritab sulgeda tabelit, mille alampuus leidub selline leht-tabel, mille tee ei sisalda vastuolu.

3.7.2. Mittesobiva reegli rakendamine

Viga tekib juhul, kui kasutaja üritab rakendada mingit reeglit valemile, mille peatehe ei vasta sellele reeglile.

3.7.3. Vigane reegli rakendamine

Vigane reegli rakendamine on suvaline viga, mis leitakse reegli rakendamise kinnitamisel. See viga klassifitseeritakse Bethi reegli rikkumisena. Selle vea teade sõltub kontekstist ja rakendatavast reeglist. Näiteks harus olemasoleva elemendi uueks deklareerimine kuulub ka siia alla.

3.7.4. Mittesuletud tabeli puhul samaselt tõesuse väitmine

See viga tekib juhul, kui kasutaja väidab valemi samaselt tõesust juhul, kui leidub selline tabel, mis ei ole suletud.

3.7.5. Kontramudelisse mitteatomaarse valemi lisamine

Kontramudeli tingimusteks saavad olla ainult atomaarsed valemid. Mitteatomaarse valemi kontramudelisse lisamine tekitab selle vea.

3.7.6. Kontramudelina valitud haru sisaldab vasturääkivusi

See viga tekib juhul, kui kasutaja kinnitab kontramudeli sellise haru jaoks, kus on olemas vastuolu.

3.7.7. Bethi tabeli konstrueerimine ei ole viidud lõpuni

See viga tekib siis, kui kasutaja kinnitab kontramudeli sellise haru jaoks, milles on võimalik reeglite rakendamine.

3.7.8. Bethi tabeli konstrueerimine ei ole viidud lõpuni (osa konstantidest on kasutamata)

See viga tekib siis, kui kasutaja kinnitab kontramudeli sellise haru jaoks, milles on võimalik reeglite (vi^a), (vii^b) või lisareegli rakendamine.

3.7.9. Kontramudeli hulk ei sisalda osa kontramudelis olemasolevatest elementidest

See viga tekib kontramudeli kinnitamisel juhul, kui leidub mingi element, mis on tingimustes atomaarse valemi predikaadi argument, aga ei ole hulgale lisatud.

3.7.10. Kontramudeli hulk sisaldab liigseid elemente

See viga kirjeldab olukorda, millal kontramudeli hulgas sisaldav element ei osale tingimuste määramises.

3.7.11. Vigane kontramudel

Juhul, kui kontramudel deklareeritakse haru alusel, milles konstrueerimine on viidud lõpuni, siis see viga tähendab ühe järgmistest

- Mingi tingimustes olev atomaarne valem puudub tema tõeväärtusele vastavas valitud haru veerus.
- Mingi valitud harus olev atomaarne valem puudub mudeli tingimuste hulgast.

Juhul, kui kasutaja väidab, et Bethi tabeli konstrueerimine läheb lõpmatusse, siis see viga tähendab, et konstrueeritud mudelis on mingil eeldusvalemil selle valemi paigutusele mittevastav tõeväärtus.

4. Programmi lühikirjeldus

Programm koosneb kahest rakendusest

- Koostaja rakendus,
- Lahendaja rakendus,

Siin on esitatud nende lühikirjeldused.

Koostaja rakendus on programm, mis on ettenähtud ainult õppejõule. Selle rakenduse abil luuakse ülesannete kogud, kogude juures määratakse trahvipunktid iga lahendamise vea tüübi jaoks, defineeritakse ülesannete kogu lahendamise režiim (kas praktikumi või kontrolltöö režiim). Kontrolltöö režiimi järelvaatuse taseme tõstmiseks saab defineerida PIN-koode, ilma milleta tudengid ei saa lahendust alustada või jätkata. Koostaja rakenduses on samuti vahend lahenduste tulemuste vaatamiseks.

Lahendaja rakendus on ettenähtud ülesannete lahendamiseks. Ülesanded hoitakse koostaja rakenduse abil loodud ülesannete failides. Kasutajal on aga võimalus ise ka ülesandeid koostada. Lahenduse tulemusi (või vahetulemusi) võib kasutaja salvestada lahenduse faili. Ülesannete lahendamine on samm-sammuline Bethi tabeli konstrueerimine, mis viib lausearvutuse või predikaatarvutuse valemi samaselt tõesuse tõestamisele või ümberlükkamisele.

Kokkuvõte

Magistritöö tulemusena oli disainitud ja realiseeritud Bethi tabelite konstrueerimise keskkond. Keskkond kujutab ennast loogiliste valemite samaselt tõesuse tõestamise või ümberlükkamise süsteemi Bethi semantiliste tabelite meetodiga. Magistritöös tehakse edukas katse ehitada sellise tõestamise keskkond, milles tõestamine on palju kiirem ja mugavam kui paberil. Kuna süsteem on mõeldud õpiprogrammina, selle põhikasutus on ülesannete lahendamine. Ülesannete lahendamisel süsteem kontrollib kõik tõestuse konstrueerimiseks kasutaja poolt vastuvõetavad tegevused ja arvestab lahendamise vigu.

Tõestamine toimub sammhaaval reeglite rakendamise kaudu. Iga reegli rakendamine toimub kolmes etapis: Bethi tabelis valemi valik, Bethi meetodi reegli valimine, vahetute alamvalemite alusel saadud järeldusvalemite paigutamine Bethi tabelisse. Ümberlükkamise puhul peab kasutaja esitama ka kontramudeli.

Ülesannete lahendamiseks on mõeldud kaks režiimi: praktikumi ja kontrolltöö režiim.

Programmis on loodud ka vahendid ülesannete koostamiseks ja ülesannete lahendamise tulemuste vaatamiseks.

Loodud programm on eestikeelse kasutajaliidesega, aga võib toetada ka muid keeli ilma koodi muutmisteta. Programmi kasutus oli proovitud aine „Matemaatiline loogika ja algoritmiteooria“ aine praktikumides Tartu Ülikoolis.

Programm on loodud JDK1.4.2 arenduskeskkonnas. Programmi töö oli testitud operatsioonisüsteemide Windows 2000 ja Windows XP all.

Learning program for Beth's tableaux construction

Master work

Anton Golovko

Abstract

The work is based on the semantic tableaux method [2]. The method is used to prove or disprove logic sequents in propositional and predicate logic. Usually the construction of proof by this method becomes too complicated when using paper. The main goal of this work is to develop an interactive system for proving formulas using this method and create learning program for it.

The main use of this program is task solving. A task in the program is a sequent that should be proved or refuted. The solution is a stepwise process that uses a set of rules defined by Beth's method. A rule application step consists of selection of a formula in the Beth's tableau, selection of an applicable rule and addition of formula's deductions to the Beth's tableau.

In case of refuting of a sequent user must provide a counter model.

Each step user performs is checked by the program in order to guarantee the correct flow of the proof and to provide an ability to assess the results of the task solution.

The solution of a task can go either in practice or exam mode.

The program contains a teacher module that is used for the creation of task collections and viewing the results of solutions made by students.

Program has Estonian graphical user interface, but can support other languages also without any changes in code. Program was used on practical lessons of „Mathematical Logic and Theory of Algorithms" in the University of Tartu.

Program was created using JDK1.4 development environment and was tested under Windows XP and Windows 2000.

Kasutatud kirjandus

[1] E. W. Beth. The foundations of mathematics. Amsterdam, 1959.

[2] Математическая теория логического вывода. Москва, 1967.

[3] R.Prank. Matemaatiline loogika ja algoritmiteooria. Tartu Ülikooli kirjastus, 2004.

[4] T. Tamme, T. Tammet, R. Prank. Loogika: Mõtlemisest tõestamiseni. Tartu, 1997.

Lisad

Lisa 1. Bethi tabelite konstrueerimise õpiprogrammi „BETH“ kasutajajuhend

Bethi tabelite konstrueerimiseks õpiprogramm „BETH“

KASUTAJAJUHEND

Sisukord

1. Programmi suunitlus

„BETH" on programm Bethi semantiliste tabelite meetodi õpetamise jaoks. Programmi põhikasutuseks on ülesannete lahendamine. Iga ülesanne programmis on sekvents, mida tuleb tõestada või ümber lükata Bethi semantiliste tabelite meetodiga. Tõestamisel konstrueeritakse Bethi tabel ja esitatakse kontramudel, kui vaja. Bethi tabeli konstrueerimiseks rakendatakse Bethi tabeli valemitele meetodi reegleid. Reegli rakendamise protsess koosneb valemi valimisest, reegli valimisest ja uute järeldusvalemite Bethi tabelis paigutamisest.

Ülesannete lahendamisel kontrollitakse iga kasutaja poolt tehtud samm ja vigade puhul antakse koheselt tagasiside. Diagnoositud vigu kasutatakse ka lahenduse hindamiseks.

2. Programmi failid ja käivitamine

Ülesannete lahendamise programm asub pakis beth1.2.2_user.jar. Programmi käivitamiseks tuleb sisestada operatsioonisüsteemi järgmine käsurida:

```
java –classpath beth1.2.2_user.jar beth.gui.MainFrame
```

või

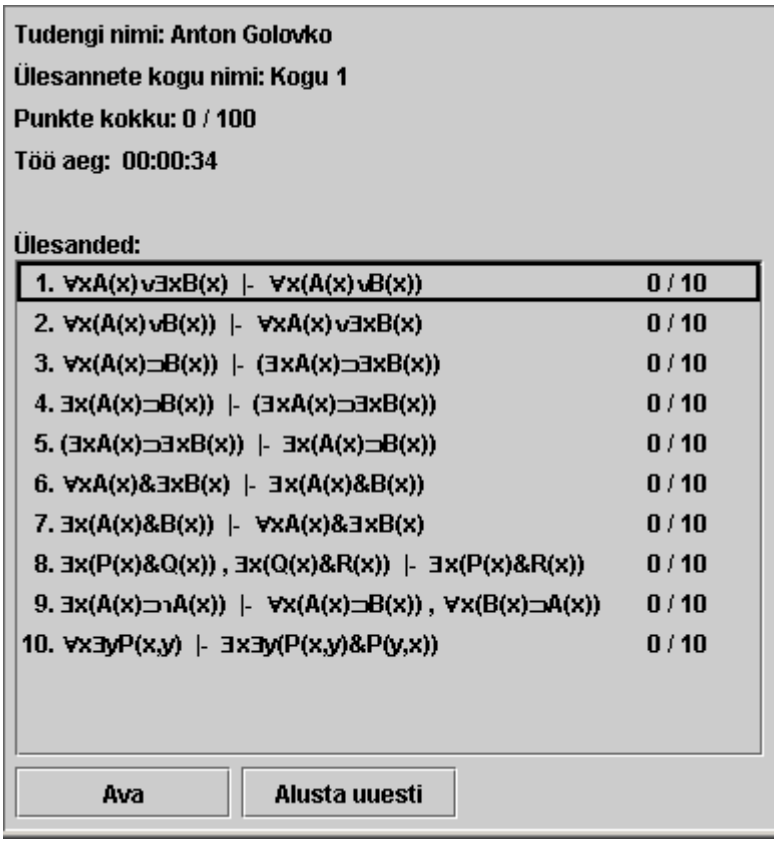
```
java –jar beth1.2.2_user.jar
```

Programmiga töötamiseks arvutil peab olema installitud Java Runtime Environment versiooniga vähemalt 1.4.

3. Ülesannete avamine

Ülesannete lahendamiseks tuleb avada ülesannete fail. Selleks tuleb valida „Ava ülesannete fail..." menüüst „Fail". Ülesannete fail on fail laiendiga `btt`. Ülesannete failis hoitakse ülesannete kogu, mis võib sisaldada suvalist arvu ülesandeid. Faili laadimisel küsib programm kasutaja nime. Kasutaja nimi peab vastama õppejõu nõudmistele (tavaliselt ees- ja perekonnanimi). Kui nimi on sisestatud, ilmub programmi akna vasakus osas ülesannete kogu paneel, mis sisaldab ülesannete nimekirja ja järgmist üldinfot:

- lahendaja nimi
- ülesannete kogu nimetus
- korjatud punktide arv
- taimer



Joonis 5 Ülesannete kogu paneel

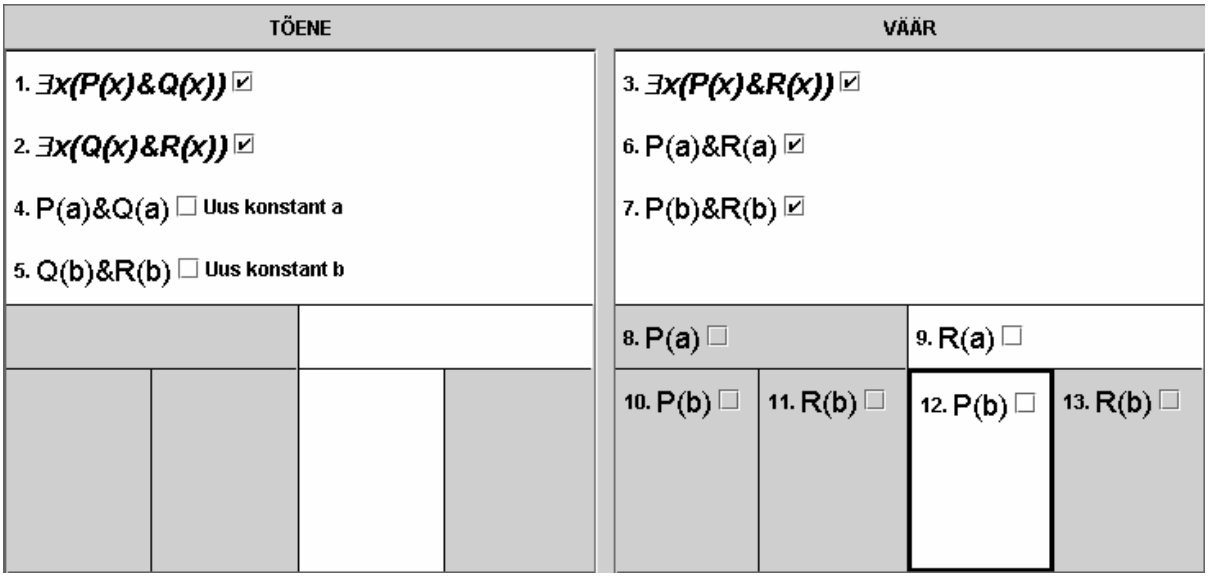
Ülesanne on sekvents, mille samaselt tõesust tuleb tõestada või ümber lükata. Ülesande juures näidatakse, kui palju punkte sai kasutaja lahenduse eest. Ülesande lahendamiseks avamiseks peab kasutaja tegema topeltklõpsu ülesande peal või valima ülesande hiirega ja vajutama nupule „Ava“. Kui kasutaja tahab ülesande lahendust uuesti alustada, tuleb kasutada nupu „Ava uuesti“. Sellele nupule vajutamisel vastava ülesande eelmine lahendus kaotatakse.

Ühe ülesande lahendamisel võib kasutaja julgesti lülitada üle teisele ülesandele, isegi kui ülesande lahendus ei ole veel lõpuni viidud. Sest programm säilib kõiki lahendustes tehtavaid muudatusi.

4. Ülesande lahendamine

4.1. Bethi tabel

Bethi tabel koosneb kahest piirkonnast: tõesuse piirkond ja vääruse piirkond. Bethi tabel on tabelite puu, milles on vähemalt juurtabel olemas. Ülesande lahendamise alguses on ainult üks tabel. Iga tabel koosneb kahest veerust: vasakust ja paremast. Vasakud veerud kuuluvad tõesuse piirkonda ja paremad veerud kuuluvad vääruse piirkonda.



Joonis 6 Bethi tabel programmis

Tabeli veerud võivad sisaldada valemeid. Kõik valemid Bethi tabelis on nummerdatud. Iga valemi kõrval on üks kast, mida kasutaja võib kasutada oma otsatarbeks, et märkida, et valem on juba analüüsitud.

4.2. Algseis

Ülesande avamisel ilmub programmi akna paremal poolel Bethi tabel, mille vasakusse veergu on paigutatud sekventsi vasakpoolsed valemid ja paremasse veergu vastavalt sekventsi parempoolsed valemid.

Need eeldusvalemid kuvatakse kursiiviga.

4.3. Tabeli valimine

Tabelit saab programmis valida hiireklõpsuga. Valitud tabeli veerud on eristatud valge värviga. Kui valitakse tabel, siis tema ülemtabelid on ka valitud.

4.4. Valemi valimine

Valemi valimiseks peab kasutaja tegema hiireklõpsu valemi peal. Siis programm valib ka haru, milles asub antud valem. Kui valemil on järeldusvalemid olemas, näitab programm valitud valemi järeldusvalemeid allakriipsutatult.

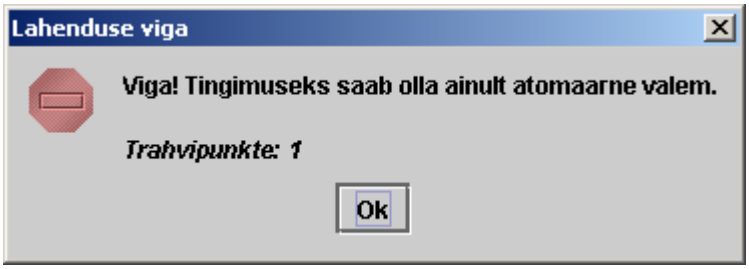
4.5. Punktid

Iga ülesande jaoks defineerib õppejõud ülesannete kogus selle ülesande kaalu. Ülesande lahendamist lõppuni viies saab kasutaja need punktid. Nendest kogutud punktidest lahutatakse aga trahvipunktid iga tehtud viga eest. Kui vigade eest lahutatav punktide arv on suurem kui ülesande kaal, saab kasutaja ülesande lahenduse eest 0 punkti. Lahendamise punktid ja ülesande kaal näidatakse ülesande juures ülesannete kogu nimekirjas.

4.6. Lahendamise vead

Programmis on defineeritud hulk lahendamisvigade tüüpe, mille tegemisel võetakse kasutajalt vastav trahvipunktide arv maha. Trahvipunktide määrad iga veatüübi jaoks

defineerib õppejõud. Vea tekkimisest informeerib programm kasutajat veateade näitamisega, milles kirjeldatakse see viga ja näidatakse, kui palju trahvipunkte on maha võetud.



Joonis 7 Vea dialoog

Vea tekkimisel tehtav operatsioon katkestatakse. Ülesande lahendamise paneelis kuvatakse jooksev trahvipunktide arv alumisel paremal nurgal.

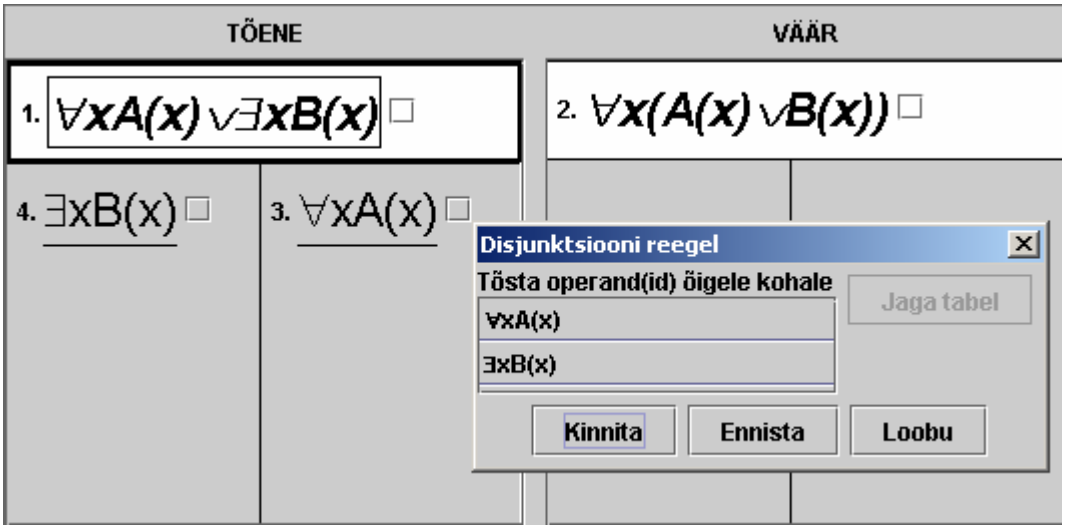
4.7. Reeglite rakendamine

Bethi tabeli konstrueerimisel peab kasutaja rakendama Bethi tabelis olevatele valemitele reegleid. Lausearvutusega seotud reeglid paiknevad menüüs „Lausearvutus“ ja predikaatarvutusega seotud reeglid menüüs „Predikaatarvutus“.

Reegli rakendamiseks valemile peab valima kasutaja selle valemi. Peale selle saab kasutaja valida sobiva reegi programmi menüüdest. Kui reegel on valitud, avatakse reegli dialoog. Mittesobiva reegli valimise puhul näidatakse veateadet. Reegli dialoog näitab valitud reegli alamvalemeid. Reegli rakendamine peab toimuma järgmiselt:

- tabeli jagamine alamtabeliteks (nupu „Jaga tabel“ kasutades), kui vaja
- predikaatarvutuse reeglite puhul indiviidmuutuja konstantse parameetriga asendamine. Siin juures peab kasutaja näitama, kas asendav konstant on uus või ei ole selles harus, mille moodustatavasse tabelisse tahab kasutaja selle alamvalemi paigutada
- alamvalemi tõstmine hiirega Bethi tabelis õigesse tabeli veergu
- Nupule „Kinnita“ vajutamine.

Teine ja kolmas samm siin võivad olla sooritatud mitu korda, sõltudes sellest, kui palju järeldusvalemeid tahab kasutaja Bethi tabelisse paigutada.



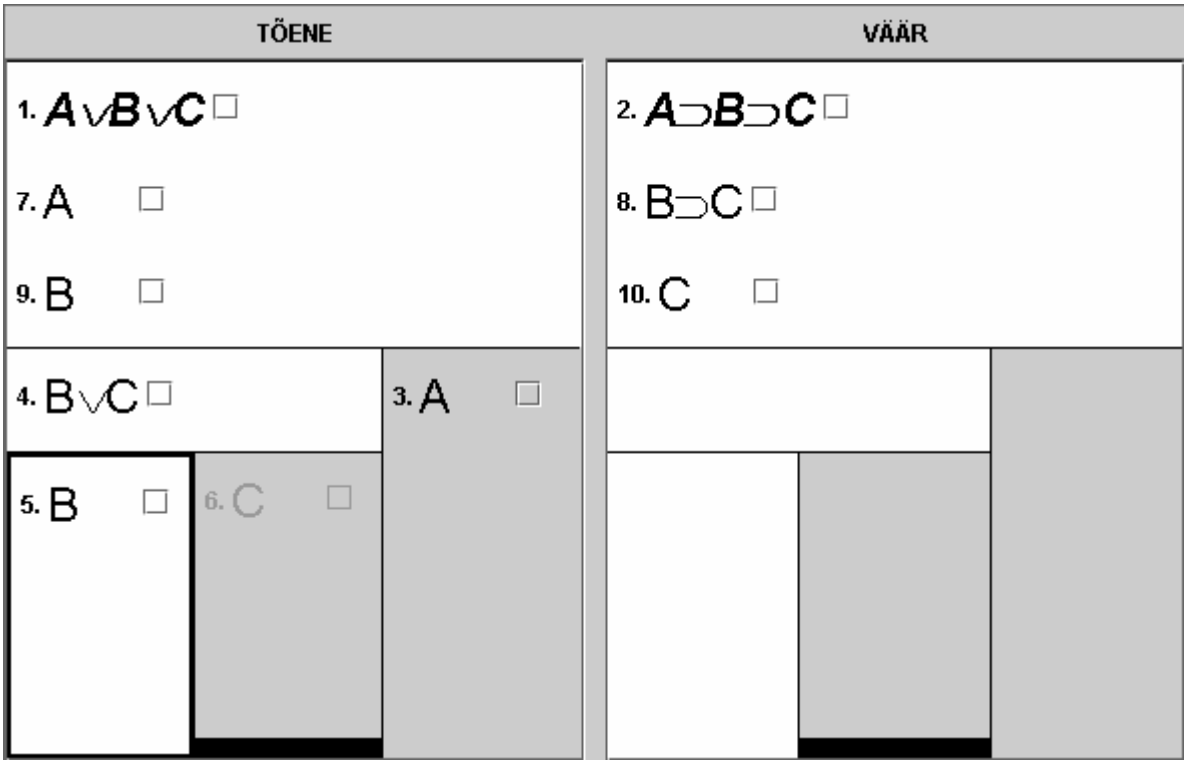
Joonis 8 Disjunktsiooni reegli dialoog

Kasutajale lubatakse teha suvalisi tõstmisi ja jagada tabeleid isegi juhul, kui reegli järgi pole vaja midagi jagada. Reegli rakendamise õigsust kontrollitakse ainult peale nupu „Kinnita“ vajutamist. Kui peale sellele nupule vajutamist programm leiab, et reegel oli vigaselt rakendatud, annab siis sellest kasutajale teada veateatega ja reegel jääb rakendamata.

Kasutajal on alati võimalus reegli rakendamisel üks samm tagasi teha. Selleks tuleb kasutada nupu „Ennista“. Kui kasutaja tahab reegli rakendamise katkestada ja taastada seisu, mis oli enne reegli väljakutset, saab ta vajutada nupule „Loobu“.

4.8. Tabeli sulgemine

Tabeli sulgemiseks tuleb valida tabel ja valida menüüst „Tabel“ punkt „Sule tabel“. Kui valitud tabelit sulgeda ei saa, teatatakse kasutajale veast. Kui tabeli sulgemine on võimalik, siis tabel ja selle tabeli kõik alamtabelid blokeeritakse ja tabeli haru alla ilmub must riba. Suletud tabel ei osale edasi Bethi tabeli konstrueerimisel. Suletud tabeli valemitele ei saa rakendada reegleid.



Joonis 9 Bethi tabel suletud haruga

4.9. Sammu tagasi võtmine

Kasutajal on alati võimalus teha samm tagasi. Selleks tuleb kasutada menüüst „Tabel“ nuppu „Samm tagasi“. Sammuks loetakse reegli rakendamist või tabeli sulgemist. Sammu tagasivõtmisega ei ole võimalik trahvipunktidest lahti saada.

4.10. Vastuse andmine

Lahendamise lõpetamiseks peab kasutaja andma vastuse, kas ülesande moodustav sekvents on samaselt tõene või ei ole. Selleks peab ta valima õige raadionupu ja vajutama nupule „Kinnita lahendus“. Juhul kui on valitud „Ei ole samaselt tõene“, siis kasutaja peab ka koostama kontramudeli ja alles siis vajutama nupule „Kinnita lahendus“.

Kui ülesande lahendus oli edukalt kinnitatud, siis kasutajale antakse punkte lahenduse eest ja ülesannete kogu paneelil ülesande juures ilmub linnuke ja märgend, mis näitab, kas sekvents on samaselt tõene (T) või ei ole (V).

4.11. Samaselt tõesuse väitmine

Samaselt tõesuse väitmiseks valitakse „Samaselt tõene“ raadionupp ja vajutatakse nupule „Kinnita lahendus“. Kui Bethi tabeli seisust ei järeldu samaselt tõesus, siis programm näitab veateadet.

4.12. Samaselt tõesuse ümberlükkamine

Samaselt tõesuse ümberlükkamiseks tuleb märkida raadionupu „Ei ole samaselt tõene“, konstrueerida kontramudel ja vajutada nupule „Kinnita lahendus“. Kui kasutaja esitab vigase kontramudeli, siis programm näitab veateadet.

Kontramudeli esitamiseks peab kasutaja valima Bethi tabelis haru, mille alusel see kontramudel konstrueeritakse. Haru moodustav tabel peab olema Bethi tabeli leht-tabeliks (puu mõttes).

Kontramudeli paneelis peab kasutaja defineerima hulga, mis on mudeli kandjaks. Selleks peab ta sisestama tekstisisendisse hulga elemendid. Tingimuste defineerimiseks peab kasutaja valima valitud harust atomaarseid valemeid ja vajutama nupule „Lisa Bethi tabelist“. Mitteatomaarse valemi tingimuste hulgasse lisamine tekitab vea.

Kui kõik vajalikud elemendid on hulgas ja kõik tingimused määratud, saab lahenduse kinnitada. Puuduliku kontramudeli kinnitamine tekitab vea.

4.13. Järelduse väitmine mittelahendatavuse juhul

Programmis on võimalik ka anda vastust juhul, kui Bethi tabeli meetodiga ei saa sekventsi samaselt tõesust tõestada või ümber lükata. Selleks peab kasutaja valima märkeruudu „Konstrueerimine läheb lõpmatusse“. Peale selle võib kasutaja kas väidata

samaselt tõesust (sellel juhul peale „Kinnita lahendus“ nupule vajutamist programm ei kontrolli vastuse õigsust) või esitada kontramudeli ja sekventsi ümber lükata.

Sellel juhul on kasutajal kontramudeli konstrueerimisel vabadus välja mõelda suvaline kontramudel, kasutades selleks hulga ja tingimuste tekstisisendeid. Nupule „Kinnita lahendus“ vajutamisel kontrollib programm, et esitatud mudel on tõesti antud sekventsi kontramudeliks. Kui ei ole, programm näitab veateadet.

Kui kasutaja arvates Bethi tabeli konstrueerimine läheb lõpmatusse, programm ei omista kasutajale punkte lahenduse eest, delegeerides selle lahenduse sobivuse kontrolli õppejõule. Sel juhul programm ainult võtab vastuse vastu ja ei loe ülesannet lahendatuks (sel põhjusel selline lahendus võib olla ka edasi jätkatud).

5. Ülesannete lahenduste salvestamine

Kasutaja saab salvestada oma lahendust faili. Selleks valitakse menüüst „Fail“ punkt „Salvesta lahendus“. Lahenduse salvestamisel säilitakse jooksev seis iga ülesande jaoks kogust. Lahenduse fail on fail laiendiga `bts`. Selle faili võib kasutaja pärast avada ja ülesannete lahendamist jätkata.

Lisa 2. Tööle lisatud CD sisu kirjeldus

Käesolevale tööle lisandub CD, mille peal asub järgmine:

1. Kataloogis `papers` on käesolev töö ja kasutajajuhend DOC ja PDF formaadis.
2. Kataloogis `release` asub
    a. Õppejõududele ettenähtud pakk `beth1.2.2_teacher.jar`, mis sisaldab kogu programmi.
    b. Tudengitele ettenähtud pakk `beth1.2.2_user.jar`, mis sisaldab ainult lahendaja rakenduse.
    c. Näidisülesannetega kataloog `sampletasks`.
3. Kataloogis `src` asub programmi lähtekood ja programmis kasutatavate tekstide fail.

Koostaja rakenduse käivitamine:

```
java -classpath <jar-fail> beth.edit.EditFrame
```

Lahendaja rakenduse käivitamine:

```
java -classpath <jar-fail> beth.gui.MainFrame
```