TARTU ÜLIKOOL
Arvutiteaduse instituut
Informaatika õppekava

Kaarel Kütt

Karnaugh’ kaardi lahendaja

Bakalaureusetöö (9 EAP)

Juhendaja Margus Rosin

Tartu 2022

Karnaugh’ kaardi lahendaja

Lühikokkuvõte:

Karnaugh’ kaart on tabelina esitatav abivahend, mida kasutatakse loogikavalemite lihtsustamisel. Varasemalt oli Karnaugh’ kaardi abil loogikavalemi minimaalse kuju leidmiseks mitmeid rakendusi, kuid nende kõigi puuduseks oli neljast enama muutujaga loogikatehete visuaalse lahenduse kuvamine, mis ei vasta tavalistele Karnaugh’ kaardi reeglitele. Selle lõputöö eesmärk oli luua rakendus, mis võimaldaks kuvada viie või kuue muutujaga võrrandite visuaalset lahendust kolmemõõtmeliselt ja parendaks olemasolevate lahenduste funktsionaalsust.

Võtmesõnad:

Karnaugh’ kaart, digitaalne loogika

CERCS: P175 - Informaatika, süsteemiteooria

Karnaugh map minimizer

Abstract:

Karnaugh map is a tabular tool used to simplify logic formulas. There were applications for finding the minimum cover of a logic formula using the Karnaugh map method before, but they had a disadvantage. When the number of variables exceeded four, they displayed visual solutions that did not follow the usual rules of Karnaugh map. The aim of this paper was to create an application that would display equations with more than four variables in three dimensions and improve the functionality of existing applications.

Keywords:

Karnaugh map, digital logic

CERCS: P175 - Informatics, systems theory

Sisukord

Sissejuhatus

Tartu Ülikoolis õpetatakse ainet „Digitaalne loogika“, kus tutvustatakse ka loogikavalemite lihtsustamist ja valemite minimaalset disjunktiivset normaalkuju. Selle aine raames kasutatakse rakendust Karnaugh Map Minimizer [1]. Sellel lahendusel esineb aga mitmeid miinuseid, mis raskendavad kasutamist ja segavad sujuvat kasutuskogemust. Tarkvaral esinevad näiteks järgnevad puudused: argumentide paigutust ei saa muuta, viie ja kuue muutuja korral pole lahenduse välja lugemine intuitiivne, samuti muutub rakendus vahel vea tõttu kasutuskõlbmatuks.

Lisaks Karnaugh Map Minimizerile on praegu olemas veel rakendusi, mis pakuvad Karnaugh’ kaardi lahendamist, kuid ka neil esineb puuduseid. Näiteks on olemas veebirakendus Logic Circuit Simplification [2]. Kuigi rakendus töötab stabiilsemalt, esinevad ka sellel sarnased miinused: lahendusi ei kuvata suuremate muutujate arvu korral ruumiliselt ning lahendust on kaardilt keeruline välja lugeda. Lisaks kuvatakse lahendust Karnaugh’ kaardil ainult korraks ja muudatuste tegemiseks tuleb kõik väärtused uuesti sisestada. Eelloetletud põhjuste tõttu muutub rakenduse kasutamine tülikaks ning ka aega nõudvaks.

Selle bakalaureusetöö eesmärk on luua rakendus, mida on võimalik kasutada Karnaugh’ kaardi lahendamiseks ja millel ei esine seniste lahenduste kitsaskohti. Valmis rakendus on kasutajasõbralik ja kuvab kogu vajalikku informatsiooni ühel vaatel. Muudatusi väärtustustes saab teha jooksvalt ning kõik leitud osavalemid on kasutaja jaoks esile tõstetud ka Karnaugh’ kaardil. Samuti peaks rakendus kuvama viie ja kuue muutuja korral lahendust ruumiliselt ehk näiteks kuue sisendi puhul 4x4x4 kuubina.

Bakalaureusetöö esimeses osas tutvustatakse Karnaugh’ kaardi põhimõtet ja selle kasutamist. Teises osas kirjeldatakse ja analüüsitakse olemasolevaid lahendusi. Kolmandas osas tuuakse välja eelnevalt rakendusele määratud funktsionaalsed ja mittefunktsionaalsed nõuded. Neljandas osas kirjeldatakse valminud rakendust ja tutvustatakse selle tööd.

Mõisted ja terminid

Primaarimplikant [3]: Plokk Karnaugh' kaardil, mis ei sisaldu üheski teises plokis. Plokk peab sisaldama nullist erinevaid väärtuseid (SOP lahenduse korral ühest erinevaid väärtuseid). Kusjuures ploki mõõtmed peavad olema kahe astmed.

Minterm [4]: Kõiki muutujaid sisaldav väärtustuste komplekt, mille väljund on üks ehk element Karnaugh' kaardil, mille väärtuseks on üks.

Maksterm [4]: Kõiki muutujaid sisaldav väärtustuste komplekt, mille väljund on null ehk element Karnaugh' kaardil, mille väärtuseks on null.

SOP [4]: (*Sum Of Products*) Implikantidest koostatud valem, kus implikandi liikmed on kokku pandud kasutades konjunktsioone ning saadud valemid omakorda kokku pandud kasutades disjunktsioone. Karnaugh' kaardi puhul kasutatakse seda lahendust juhul kui implikandid koosnevad nullist erinevatest väärtustest (üks või *don-not-care* väärtus).

POS [4]: (*Product Of Sums*) Implikantidest koostatud valem, kus implikandi liikmed on inverteeritud ja kokku pandud kasutades disjunktsioone ning saadud valemid omakorda kokku pandud kasutades konjunktsioone. Karnaugh' kaardi puhul kasutatakse seda lahendust juhul kui implikandid koosnevad ühest erinevatest väärtustest (null või *don-not-care* väärtus).

Loogikavärat [4]: Digitaalseid signaale kombineeriv elektroonika komponent, mis kasutab binaarloogika tehteid.

1. Karnaugh' kaart ja selle kasutamine

Järgnevates alampeatükkides tutvustatakse Karnaugh' kaarti ning tuuakse välja, milleks ja kuidas seda kasutatakse. Lisaks tuuakse välja viie ja kuue muutujaga Karnaugh' kaardi kujutamise viisid ning miks peaks eelistama üht meetodit teisele.

1.1 Karnaugh' kaart

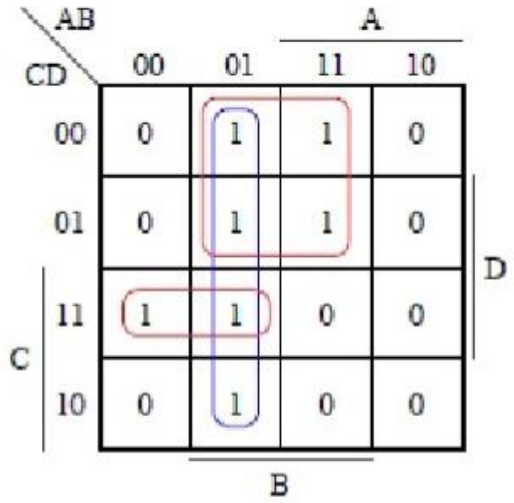
Palm jt [3] kirjeldavad Karnaugh' kaarti kui tabelit lahtritest, kus igale lahtrile vastab üks muutujate väärtustus valemi täielikus disjunktiivses normaalkujus, kusjuures kõrvuti asetsevad elemendid erinevad alati ainult ühe muutuja väärtuse poolest. Lisatakse veel, et kõikide ridade algused ja lõpud on mõttelises ühenduses ning samamoodi kõikide veergude algused ja lõpud, mis tähendab, et kõigil tabeli elementidel on kuni neli naaberelementi.

Valemite täieliku disjunktiivse normaalkuju leidmisel on eesmärgiks valemite ühtsele kujule viimine. Selle tulemuseks on konjunktsioonide disjunktsioon, kus iga konjunktsioon vastab ühele muutujate väärtustusele, mille puhul on valem tõene. Seetõttu on disjunktiivsest normaalkujust lihtne välja lugeda kõik väärtustused, mille puhul on valem tõene. Disjunktiivne normaalkuju leidub kõigile valemitele, mis on vähemalt ühel väärtustusel tõesed. Minimaalne disjunktiivne normaalkuju on, nagu nimigi ütleb, täieliku normaalkuju minimeeritud kuju. Minimaalses disjunktiivses normaalkujus ei pea konjunktsioonid sisaldama kõiki muutujaid nagu täielikus normaalkujus.

Palm jt [3] väidavad, et valemite minimaalsele disjunktiivsele normaalkujule viimiseks on üks võimalus kasutada ameerika matemaatiku Maurice Karnaugh välja mõeldud Karnaugh' kaarti. Samuti tuuakse allikas välja, et Karnaugh' kaardi meetod sobib pigem väikese muutujate arvuga valemite puhul. Ndjountche [5] täpsustab, et Karnaugh' kaardi kasutamine on mõistlik valemitega, kus on kuni kuus muutujat, rohkemate muutujatega valemite puhul kasutatakse tihti näiteks Quine-McCluskey meetodit (algoritmist täpsemalt peatükis 4.3). Kui muutujate arv on alla seitsme, siis võib kasutada minimaalse disjunktiivse normaalkuju saamiseks Karnaugh' kaarti. Selleks tuleb allika sõnul leida minimaalne arv primaarimplikante, mis kataksid kõik mintermid. Palm jt [3] toovad välja, et primaarimplikandid võivad osaliselt kattuda ehk sisaldada samu minterme, kui see aitab lõplikku normaalkuju lühemaks teha.

1.2 Karnaugh' kaardi kasutamine

Nagu kirjeldatud, siis kuni nelja muutujaga Karnaugh' kaart on kaheteljeline tabel, kust tuleb lahenduse saamiseks leida minimaalne arv primaarimplikante. Palm jt [3] toovad välja, et primaarimplikandi ploki mõõtmed peavad alati olema arvu 2 astmed. Allika järgi peavad need plokid olema võimalikult suured, sest mida suuremad plokid, seda vähem neid on, mis omakorda tagab tulemuseks võimalikult lühikese normaalkuju. Joonisel 1 on näha Karnaugh' kaart ja sellelt leitud primaarimplikandid, millest saab kokku panna SOP lahenduse: ¬AB + B¬C + ¬ACD.



Joonis 1. Karnaugh' kaart leitud primaarimplikantidega [5].

Kuigi varem loodud Karnaugh' kaardi lahendused kasutavad viie ja kuue muutuja korral samuti kahemõõtmelist tabelit, siis Maurice Karnaugh [6] on ka ise arvanud, et rohkem kui nelja muutuja korral on kõige mõistlikum kasutada ruumilist vaadet. Nii saab muutujate arvu kasvatada neljalt kuuele ilma, et peaks muutma algseid reegleid. Teine Karnaugh poolt välja pakutud võimalus, mis on laialdasemalt kasutust leidnud, oleks suurendada ühel teljel olevate muutujate arvu. Sellisel juhul poleks aga vaatlusel enam eriti lihtne primaarimplikante leida, kuna ühe muutuja võrra erinevad väärtustused ei asu enam alati kõrvuti. Näiteks viie muutujaga võrrandi puhul on üheks väärtustuseks 00000, millele leidub viis väärtustust, mis erinevad ainult ühe muutuja poolest: 10000, 01000, 00100, 00010, 00001. Tavalisel kaheteljelisel kaardil saab ühel elemendil ehk väärtustusel olla aga ainult kuni neli naaberelementi, seega peab viies asuma sellest kuskil eemal. See aga teebki primaarimplikantide leidmise raskeks ja vähem intuitiivseks, kui ühel teljel on rohkem kui kaks muutujat. Kui aga viies ja kuues muutuja panna kolmandale teljele ja muuta Karnaugh' kaart kolmemõõtmeliseks, siis oleks igal elemendil kuni kuus naaberelementi – lisaks üleval, all ja vasakul,

paremal asuvatele elementidel on nüüd elemendid ka ees ja taga. See teeb ka primaarimplikantide leidmise lihtsamaks, kuna nüüd saavad kõik ühe muutuja võrra erinevad väärtustused asuda selle elemendi naabruses.

2. Olemasolevad lahendused

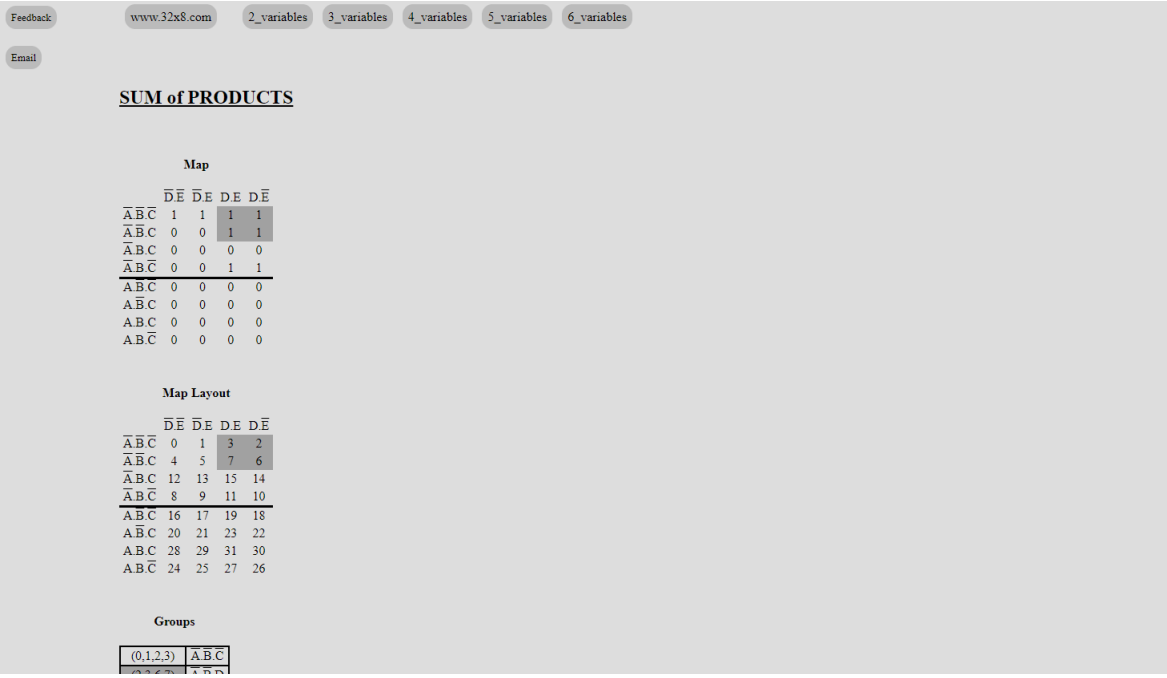
Järgnevates alampeatükkides tutvustatakse enne antud bakalaureusetööd valminuid rakendusi Karnaugh' kaardi lahendamiseks. Analüüsitakse kahte rakendust: Logic Circuit Simplification ja Karnaugh Map Minimizer. Mõlema rakenduse puhul tuuakse välja nii nende plussid kui ka miinused.

2.1 Logic Circuit Simplification

Logic Circuit Simplification [2] on veebirakendus, mis võimaldab kasutajal leida suvalise väärtustuste komplekti minimaalse disjunktiivse normaalkuju, samuti võimaldab rakendus leida ka minimaalse konjunktiivse normaalkuju. Muutujate arv võib olla alates kahest kuni kuueni.

Rakenduse plussiks on kindlasti lihtne kättesaadavus, vaja on ainult veebibrauseris vastavale lehele minna ja midagi alla laadida pole vaja. Lisaks genereerib rakendus ka loogikaväratite skeemi.

Rakenduse suurimaks puuduseks on viie ja kuue muutujaga Karnaugh' kaardi kuvamise meetod. Seda tehakse tavalises kahemõõtmelises tabelis, mis tähendab, et primaarimplikantide leidmine ei käi enam tavaliste reeglite järgi. Seega ei piisa ainult kõrvuti asetsevate elementide vaatamisest. Lisaks kuvatakse leitud implikante Karnaugh' kaardil ainult korraks ja nende nägemiseks peab enne ka vastava valiku tegema. Samuti raskendab rakenduse



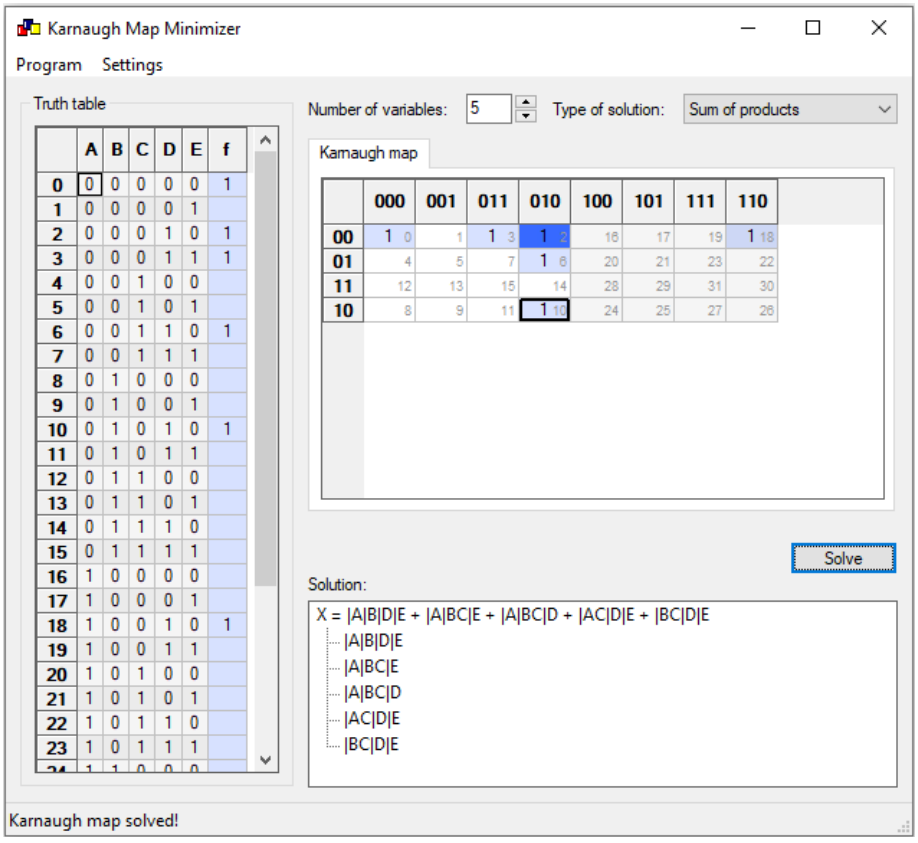
Joonis 2. Veebirakenduse Logic Circuit Simplification kuvatõmmis.

sujuvat kasutamist see, et väljundite väärtused tuleb iga kord uuesti sisestada, isegi kui kasutaja tahaks ainult ühte väljundit muuta või SOP lahenduse asemel POS lahendust näha. Joonisel 2 on näha korraga ekraanile mahtuva rakenduse osa ning primaarimplikantide kuvamine, mida tehakse tsüklis läbi ühe korra.

Lisaks on veebirakendusel palju erinevaid valikuid ja vaateid, mis teevad selles orienteerumise keeruliseks. Esialgne tõeväärtustabel kaob, kui hakatakse tulemust otsima ja kui kasutaja tahab esialgses sisestuses muudatusi teha, peab ta kogu tõeväärtus tabeli uuesti täitma. Peale selle kuvatakse tulemused üksteise all reas ning vajaliku informatsiooni leidmine pole intuitiivne.

2.2 Karnaugh Map Minimizer

Karnaugh Map Minimizer [1] on Robert Kovačevići loodud programm Karnaugh' kaardi lahendamiseks, mis on kirjutatud C++ keeles. Rakendus võimaldab kasutada kuni kaheksat muutujat, mis tundub küll positiivne, kuid sellise arvu muutujate korral on Karnaugh' kaardi kasutamine juba ebaefektiivne, sest elemendi ühe võrra erinevaid naabreid on kaardilt raske leida. Naaberelemente on juba kaheksa tükki, millest ainult neli asuvad elemendi kõrval.



Joonis 3. Rakenduse Karnaugh Map Minimizer kuvatõmmis.

Rakenduse üldine kasutusmugavus on suurem kui Logic Circuit Simplificationi veebirakendusel. Muudatusi saab tõeväärtustabelis teha korduvalt, ka pärast lahenduse vaatamist. Samuti on kogu informatsioon nähtaval ühel vaatel ning puudub vajadus erinevate kuvade vahel navigeerida.

Rakendusel on aga sama probleem, mis Logic Circuit Simpification veebirakendusel. Viie ja kuue muutuja korral näidatakse Karnaugh' kaarti samuti ainult kahemõõtmelises tabelis ning primaarimplikante on Karnaugh' kaardilt raske eristada ilma, et neid ükshaaval vaataks. Samuti pole viie ja kuue muutuja puhul isegi tabelis kõrvuti olevad elemendid alati ühe võrra erinevad, mis jällegi raskendab lahenduse loetavust. Joonisel 3 on näha element 00010 ja kõik tema naaberelemendid.

Lisaks esineb rakenduses viga, mille tõttu kuvatakse nii tõeväärtustabeli kui ka Karnaugh' kaardi lahtreid üksteise peal. Rakenduse testimise käigus esines seda mitu korda. Vea esinemine muudab aga rakenduse kasutuskõlbmatuks ning see tuleb taaskäivitada, et uuesti väärtuseid sisestada ja Karnaugh' kaarti näha.

3. Nõuded rakendusele

Järgnevalt tutvustatakse nõudeid loodavale rakendusele. Nõuded lepiti kokku juhendajaga enne rakenduse arenduse alustamist ning nende määramisel lähtuti rakenduse funktsionaalsusest ja olemasolevate rakenduste vigade parandamisest. Samuti võeti arvesse, et rakendust hakkavad kasutama ka aine „Digitaalne loogika“ tudengid õppe-eesmärgil. Seega kuvatakse näiteks rakenduses lisaks minimaalsele kujule ka kõik leitud primaarimplikandid.

Tarkvara funktsionaalsed nõuded kirjeldavad, mida süsteem teeb ja kuidas see toimib [7]. Loodavale rakendusele määratud funktsionaalsed nõuded on järgmised:

1. Rakendus peab suutma lahendada kahe kuni kuue muutujaga Karnaugh’ kaarti.
2. Rakendus peab võimaldama leida nii SOP kui ka POS lahendeid.
3. Rakendus peab näitama Karnaugh’ kaardi lahendust graafiliselt, kuvama peab nii SOP kui ka POS lahendeid.
4. Rakendus peab genereerima tõeväärtustabeli automaatselt.
5. Rakendus genereerib tõeväärtustabeli vastavalt muutujate arvule.
6. Rakendus peab suutma töödelda ka tõeväärtustabeli *do-not-care* väärtuseid.
7. Rakendus peab suutma automaatselt täita kõik tõeväärtustabeli väljundid kasutaja valitud väärtusega (0, 1 või *do-not-care*).
8. Kasutaja peab saama tõeväärtustabeli kõiki väljundväärtuseid muuta.
9. Rakendus peab kuvama kasutajale neli funktsiooni:
   9.1. kõiki primaarimplikante sisaldav SOP funktsioon,
   9.2. minimaalne SOP funktsioon,
   9.3. kõiki primaarimplikante sisaldav POS funktsioon,
   9.4. minimaalne POS funktsioon.
10. Rakendus peab leitud primaarimplikandid kuvama Karnaugh’ kaardil erinevate värvidega.
11. Rakendus peab näitama iga leitud primaarimplikanti Karnaugh’ kaardil ja vastavat osa funktsioonis sama värviga.
12. Rakendus peab võimaldama kuvada viie või kuue muutujaga Karnaugh’ kaarti ruumiliselt.
13. Rakendus peab töötama Windows operatsioonisüsteemiga arvutil.
14. Rakendus peab olema käivitatav fail, mis ei vaja installeerimist.

Lisaks määrati ka mittefunktsionaalsed nõuded, mis panevad rakendusele piirangud, näiteks kui täpne, kiire ja turvaline peab rakendus olema [7]. Rakendusele määratud mittefunktsionaalsed nõuded on:

1. Rakendus peab olema kasutajasõbralik.
2. Rakendus peab suutma avakuva ära laadida vähem kui ühe sekundiga.
3. Rakendus peab suutma kahe kuni neli muutujaga Karnaugh' kaardi lahendada vähem kui ühe sekundiga.
4. Rakendus peab suutma viie ja kuue muutujaga Karnaugh' kaardi lahendada vähem kui kahe sekundiga.
5. Rakendus peab olema ingliskeelne.

Rakenduse loomisel on püütud järgida kõiki sellele määratud nõudeid.

4. Karnaugh' kaardi minimeeritud kuju leidmise rakendus

Valminud Karnaugh' kaardi lahendamise rakendus (edaspidi rakendus) on kättesaadav aadressil https://sourceforge.net/projects/karnaugh-map-solver/. See on alla laetav käivitatava failina, mida kasutaja eraldi installeerima ei pea. Rakenduse keeleks valiti inglise keel, sest see võimaldab rakenduse laialdasemat kasutamist.

Rakenduse loomisel kasutati töövahendeid GitHub[1] ja IntelliJ[2]. IntelliJ on ettevõtte JetBrains loodud IDE (programmeerimiskeskkond), mis on mõeldud Javaga töötamiseks ning GitHub on versioonihaldustarkvara, kus säilitatakse rakenduse erinevaid versioone.

4.1 Rakenduses kasutatud tehnoloogiad

Rakenduse loomisel kasutati programmeerimiskeelena Javat ning kasutajaliidese loomiseks JavaFX platvormi ja tööriista JavaFX Scene Builder. Java projekti ehitamiseks kasutati tööriista Gradle[3] ja käivitatava (*executable*) programmi saamiseks Gradle'i pistikprogrammi Shadow[4]. Eelpool mainitud tehnoloogiad valiti, sest need sobisid loodud rakenduse funktsionaalsustega ja rakenduse looja oli nendega varasemalt juba tuttav.

4.2 Rakenduse funktsionaalsused ja kasutamine

Rakenduse avamisel tuleb esimese asjana valida muutujate arv, mis jääb vahemikku kahest kuni kuueni. Seejärel genereeritakse akna vasakusse äärde tõeväärtustabel (*truth tabel*) vastavalt muutujate arvule. Peale seda saab kasutaja valida erinevate väärtustuskomplektide väljundid, mis kuvatakse kohe ka Karnaugh' kaardil. Kasutaja saab valida kolme erineva väljundi vahel:

- „0" (väär),
- „1" (tõene) või
- „?" (*do-not-care*).

Väärtuse muutmiseks peab vajutama hiirega vastava lahtri peal ning seejärel muudetakse vastavat väärtust. Samuti on kasutajal võimalik kõik väljundid korraga üle kirjutada, milleks

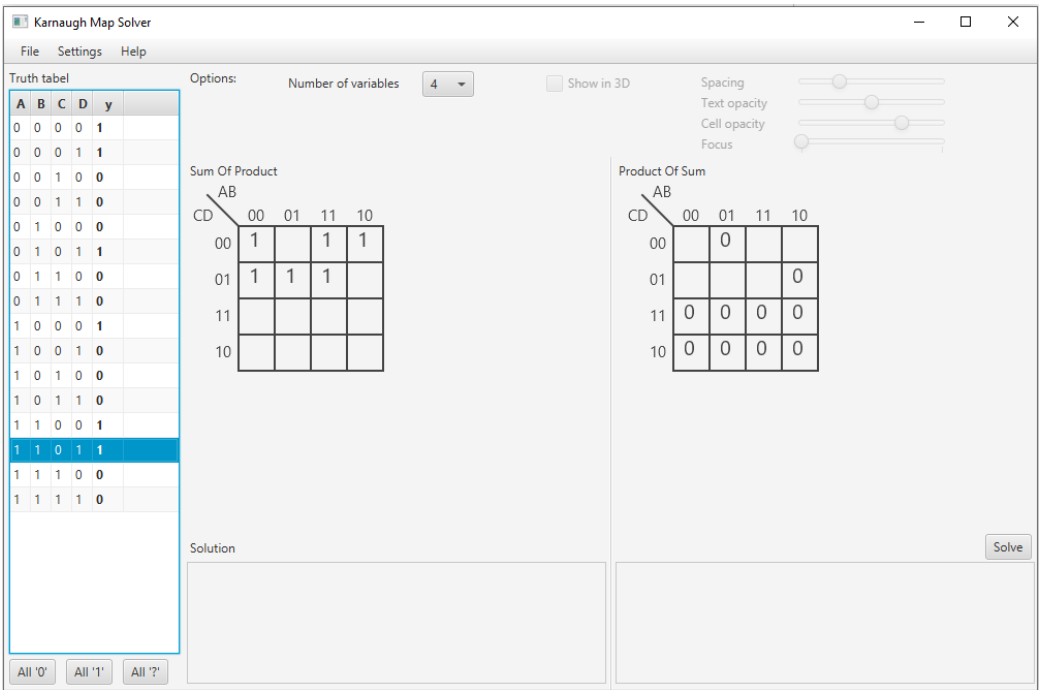
[1] GitHub. https://github.com/ (10.04.2022)

[2] IntelliJ. https://www.jetbrains.com/idea/ (10.04.2022)

[3] Gradle. https://docs.gradle.org/current/userguide/userguide.html (10.04.2022).

[4] Gradle Shadow Plugin. https://imperceptiblethoughts.com/shadow/introduction/ (10.04.2022).

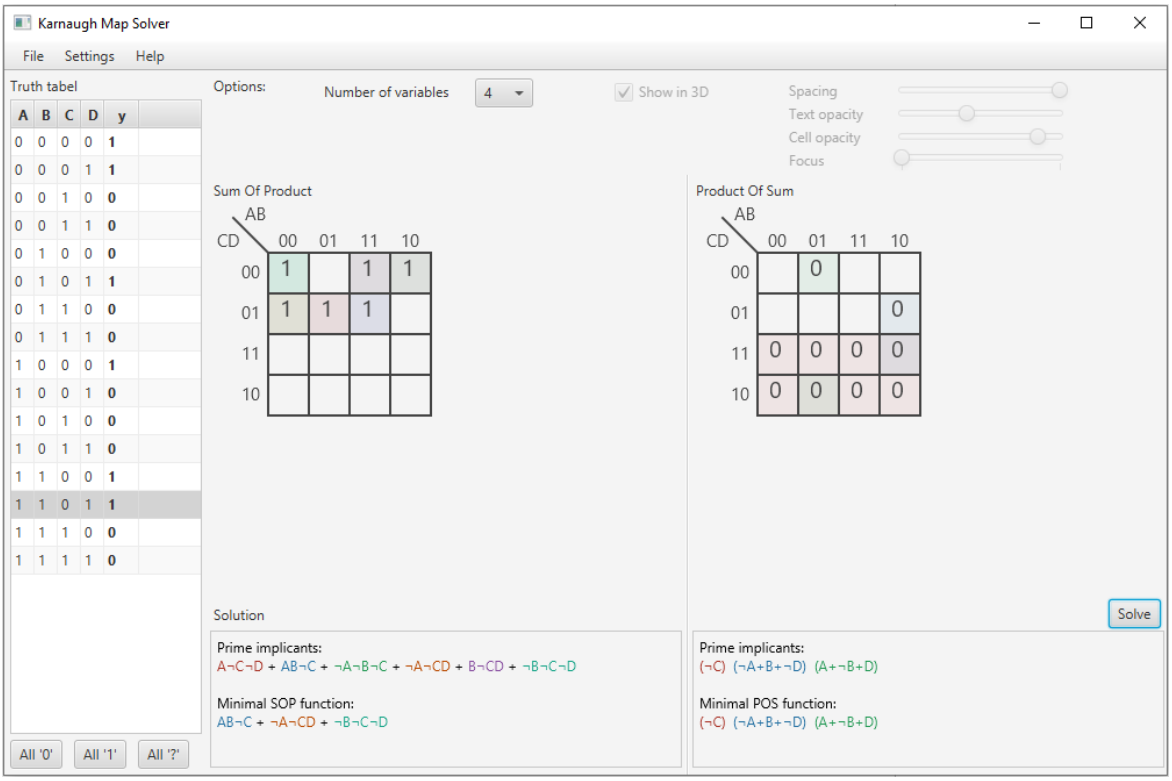
on tõeväärtustabeli all kolm nuppu: „All '0'“, „All '1'“ ja „All '?'“. Samal ajal kui kasutaja tõeväärtustabelit täidab ilmuvad vastavad väärtused ka Karnaugh' kaardile. Lisaks toob rakendus esile väärtuse asukoha Karnaugh' kaardil, kui kasutaja hiirega tõeväärtustabelis selle



Joonis 4. Rakenduse kuvatõmmis väljundite valimisel.

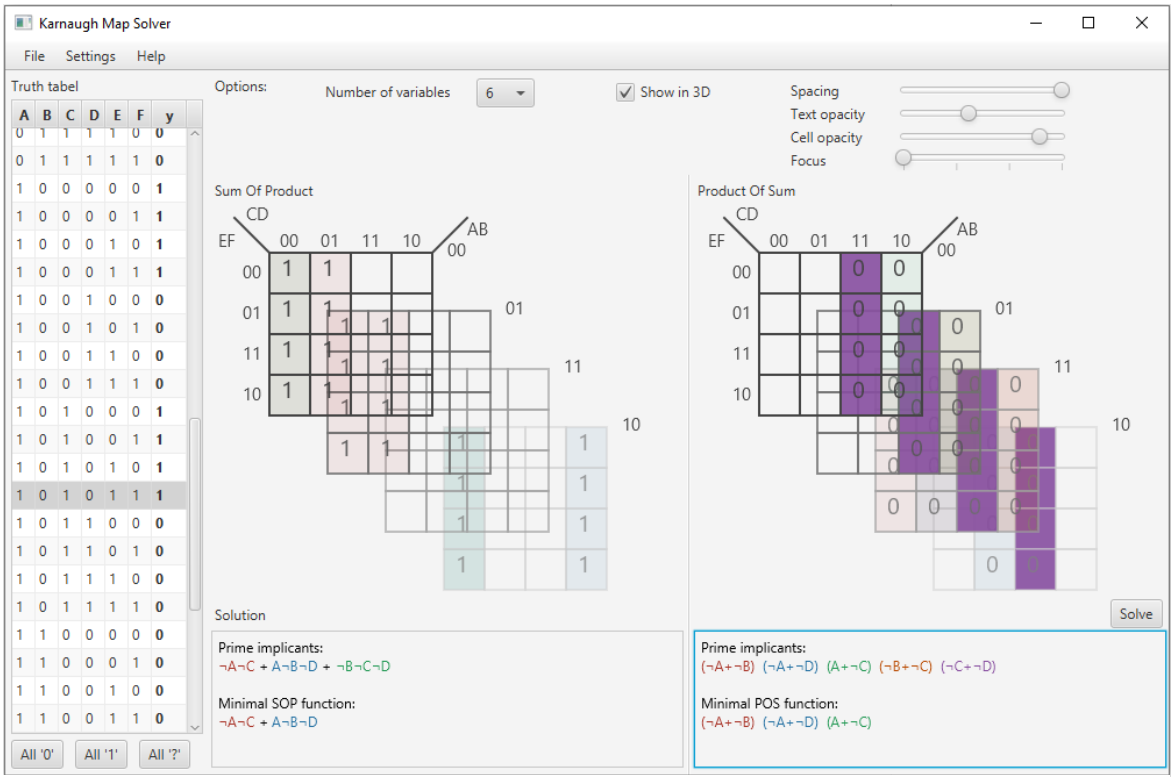
peale läheb. Karnaugh' kaarte kuvatakse korraga kaks (vt joonis 4), et leida nii SOP kui ka POS lahendus.

Kui kasutaja on sisestanud kõik soovitud väärtused, siis vajutades nuppu „Solve“, leitakse mõlemad eelmainitud lahendused. Karnaugh' kaardil kuvatakse erinevate värvidega vastava lahenduse kõik primaarimplikandid. Lahenduse (*Solution*) juures kuvatakse mõlema lahenduse tüübi kohta kaks funktsiooni: kõiki primaarimplikante sisaldav valem (*Prime implicants*) ja minimaalset arvu primaarimplikante sisaldav valem (*Minimal SOP/POS function*). Kusjuures kõik valemi osad, mis tähistavad primaarimplikante, on sama värvi sellele vastava primaarimplikandile Karnaugh' kaardil. Rohkemate primaarimplikantide puhul on Karnaugh' kaardil palju värvilisi plokke ning selles olukorras on kasutajal võimalik ka erinevaid primaarimplikante esile tõsta. Selleks peab hiirega vajutama valemis vastava osa peale ning sellele vastav primaarimplikant kuvatakse kaardil erksama värviga. Joonisel 4 toodud Karnaugh' kaardi lahendus on näha joonisel 5.



Joonis 5. Rakenduse kuvatõmmis leitud valemitega.

Kui kasutaja on valinud muutujate arvuks viis või kuus, siis on võimalik lahendust ka ruumiliselt kuvada. Selleks tuleb märkida akna üleval servas olev valik „Show in 3D“. Seejärel kuvatakse Karnaugh' kaart kolmel teljel, mille näidet on näha joonisel 6, kus on lisaks näha, kuidas on esiletõstetud osavalem (¬C + ¬D). Lisaks avanevad lisavalikud Karnaugh' kaardi vaate muutmiseks. Esiteks saab kasutaja liugnupuga „Spacing“ valida, kui kaugel erinevad kihid üksteisest kuvatakse. Teiseks saab kasutaja valida liugnupuga „Focus“ millist kihti ta tahab esile tõsta, valitud kiht kuvatakse teistest tumedamalt. Lisaks on veel kaks liugnuppu, millega saab muuta esiletõstmata kihtide läbipaistvust, eraldi nii teksti kui ka lahtrite jaoks.



Joonis 6. Rakenduse kuvatõmmis kuue muutujaga Karnaugh' kaardi kolmemõõtmelisest vaatest.

Rakenduse menüüs „Settings" saab muuta rakenduse sätteid. Valides „Show only relevant values", näidatakse kasutajale Karnaugh' kaardil ainult väärtuseid, mida priomaarimplikatide koostamisel kasutatakse. SOP lahenduse korral kuvatakse ühtesid ja *do-not-care* väärtuseid, POS lahenduse korral nulle ja *do-not-care* väärtuseid. Teiseks saab kasutaja valida „Show indexes", et tabelis näidataks ka lahtrite indekseid. Lisaks saab kasutaja valida, milliste väärtustega tõeväärtustabel algväärtustatakse.

4.3 Rakenduse ülesehitus

Rakendus on ülesehitatud kahele klassile. Esimene hoiab endas üht termi ehk märkide massiivi ja selle väärtust: üks, null või „?". Lisaks kasutatakse seda ka implikantide loomiseks, sellisel juhul pannakse ebaolulise muutuja kohale sidekriips. Näiteks nelja muutuja korral oleks implikandi ¬ACD kujuks „0-11". Teine klass on Karnaugh' kaardi hoidmiseks. Selle isend luuakse vastavalt valitud muutujate arvule ning selles hoitakse eelmainitud termi isendeid. Lisaks väärtustustele hoiab see endas nelja järjendit: nii POS kui ka SOP lahenduste kõiki primaarimplikante ning nende minimaalseid funktsioone.

Sama termi isendit kasutatakse mitmes kohas, seega kui kasutaja muudab tõeväärtustabelis mõnda väärtust, siis muutub ka Karnaugh' kaardil olev väärtus. Juhul kui kasutaja vajutab

„Solve“ nuppu, siis kutsutakse välja funktsioon, mis leiab antud Karnaugh’ kaardile vastavad neli funktsiooni, mis kuvatakse seejärel ka rakenduse aknas.

Valemi minimaalse kuju leidmiseks on vaja leida kõigepealt primaarimplikandid ja nende seast asendamatud primaarimplikandid (*essential prime implicants*). Nende leidmiseks kasutatakse Quine-McCluskey meetodit. Enne algoritmi kasutamist peab leidma kõik min- või makstermid olenevalt, kas leida tahetakse SOP või POS lahendit. Ülejäänud algoritm töötab mõlemal juhul samamoodi. Järgnevalt vaadatakse läbi, kuidas leida SOP lahendeid.

Petrík [8] väidab oma artiklis, et primaarimplikantide leidmiseks peab läbi vaatama kõik mintermid ja *do-not-care*-termid (ehk kõik termid, mille väärtus pole null) ning moodustama neist uue implikandi, kui need erinevad ainult ühe numbri poolest, sellisel juhul asendatakse erinev number sidekriipsuga. Näiteks mintermidest 1100 ja 1101 saab luua implikandi 110-. Petrík [8] lisab, et kõik mintermid, millest sai luua uue implikandi, tuleb ära märgistada, et neid edaspidi enam mitte kasutada, kuna need on kaetud juba suurema implikandi poolt ja pole järelikult primaarimlikandid. Rakenduses on see lahendatud nii, et mintermid, millele ei õnnestunud leida ühe võrra erinevat paarilist, lisatakse kohe primaarimplikantide hulka, kuna see ei saa enam järelikult suuremas grupis paikneda. Petrík [8] jätkab, et nüüd tuleb eelmine samm uuesti läbi teha kõigi uute leitud implikantidega, kuni enam uusi implikante luua ei saa. Näiteks saab teise sammuna implikantidest 110- ja 010- luua uue implikandi -10-. Juhul kui enam uusi implikante leida ei ole võimalik, siis ongi leitud kõik primaarimplikandid.

Järgmiseks tuleb leida primaarimplikantide hulgast asendamatud primaarimplikandid. Petrík [8] väidab, et selleks tuleb vaadata läbi eelnevalt leitud primaarimplikandid ning kui selles on mõni minterm, mis üheski teises primaarimplikandis ei esine, siis järelikult on see asendamatu. Allika sõnul tuleb eemaldada, siis vaatluse alt nii implikant kui ka minterm. Rakenduses on iga mintermi juurde salvestatud kõik primaarimplikandid, milles ta sisaldub ning kui neid on ainult üks, siis see üks lisatakse asendamatute primaarimplikantide hulka. Lisaks eemaldatakse antud minterm kõige teiste implikantide juurest, kuhu ta salvestatud oli. Kõik leitud asendamatud primaarimplikandid kuuluvad ka minimaalsesse funktsiooni.

Seejärel saab leida ülejäänud minimaalse funktsiooni osad. Petríki [8] järgi peab selleks allesjäänud primaarimplikantide hulgast eemaldama kõik sellised primaarimplikandid, mis nüüd kuuluvad täielikult mõne teise alla. Rakenduses vaadatakse iga alles jäänud primaarimplikandi minterme ning kui kõik need kõik on lisatud ka mõnele teisele, siis järelikult on

see primaarimplikant ebavajalik ja selle võib vaatluse alt eemaldada. Järgmiseks tuleb Petríki [8] järgi kuidagi leida ülejäänud ebavajalikud primaarimplikandid. Loodud rakenduses valitakse alles jäänud primaarimplikantidest selline, mis sisaldab kõige rohkem veel vaatluse all olevaid minterme ja lisatakse see minimaalse funktsiooni osade hulka ja kõik selle juures olevad mintermid eemaldatakse teistelt implikantidelt. Seejärel korratakse asendamatute implikantide leidmist ja ebavajalike eemaldamist nii kaua, kuni vaatluse all pole enam ühtegi mintermi.

4.4 Võimalikud edasiarendused

Loogikavalemite lihtsustamine on tihti seotud loogikaskeemidega ja seeläbi ka loogikaväratitega. Samuti räägitakse valemite lihtsustamise seosest loogikaväratitega ka aines „Digitaalne loogika“. Seega võiks edasiarendusena rakendus ka loogikaväratite skeemi luua, mis vastaks leitud minimaalsele funktsioonile. Selleks oleks vaja võtta minimaalse funktsiooni osavalemid ja need vastavalt lahenduse tüübile OR- (POS) või AND-väratite (SOP) sisendiks anda ja nende väratite sisendid oma korda AND- (POS) või OR-värati (SOP) sisendiks anda. Eelnevalt kirjeldatud skeemi võiks kasutajale kuvada eraldi aknas ja selle näitamiseks võiks olla rakenduses eraldi valik.

Kokkuvõte

Bakalaureusetöö eesmärk oli koostada rakendus Karnaugh' kaardi lahendamiseks, mis parendaks olemasolevaid lahendusi. Seniste rakenduste suureks puuduseks oli viie ja kuue muutujaga Karnaugh' kaardi kuvamine, mida tehti kahemõõtmelise tabeliga. Samuti ei olnud need eriti kasutajasõbralikud ning primaarimlikantide kuvamine Karnaugh' kaardil polnud kõige paremini lahendatud.

Valminud rakendus suudab leida suvalise väärtustuste komplekti minimaalse disjunktiivse ja konjunktiivse normaalkuju. Erinevalt varem loodud lahendustest on uues rakenduses võimalik viie ja kuue muutujaga Karnaugh' kaarti kuvada ka kolmemõõtmeliselt, mis lihtsustab kaardilt lahenduse lugemist ning järgib endiselt kõiki Karnaugh' kaardi reegleid. Lisaks saab kasutaja kõik valemis olevad primaarimplikandid lihtsalt üles leida ka Karnaugh' kaardilt, kui ta nende peale hiirega vajutab.

Karnaugh' kaardi lahendamise rakendus on kättesaadav veebiaadressil https://sourceforge.net/projects/karnaugh-map-solver/. Seda saab alla laadida igaüks ning see ei vaja eraldi arvutisse installeerimist.

Rakenduse lähtekood on nähtav GitHub repositooriumis https://github.com/kaarelkytt/KarnaughMapSolver.

Edasiarendusena võiks rakendusele lisada võimaluse kuvada ka leitud funktsioonile vastavat loogikaväratitest skeemi.

Viidatud kirjandus

[1] Karnaugh map minimizer. http://k-map.sourceforge.net/ (14.12.2021).

[2] Logic circuit simplification (SOP and POS). http://www.32x8.com/index.html. (14.12.2021).

[3] Palm R. ja Prank R. Normaalkujude minimeerimine. *Sissejuhatus matemaatilisse loogikasse*. H. Nestra. Tartu: Tartu Ülikooli kirjastus, 2004, 31-36.

[4] Ndjountche T. Logic Gates. *Digital Electronics 1: Combinational Logic Circuits*. R. Baptist. London: John Wiley & Sons, Incorporated, 2016, 67-129.

[5] Ndjountche T. Systematic Methods for the Simplification of Logic Functions. *Digital Electronics 1: Combinational Logic Circuits*. R. Baptist. London: John Wiley & Sons, Incorporated, 2016, 222-274.

[6] Karnaugh M. The map method for synthesis of combinational logic circuits. *Transactions of the American Institute of Electrical Engineers, Part I: Communication and Electronics*. 1953, 593-598.

[7] Aurum A. Requirements Engineering: Setting the Context. *Engineering and Managing Software Requirements*. C. Wohlin. Berlin: Springer Berlin Heidelberg, 2005, 1-15.

[8] Petrík M. Quine–McCluskey method for many-valued logical functions. *Soft Comput*. 2007, 393-402.

Lisad

I. Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja üldsusele kättesaadavaks tegemiseks

Mina, Kaarel Kütt

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) minu loodud teose

   Karnaugh' kaardi lahendaja,

mille juhendaja on Margus Rosin,

reprodutseerimiseks eesmärgiga seda säilitada, sealhulgas lisada digitaalarhiivi DSpace kuni autoriõiguse kehtivuse lõppemiseni.

2. Annan Tartu Ülikoolile loa teha punktis 1 nimetatud teos üldsusele kättesaadavaks Tartu Ülikooli veebikeskkonna, sealhulgas digitaalarhiivi DSpace kaudu Creative Commonsi litsentsiga CC BY NC ND 3.0, mis lubab autorile viidates teost reprodutseerida, levitada ja üldsusele suunata ning keelab luua tuletatud teost ja kasutada teost ärieesmärgil, kuni autoriõiguse kehtivuse lõppemiseni.

3. Olen teadlik, et punktides 1 ja 2 nimetatud õigused jäävad alles ka autorile.

4. Kinnitan, et lihtlitsentsi andmisega ei riku ma teiste isikute intellektuaalomandi ega isikuandmete kaitse õigusaktidest tulenevaid õigusi.

Kaarel Kütt

10.05.2022