TARTU ÜLIKOOL
Matemaatika-informaatikateaduskond
Arvutiteaduse instituut

Mati Tombak

KEERUKUSTEOORIA

TARTU 2007

TARTU ÜLIKOOL
MATEMAATIKA-INFORMAATIKATEADUSKOND
ARVUTITEADUSE INSTITUUT

Mati Tombak

KEERUKUSTEOORIA

Tartu 2007

Käesoleva õpiku trükkimist on toetanud Eesti Infotehnoloogia Sihtasutus projekti "Tiigriülikool" raames.

Käesoleva õpiku väljaandmist on toetanud Euroopa Liit. Meede 1.1. Projekt *Infotehnoloogiaalase hariduse kvaliteedi tagamine Tartu Ülikoolis ja Tartu Kutsehariduskeskuses*.

Toimetaja: Tõnu Tamme

ISBN
Tartu Ülikooli Kirjastus
www.tyk.ee
Tellimus nr.

Sisukord

Eessõna

Tartu Ülikooli Matemaatikateaduskonna esimene dekaan Endel Jürimäe juhtis kunagi kolleegide tähelepanu sellele, et kuigi matemaatilise teooria ideaal on deduktiivne teooria, on õpetamine tulemuslikum kui kasutada induktiivset lähenemist. Käesoleva õppevahendi autor jagab täielikult seda seisukohta ja sellest tulenevalt ei alga raamat aja-ja mälukeerukuse formaalse definitsiooniga ja keerukusteooria deduktiivse ülesehitamisega. Selle asemel uurime Boole'i funktsioonidega seotud arvutuslikke probleeme, vastavaid algoritme ja nende keerukust. Seejärel, lähtudes vajadusest võrrelda erinevate ülesannete keerukust, toome sisse polünomiaalse ja Turingi taandamise mõisted, keerukusklassid **P**, **NP**, **coNP** ning polünomiaalse hierarhia ja uurime nende omavahelisi vahekordi. Raamat lõpeb kahe intrigeeriva peatükiga Pehouseki teoreemist ja interaktiivsetest protokollidest, mis peaksid ärgitama huvilisi tegelema edasi keerukusteooria kaasaegsemate osadega.

Keerukusteooria alal on ilmunud hulgaliselt õpikuid ja monograafiaid. Nimetame siin olulisemaid, mis on saadaval ka Tartu Ülikooli Matemaatika-informaatikateaduskonna erialaraamatukogus. Kõigepealt [GJ79],mis oli esimene monograafia selles valdkonnas ja sisaldab umbes 300 **NP**-täieliku probleemi kirjeldusi, ilmunud on ka tõlge vene keelde aastast 1981. Järgmistena märgime kolme monograafiat, mis ilmusid peaaegu samaaegselt: [Pap94, BC94, BDG95]. Internetist on saadav [Yap87] ning uuemad monograafiad on [Sip05] ja [HO02]. Eesti keeles on ilmunud Valdo Prausti õpik [Pra96].

Käesolev õppevahend on valminud autori loengukursuse "Keerukusteooria" põhjal. 2003/2004 õppeaastal koostas nimetatud loengute järgi konspekti Erkki Hermann, mis saigi käesoleva raamatu aluseks. Konspekti esialgne sisu muutus vormistamise käigus küll oluliselt, kuid peaaegu kõik joonised on säilitanud oma esialgse, Erkki Hermanni antud kuju. Peatüki "Turingi taandamine" on suuremas osas kirjutanud Peeter Laud ning peatüki "DPLL halvima juhu eksponentsiaalne keerukus" Margus Niitsoo, kelle originaallooming on ka lemmad 4.3, 4.4 ja teoreem 4.5 koos tõestustega. Paljudele sisulistele ja vormistamisvigadele juhtisid tähelepanu 2006/2007. õppeaasta kevadsemestril käesolevat kursust kuulanud üliõpilased, eriti Margus Niitsoo, Janno Veldemann ja Maria Lorents. Oluliselt paremaks muutus käsikiri toimetaja Tõnu Tamme töö tulemusena. Tahaks tänada kõiki nimetatuid nende panuse eest. Veel tahaks tänada oma abikaasat Tiiu Tombakut ja TÜ arvutiteaduse instituudi juhatajat Tiit Roosmaad, kelle sõbraliku surveta poleks käesolev õppevahend valminud arvatavasti enne järgmist kümnendit.

1. Tagurdusalgoritm kehtestatavuse testimiseks

Tähistame tõeväärtusi **tõene** ja **väär** vastavalt arvudega 1 ja 0. n muutuja Boole'i funktsioon on funktsioon $f\colon \{0,1\}^n \to \{0,1\}$. Boole'i funktsiooni loomulik esitusviis on tõetabelina, milles igale argumentide väärtuste kombinatsioonile (väärtustusele) seatakse vastavusse funktsiooni tõeväärtus. Erinevaid väärtustusi n muutujaga Boole'i funktsiooni jaoks on 2^n, seega sisaldab tõetabel 2^n rida. Kui me fikseerime tõetabeli ridade järjekorra, näiteks väärtustustele vastavate kahendarvude kasvamise järjekorras, siis on väärtustuse näitamine ilmselt üleliigne – tõeväärtuse positsiooni järjekorranumber (kahendsüsteemis) annab vastava väärtustuse. Taoline esitusviis on ökonoomseim suvalise Boole'i funktsiooni esitamiseks, kuna 2^{2^n}-elemendise hulga elementide nummerdamiseks läheb vaja 2^n bitti.

Näide. Kahe muutujaga Boole'i funktsioonide tõetabeleid on 16.

1	1	0	0	x
1	0	1	0	y
0	0	0	0	0
0	0	0	1	$x \overline{\vee} y$
0	0	1	0	$x \not\subset y$
0	0	1	1	$\overline{x}$
0	1	0	0	$x \not\supset y$
0	1	0	1	$\overline{y}$
0	1	1	0	$x \oplus y$
0	1	1	1	$x \mathbin{\overline{\&}} y$
1	0	0	0	$x \mathbin{\&} y$
1	0	0	1	$x \sim y$
1	0	1	0	y
1	0	1	1	$x \supset y$
1	1	0	0	x
1	1	0	1	$x \subset y$
1	1	1	0	$x \vee y$
1	1	1	1	1

Paljude rakenduste jaoks ei ole see siiski piisav. Näiteks 32-bitise protsessori aritmeetikaskeemi esitamiseks on vaja kolmkümmend kaks 64-muutuja Boole'i funktsiooni (üks funktsioon 32-bitise resultaadi iga kahendkoha jaoks, igal funktsioonil peab olema 32 muutujat kumbagi argumendi jaoks). Seega ei piisaks projekteeritava protsessori ressurssidest selle enda projekteerimiseks. Õnneks on rakendustes kasutatavad Boole'i funktsioonid lihtsama struktuuriga ning nende esitamiseks saab kasutada teisi vahendeid – lausearvutuse valemit ja selle erijuhte: disjunktiivset ja konjunktiivset normaalkuju.

Tähistame lausemuutujaid tähestiku lõpuosa väiketähtedega (ka koos naturaalarvuliste indeksitega).

Definitsioon.

1° Iga lausemuutuja ja konstant 0 või 1 on lausearvutuse valem.

2° Kui A ja B on lausearvutuse valemid, siis on lausearvutuse valemid ka $(\neg A)$, $(A \mathbin{\&} B)$, $(A \vee B)$, $(A \supset B)$, $(A \oplus B)$ ja $(A \sim B)$.

Teooriast on teada, et suvalist Boole'i funktsiooni saab esitada kahe muutuja Boole'i funktsioonide superpositsioonina, kusjuures piisab, kui kasutada *täielikke funktsioonide komplekte* [Kul64]). Minimaalseteks täielikeks komplektideks on näiteks $\{\neg, \vee\}$, $\{\supset, 0\}$ või $\{\overline{\&}\}$. Tegelikus elus ei kasutata lausearvutuse valemit defineerides funktsioonide miinimumkomplekte, aga ka mitte kõiki 16 kahe muutuja Boole'i funktsiooni. Mõlema äärmuse puhul saadakse inimmõistusele raskesti hoomatav, *write only* valem. Ülaltoodud definitsioon on üheks kompromissiks nimetatud äärmuste vahel. Sõltuvalt eesmärgist kasutatakse ka komplekte $\{\neg, \vee, \&\}$, $\{\neg, \oplus, \&\}$ või lisatakse komplektile koguni kolme ja enama muutuja funktsioone, nagu näiteks **if** x **then** y **else** z.

Vaatleme kolme ülesannet seoses Boole'i funktsioonidega ja püüame hinnata nende arvutuslikku keerukust, kasutades mitteformaalset lähenemist.

1. Väärtustuse testimine. Antud on Boole'i funktsioon $f\colon \{0,1\}^n \to \{0,1\}$ ja väärtustus $\alpha = (\alpha_1, \ldots, \alpha_n)$, $\alpha \in \{0,1\}^n$. Leida funktsiooni f väärtus kohal α.

Kui funktsioon on esitatud tõetabelina, tuleb arvutada väärtuse aadress väärtustuse α järgi, milleks kulub n sammu. Tõetabel ise on aga pikkusega $p = 2^n$. Seega intuitiivse algoritmi keerukus testimisülesande lahendamiseks on $O(n)$ muutujate arvu n suhtes ja $O(\log_2 p)$ algandmete pikkuse p suhtes.

Kui funktsioon on esitatud lausearvutuse valemina, siis tuleb testimiseks asendada valemis muutujate $x_1, \ldots, x_n$ kõik esinemised vastavalt väärtustega $\alpha_1, \ldots, \alpha_n$ ning sooritada tehted. Selleks kulub $O(p)$ sammu, kus p on valemi pikkus. Algoritmi töökiiruse sõltuvus muutujate arvust n ei ole selge, kuna me ei tea, kuidas sõltub valemi pikkus muutujate arvust.

2. Kehtestatavus. Antud on Boole'i funktsioon $f(x_1, \ldots, x_n)$. Kas leidub väärtustus $\alpha = (\alpha_1, \ldots, \alpha_n)$, nii et $f(\alpha_1, \ldots, \alpha_n) = 1$

Kui funktsioon on esitatud tõetabelina, siis see on kehtestatav, kui tabelis leidub 1. Selle tuvastamiseks kulub halvimal juhul $O(2^n)$ sammu. Kui funktsioon on esitatud lausearvutuse valemina pikkusega p, siis tuleb genereerida järjest kõik väärtustused ja testida funktsiooni kuni esimese positiivse vastuseni. Kiirusehinnang on $O(p \cdot 2^n)$ sammu. Siin tuleb märkida, et see on *halvima juhu* hinnang. Kui me oskame valida kohe alguses tõese väärtustuse (või kui meil lihtsalt veab), saame positiivse vastuse kohe. Sellest tekib idee konstrueerida algoritm, mis tegeleks kehtestatava väärtustuse otsimisega süstemaatiliselt.

3. Lahendite loendamine. Antud on Boole'i funktsioon $f \colon \{0,1\}^n \to \{0,1\}$. Leida võrrandi $f(x_1, \ldots, x_n) = 1$ lahendite arv, s.t. leida, mitmel väärtustusel on funktsioon tõene. Tähistame funktsiooni $f(x_1, \ldots, x_n)$ tõeste väärtustuste arvu $\#f$. Ilmselt ei ole lahendite loendamise ülesanne arvutuslikult kergem kui kehtestatavus, kuna $f(x_1, \ldots, x_n)$ on kehtestatav parajasti siis, kui $\#f > 0$.

Konstrueerime tagurdusalgoritmi kehtestatavuse testimiseks üldkujuliste lausearvutuse valemite jaoks. Tagurdusmeetod (*backtracking method*) on universaalne meetod otsimisülesannete lahendamiseks. Meetodi idee seisneb ülesande jaotamises väiksemateks alamülesanneteks, mida töödeldakse rekursiivselt otsitava tulemuse leidmiseni. Idee kehtestatavusülesande jaotamiseks alamülesanneteks annab Boole-Shannoni lahutus.

Olgu $f \colon \{0,1\}^n \to \{0,1\}$ n muutuja Boole'i funktsioon ja x_i selle muutuja. Fikseerides muutuja x_i väärtuse saame funktsioonist f kaks $n-1$ muutuja funktsiooni:

$$f_{x_i}(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) = f(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n)$$

ning

$$f_{\overline{x}_i}(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) = f(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_n).$$

Funktsioonide f_x ja $f_{\overline{x}}$ arvutamiseks tuleb funktsiooni f esitavas valemis asendada kõik muutuja x esinemised vastavalt konstantidega 1 või 0 ja saadud valemit lihtsustada, kasutades lausearvutusest tuttavaid ekvivalentsiseoseid.

Teoreem 1.1. *(Boole-Shannoni lahutus)*

Olgu $f(x_1,...,x_n)\colon \{0,1\}^n \to \{0,1\}$ Boole'i funktsioon, siis iga i jaoks $(1 \leqslant i \leqslant n)$ kehtib

$$f(x_1,...,x_{i-1},x_i,x_{i+1},...,x_n) \equiv (x_i \mathbin{\&} f_{x_i}) \vee (\overline{x}_i \mathbin{\&} f_{\overline{x}_i}).$$

Tõestus. Olgu $\alpha \in \{0,1\}^n$ suvaline väärtustus.

$$\begin{aligned}
& [(x_i \mathbin{\&} f_{x_i}) \vee (\overline{x}_i \mathbin{\&} f_{\overline{x}_i})](\alpha) = \\
= \; & [(x_i \mathbin{\&} f(x_1,\dots,x_{i-1},1,x_{i+1},\dots,x_n)) \vee \\
& \vee(\overline{x}_i \mathbin{\&} f(x_1,\dots,x_{i-1},0,x_{i+1},\dots,x_n))](\alpha) = \\
= \; & (\alpha_i \mathbin{\&} f(\alpha_1,\dots,\alpha_{i-1},1,\alpha_{i+1},\dots,\alpha_n)) \vee \\
& \vee(\overline{\alpha}_i \mathbin{\&} f(\alpha_1,\dots,\alpha_{i-1},0,\alpha_{i+1},\dots,\alpha_n)).
\end{aligned}$$

1) Kui $\alpha_i = 1$, siis

$$\begin{aligned}
& [x_i \mathbin{\&} f_{x_i}) \vee (\overline{x}_i \mathbin{\&} f_{\overline{x}_i}](\alpha) = \\
= \; & (1 \mathbin{\&} f(\alpha_1,\dots,\alpha_{i-1},1,\alpha_{i+1},\dots,\alpha_n)) \vee \\
& \vee(0 \mathbin{\&} f(\alpha_1,\dots,\alpha_{i-1},0,\alpha_{i+1},\dots,\alpha_n)) = \\
= \; & f(\alpha_1,\dots,\alpha_{i-1},1,\alpha_{i+1},\dots,\alpha_n) = \\
= \; & f(\alpha_1,\dots,\alpha_{i-1},\alpha_i,\alpha_{i+1},\dots,\alpha_n).
\end{aligned}$$

2) Kui $\alpha_i = 0$, siis

$$\begin{aligned}
& [(x_i \mathbin{\&} f_{x_i}) \vee (\overline{x}_i \mathbin{\&} f_{\overline{x}_i})](\alpha) = \\
= \; & (0 \mathbin{\&} f(\alpha_1,...,\alpha_{i-1},1,\alpha_{i+1},...,\alpha_n)) \vee \\
& \vee(1 \mathbin{\&} f(\alpha_1,\dots,\alpha_{i-1},0,\alpha_{i+1},\dots,\alpha_n)) = \\
= \; & f(\alpha_1,\dots,\alpha_{i-1},0,\alpha_{i+1},\dots,\alpha_n) = \\
= \; & f(\alpha_1,\dots,\alpha_{i-1},\alpha_i,\alpha_{i+1},\dots,\alpha_n).
\end{aligned}$$

□

Järeldus 1.2. *$f(x_1,...,x_n)$ on kehtestatav parajasti siis kui f_{x_i} on kehtestatav või $f_{\overline{x}_i}$ on kehtestatav.*

Tõestus.
$\Longrightarrow$ Olgu $\alpha = (\alpha_1,\dots,\alpha_{i-1},\alpha_i,\alpha_{i+1},\dots,\alpha_n)$ funktsiooni $f(x_1,\dots,x_n)$ kehtestav väärtustus, s.t. $f(\alpha) = 1$. Boole-Shannoni lahutusest on näha, et vähemalt üks funktsioonidest f_{x_i}, $f_{\overline{x}_i}$ peab olema tõene väärtustusel $(\alpha_1,\dots,\alpha_{i-1},\alpha_{i+1},\dots,\alpha_n)$.

$\Longleftarrow$ Oletame, et $f_{x_i}(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$ on kehtestatav, s.t. leidub väärtustus $(\alpha_1, \ldots, \alpha_{i-1}, \alpha_{i+1}, \ldots, \alpha_n)$, mis muudab funktsiooni f_{x_i} tõeseks Moodustame väärtustuse α', laiendades väärtustust α muutujale x_i konstandiga 1, mis muudab Boole-Shannoni lahutuse parema poole tõeseks. Seega peab olema tõene ka vasak pool, s.t. $\alpha' = (\alpha_1, \ldots, \alpha_{i-1}, 1, \alpha_{i+1}, \ldots, \alpha_n)$ on funktsiooni f kehtestav väärtustus. Analoogiliselt, kui väärtustus $(\alpha_1, \ldots, \alpha_{i-1}, \alpha_{i+1}, \ldots, \alpha_n)$ muudab tõeseks funktsiooni $f_{\overline{x}_i}(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$, siis on $(\alpha_1, \ldots, \alpha_{i-1}, 0, \alpha_{i+1}, \ldots, \alpha_n)$ funktsiooni f kehtestav väärtustus. $\Box$

Algoritm 1.3. *Kehtestatavusalgoritm lausearvutuse valemitele.*

function $SAT(F : valem) : BOOLEAN$

begin
    if $F = 0$ **then return** (0) **fi**
    if $F = 1$ **then return** (1) **fi**
    vali valemi F muutuja x;
    return $(SAT(F_x) \vee SAT(F_{\overline{x}}))$;
end SAT

Esitatud algoritmis tuleb mõista võrdusi $F = 0$ ja $F = 1$ süntaktiliste võrdustena, s.t. valem koosneb konstandist 0 või 1.

2. Disjunktiivne ja konjunktiivne normaalkuju

Lihtsamate ja kiiremate töötlusalgoritmide saamiseks kasutatakse Boole'i funktsioonide esitamiseks disjunktiivset ja konjunktiivset normaalkuju.

Disjunktiivne normaalkuju (*disjunctive normal form*, DNF). Lausearvutuse valem F on disjunktiivsel normaalkujul, kui ta on esitatav järgmiselt:

$$F(x_1, ..., x_n) = \bigvee_{i=1}^{p} C_i,$$

kus C_i on elementaarkonjunktsioon (*term*), mis on esitatav kujul

$$C_i = \mathop{\&}_{j=1}^{m_i} l_{ij},$$

kus l_{ij} on literaal. Literaal on muutuja või selle eitus.

Üldisust kitsendamata võime eeldada, et disjunktiivse normaalkuju kõik termid on erinevad ja iga muutuja esineb termis ülimalt üks kord. Tõepoolest, kui term sisaldab literaale x ja $\overline{x}$, siis on term samaselt väär ja selle võib kustutsda. Korduvad literaalid ei muuda aga termi tõeväärtust. Samuti võib eeldada, et kõik termid disjunktiivses normaalkujus on erinevad. DNF on *täielik*, kui igas termis on täpselt n literaali, s.t. $m_i = n$.

Konjunktiivne normaalkuju (*conjunctive normal form*, CNF). Lausearvutuse valem F on konjunktiivsel normaalkujul, kui ta on esitatav kujul

$$F(x_1, ..., x_n) = \mathop{\&}_{i=1}^{p} D_i$$

kus D_i on elementaardisjunktsioon (disjunkt), mis on esitatav kujul

$$D_i = \bigvee_{j=1}^{m_i} l_{ij}$$

kus l_{ij} on literaal.

Üldisust kitsendamata võib eeldada, et iga muutuja esineb igas disjunktis ülimalt üks kord. Tõepoolest, kui disjunkt sisaldab literaale x ja $\overline{x}$, siis on disjunkt samaselt tõene ja selle võib kustutada.

Korduvad literaalid aga ei muuda disjunkti tõeväärtust. Analoogiliselt võib eeldada, et kõik disjunktid konjunktiivses normaalkujus on erinevad. CNF on *täielik*, kui igas disjunktis on täpselt n literaali, s.t. $m_i = n$.

Järgnevalt vaatleme, kuidas DNF ja CNF väljendavad kõikide väärtustuste hulga omadusi. Tähistame $x^1 = x$ ja $x^0 = \overline{x}$. Kui $l \in \{x, \overline{x}\}$, siis nimetame muutujat x literaali l muutujaks ja tähistame $var(l)$. Literaali eituselt eemaldame kahekordse eituse, s.t.

$$\bar{l} = \begin{cases} \overline{x}, & \text{kui } l = x, \\ x, & \text{kui } l = \overline{x} \end{cases}$$

Olgu DNF F muutujate hulgaga $X = \{x_1, \ldots, x_n\}$ ning T valemi F term kujul $T = y_1^{\alpha_1} \mathbin{\&} \ldots \mathbin{\&} y_k^{\alpha_k}$, kus $y_i \in X$ $(1 \leqslant i \leqslant k)$. Olgu $z_1, \ldots, z_{n-k}$ hulga X muutujad, mis ei esine termis T. DNF F on tõene parajasti siis, kui vähemalt üks term on tõene. Term T on tõene parajasti siis, kui kõik selle literaalid on tõesed, s.t. iga väärtustuse jaoks, milles $y_1 = \alpha_1, \ldots, y_k = \alpha_k$, sõltumata ülejäänud muutujate väärtustest. Seega on term T tõene 2^{n-k} väärtustuse jaoks. Täielik term, milles esinevad kõik muutujad, on tõene ühe väärtustuse jaoks. Kui täielik term T on kujul $x_1^{\alpha_1} \mathbin{\&} \ldots \mathbin{\&} x_n^{\alpha_n}$, siis ainus väärtustus, mis muudab termi T tõeseks, on $(\alpha_1, \ldots, \alpha_n)$. Term $T = y_1^{\alpha_1} \mathbin{\&} \ldots \mathbin{\&} y_k^{\alpha_k}$ on seega samaväärne 2^{n-k} täieliku termi disjunktsiooniga

$$T \equiv \bigvee_{(\beta_1, \ldots, \beta_{n-k}) \in \{0,1\}^{n-k}} y_1^{\alpha_1} \mathbin{\&} \ldots \mathbin{\&} y_k^{\alpha_k} \mathbin{\&} z_1^{\beta_1} \mathbin{\&} \ldots \mathbin{\&} z_{n-k}^{\beta_{n-k}}.$$

Seega esitab DNF tõeste, e. *lubatud* väärtustuste loetelu.

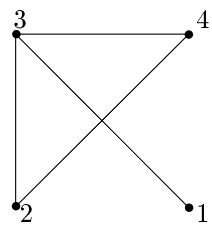
Olgu CNF F muutujate hulgaga $X = \{x_1, \ldots, x_n\}$ ning D valemi F disjunkt kujul $D = y_1^{\alpha_1} \vee \ldots \vee y_k^{\alpha_k}$, kus $y_i \in X$ $(1 \leqslant i \leqslant k)$. Olgu $z_1, \ldots, z_{n-k}$ hulga X muutujad, mis ei esine disjunktis D. CNF F on väär parajasti siis, kui vähemalt üks disjunkt on väär. Disjunkt D on väär parajasti siis, kui kõik selle literaalid on väärad, s.t. iga väärtustuse jaoks, milles $y_1 = \overline{\alpha}_1, \ldots, y_k = \overline{\alpha}_k$, sõltumata ülejäänud muutujate väärtustest. Seega on disjunkt D väär 2^{n-k} väärtustuse jaoks. Täielik disjunkt, milles esinevad kõik muutujad, on väär ühe väärtustuse jaoks. Kui täielik disjunkt D on kujul $x_1^{\alpha_1} \vee \ldots \vee x_n^{\alpha_n}$, siis ainus väärtustus, mis muudab disjunkti D vääraks, on $(\overline{\alpha}_1, \ldots, \overline{\alpha}_n)$. Disjunkt $D = y_1^{\alpha_1} \vee \ldots \vee y_k^{\alpha_k}$ on seega samaväärne 2^{n-k} täieliku disjunkti konjunktsiooniga

$$D \equiv \mathop{\&}_{(\beta_1, \ldots, \beta_{n-k}) \in \{0,1\}^{n-k}} y_1^{\alpha_1} \vee \ldots \vee y_k^{\alpha_k} \vee z_1^{\beta_1} \vee \ldots \vee z_{n-k}^{\beta_{n-k}}$$

Seega esitab CNF väärate e. *keelatud* väärtustuste loetelu.

Vaatleme, kuidas väljendada DNF-i ja CNF-i abil graafi klikkide struktuuri. Graafi $G = (V, E)$ klikk on G tippude hulga V alamhulk V', mis indutseerib täieliku alamgraafi. Graafi G maksimaalne klikk on selline klikk $V' \subseteq V$, et ükski tippude hulk V'', selline et $V' \subset V'' \subseteq V$, ei ole klikk. Tipule $i \in V$ vastaku muutuja $x_i \in X$, niiviisi võime vaadelda väärtustusi kui hulga V alamhulkade karakteristlikke vektoreid. Konstrueerime DNF-i DF_G ja CNF-i CF_G, mille tõesed väärtustused on graafi G klikkide karakteristlikud vektorid. Alustame konkreetse näite uurimisest.

Näide. Olgu graaf G:



Graafi G klikid on $\{\}$, $\{1\}$, $\{2\}$, $\{3\}$, $\{4\}$, $\{1,3\}$, $\{2,3\}$, $\{2,4\}$, $\{3,4\}$, $\{2,3,4\}$ ning maksimaalsed klikid on $\{1,3\}$ ja $\{2,3,4\}$.

Koostame tõetabeli:

x_1	x_2	x_3	x_4	f_G	klikk
0	0	0	0	1	$\emptyset$
0	0	0	1	1	$\{x_4\}$
0	0	1	0	1	$\{x_3\}$
0	0	1	1	1	$\{x_3, x_4\}$
0	1	0	0	1	$\{x_2\}$
0	1	0	1	1	$\{x_2, x_4\}$
0	1	1	0	1	$\{x_2, x_3\}$
0	1	1	1	1	$\{x_2, x_3, x_4\}$
1	0	0	0	1	$\{x_1\}$
1	0	0	1	0	–
1	0	1	0	1	$\{x_1, x_3\}$
1	0	1	1	0	–
1	1	0	0	0	–
1	1	0	1	0	–
1	1	1	0	0	–
1	1	1	1	0	–

Kirjutades tõetabeli järgi välja funktsiooni f_G täieliku disjunktiivse normaalkuju ja minimeerides seda, saame disjunktiivse normaalkuju

$$DF_G = (\overline{x}_2 \,\&\, \overline{x}_4) \vee (\overline{x}_1).$$

Paneme tähele, et muutujad saadud DNF-i esimeses termis moodustavad maksimaalse kliki $\{x_1, x_3\}$ täiendi ja muutujad teises termis maksimaalse kliki $\{x_2, x_3, x_4\}$ täiendi. Tehes julge hüpoteesi, et see ei ole juhuslik, saame graafi G iga maksimaalse kliki $V' \subseteq V$ järgi termi

$$T_{V'} = \underset{i \in V \setminus V'}{\&} \overline{x}_i.$$

Valem DF_G on kõikide selliste termide disjunktsioon:

$$DF_G = \bigvee_{V' = \mathrm{m}axklikk} T_{V'}.$$

Osutub, et esitatud hüpotees peab paika.

Teoreem 2.1. *Olgu $G = (V, E)$ graaf, mille tippude hulk on $V = \{1, \ldots, n\}$. Kahendvektor $\alpha \in \{0,1\}^n$ on graafi G kliki karakteristlik vektor parajasti siis, kui*

$$DF_G(\alpha) = \left[\bigvee_{V' = maxklikk} \left(\underset{i \in V \setminus V'}{\&} \overline{x}_i \right) \right] (\alpha) = 1.$$

Tõestus. 1)$\Rightarrow$ Olgu $V' \subseteq V$ graafi G klikk ja $\alpha = (\alpha_1, \ldots, \alpha_n)$ hulga V' karakteristlik vektor. Graafis G peab leiduma maksimaalne klikk V'', mis sisaldab klikki V'. Valem DF_G sisaldab termi

$$T_{V''} = \underset{i \in V \setminus V''}{\&} \overline{x}_i.$$

Term $T_{V''}$ on ilmselt tõene hulga V'' karakteristlikul vektoril β, kuna iga $i \in V \setminus V''$ jaoks $\beta_i = 0$. V' on hulga V'' alamhulk, seega võrdusest $\beta_i = 0$ järeldub $\alpha_i = 0$. Järelikult $T_{V''}(\alpha) = 1$ ja $DF_G(\alpha) = 1$.

2) $\Leftarrow$ Olgu $DF_G(\alpha) = 1$. Siis peab leiduma graafi G maksimaalne klikk V'' ja sellele vastav term

$$T_{V''} = \underset{i \in V \setminus V''}{\&} \overline{x}_i,$$

niisugune et $T_{V''}(\alpha) = 1$. See on võimalik ainult juhul, kui $\alpha_i = 0$ iga $i \in V \setminus V''$ jaoks. Siis aga $V' \subseteq V''$ ja V' on graafi G klikk. $\square$

Kirjutades eelmise näite funktsiooni f_G tõetabeli järgi välja täieliku konjunktiivse normaalkuju ja minimeerides seda, saame tulemuseks valemi $(\overline{x}_1 \vee \overline{x}_2) \,\&\, (\overline{x}_1 \vee \overline{x}_4)$. Paneme tähele, et siin on disjunktideks puuduvate servade otspunktidele vastavate muutujate eituste disjunktsioonid. Seega oleks üldkujuline valem

$$CF_G = \mathop{\&}_{\{i,j\} \notin E} (\overline{x}_i \vee \overline{x}_j).$$

Teoreem 2.2. *Olgu $G = (V, E)$ graaf, mille tippude hulk on $V = \{1, \ldots, n\}$. Kahendvektor $\alpha \in \{0,1\}^n$ on graafi G kliki karakteristlik vektor parajasti siis, kui*

$$CF_G(\alpha) = \left[\mathop{\&}_{\{i,j\} \notin E} (\overline{x}_i \vee \overline{x}_j) \right] (\alpha) = 1.$$

Tõestus. Tõestuse jätame iseseisvaks harjutuseks (vt. ülesanne 5). □

Analüüsides saadud valemeid võime öelda, et probleemi kirjeldamine DNF abil töötab nagu autokraatliku riigi seadusandlus – kõik, mis ei ole lubatud, on keelatud. CNF väljendab aga demokraatlikku seadusandlust – kõik, mis ei ole keelatud, on lubatud.

3. Tagurdusalgoritm konjunktiivsele normaalkujule

Selleks, et rakendada tagurdusalgoritmi konjunktiivsele normaalkujule C, uurime kõigepealt valemite C_x ja $C_{\overline{x}}$ leidmist.

Teoreem 3.1. *Kui C on CNF ja l on literaal, siis CNF C_l on valem C, millest on eemaldatud kõik disjunktid, mis sisaldavad literaali l ning literaal $\overline{l}$ ülejäänud disjunktidest.*

Tõestus. C_l on definitsiooni kohaselt valem C, milles on literaal l asendatud konstandiga 1 ja $\overline{l}$ konstandiga 0. Disjunktid, mis sisaldavad konstanti 1 on tõesed ja ei mõjuta valemi C_l tõeväärtust. Konstant 0 aga ei mõjuta seda sisaldava disjunkti tõeväärtust. □

Lisaks osavalemite C_x ja $C_{\overline{x}}$ lihtsustumisele võrreldes üldkujuliste lausearvutuse valemitega, pakub CNF ka teisi võimalusi tagurdusalgoritmi kiirendamiseks.

Fakt. Kui F sisaldab ühikdisjunkti l (disjunkt, mis koosneb ainult ühest literaalist), siis iga kehtestav väärtustus peab muutma disjunkti l tõeseks.

Definitsioon. Literaal l on puhas literaal valemis F, kui F ükski disjunkt ei sisalda literaali $\overline{l}$.

Teoreem 3.2. *Kui l on puhas literaal valemis F ja F on kehtestatav, siis leidub kehtestav väärtustus, milles literaal l on tõene.*

Tõestus. Olgu l puhas literaal ja $var(l) = x_i$. Kirjutame valemi F kujul, kus kõik literaali l esinemised on esile tõstetud, saame: $F = D_1 \,\&\, ... \,\&\, D_k \,\&\, (D_{k+1} \vee l) \,\&\, ... \,\&\, (D_{k+m} \vee l)$, kus $D_1, ..., D_{k+m}$ on disjunktid, mis ei sisalda literaale $l, \overline{l}$.

Olgu α suvaline väärtustus $\alpha = (\alpha_1, ..., \alpha_n)$, nii et $F(\alpha) = 1$. Kui $l(\alpha) = 1$, siis see ongi nõutud väärtustus. Kui $l(\alpha) = 0$, siis peavad disjunktid $D_1, ..., D_{k+m}$ olema tõesed väärtustusel α. Kuna ükski nimetatud disjunktidest ei sisalda literaali l, ega ka selle eitust, siis jäävad need tõeseks ka väärtustusel

$$\alpha' = (\alpha_1, \ldots, \alpha_{i-1}, \overline{\alpha}_i, \alpha_{i+1}, \ldots, \alpha_n),$$

mille jaoks $l(\alpha') = 1$. □

Definitsioon. Disjunkt D neelab disjunkti C, kui $D \subseteq C$, s.t. $C = D \vee l_1 \vee \ldots \vee l_r$ mingite literaalide $l_1, \ldots, l_r$ jaoks.

Teoreem 3.3. *Kui disjunkt D neelab disjunkti C, siis $D \& C \equiv D$.*

Tõestus. Olgu α suvaline väärtustus. $D(\alpha) = 1$ siis ja ainult siis, kui leidub literaal $l \in D$, mis on tõene väärtustusel α. Neelamise tõttu kuulub literaal l ka disjunkti C literaalide hulka, seega $C(\alpha) = 1$ ja $[D \,\&\, C](\alpha) = 1$. Kui aga $D(\alpha) = 0$, siis ka $[D \,\&\, C](\alpha) = 0$. $\square$

Konjunktiivse normaalkuju F disjunkti E nimetatakse neelatavaks, kui leidub $D \in F$ mis neelab disjunkti E. Arusaadavalt on mõistlik enne CNF-i töötlemist eemaldada kõik neelatavad disjunktid. Järgnevas algoritmis tähistatase sümboliga $\square$ tühja disjunkti. Eelnevast teame, et tühi disjunkt keelab ära kõik väärtustused, s.t valem, mis sisaldab tühja disjunkti on samaselt väär. Tühja hulga sümbol, $\emptyset$, tähistab tühja valemit, s.t. valemit, mis ei keela ühtegi väärtustust ja on seega samaselt tõene.

Järgnev algoritm on saanud oma nime autorite nimede esitähtede järgi, (Davis, Putnam, Logemann, Loveland, vt. [DP60, DLL62]).

Algoritm 3.4. *algoritm!DPLL*

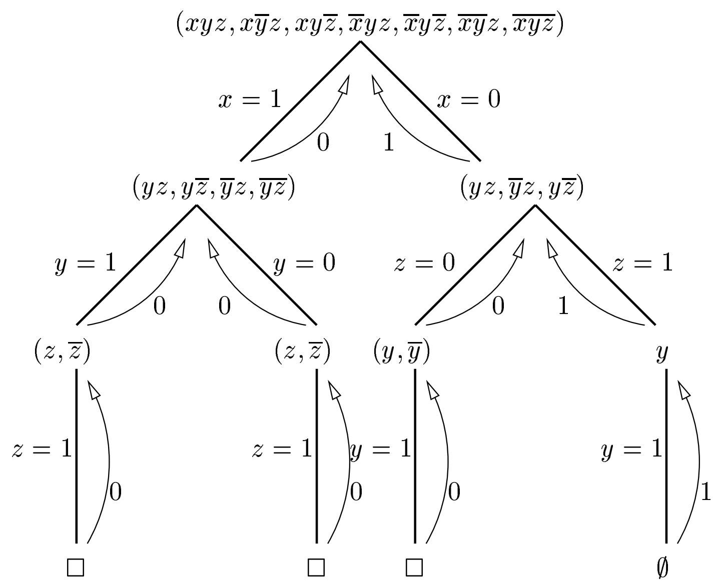
function DPLL(F: CNF): BOOLEAN
begin S0: Eemalda valemist F neelatavad disjunktid. S1: **if** $F = \emptyset$ **then return**(1); S2: **if** $\square \in F$ **then return**(0); S3: **if** $\{l\} \in F$ **then return**(DPLL(F_l)); S4: **if** l on puhas literaal **then return**(DPLL(F_l)); S5: Vali literaal l; **return**($DPLL(F_l) \vee DPLL(F_{\overline{l}})$); **end**

Algoritmi sammu S0 nimetatakse neelamisreegliks, sammu S3 ühikdisjunkti reegliks ning sammu S4 puhta literaali reegliks. Nende reeglite lubatavus järeldub teoreemidest 3.2 ja 3.3. Algoritm DPLL esindab tegelikult tervet algoritmide peret, kuna algoritmi sammus S5 (hargnemisreeglis) on käsk ”vali literaal l”, ilma et oleks täpsustatud, kuidas valikut teha. Erinevad valikustrateegiad annavad erineva jõudlusega algoritme. Valiku kriteeriumid varieeruvad lihsatest (”vali literaal, mis esineb kõige suuremas arvus disjunktides”, ”vali literaal, mille eitus esineb enam kordi lühemates disjunktides”) kuni nii keerulisteni, et nende kontrollimiseks kuluv tööaeg ületab saadava kasu.

Näide. Olgu antud CNF

$$F = (x \vee y \vee z) \mathbin{\&} (x \vee \overline{y} \vee z) \mathbin{\&} (x \vee y \vee \overline{z}) \mathbin{\&}$$
$$\mathbin{\&} (\overline{x} \vee y \vee z) \mathbin{\&} (\overline{x} \vee y \vee \overline{z}) \mathbin{\&} (\overline{x} \vee \overline{y} \vee z) \mathbin{\&} (\overline{x} \vee \overline{y} \vee \overline{z}).$$

Kasutades algoritmi 3.4 koostame puu (vaata joonist 3.1). Joonisel on puu tippudes jääkvalem, millele rakendatakse rekursiivselt algoritmi $DPLL$. Eelviimasel tasemel literaali järgi hargnemisi ei toimu, kuna rakendub ühikdisjunkti reegel. Tagasinoolte märgenditeks on käsu **return** poolt tagastatav väärtus.



Joonis 3.1: DPLL($(xyz, x\overline{y}z, xy\overline{z}, \overline{x}yz, \overline{x}y\overline{z}, \overline{xy}z, \overline{xyz})$);

4. DPLL halvima juhu eksponentsiaalne keerukus

Tagurdusalgoritmide jaoks saab eksponentsiaalne näide olla ainult samaselt väär valem, kuna suvalise kehtestatava valemi tõese väärtustuse järgi saab algoritmides SAT ja DPLL valida hargnemisliteraali vastavalt väärtustusele ja sellega tagada otsetee lahenduspuu tõese leheni. Käesolevas peatükis konstrueerime ülesannete pere, mille korral $DPLL$ töötab alati eksponentsiaalses ajas. Selleks vaatleme *tuvipesa probleemi* (inglise keeles *pidgeonhole problem*). Probleem on selles, et paigutada m tuvi n tuvipesasse, nii et igas pesas oleks ülimalt üks tuvi. Lausearvutuse valemit, mis kirjeldab kõikvõimalikke paigutusi, tähistame PHP_n^m. Arusaadavalt on ülesandel lahend ainult juhul kui $m \leqslant n$. Vastupidisel juhul on valem PHP_n^m samaselt väär. Valemit $\neg PHP_n^{n+1}$ (mis on tautoloogia) nimetatakse *tuvipesa printsiibiks* ja valemit $\neg PHP_n^m$, kus $m > n$ nimetatakse *üldistatud tuvipesa printsiibiks*. Meid huvitab tuvipesade printsiibi eitus, s.t. samaselt väär valem PHP_n^{n+1} Selle konstrueerimiseks vajame mõningaid abivahendeid (mida läheb vaja ka järgnevates peatükkides).

Defineerime valemi $atleast$:

$$atleast(k, x_1, ..., x_n) = \mathop{\&}_{\{i_1,...,i_{n-k+1}\} \subseteq \{1,...,n\}} (x_{i_1} \vee \ldots \vee x_{i_{n-k+1}}).$$

Näide. $atleast(2, x_1, x_2, x_3) = (x_1 \vee x_2) \,\&\, (x_1 \vee x_3) \,\&\, (x_2 \vee x_3)$ $n = 3,\ k = 2,\ n - k + 1 = 2.$

x_1	x_2	x_3	$x_1 \vee x_2$	$x_1 \vee x_3$	$x_2 \vee x_3$	$atleast(2, x_1, x_2, x_3)$
0	0	0	0	0	0	0
0	0	1	0	1	1	0
0	1	0	1	0	1	0
0	1	1	1	1	1	1
1	0	0	1	1	0	0
1	0	1	1	1	1	1
1	1	0	1	1	1	1
1	1	1	1	1	1	1

Vektori $\alpha \in \{0,1\}^n$ Hammingi kaal, $\|\alpha\|$, on vektori ühtede arv: $\|\alpha\| = \Sigma_{i=1}^n \alpha_i$.

Teoreem 4.1. $[atleast(k, x_1, ..., x_n)](\alpha_1, ..., \alpha_n) = 1$ *parajasti siis, kui* $\|\alpha\| \geqslant k$.

Tõestus. 1) Olgu α suvaline väärtustus, selline et

$$[Atleast(k, x_1, \ldots, x_n)](\alpha_1, \ldots, \alpha_n) = 1.$$

Oletame vastuväiteliselt, et $\|\alpha\| < k$, α sisaldab $l > n-k$ nulli, ehk vähemalt $n-k+1$ nulli. Olgu need positsioonidel $i_1, \ldots, i_{n-k+1}$. Disjunkt $(x_{i_1} \vee \ldots \vee x_{i_{n-k+1}}) \in atleast(k, x_1, \ldots, x_n)$. Kuna $(x_{i_1} \vee ... \vee x_{i_{n-k+1}})(\alpha) = 0$, siis $[atleast(k, x_1, ..., x_n)](\alpha) = 0$

2) Olgu α väärtustus,selline et $\|\alpha\| \geqslant k$, siis α sisaldab kuni $n-k$ nulli. Selleks, et mingi disjunkt oleks väär, peab väärtustuses olema vähemalt $n-k+1$ nulli. $\square$

Edaspidises läheb meil vaja ka analoogilist valemit, "ülimalt k ühte".

$$atmost(k, x_1, ..., x_n) = \underset{\substack{\alpha \in \{0,1\}^n, \\ \|\alpha\| = k+1}}{\&} \Big(\bigvee_{\alpha_i = 1} \overline{x}_i \Big)$$

Teoreem 4.2. $[atmost(k, x_1, ..., x_n)](\alpha_1, ..., \alpha_n) = 1$ *parajasti siis, kui* $|\alpha| \leqslant k$.

Tõestus. Tõestuse jätame iseseisvaks harjutuseks, vt. ülesanne 7. $\square$

Viimased kaks teoreemi kokku lubavad defineerida valemi

$$exactly(k, x_1, \ldots, x_n) = atleast(k, x_1, \ldots, x_n) \& atmost(k, x_1, \ldots, x_n),$$

mis on tõene väärtustusel α siis ja ainult siis, kui $\|\alpha\| = k$.

Tähistame $unique(x_1, \ldots, x_n)$ valemit, mis on tõene väärtustustel, mis sisaldavad täpselt ühte väärtust "1":

$$\begin{aligned} unique(x_1, \ldots, x_n) &= exactly(1, x_1, \ldots, x_n) = \\ &= atleast(1, x_1, \ldots, x_n) \; \& \; atmost(1, x_1, \ldots, x_n) = \\ &= (x_1 \vee \ldots \vee x_n) \; \& \left(\underset{1 \leqslant i < j \leqslant n}{\&} (\overline{x}_i \vee \overline{x}_j) \right) \end{aligned}$$

Lemma 4.3. *Valemi* $unique(x_1, ..., x_n)$ *korral kehtib:*

1) *Valem on sümmeetriline literaalide vahetuse suhtes.*

2) $n \geqslant 2$ *korral esineb iga literaal* x_i *nii positiivselt kui negatiivselt ning ei esine kunagi üksi (vaid alati disjunktis mõne teise muutujaga).*

3) $unique(x_1, \ldots, x_n)$ *neelab mõnda* $unique(y_1, \ldots, y_m)$ *osavalemit siis ja ainult siis, kui* $\{x_1, \ldots, x_n\} \subset \{y_1, \ldots, y_m\}$.

4)

$$unique_{x_1}(x_2,\dots,x_n) = \mathop{\&}_{i=2,\dots,n} \overline{x}_i.$$

5)

$$unique_{\overline{x}_1}(x_2,\dots,x_n) = unique(x_2,\dots,x_n).$$

Tõestus. 1) ja 2) on ilmsed. 3), 4) ja 5) on ülesandeks lugejale (vt. ülesannet 8). □

Konstrueerime nüüd valemite pere PHP_n^{n+1} parameetri n põhjal. Muutujate hulk $x_{1,1},\dots,x_{n,n+1}$ moodustab $n \times (n+1)$ tabeli. Järgnev valem väidab sisuliselt seda, et tabeli igas reas ja igas veerus on täpselt üks muutuja väärtustatud tõeseks:

$$\begin{aligned} PHP_n^{n+1}(x_{1,1},\dots,x_{n,n+1}) = \\ = \left(\mathop{\&}_{i=1,\dots,n} unique(x_{i,1},\dots,x_{i,n+1}) \right) \& \\ \& \left(\mathop{\&}_{i=1,\dots,n+1} unique(x_{1,i},\dots,x_{n,i}) \right). \end{aligned}$$

Valem on ilmselgelt samaselt väär, sest veerge on ühe võrra rohkem, kui ridu.

Lemma 4.4. *Olgu T valem, mis avaldub valemite unique konjunktsioonina, kusjuures T iga muutuja x sisaldub täpselt kahes alamvalemis unique, mis mõlemad sõltuvad veel vähemalt n teisest muutujast, kusjuures ükski muutuja peale x ei sisaldu mõlemas neis kahes valemis. Siis teeb $DPLL$ valemi T väärtuse otsimisel igas vaadeldavas harus vähemalt n hargnemist.*

Tõestus. Tõestame väite induktsiooniga:

Baas: $n = 0$ korral on väide ilmne (sest teoreem ei väida sel juhul sisuliselt midagi).

Samm: $n \geqslant 1$. Teoreemi eeldustest lähtuvalt on selge, et iga valem *unique* sõltub vähemalt kahest muutujast ja iga muutuja sisaldub täpselt kahes valemis *unique*. Lemma 4.3 p.2 tõttu ei saa seega kasutada ei ühikdisjunkti ega ka puhta literaali reeglit ning p.3 välistab neelamisreegli (sest üheski kahes valemis *unique* pole üle ühe ühise muutuja ning muutujaid on kõigis vähemalt 2). Seega on vaja rakendada hargnemisreeglit. Vaatleme eraldi mõlemas harus toimuvat, kui fikseeritakse suvaline muutuja x:

$x = 1$ Lemma 4.3 p.4 põhjal teisenevad mõlemad muutujat x sisaldanud valemid $unique$ negatiivsete literaalide konjunktsiooniks. $DPLL$ kasutab enne oma järgmist hargnemist kõigi nende literaalide jaoks ühikdisjunkti reeglit ja fikseerib nad negatiivse väärtusega. Selle tulemusena lühenevad kõik neid sisaldanud valemid $unique$ ühe muutuja võrra (Lemma 4.3 p.5). Et aga ükski teine muutuja peale x ei sisaldu mõlemas valemis $unique$, ei saa ükski neist lüheneda rohkem kui 1 muutuja jagu.

$x = 0$ Lemma 4.3 pt 5 tõttu lühenevad mõlemad muutujat x sisaldanud valemid $unique$ lihtsalt ühe muutuja võrra.

Paneme tähele, et mõlemal juhul taandab $DPLL$ valemi T valemiks T', mis rahuldab kõiki teoreemi eeldusi juhul $n-1$, mis tähendab induktsiooni hüpoteesi järgi, et selles tuleb teha veel vähemalt $n-1$ hargnemist igas harus. Lisades sellele juurde muutuja x fikseerimisel tehtud hargnemise saamegi vajalikud n sammu. □

Teoreem 4.5. *Algoritm $DPLL$ töötab valemitel PHP_n^{n+1} sisendi pikkuse suhtes eksponentsiaalses ajas.*

Tõestus. Valem PHP_n^{n+1} on $unique$-valemite konjunktsioon, mis rahuldab lemma 4.4 eeldusi $n-1$ korral. Seega tehakse igas vaadeldud harus vähemalt $n-1$ hargnemist. Et PHP on samaselt väär, on $DPLL$ sunnitud läbi vaatama kõik harud. Seega teeb algoritm vähemalt $\sum_{i=0}^{n-1} 2^i = 2^n - 1$ sammu, mis on eksponentsiaalne PHP_n^{n+1} pikkuse ($O(n^3)$) suhtes

. □

5. Lahendite loendamine elimineerimismeetodil

Lahendite loendamiseks ei ole tingimata vajalik valemi kõigi tõeste väärtustuste leidmine. Disjunktiivne ja konjunktiivne normaalkuju sisaldavad loendamiseks vajalikku informatsiooni, vaja on seda kasutades teha vastavad arvutused. Käesolevas peatükis esitatud meetod on võetud K. Iwama artiklist [Iwa89].

Disjunktiivset ja konjunktiivset normaalkuju võib vaadelda literaalide hulkade hulkadena, erinev on ainult semantika. Seetõttu on mõistlik uurida nende omadusi koos. Unustame ajutiselt tehtemärgid ja tegeleme literaalide hulkadega. Olgu muutujate hulk

$$X = \{x_1, \ldots, x_n\}.$$

Definitsioon. Literaalide hulk $\{l_1, ..., l_k\}$ sisaldab *kontraarset paari*, kui selles hulgas leidub mingi muutuja ja tema eitus.

Definitsioon. Literaalide hulk on *kooskõlaline*, kui see ei sisalda kontraarset paari.

Definitsioon. *Normaalkuju* on kooskõlaliste literaalihulkade hulk.

Definitsioon. Kooskõlaline literaalide hulk on *täielik*, kui selle võimsus on võrdne muutujate arvuga n.

Definitsioon. Normaalkuju on *täielik,* kui selle iga literaalihulk on täielik.

Kui $L = \{l_1, \ldots, l_k\}$ on kooskõlaline literaalide hulk ning $y_1, ..., y_{n-k}$ on muutujad, mis ei esine hulgas L ehk

$$X \setminus \{var(l_1), \ldots, var(l_k)\} = \{y_1, ..., y_{n-k}\},$$

siis hulga L täielik normaalkuju on

$$gen(L) = \{\{l_1, \ldots, l_k, y_1^{\alpha_1}, \ldots, y_{n-k}^{\alpha_{n-k}}\} : \alpha \in \{0,1\}^{n-k}\}.$$

Normaalkuju $gen(L)$ koosneb ilmselt 2^{n-k} täielikust literaalide hulgast. Kui normaalkuju $N = \{L_1, \ldots, L_p\}$, siis vastav täielik normaalkuju on literaalihulkade täielike normaalkujude hulgateoreetiline summa

$$gen(N) = \bigcup_{i=1}^{p} gen(L_i).$$

Kui N esitab disjunktiivset normaalkuju, siis selle tõeste väärtustuste arv on $|gen(N)|$. Kui N esitab konjunktiivset normaalkuju, siis selle väärade väärtustuste arv on $|gen(N)|$ ning tõeste väärtustuste arv $2^n - |gen(n)|$. Mõlemal juhul tuleb lahendite loendamiseks osata arvutada suurust $|gen(N)|$. Literaalide hulga $L = \{l_1, \ldots, l_k\}$ jaoks on probleem lihtne – $|gen(L)| = 2^{n-k}$. Selleks, et arvutada $|gen(N)|$ tuleb aga arvestada, et $|gen(L_i)|$ ja $|gen(L_j|$ võivad sisaldada ühiseid elemente. Seega tuleb kasutada kombinatoorikast tuntud elimineerimisteoreemi:

Kui $X = X_1 \cup \ldots \cup X_p$, siis

$$|X| = \sum_{k=1}^{p} (-1)^{k-1} \left(\sum_{\{i_1,\ldots,i_k\} \subseteq \{1,\ldots,p\}} |X_{i_1} \cap \ldots \cap X_{i_k}| \right).$$

Elimineerimisteoreemi rakendamiseks meie juhule peame oskama arvutada suurusi $|gen(L_{i_1}) \cap \ldots \cap gen(L_{i_k})|$.

Teoreem 5.1.

$$gen(L_1) \cap ... \cap gen(L_m) = \begin{cases} \emptyset, \textit{kui } L_1 \cup ... \cup L_m \textit{ sisaldab} \\ \quad \textit{kontraarset literaalipaari,} \\ gen(L_1 \cup ... \cup L_m), \textit{vastupidisel juhul} \end{cases}$$

Tõestus.

1) Sisaldagu $L_1 \cup ... \cup L_m$ kontraarset paari $(x, \overline{x})$, s.t. leiduvad i, j nii et $x \in L_i$, $\overline{x} \in L_j$. Siis $gen(L_i)$ iga literaalihulk sisaldab literaali x ja $gen(L_j)$ iga literaalihulk sisaldab literaali $\overline{x}$, seega $gen(L_i) \cap gen(L_j) = \emptyset$, aga siis on ka $gen(L_1) \cap ... \cap gen(L_m) = \emptyset$.

2) $L_1 \cup ... \cup L_m$ ei sisalda kontraarset paari. Tõestame induktsiooniga m järgi.

Baas: m=1, $gen(L_1) = gen(L_1)$.

Samm: Rakendame hulkade ühisosale

$$gen(L_1) \cap \ldots \cap gen(L_{m-1}) \cap gen(L_m)$$

induktsiooni hüpoteesi:

$$\begin{aligned}(gen(L_1) \cap \ldots \cap gen(L_{m-1})) \cap gen(L_m) = \\ = gen(L_1 \cup ... \cup L_{m-1}) \cap gen(L_m).\end{aligned}$$

Kuna vaadeldavad literaalide hulgad ei sisalda kontraarset paari,võime need esitada kujul

$$\begin{aligned} L_1 \cup \ldots \cup L_{m-1} &= \{a_1, \ldots, a_r, b_1, \ldots, b_s\}, \\ L_m &= \{a_1, \ldots, a_r, c_1, \ldots, c_t\}. \end{aligned}$$

Literaalid $a_1, \ldots, a_r$ on nendele hulkadele ühised. Tähistame $p = n - (r + s + t)$ ning $y_1, \ldots, y_p$ olgu muutujad, mis ei sisaldu hulgas

$$\{var(a_1), ..., var(a_r), var(b_1), ..., var(b_s), var(c_1), ..., var(c_t)\},$$

siis

$$\begin{aligned} & gen(L_1) \cap \ldots \cap gen(L_{m-1}) \cap gen(L_m) = \\ & = gen(L_1 \cup \ldots \cup L_{m-1}) \cap gen(L_m) = \\ & = \{\{a_1, \ldots, a_r, b_1, \ldots, b_s, c_1^{\alpha_1}, \ldots, c_t^{\alpha_t}, y_1^{\beta_1}, \ldots, y_p^{\beta_p}\} : \\ & \qquad : \alpha \in \{0,1\}^t,\ \beta \in \{0,1\}^p\} \cap \\ & \cap \{\{a_1, \ldots, a_r, b_1^{\phi_1}, \ldots, b_s^{\phi_s}, c_1, \ldots, c_t, y_1^{\beta_1}, \ldots, y_p^{\beta_p}\} : \\ & \qquad : \phi \in \{0,1\}^s,\ \beta \in \{0,1\}^p\} = \\ & = \{\{a_1, \ldots, a_r, b_1, \ldots, b_s, c_1, \ldots, c_t, y_1^{\beta_1}, \ldots, y_p^{\beta_p}\} : \beta \in \{0,1\}^p\} = \\ & \qquad = gen(L_1 \cup \ldots \cup L_m). \end{aligned}$$

$\square$

Näide. $F = \{\underbrace{\{x, \overline{y}\}}_{L_1}, \underbrace{\{\overline{x}\}}_{L_2}, \underbrace{\{x, z\}}_{L_3}\}$, $X = \{x, y, z\}$.
$gen(L_1) = \{\{x, \overline{y}, z\}, \{x, \overline{y}, \overline{z}\}\}$.
$gen(L_2) = \{\{\overline{x}, y, z\}, \{\overline{x}, y, \overline{z}\}, \{\overline{x}, \overline{y}, z\}, \{\overline{x}, \overline{y}, \overline{z}\}\}$.
$gen(L_3) = \{\{x, y, z\}, \{x, \overline{y}, z\}\}$.
$|gen(L_1)| = 2,\ |gen(L_2)| = 4,\ |gen(L_3)| = 2$.
$gen(L_1) \cap gen(L_2) = \emptyset$.
$gen(L_2) \cap gen(L_3) = \emptyset$.
$gen(L_1) \cap gen(L_3) = \{x, \overline{y}, z\}$.
$|gen(F)| = |gen(L_1)| + |gen(L_2)| + |gen(L_3)| - (|gen(L_1) \cap gen(L_2)| +$
$|gen(L_1) \cap gen(L_3)| + |gen(L_2) \cap gen(L_3)|) +$
$|gen(L_1) \cap gen(L_2) \cap gen(L_3)| =$
$2^1 + 2^2 + 2^1 - (0 + 2^0 + 0) + 0 = 7$.

6. Ortogonaliseerimine

Elimineerimisteoreemi kasutamist on võimalik vältida, teisendades sobival viisil normaalkuju. Meetod, mida siin esitame on võetud artiklitest [Dub91, Tom93].

Definitsioon. Kooskõlalised literaalide hulgad D ja E on ortogonaalsed, kui

$$gen(D) \cap gen(E) = \emptyset.$$

Fakt. D ja E on ortogonaalsed siis ja ainult siis, kui $D \cup E$ sisaldab kontraarset literaalipaari (järeldub teoreemist **??**).

Definitsioon. Normaalkuju F on ortogonaalne, kui F literaalide hulgad on paarikaupa ortogonaalsed.

Ortogonaalse normaalkuju $F = \{X_1, \ldots, X_p\}$ puhul kehtib $gen(X_i) \cap gen(X_j) = \emptyset$ iga i, j jaoks $1 \leqslant i < j \leqslant p$ ja seetõttu

$$|gen(X)| = \sum_{i=1}^{p} |gen(X_i)|.$$

Kui tõlgendada ortogonaalset normaalkuju F disjunktiivse normaalkujuna, saame tõeste väärtustuste arvuks

$$\#F = \sum_{i=1}^{p} 2^{n-|X_i|}$$

ning tõlgendades seda konjunktiivse normaalkujuna saame

$$\#F = 2^n - \sum_{i=1}^{p} 2^{n-|X_i|}.$$

Siit tuleb idee teisendada konjunktiivne normaalkuju ortogonaalseks, et vältida elimineerimisteoreemi rakendamise vajadust.

Algoritm 6.1. *Disjunktipaari ortogonaliseerimine*

function Ort(D,E: disjunktid): CNF

begin
    if(D ja E on ortogonaalsed) **then return**(D & E) **fi**
    if D neelab E **then return**(D) **fi**
    $D = a_1 \vee \ldots \vee a_k \vee c_1 \vee \ldots \vee c_l$
    $E = a_1 \vee ... \vee a_k \vee b_1 \vee ... \vee b_r$
      kus $var(c_i) \neq var(b_j)$, $1 \leqslant i \leqslant l$, $1 \leqslant j \leqslant r$.
      (Olgu $r \geqslant l$)
    return (D & E_1 & E_2 & ... & E_l), kus $E_1 = E \vee \overline{c}_1$
$E_2 = E \vee c_1 \vee \overline{c}_2$
$E_3 = E \vee c_1 \vee c_2 \vee \overline{c}_3$
....
$E_l = E \vee c_1 \vee ... \vee c_{l-1} \vee \overline{c}_l$
    fi
end

Näide $\overbrace{(\underbrace{x}_{a_1} \vee \underbrace{y}_{a_2} \vee \underbrace{\overline{z}}_{c_1} \vee \underbrace{v}_{c_2})}^{D} \& \overbrace{(\underbrace{x}_{a_1} \vee \underbrace{y}_{a_2} \vee \underbrace{w}_{b_1} \vee \underbrace{\overline{u}}_{b_2})}^{E}$

Kasutades algoritmi 6.1, saame
$D = (x \vee y \vee \overline{z} \vee v)$
$E_1 = (x \vee y \vee w \vee \overline{u} \vee z)$
$E_2 = (x \vee y \vee w \vee \overline{u} \vee \overline{z} \vee \overline{v})$

Teoreem 6.2. *Ort(D, E) on ortogonaalne CNF, mis on ekvivalentne D & E-ga.*

Tõestus.

Kui D ja E on ortogonaalsed, siis algoritmi väljund on D & E ja teoreemi väited kehtivad. Samuti on ilmne juhtum, kus üks disjunkt neelab teist. Mittetriviaalsel juhul on iga i jaoks ($1 \leqslant i \leqslant l$) disjunkt D disjunktiga E_i ortogonaalne (kontraarne paar $c_i \in D$ ning $\overline{c}_i \in E_i$). Iga i, j jaoks ($1 \leqslant i < j \leqslant l$) on E_i ja E_j ortogonaalsed, kuna $\overline{c}_i \in E_i$ ja $c_i \in E_j$.

Näitame, et $D \& E \equiv D \& E_1 \& ... \& E_l$. Olgu α suvaline väärtustus $\alpha \in \{0,1\}^n$. Kui $[D \& E](\alpha) = 1$, siis $D(\alpha) = 1$ *ja* $E(\alpha) = 1$, seega ka $E_i(\alpha) = [E \vee c_1 \vee ... \vee c_{i-1} \vee \overline{c}_i](\alpha) = 1$, kuna $E(\alpha) = 1$. Kui $[D \& E](\alpha) = 0$, siis kas $D(\alpha) = 0$, või $D(\alpha) = 1$ ja $E(\alpha) = 0$. Esimesel juhul $[D \& E_1 \& ... \& E_l](\alpha) = 0$, teisel juhul $a_1(\alpha) = 0, \ldots, a_k(\alpha) = 0$. Siis vähemalt üks literaalidest $c_i = 1$. Olgu i_0 vähim i, et

$c_{i_0}(\alpha) = 1$, siis $E_{i_0}(\alpha) = E(\alpha) \vee c_1(\alpha) \vee ... \vee c_{i_0-1}(\alpha) \vee \overline{c}_{i_0(\alpha)} = 0$, seega ka $[D \,\&\, E_1 \,\&\, ... \,\&\, E_l](\alpha) = 0$. □

Algoritm 6.3. *Ortogonaliseerimine*

function Orthogonalize(F:CNF):CNF
Algoritm kasutab disjunktide hulka L, milles on
märgendatud ortogonaliseeritud disjunktid

begin
    $L := F$
    while L sisaldab märgendamata disjunkte
        do Vali märgendamata $D \in L$, märgenda D
        for $E \in \{$valemi L märgendamata disjunktid$\}$
            do eemalda E, lisa ort$(D \,\&\, E)\backslash\{D\}$
        end for
    end while
    return L
end

7. Ortogonaalne DNF

Kui analüüsida algoritmi SAT esimesest peatükist, siis näeme, et literaalid, mis märgendavad lahenduspuu teed kuni leheni, milles jääkvalem on 1 (tõene), moodustavad disjunktiivse normaalkuju termi ning erinevatele lehtedele vastavad termid sisaldavad kontraarset literaalipaari. Siit tuleb idee modifitseerida algoritmi SAT nii, et väljundiks oleks ortogonaalne DNF. Algoritmi esimene parameeter on üldkujuline lausearvutuse valem, teine parameeter on sõnemuutuja, mis on algselt tühi sõne Λ (mis tähistab tühja termi) ja väljund on samuti sõne. Funktsioonist väljumisel käsuga **return** teostatakse sõnemuutujate ja sõnede konkatenatsiooni.

Algoritm 7.1. *SATO. Valem $\Rightarrow$ ortogonaalne DNF.*

function SATO(F:formula,C:string):string

begin
    if $F = 1$ **then return**(C) **fi**
    if $F = 0$ **then return**$(\emptyset)$ **fi**
    vali muutuja x;
    return$(SATO(F_x, C\smile' \&' \smile' x')\smile' \vee' \smile$
    $\smile SATO(F_{\overline{x}}, C\smile' \&' \smile' \overline{x}'))$;
end

Teoreem 7.2. *SATO(F, Λ) on ortogonaalne DNF, mis on ekvivalentne valemiga F.*

Tõestus.

1) Ortogonaalsus. Algoritmi kolmas samm tagastab $SATO(F_x, C\smile' \&' \smile' x')$, kus iga term sisaldab literaali x ja $SATO(F_{\overline{x}}, C\smile' \&' \smile' \overline{x}')$, kus iga term sisaldab literaali $\overline{x}$, millega on ortogonaalsus tagatud.

2) Ekvivalentsus. Tõestame induktsiooniga muutujate arvu n järgi.

Baas: $n = 1$ Pärast lihtsustamisi on neli erinevat ühe muutujaga valemit: 0, x, $\overline{x}$ ja 1. Algoritmi SATO väljundid on vastavalt $\emptyset, x, \overline{x}$ ja Λ.

Samm: Olgu valitav muutuja x_1, siis

$$\begin{aligned} SATO(F_{x_1}(x_2, ..., x_n),' x_1')\smile' \vee' \smile SATO(F_{\overline{x}_1}(x_2, ..., x_n),' \overline{x}_1') = \\ = SATO(F_{x_1}(x_2, ..., x_n), \Lambda)\smile' \&' \smile' x_1' \smile' \vee' \smile \\ \smile SATO(F_{\overline{x}_1}(x_2, ..., x_n), \Lambda)\smile' \&' \smile' \overline{x}_1'. \end{aligned}$$

Induktsiooni hüpoteesi kohaselt
$SATO(F_{x_1}(x_2,...,x_n),\Lambda) \equiv F_{x_1}$,
$SATO(F_{\overline{x}_1}(x_2,...,x_n),\Lambda) \equiv F_{\overline{x}_1}$ ning $F \equiv F_{x_1} \,\&\, x_1 \vee F_{\overline{x}_1} \,\&\, \overline{x}_1$. $\square$

Sama algoritmi saame rakendada ka konjunktiivsele normaalkujule, kasutades kiirendavaid ühikdisjunkti ja neelamise reegleid. Puhta literaali reegli kasutamine annaks vale tulemuse, kuna see võib lõigata ära lahenduspuu mõned tõesed harud.

Algoritm 7.3. *DPLLO. CNF ⇒ ortogonaalne DNF.*

```
function DPLLO(F:CNF,C:string):string
begin
        S0: Eemalda valemist F neelatavad disjunktid.
        S1: if F = ∅ then return(C) fi
        S2: if F = □ then return(Λ) fi
        S3: if {l} ∈ F then return (DPLL(F_l, C␣' &'␣' l')) fi
        S4: vali muutuja x;
            return((DPLLO(F_x, C␣' &'␣' x')␣' ∨'␣
                    ␣DPLLO(F_x̄, C␣' &'␣' x̄'))
end
```

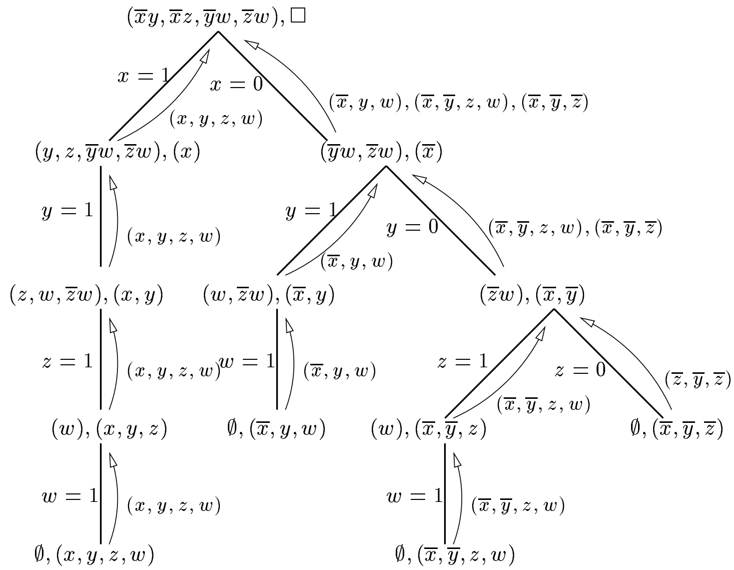
Näide. $F = ((\overline{x} \vee y) \,\&\, (\overline{x} \vee z) \,\&\, (\overline{y} \vee w) \,\&\, (\overline{z} \vee w)), C = \Lambda$. Kasutades algoritmi 7.3 koostame puu (vaata joonist 7.1).

Mööda puud üles liikudes saame valemi
$G = (x \,\&\, y \,\&\, z \,\&\, w) \vee (\overline{x} \,\&\, y \,\&\, w) \vee (\overline{x} \,\&\, \overline{y} \,\&\, z \,\&\, w) \vee (\overline{x} \,\&\, \overline{y} \,\&\, \overline{z})$.

Kui meie eesmärk on tõeste väärtustuste loendamine, siis saame algoritmides SATO ja DPLLO loobuda ortogonaalse DNF-i väljastamisest, kuna lahenduspuu tõese lehe kaugus puu juurest on ka vastava termi literaalide arv. See inspireerib konstrueerima järgmisi algoritme.

Algoritm 7.4. *Lausearvutuse valemi lahendite loendamine.*

```
function #SAT(F:formula,k:int):int
begin
        if F = 1 then return 2^k fi
        if F = 0 then return(0) fi
        vali muutuja x;
        return(#SAT(F_x, k − 1) + #SAT(F_x̄, k − 1));
end
```



Joonis 7.1: DPLLO$((\overline{x}y, \overline{x}z, \overline{y}w, \overline{z}w), \square)$;

Algoritm 7.5. *CNF-i lahendite loendamine*

function #DPLL(F:CNF,k:INT):INT
begin
S0: Eemalda valemist F neelatavad disjunktid.
S1: **if** $F = \emptyset$ **then return**(2^k) **fi**
S2: **if** $\square \in F$ **then return**(0) **fi**
S3: **if** $\{l\} \in F$ **then return**$(\#DPLL(F_l, k-1)$ **fi**
S4: vali muutuja x;
return$(\#DPLL(F_x, k-1) + \#DPLL(F_{\overline{x}}, k-1))$;
end

Teine parameeter, k näitab veel väärtustamata muutujate arvu. Algoritmide $\#SAT$ ja $\#DPLL$ väljundiks on valemi F tõeste väärtustuste arv.

Näide.

$\#DPLL((\overline{x} \vee y) \& (\overline{x} \vee z) \& (\overline{y} \vee w) \& (\overline{z} \vee w), 4) = 2^0 + 2^1 + 2^0 + 2^1 = 6.$

Algoritmide $\#SAT$ ja$\#DPLL$ õigsus järeldub teoreemist 7.2, kuna lahenduspuu 1-lehtedele vastavad valemiga F ekvivalentse ortogonaalse disjunktiivse normaalkuju termid.

8. Moon-Moseri graafid

Kõik eelmistes peatükkides esitatud loendamisalgoritmid on halvimal juhul eksponentsiaalse tööajaga. Tagurdusmeetodil põhinevate loendamisalgoritmide jaoks sobib eksponentsiaalset aega nõudvaks näiteks valem PHP_n^{n+1}, kuna loendamisalgoritm peab kokku liitma ka kõik nullid, s.t. läbima kõik lahenduspuu lehed, mis vastavad väärale väärtustusele. Niisuguseid lehti on valemi PHP_n^{n+1} lahenduspuus teoreemi 4.5 põhjal vähemalt $2^n - 1$. Siiski jääb väike kahtlus: äkki leidub eksponentsiaalset tööaega nõudev näide ainult samaselt väärate valemite hulgas?

Käesolevas peatükis esitame valemite pere, mille puhul ei ole tegemist samaselt väärate valemitega ning mis nõuab eksponentsiaalset tööaega kõigilt seniesitatud loendamisalgoritmidelt. Selleks tuleb kõigepealt tegeleda monotoonsete Boole'i funktsioonide omaduste uurimisega.

Definitsioon. Olgu $\alpha = (\alpha_1, \ldots, \alpha_n)$ ja $\beta = (\beta_1, \ldots, \beta_n)$ kaks väärtustust pikkusega n. Me ütleme, et α *eelneb vahetult* väärtustusele β (tähistame $\alpha \prec \beta$), kui

$$(\exists i)[\alpha_i < \beta_i \;\&\; (\forall j)((j \neq i) \supset (\alpha_j = \beta_j))].$$

Relatsiooni $\prec$ transitiivset sulundit tähistame $\prec^+$ ja nimetame *eelneb*.

Definitsioon. Boole'i funktsioon $f : \{0,1\}^n \to \{0,1\}$ on *monotoonne*, kui iga $\alpha \prec^+ \beta$ jaoks $f(\alpha) \leqslant f(\beta)$.

Definitsioon. $f : \{0,1\}^n \to \{0,1\}$ on *antimonotoonne*, kui iga $\alpha \prec^+ \beta$ jaoks $f(\alpha) \geqslant f(\beta)$.

Teoreem 8.1. $f : \{0,1\}^n \to \{0,1\}$ *on monotoonne siis ja ainult siis, kui* $\neg f$ *on antimonotoonne.*

Tõestus. Tõestuseks märgime, et iga $\alpha, \beta \in \{0,1\}^n$ jaoks $f(\alpha) \leqslant f(\beta)$ siis ja ainult siis, kui $\neg f(\alpha) \geqslant \neg f(\beta)$. Loomulikult kehtib see ka juhul, kui $\alpha \prec^+ \beta$. $\square$

Definitsioon. Boole'i funktsiooniga $f(x_1, \ldots, x_n)$ *duaalne* funktsioon $f_{dual}(x_1, \ldots, x_n) = \neg f(\overline{x}_1, \ldots, \overline{x}_n)$.

Teoreem 8.2. *Kui funktsioon f on monotoonne, siis on monotoonne ka f_{dual}.*

Tõestus. Olgu α ja β kaks väärtustust nii, et $\alpha \prec^{+} \beta$. Siis $\overline{\beta} \prec^{+} \overline{\alpha}$. Funktsiooni f monotoonsuse tõttu $f(\overline{\beta}) \leqslant f(\overline{\alpha})$. Siis aga $\neg f(\overline{\alpha}) \leqslant \neg f(\overline{\beta})$, s.t. $f_{dual}(\alpha) \leqslant f_{dual}(\beta)$. □

Teoreem 8.3. *Kui Boole'i funktsiooni $f(x_1, \ldots, x_n)$ konjunktiivne normaalkuju on C ja disjunktiivne normaalkuju on D, siis funktsiooni $f_{dual}(x_1, \ldots, x_n)$ disjunktiivse normaalkuju saame valemist C, asendades kõik tehtemärgid ' & ' tehtemärkidega ' $\vee$ ' ja vastupidi ning funktsiooni $f_{dual}(x_1, \ldots, x_n)$ konjunktiivse normaalkuju saame samal viisil valemist D.*

Tõestus. Tõestada ise! vt. ülesanne 15. □

Tuletame pisut meelde Boole'i funktsiooni f minimaalse disjunktiivse normaalkuju leidmist Quine-McCluskey meetodil. *Lihtimplikant* on disjunktiivse normaalkuju term, millest ei saa jätta välja ühtegi literaali, ilma et muudetud normaalkuju kaotaks samaväärsust esialgsega.

Termide ühildumisteisendus on operatsioon, mis teisendab termipaari $T \mathbin{\&} x$, $T \mathbin{\&} \overline{x}$ termiks T.

Quine-McCluskey meetod koosneb kahest etapist:

1) Leida funktsiooni f täieliku disjunktiivse normaalkuju sulund termide ühildumisteisenduse suhtes ja eemaldada sellest kõik neelatavad termid. Tulemus on funktsiooni f kõigi lihtimplikantide hulk.

2) Leida lihtimplikantide hulga minimaalne kate, s.t. selline alamhulk, milles on vähim arv literaale ja mis on ekvivalentne funktsiooniga f.

Teoreem 8.4. *Monotoonse Boole'i funktsiooni minimaalne DNF koosneb kõigist selle lihtimplikantidest ja sisaldab ainult positiivseid literaale.*

Tõestus. Olgu $l_1 \mathbin{\&} \ldots \mathbin{\&} l_k$ üks monotoonse Boole'i funktsiooni f lihtimplikantidest. Oletame vastuväiteliselt, et literaal $l_k = \overline{x}_i$ mingi muutuja $x_i \in X$ jaoks. Siis igas väärtustuses $\alpha \in \{0,1\}^n$, sellises et $[l_1 \mathbin{\&} \ldots \mathbin{\&} l_k](\alpha) = 1$ peab $\alpha_i = 0$, s.t.

$$\alpha = (\alpha_1, \ldots, \alpha_{i-1}, 0, \alpha_{i+1}, \ldots, \alpha_n).$$

Funktsiooni f monotoonsuse tõttu $f(\alpha_1, \ldots, \alpha_{i-1}, 1, \alpha_{i+1}, \ldots, \alpha_n) = 1$. Siis aga on funktsiooni f lihtimplikantide hulgas $l_1 \mathbin{\&} \ldots \mathbin{\&} l_{k-1}$, mis on vastuolus eeldusega, et $l_1 \,\&\, \ldots \,\&\, l_k$ on lihtimplikant. Väärtustus, mis annab literaalidele $l_1, \ldots, l_k$ väärtuse 1 ja kõigile ülejäänutele väärtuse 0, muudab vääraks kõik ülejäänud lihtimplikandid. Seega lihtimplikant $l_1 \,\&\, \ldots \,\&\, l_k$ peab kuuluma funktsiooni f minimaalsesse DNF-i. □

Järeldus 8.5. *Monotoonse Boole'i funktsiooni minimaalne CNF sisaldab ainult positiivseid literaale.*

Tõestus. Olgu D monotoonse funktsiooniga f duaalse funktsiooni f_{dual} minimaalne DNF. Teoreemide 8.2 ja 8.4 põhjal sisaldab D ainult positiivseid literaale. Valem D_{dual} on funktsiooni f konjunktiivne normaalkuju, mis sisaldab samuti ainult positiivsed literaale. Oletame, et funktsioonil f leidub CNF E, mis sisaldab vähem literaale kui D_{dual}. Siis E_{dual} on funktsiooni f_{dual} DNF ning D ei ole funktsiooni f_{dual} minimaalne DNF. Seega on D_{dual} funktsiooni f minimaalne CNF. □

Järeldus 8.6. *Antimonotoonse funktsiooni minimaalne CNF ja DNF sisaldavad ainult negatiivseid literaale.*

Tõestus. Tõestada ise, vt. ülesanne 16. □

Lähtudes nendest minimaalse normaalkuju omadustest on vaja otsida monotoonset (või antimonotoonset) Boole'i funktsiooni f, mille minimaalse DNF pikkus on eksponentsiaalne selle CNF-i pikkusest. Siis on funktsiooni f iga ortogonaalse DNF-i pikkus samuti eksponentsiaalne ning algoritm $\#DPLL$ sammude arv tuleb eksponentsiaalne. Tuletades meelde näiteid graafi klikkide struktuuri väljendamisest DNF-i ja CNF-i abil, oleks vaja leida graafide pere, mille maksimaalsete klikkide arv oleks eksponentsiaalne graafi servade arvust. Selline graafide pere on tõepoolest teada, need on *Moon-Moseri* graafid ([MM65]), mis on kohustuslikuks testülesandeks igale suurima kliki leidmise algoritmile.

Definitsioon. Graafi $MM(m, l) = (V, E)$, kus

$$V = \begin{pmatrix} x_{11} & \ldots & x_{1l} \\ \vdots & & \vdots \\ x_{m1} & \ldots & x_{ml} \end{pmatrix}$$

ja $E = \{\{x_{ij}, x_{rs}\} : j \neq s\}$ nimetatakse (m, l)-Moon-Moseri graafiks.

Piltlikult: G on $MM(m,l)$-graaf, kui graafi G täiendgraaf $co-G$ koosneb l isoleeritud m-klikist K_m. Graafi G iga maksimaalne klikk sisaldab täpselt ühe tipu (suvalise) igast coG klikist K_m. Graafis G on seega m^l maksimaalset klikki.

Teoreem 8.7. *Leidub konjunktiivsel normaalkujul olevate valemite pere $\{M_l\}$, millel algoritmide $\#SAT$ ja $\#DPLL$ tööaeg on eksponentsiaalne.*

Tõestus. Tähistame $M_l = MM(2,l)$. Graafide M_l ja $co-M_l$ tippude arv on $n = 2l$, graafi $co-M_l$ servade arv on l ja graafi M_l maksimaalsete klikkide arv on 2^l. Graafi M_l klikkide struktuuri väljendava CNF-i

$$CF_{M_l} = \mathop{\&}_{i=1}^{l} (\overline{x}_{1i} \vee \overline{x}_{2i})$$

disjunktide arv on l, igaühes 2 literaali. Sama funktsiooni DNF

$$DF_{M_l} = \bigvee_{(i_1,\ldots,i_l)\in\{1,2\}^l} (\overline{x}_{i_1 1} \,\&\, \ldots \,\&\, \overline{x}_{i_l l})$$

sisaldab 2^l termi, igaühes l literaali. Mõlemas valemis esinevad muutujad ainult negatiivselt ning ükski disjunkt (term) ei neela teisi, seega on need normaalkujud järelduse 8.6 põhjal minimaalsed.

Algoritmide #SAT ja #DPLL lahenduspuu lehtede arv on võrdne algoritmide SATO ja DPLLO poolt väljastatava ortogonaalse disjunktiivse normaalkuju termide arvuga. Kuna DF_{M_l} on minimaalne, siis algoritmi #DPLL sammude arv on vähemalt 2^l. □

Teoreem 8.8. *Valemi CF_{M_l} ortogonaliseerimisel tekib $2^l - 1$ disjunkti.*

Tõestus. Tõestame induktsiooniga valemi CF_{M_l} disjunktide arvu l järgi.

Baas: $l = 1$ on ilmne.

Samm: Induktsiooni hüpoteesi kohaselt koosneb CNF $orthogonalize(CF_{M_{l-1}})$ $2^{l-2}-1$ disjunktist. Lisame valemile $CF_{M_{l-1}}$ disjunkti $\overline{x}_{1l} \vee \overline{x}_{2l}$ Kuna $CF_{M_{l-1}}$ ei sisalda literaale $\overline{x}_{1l}, x_{2l}$, siis kahekordistab $orthogonalize(CF_{M_l})$ valemi $orthogonalize(CF_{M_{l-1}})$ disjunktide arvu. Tulemuses on $2 \cdot (2^{l-1} - 1) + 1 = 2^l - 1$ disjunkti. □

Teoreem 8.9. *Valemite pere CF_{M_l} lahendite loendamisel elimineerimismeetodil on tööaeg eksponentsiaalne disjunktide arvust l.*

Tõestus. Kuna valemi CF_{M_l} disjunktide muutujad on kõik erinevad, siis iga $\{i_1, \ldots, i_k\} \in \{1, \ldots, l\}$ jaoks

$$\bigcap_{j=1}^{k} gen(\overline{x}_{1i_j} \vee \overline{x}_{2i_j}) \neq \emptyset,$$

kuna disjunktid ei sisalda kontraarset literaalipaari. Seega on elimineerimisvalemis liidetavaid

$$\binom{l}{1} + \binom{l}{2} + \ldots + \binom{l}{l} = 2^l - 1.$$

$\square$

9. Dedekindi arvud

Vaatleme ühte loendamisalgoritmide rakendust kombinatoorikas. Me oleme tegelenud mitmes kontekstis monotoonsete Boole'i funktsioonidega. Nüüd küsime: kui palju on monotoonseid n-muutuja Boole'i funktsioone?

Probleemi püstitas R. Dedekind 1897. aastal ([Ded97]) ning leidis arvud kuni nelja muutuja funktsioonide jaoks. Hoolimata probleemi näilisest lihtsusest ei õnnestunud tuletada valemit $D(n)$ arvutamiseks. Tõestati mitmeid hinnanguid, millest esitame lühikese ajaloolise ülevaate.

- M. Ward, 1946 ([War46]) – $D(n) > 2^{\binom{n}{[n/2]}}$.
- K. Yamamoto, 1953 ([Yam53]) – $D(n)$ on paarisarv, kui n on paarisarv.
- E. Gilbert, 1954 ([Gil54]) – $D(n) < n^{\binom{n}{[n/2]}} + 2$.
- K. Yamamoto, 1954 ([Yam54]) –

$$log_2 D(n) < \binom{n}{[n/2]} \left(1 + \mathcal{O}\left(n^{-1}\right)\right) log_2 \sqrt{\frac{\pi n}{2}}.$$

- V. Korobkov, 1965 ([Kor65]) – $D(n) < 2^{4{,}24\binom{n}{[n/2]}}$.
- G. Hansel, 1966 ([Han66]) – $D(n) < 3^{\binom{n}{[n/2]}}$.
- D. Kleitman, 1969 ([Kle69]) –

$$2^{(1+\alpha_n)\binom{n}{[n/2]}} \leqslant D(n) \leqslant 2^{(1+\beta_n)\binom{n}{[n/2]}},$$

  kus $\alpha_n = ce^{-n/4}$, $\beta_n = c'\left(\log n\right)/n^{1/2}$.

- A. Korshunov, 1981 ([Kor81]) – Kui n on paarisarv ja $(n \to \infty)$:

$$D(n) \sim 2^{\binom{n}{[n/2]}} \cdot \exp\left(\binom{n}{\frac{n}{2}-1} \cdot \left(\frac{1}{2^{n/2}} + \frac{n^2}{2^{n+5}} - \frac{n}{2^{n+4}}\right)\right) \quad ,$$

  Kui n on paaritu:

$$D(n) \sim 2 \cdot 2^{\binom{n}{(n-1)/2}} \exp \left(\binom{n}{(n-3)/2} \cdot \right.$$
$$\cdot \left(\frac{1}{2^{(n+3)/2}} - \frac{n^2}{2^{n+6}} - \frac{n}{2^{n+3}} \right) +$$
$$\left. + \binom{n}{(n-1)/2} \cdot \left(\frac{1}{2^{(n+1)/2}} + \frac{n^2}{2^{n+4}} \right) \right) \quad .$$

- A. Kisielewicz, 1988 ([Kis88]) –

$$D(n) = \sum_{k=1}^{2^{2^n}} \prod_{j=1}^{2^n-1} \prod_{i=0}^{j-1} \left(1 - b_i^k b_j^k \prod_{m=0}^{\log_2 i} \left(1 - b_m^i + b_m^i b_m^j \right) \right) \quad ,$$

kus $b_i^k = \left[k/2^i \right] - 2 \left[k/2^{i+1} \right]$.

Viimane valem on küll täpne, nn. "kinnine valem", kuid lähemal vaatlusel selgub, et see ei sobi arvutamiseks, juba välimine summa sisaldab 2^{2^n} liidetavat. Lihtsad arvutused näitavad, et $D(6)$ arvutamine Kisielewiczi valemi järgi käib tänapäeva arvutitele üle jõu.

Paralleelselt teoreetiliste otsingutega jätkasid mõned entusiastid ka konkreetsete Dedekindi arvude leidmist, alguses käsitsi, hiljem arvutite abil. Arvutuste ajalugu on sama muljetavaldav, kui teoreetiliste uuringute ajalugu.

0	2	R. Dedekind, 1897 ([Ded97])
1	3	R. Dedekind, 1897 ([Ded97])
2	6	R. Dedekind, 1897 ([Ded97])
3	20	R. Dedekind, 1897 ([Ded97])
4	168	R. Dedekind, 1897 ([Ded97])
5	7581	R. Church, 1940 ([Chu40])
6	7828354	M. Ward, 1946 ([War46])
7	2414682040998	R. Church, 1965 ([Chu65])
8	56130437228687557907788	D. Wiedemann, 1991 ([Wie91])

Järgnevas esitame Dedekindi arvude ülesande lausearvutuse valemi lahendite loendamisülesandena. Kõrvalproduktina saame ülaltoodud Kisielewiczi valemile uue, lihtsama tõestuse. Käsitlus on võetud artiklist [TIT01]. Kasutame Boole'i funktsiooni esitust tõetabelina: funktsiooni
$f : \{0,1\}^n \to \{0,1\}$ esitab bittvektor $f_0, f_1, ..., f_{2^n-1}$, kus $f(i) = f_i$ $(0 \leqslant i \leqslant 2^n - 1)$.

Teoreem 9.1. *Boole'i funktsioon* $f : \{0,1\}^n \to \{0,1\}$ *tõetabeliga* $f_0, ..., f_{2^n-1}$ *on monotoonne siis ja ainult siis, kui väärtustus*

$(f_0, ..., f_{2^n-1})$ *rahuldab lausearvutuse valemit*

$$M_n(x_0, .., x_{2^n-1}) = \underset{\substack{\alpha,\beta \in \{0,1\}^n \\ \alpha \prec \beta}}{\&} (\overline{x}_\alpha \vee x_\beta).$$

Tõestus. $\Rightarrow$ Olgu $f : \{0,1\}^n \to \{0,1\}$ tõetabeliga $f_0, ..., f_{2^n-1}$ ning f monotoonne. Oletame vastuväiteliselt, et $M_n(f_0, ..., f_{2^n-1}) = 0$. M_n sisaldab vähemalt ühte disjunkti, mis on väär. Olgu selleks $\overline{x}_i \vee x_j(f_0, ..., f_{2^n-1}) = 0$ ehk $\overline{f}_i \vee f_j = 0$, seega $f(i) = 1$ ja $f(j) = 0$. Valemi M_n definitsiooni järgi $i \prec j$, mis on aga vastuolus.
$\Leftarrow$ Olgu $M_n(f_0, ..., f_{2^n-1}) = 1$. Oletame vastuväiteliselt, et f ei ole monotoonne. Monotoonsuse definitsiooni kohaselt leiduvad bittjadad $\alpha = (\alpha_0, ..., \alpha_n)$ ja $\beta = (\beta_0, ..., \beta_n)$, et $\alpha \prec^+ \beta$ ja $f(\alpha) > f(\beta)$. Transitiivse sulundi definitsiooni kohaselt $\exists \varphi_0, \varphi_1, ..., \varphi_k$, et $\alpha = \varphi_0$ ja $\beta = \varphi_k$ ja $\varphi_i \prec \varphi_{i+1}$ $(0 \leqslant i < k)$. $f(\varphi_0) = 1$ ja $f(\varphi_k) = 0$. Leidub i, et $f(\varphi_i) = 1$ ja $f(\varphi_{i+1}) = 0$ ning seega $\overline{f}_{\varphi_i} \vee f_{\varphi_{i+1}} = 0$. Järelikult peab disjunkt $\overline{x}_{\varphi_i} \vee x_{\varphi_{i+1}} \in M_n$, mis on aga vastuolus. $\square$

Kisielewiczi valemi tuletamiseks peame kirjeldama Dedekindi arve antiahelate kaudu.

Definitsioon. *Antiahel* on hulga $\{0,1\}^n$ selline alamhulk, mille suvalised kaks elementi pole võrreldavad relatsiooni $\prec^+$ abil.

Kui $f : \{0,1\}^n \to \{0,1\}$ on monotoonne Boole'i funktsioon, siis selle minimaalsed ühed (s.t. väärtustused $\alpha \in \{0,1\}^n$ nii et $f(\alpha) = 1$ ja iga $\beta \prec^+ \alpha$ jaoks $f(\beta) = 0$) moodustavad antiahela. Seega on antiahelad ja monotoonsed funktsioonid üksüheses vastavuses ja Dedekindi arve on võimalik leida, loendades antiahelaid.

Teoreem 9.2. *Boole'i funktsioon tõetabeliga* $f_0, .., f_{2^n-1}$ *on antiahel siis ja ainult siis, kui vektor* $f_0, .., f_{2^n-1}$ *rahuldab valemit*

$$A_n(x_0, ..., x_{2^n-1}) = \underset{\substack{\alpha,\beta \in \{0,1\}^n \\ \alpha \prec^+ \beta}}{\&} (\overline{x}_\alpha \vee \overline{x}_\beta).$$

Tõestus. $\Rightarrow$ Olgu f antiahel. Oletame vastuväiteliselt, et $A_n(f_0, ..., f_{2^n-1}) = 0$, siis vähemalt üks disjunkt A_n-s peab olema väärtustusel $f_0, ..., f_{2^n-1}$ väär, olgu selleks $\overline{x}_\alpha \vee \overline{x}_\beta$. Seega peab $f_\alpha = 1$ ja $f_\beta = 1$. A_n definitsiooni kohaselt $\alpha \prec^+ \beta$ ja seega ei saa f olla antiahel.
$\Leftarrow$ Olgu $A_n(f_0, ..., f_{2^n-1}) = 1$, oletame vastuväiteliselt, et f ei ole antiahel, siis leiduvad $\alpha, \beta \in \{0,1\}^n$ nii et $\alpha \prec^+ \beta$ ja $f(\alpha) = 1$ ja

$f(\beta) = 1$. Seega disjunkt $\overline{x}_\alpha \vee \overline{x}_\beta$ on väär ning A_n definitsiooni kohaselt saame vastuolu. $\square$

Teoreem 9.3. *([Kis88]) Iga* $n \geqslant 1$

$$D(n) = \sum_{k=1}^{2^{2^n}} \prod_{j=1}^{2^n-1} \prod_{i=0}^{j-1} (1 - b_i^k b_j^k \prod_{m=0}^{\log_2 i} (1 - b_m^i + b_m^i b_m^j)) \ ,$$

kus $b_i^k = [k/2^i] - 2[k/2^{i+1}]$.

Tõestus. Viime lausearvutuse valemi $A(x_1, ..., x_n)$ aritmeetilisele kujule, kasutades selleks kahte sammu:

1) Teisendame lausearvutusvalemi ekvivalentseks lausearvutusvalemiks, mis sisaldab ainult binaarseid tehteid $\&$, $\supset$ ja $\neg$.

2) Teisendame lausearvutuse valemid aritmeetilisteks (lausearvutuse valemi F aritmetiseeritud vastet tähistame $[F]$).

$[B\&C] \ \Rightarrow \ [B]\cdot[C], [B \supset C] \ \Rightarrow \ 1-[B]+[B]\cdot[C], [\neg B] \ \Rightarrow \ 1-[B]$, kus B ja C on lausearvutusvalemid ning $[B]$ ja $[C]$ aritmeetikavalemid.

Kasutame teoreemi 9.2 valemit:

$$A_n(x_0, ..., x_{2^n-1}) \equiv \mathop{\&}_{\alpha,\beta \in \{0,1\}^n; \alpha \prec^+ \beta} (\overline{x}_\alpha \vee \overline{x}_\beta) \equiv$$

$$\mathop{\&}_{\alpha \in \{0,1\}^n} \mathop{\&}_{\beta \in \{0,1\}^n} (\alpha \prec^+ \beta) \supset (\overline{x}_\alpha \vee \overline{x}_\beta) \ .$$

Kasutades omadust, et $\alpha \ \prec^+ \ \beta \ \equiv \ \&_{m=1}^n (\alpha_m \ \leqslant \ \beta_m)$ ja iga $x, y \in \{0, 1\}$, $x \leqslant y \equiv x \supset y$ saame

$$A_n(x_0, ..., x_{2^n-1}) \equiv \mathop{\&}_{\alpha \in \{0,1\}^n} \mathop{\&}_{\beta \in \{0,1\}^n} (\mathop{\&}_{m=1}^{n} (\alpha_m \supset \beta_m)) \supset (\overline{x}_\alpha \vee \overline{x}_\beta) \ .$$

Kasutades teisendust $x \supset (\overline{y} \vee \overline{z}) \equiv \neg(x \ \& \ y \ \& \ z)$, saame

$$A_n(x_0, ..., x_{2^n-1}) \equiv \mathop{\&}_{\alpha \in \{0,1\}^n} \mathop{\&}_{\beta \in \{0,1\}^n} \neg(x_\alpha \ \& \ x_\beta \& (\mathop{\&}_{m=1}^{n} (\alpha_m \supset \beta_m))) \ .$$

Peale aritmetiseerimist saame

$$[A_n(x_0, ..., x_{2^n-1})] = \prod_{\beta \in \{0,1\}^n} \prod_{\alpha \in \{0,1\}^n} (1 - x_\alpha x_\beta \prod_{m=1}^{n} (1 - \alpha_m + \alpha_m \beta_m)).$$

Olgu täisarv k, mille kahendesitus on
$k_2 = b_l^k b_{l-1}^k ... b_1^k b_0^k$, siis
$k = b_l^k 2^l + b_{l-1}^k 2^{l-1} + ... + b_{i+1}^k 2^{i+1} + b_i^k 2^i + ... + b_0^k 2^0$,
$\left[\frac{k}{2^i}\right] = b_l^k 2^{l-i} + b_{l-1}^k 2^{l-i-1} + ... + b_{i+1}^k 2^1 + b_i^k 2^0$,
$\left[\frac{k}{2^{i+1}}\right] = b_l^k 2^{l-i-1} + b_{l-1}^k 2^{l-i-2} + ... + b_{i+1}^k 2^0$,
$2\left[\frac{k}{2^{i+1}}\right] = b_l^k 2^{l-i} + b_{l-1}^k 2^{l-i-1} + ... + b_{i+1}^k 2^1$,
$b_i^k = \left[\frac{k}{2^i}\right] - 2\left[\frac{k}{2^{i+1}}\right]$. Nüüd saame kasutada bittvektorite asemel naturaalarve:

$$[A_n(x_0, ..., x_{2^n-1})] = \prod_{j=0}^{2^n-1} \prod_{i=0}^{2^n-1} (1 - x_i x_j \prod_{m=0}^{n} (1 - b_m^i + b_m^i b_m^j)) \ .$$

Iga $i, j (0 \leqslant i, j \leqslant 2^n)$, $b_n^i ... b_1^i b_0^i \prec^+ b_n^j ... b_1^j b_0^j$ viitab, et $i \leqslant j$, seega piisab kui arvutada korrutis $\prod_{m=0}^{n}(1 - b_m^i + b_m^i b_m^j)$ ainult $m \leqslant \log_2 i$ jaoks:

$$[A_n(x_0, ..., x_{2^n-1})] = \prod_{j=1}^{2^n-1} \prod_{i=0}^{j-1} (1 - x_i x_j \prod_{m=0}^{\log_2 i} (1 - b_m^i + b_m^i b_m^j)) \ .$$

Bittvektor $f = f_0 ... f_{2^n-1}$ on mingi antiahela tõetabel siis ja ainult siis, kui $[A_n(f_0, ..., f_{2^n-1})] = 1$. Seega arvutades $D(n)$, peame liitma need väärtused kõigi naturaalarvude k korral 0 kuni $2^{2^n} - 1$ ehk

$$D(n) = \sum_{k=0}^{2^{2^n}-1} \prod_{j=1}^{2^n-1} \prod_{i=0}^{j-1} (1 - b_i^k b_j^k \prod_{m=0}^{\log_2 i} (1 - b_m^i + b_m^i b_m^j)) \ .$$

$D(n)$ väärtus ei muutu, kui me võtame summa $\sum_{k=1}^{2^{2^n}}$:

$$D(n) = \sum_{k=1}^{2^{2^n}} \prod_{j=1}^{2^n-1} \prod_{i=0}^{j-1} (1 - b_i^k b_j^k \prod_{m=0}^{\log_2 i} (1 - b_m^i + b_m^i b_m^j)) \ .$$

$\square$

Esitatud tõestus Kisielewiczi valemile on mõnevõrra läbipaistvam kui originaaltõestus. Resümeerides: esitatud tuletuskäik koosnes kolmest etapist. Kõigepealt me kirjeldasime monotoonse Boole'i funktsiooni mõistet lausearvutuse valemi abil. Teiseks aritmetiseerisime saadud lausearvutuse valemi. Kolmandaks summeerisime üle kõikvõimalike väärtustuste, s.t. rakendasime lahendite loendamiseks tõetabeli ammendavat läbivaatust. Tulemuseks on valem, mis on küll formaalselt korrektne, kuid arvutamiseks kõlbmatu.

Ilmselt on mõistlik loendamisülesannete kirjeldamisel loogika abil piirduda lausearvutuse valemiga. Algoritm #DPLL arvutab ilma eriliste raskusteta $D(7)$, kusjuures aritmetiseeritud valem on samaväärne tõetabeli läbivaatamisega ja ei võimalda enamat arvust $D(5)$.

10. Polünomiaalne taandamine

Terminoloogia ja tähistuste täpsustamiseks esitame algoritmiteooriast tuttava Turingi masina definitsiooni.

Turingi masina käsk. Ühelindilise Turingi masina käsk on defineeritud kui kolmik $q_i a_j d$, kus

$$q_i \in Q \text{ (olekute hulk)},$$

$$a_j \in \Sigma \cup \{\Box\} \text{ (tähestik)},$$

$$d \in \{-1, 0, +1\} \text{ (pea liikumine)}.$$

m-lindilise Turingi masina korral on käsk defineeritud kui

$$q_{1,i} a_{1,j} d_1, q_{2,i} a_{2,j} d_2, ..., q_{m,i} a_{m,j} d_m.$$

m-lindilise Turingi masina puhul eristatakse sageli ühesuunalisi sisend- ja väljundlinte, $m-2$ linti on seejuures töölindid. Eristame kahte tüüpi Turingi masinaid: testmasin, millel on kaks lõppolekut q_a – *accept, aktsepteerida*, q_r – *reject, tagasi lükata*), ning transduktor e. teisendav Turingi masin, millel on üks lõppolek q_0.

Mittedeterministlik Turingi masin. Mittedeterministliku Turingi masina korral on lubatud ühe ja sama aadressosaga mitu erinevat käsku. Turingi masina programmi tabelesituse puhul tähendab see, et tabeli lahtris võib olla mitu käsku, millest täidetakse mittedeterministlikult üks.

Näide. Konstrueerime ühelindilise Turingi masina *bitstring*, mis saab töölindil ette naturaalarvu n ühendsüsteemis (n sümbolit 1) ning väljastab mittedetermineeritult väärtustuse $\alpha \in \{0,1\}^n$.

bitstring	$\Box$	0	1
q_1	$q_2\Box-1$	—	$q_1 0+1$ $q_1 1+1$
q_2	$q_0\Box+1$	$q_2 0-1$	$q_2 1-1$

Konfiguratsioon. Turingi masina konfiguratsioon kajastab selle lintide seisu mingil ajahetkel. Ühelindilise Turingi masina puhul esitame konfiguratsiooni kujul $\Box a_1 a_2 \ldots a_{i-1} q_j a_i \ldots a_n \Box$. Tõlgendame seda kirjutist nii, et lindil on alates pesast number 1 sümbolid $a_1, \ldots, a_n$, lindi pesad alates pesast $n+1$ on tühjad ning lugemispea on olekus q_j ja vaatleb pesa i.

Arvutustee on konfiguratsioonide jada, mis vastab Turingi masina järjestikustele sammudele. Mittedeterministliku Turingi masina puhul on tegemist arvutuspuuga, kuna mittedeterministliku käsu täitmisel tekib mitu järgnevat konfiguratsiooni. Turingi masina tööaeg sõna $x \in \Sigma^*$ jaoks on arvutustee pikkus alates algkonfiguratsioonist $\Box q_1 x \Box$ kuni lõppolekule vastava konfiguratsioonini. Mittedeterministliku Turingi masina tööaeg on arvutuspuu sügavus.

Turingi masina aja- ja mälukeerukus. Olgu $f(n)$ mingi funktsioon. Me ütleme, et Turingi masina T ajakeerukus on f, kui suvalise sõna $x \in \Sigma^*$ jaoks $T(x)$ tööaeg on väiksem või võrdne, kui $f(|x|)$. Turingi masina T mälukeerukus on f, kui maksimaalne konfiguratsiooni pikkus $T(x)$ arvutusteel on väiksem või võrdne kui $f(|x|)$. Suvalise Turingi masina mälukeerukus ei saa olla suurem ajakeerukusest, kuna iga sammuga saab lugemispea liikuda algasendist ülimalt ühe sammu paremale.

Keel. Olgu $\Sigma = \{a_1, \ldots, a_n\}$ lõplik tähestik. Suvalist tähestiku Σ sõnade hulka $A \subseteq \Sigma^*$, nimetame keeleks. Ütleme, et deterministlik Turingi masin M tuvastab keele A (tähistame $L(M) = A$), kui iga sisendi $x \in \Sigma^*$ korral $M(x)$ arvutustee lõpeb olekus q_a siis ja ainult siis, kui $x \in A$.

Mittedeterministliku Turingi testmasina rakendamisel sõnale x kirjeldab rakendamise protsessi arvutuspuu, mille erinevad teed võivad lõppeda erinevates lõppolekutes, s.t. osa lõppkonfiguratsioone võivad olla aktsepteerivad, osa tagasilükkavad. Sõna x loetakse aktsepteerituks mittedeterministliku Turingi masina M poolt, kui $M(x)$ arvutuspuu lehtede hulgas leidub aktsepteeriv konfiguratsioon.

Ütleme, et mittedeterministlik Turingi masin M tuvastab keele A (tähistame $L(M) = A$), kui iga sisendi $x \in \Sigma^*$ korral $M(x)$ arvutusteede hulgas leidub tee, mis lõpeb olekus q_a, siis ja ainult siis, kui $x \in A$.

Keerukusklass P. Keel A kuulub keerukusklassi **P**, kui leidub polünoom $p(x)$ ja deterministlik Turingi (test)masin M ajakeerukusega $p(x)$, mis tuvastab keele A.

Suvalise polünoomi $p(x)$ jaoks leiduvad konstandid c ja k, nii et $p(x) = O(cx^k)$. Seega on keerukusklass **P** tegelikult tõkestatud astmefunktsiooniga, termin *polünomiaalne keerukus* on kasutusel ajaloolistel põhjustel. Kuigi keerukusklass **P** on defineeritud Turingi masina terminites, on võimalik näidata, et otseadresseeritava mäluga imperatiivset programmeerimiskeelt saab modelleerida mitmelindisel Turingi masinal nii, et tööaeg suureneb ülimalt polünomiaalselt sisendi pikkusest. Samuti saab modelleerida mitmelindilist Turingi masinat ühelindisel samade kadudega. Tehnilised detailid ei mahu käesoleva kursuse raamesse, nende probleemide korrektse käsitluse võib leida C. K. Yapi monograafiast [Yap87].

Näide. Olgu antud lausearvutuse valem $F(x_1, ..., x_n)$ ja väärtustus $\alpha \in \{0,1\}^n$. Kas $F(\alpha) = 1$?

Sõnastame kõigepealt selle arvutusliku probleemi keele tuvastamise probleemina. Tähestik $\Sigma = \{\&, \vee, \neg, x, 0, 1, (,), [,], ;\}$. Kõigepealt defineerime korrektse sisendi kontekstivaba grammatika G abil. $G = (N, \Sigma, P, < sisend >)$, kus $N = \{< sisend >, < valem >,$ $< bittjada >, < lause >, < term >, < muutuja >, < indeks >\}$.

$$P = \left\{ \begin{array}{lll} < sisend > & \longrightarrow & < valem >; < bittjada > \\ < valem > & \longrightarrow & < valem > \vee < lause > \\ < valem > & \longrightarrow & < lause > \\ < lause > & \longrightarrow & < term > \& < lause > \\ < lause > & \longrightarrow & < term > \\ < term > & \longrightarrow & < muutuja > \\ < term > & \longrightarrow & \neg < muutuja > \\ < term > & \longrightarrow & (< valem >) \\ < muutuja > & \longrightarrow & x[< indeks >] \\ < indeks > & \longrightarrow & 1 \\ < indeks > & \longrightarrow & < indeks > 0 \\ < indeks > & \longrightarrow & < indeks > 1 \\ < bittjada > & \longrightarrow & 0 \\ < bittjada > & \longrightarrow & 1 \\ < bittjada > & \longrightarrow & < bittjada > 0 \\ < bittjada > & \longrightarrow & < bittjada > 1 \end{array} \right\}$$

Süntaktiliselt on korrektne sisend paar $< valem >; < bittjada >$ kus $< valem >$ on lausearvutuse valem, mille muutujad on tähistatud tähega x, millele järgneb nurksulgudes muutuja järjekorranumber kahendsüsteemis. $< bittjada >$ esitab väärtustuse. Kasutades süntaksi analüüsi teooriast tuntud meetodeid on lihtne veenduda, et esitatud grammatika kuulub klassi $SLR(k)$ ning suvalise sõna süntaktiline korrektsus on seetõttu kontrollitav lineaarse tööajaga sisendsõna pikkusest. Mõistlik on nõuda, et valem ja väärtustus oleksid kooskõlas, s.t. väärtustuse bittide (tõeväärtuste) arv peab olema võrdne valemi muutujate maksimaalse järjekorranumbriga. See ei ole kontrollitav süntaksi analüüsi meetoditega; kirjutame lihtsa programmi, mille tööaeg on samuti lineaarne.

Nüüd võib defineerida meid huvitava keele: $BOOLEANVALUE$ on nende süntaktiliselt ja semantiliselt korrektsete paaride $< valem >; < bittjada >$ hulk, milles valem on antud väärtustuse jaoks tõene. Keelde kuuluvuse testist on kirjeldamata valemi väärtuse kontroll. Selleks rakendame varemõpitut: eeldame, et süntaksi analüsaator koostab avaldise puu ja programmeerime selle interpretaatori. Ka interpretaatori tööaeg on lineaarne valemi pikkusest. Modelleerides esitatud algoritme Turingi masinal, saame polünomiaalses ajas töötava testmasina meie keele liikmelisuse kontrolliks.

Keerukusklass NP. Keel A kuulub keerukusklassi **NP** kui leidub polünoom $p(x)$ ja mittedeterministlik Turingi (test)masin M ajakeerukusega $p(x)$, mis tuvastab keele A.

Näide. Olgu antud lausearvutuse valem $F(x_1, ..., x_n)$. Kas F on kehtestatav?

Keel SAT , mis vastab vaadeldavale probleemile on esitatav eelmises näites antud grammatika abil, milles aksioomiks on $< valem >$. Keelt SAT tuvastavat mittedeterministlikku algoritmi saab kirjeldada järgmiselt.

S1. Kontrollida, kas x on valem.

S2. Leida valemi muutujate maksimaalne indeks n.

S3. Lisada valemile semikooloni järele n sümbolit ”1”.

S4. Genereerida mittedeterministlikult väärtustus pikkusega n, kasutades Turingi masinat *bitstring* üle-eelmisest näitest.

S5. Arvutada valemi tõeväärtus genereeritud väärtustuse jaoks.

Sammud S1, S2, S3 ja S5 on teostatavad polünomiaalse ajaga. Analüüsides mittedeterministlikku Turingi masinat *bitstring* näeme, et selle kõik arvutusteed on ühepikkused, sisaldades $2n + 3$ konfiguratsiooni. Järelikult kuulub keel SAT keerukusklassi **NP**.

Abstraheerudes kõigest väheolulisest saame veelgi lakoonilisema mittedeterministliku tuvastamisalgoritmi:

input$F$: lausearvutuse valem.
guess väärtustus α
check if $F(\alpha) = 1$

Juhime tähelepanu sellele, et esitatud algoritm eeldab, et sisendiks on süntaktiliselt korrektne lausearvutuse valem. Sellisel viisil saab abstraheeruda süntaktilistest detailidest, mis on keerukusteooria seisukohast ebaolulised. Seetõttu defineeritakse keele A täiend, A^c kui nende Σ^* sõnade hulk, mis on süntaktiliselt korrektsed, kuid ei rahulda keelde A kuulumise tingimust.

Keerukusklass PSPACE. Keel A kuulub keerukusklassi **PSPACE**, kui leidub polünoom $p(x)$ ja deterministlik Turingi masin M mälukeerukusega $p(x)$, mis tuvastab keele A.

Definitsioon. Keel A_1 on polünomiaalselt (mitu-üks) taandatav keelele A_2 (tähistatame $A_1 \leqslant_m^P A_2$), kui leidub polünomiaalses ajas töötav deterministlik Turingi masin $T: \Sigma^* \to \Sigma^*$, nii et iga $x \in \Sigma^*$ korral

$$x \in A_1 \Leftrightarrow T(x) \in A_2.$$

Teoreem 10.1. *Polünomiaalse taandamise omadused.*

*1. Kui $A_1 \leqslant_m^P A_2$ ja $A_2 \in$**P**, siis $A_1 \in$**P** (**P** on kinnine $\leqslant_m^P$ suhtes).*

*2. Kui $A_1 \leqslant_m^P A_2$ ja $A_2 \in$**NP**, siis $A_1 \in$**NP**. (**NP** on kinnine $\leqslant_m^P$ suhtes).*

3. Kui $A_1 \leqslant_m^P A_2$ ja $A_2 \leqslant_m^P A_3$, siis $A_1 \leqslant_m^P A_3$. ($\leqslant_m^P$ on transitiivne).

4. Kui $A_1 \leqslant_m^P A_2$, siis $A_1^c \leqslant_m^P A_2^c$.

Tõestus. 1. Olgu T Turingi masin, mis realiseerib polünomiaalse taandamise $A_1 \leqslant_m^P A_2$ ning M Turingi masin, mis testib keelt A_2. Olgu $p(n)$ ja $q(n)$ polünoomid, mis tõkestavad vastavalt T ja M sammude arve. Keelt A_1 aktsepteeriv Turingi masin on nende kompositsioon $T \circ M$, mille tööaeg on sõna $x \in \Sigma^*$ testimisel tõkestatud polünoomiga $p(|x|) + q(|T(x)|) = p(|x|) + q(p(|x|))$.

2. Analoogiliselt punktile 1.

3. Kui Turingi masin S tööajaga $p(n)$ realiseerib taandamise $A_1 \leqslant_m^P A_2$ ning T tööajaga $q(n)$ realiseerib taandamise $A_2 \leqslant_m^P A_3$, siis nende kompositsioon $S \circ T$ tööajaga $p(n) + q(p(n))$ realiseerib taandamise $A_1 \leqslant_m^P A_3$.

4. Realiseerigu Turingi masin T taandamise $A_1 \leqslant_m^P A_2$. Kui $x \in A_1^c$, siis $x \notin A_1$. Polünomiaalse taandamise definitsioonist järeldub, et $T(x) \notin A_2$, seega $T(x) \in A_2^c$. Kui $x \notin A_1^c$, siis $x \in A_1$ ja polünomiaalse taandamise definitsiooni põhjal $T(x) \in A_2$, seega $T(x) \notin A_2^c$. $\square$

Definitsioon. L_1 ja L_2 on polünomiaalselt ekvivalentsed (tähistame $L_1 \equiv_m^P L_2$), kui $L_1 \leqslant_m^P L_2$ ja $L_2 \leqslant_m^P L_1$.

Intuitiivselt võiks tõlgendada käesolevas peatükis defineeritud mõisteid järgmiselt:

- kui $L_1 \leqslant_m^P L_2$, siis keel L_1 ei ole raskemini tuvastatav (polünomiaalse tööaja täpsuseni) kui keel L_2,

- kui $L_1 \equiv_m^P L_2$, siis keeled on tuvastamiskeerukuselt samaväärsed (jällegi polünomiaalse tööaja täpsuseni).

11. Kehtestatavuse probleemid

Esimeste näidetena polünomiaalsest taandamisest vaatleme probleemi SAT mõningaid erijuhte.Varasemast on tuttavad probleemid SAT ja $CNF - SAT$.

$kCNF - SAT$
Antud: konjunktiivsel normaalkujul valem F, milles iga disjunkti pikkus (literaalide arv disjunktis) on võrdne konstandiga k.
Omadus: kas F on kehtestatav?

Ilmne on, et iga $k > 0$ jaoks $kCNF - SAT \leqslant_m^P CNF - SAT$, kuna valem kujul kCNF on erijuht valemist kujul CNF. Samal põhjusel kehtib $CNF - SAT \leqslant_m^P SAT$. Polünomiaalne taandamine vastupidises suunas ei ole aga sugugi ilmne.

Teoreem 11.1. *$CNF - SAT \leqslant_m^p 3CNF - SAT$.*

Tõestus. Olgu $F \in$CNF muutujatega $X = \{x_1, ..., x_n\}$,

$$F = \mathop{\&}_{i=1}^{p} D_i,$$

kus D_i on disjunktid, milles $0 < |D_i| \leqslant n$. Olgu D suvaline disjunkt valemis F, mis sisaldab m literaali. Teisendame disjunkti D 3CNF-ks $c(D)$ nii et

$$F = \mathop{\&}_{i=1}^{p} c(D_i)$$

on kehtestatav siis ja ainult siis, kui F on kehtestatav.

1. $m = 3$. Disjunkt D on juba nõutaval kujul ja $c(D) = D$.

2. $m = 2$. Disjunkt D on kujul $l_1 \vee l_2$, kus l_1 ja l_2 on literaalid. Võtame $c(D) = (l_1 \vee l_2 \vee u) \,\&\, (l_1 \vee l_2 \vee \overline{u})$. Lausearvutusest on teada, et valemid D ja $c(D)$ on ekvivalentsed.

3. $m = 1$. Disjunkt D koosneb ühest literaalist l. Võtame $c(D) = (l \vee u \vee v) \,\&\, (l \vee u \vee \overline{v}) \,\&\, (l \vee \overline{u} \vee v) \,\&\, (l \vee \overline{u} \vee \overline{v})$. Rakendades kaks korda eelmise punkti ekvivalentsiseost näeme, et D ja $c(D)$ on ekvivalentsed.

4. $m > 3$. Võtame $c(D) = (l_1 \vee l_2 \vee u_1) \,\&\, (\overline{u}_1 \vee l_3 \vee u_2) \,\&\, ...$
$\&(\overline{u}_{m-4} \vee l_{m-2} \vee u_{m-3}) \,\&\, (\overline{u}_{m-3} \vee l_{m-1} \vee l_m)$, kus $u_1, \ldots, u_{m-3}$ on uued muutujad. Näitame, et D ja $c(D)$ on samaaegselt kehtestatavad.

Olgu D kehtestatav ja $\alpha = (\alpha_1, ..., \alpha_m)$ väärtustus, mis muudab D tõeseks. Siis leidub disjunktis D literaal, olgu see l_i, nii et $l_i(\alpha) = 1$. Laiendame väärtustust muutujatele $u_1, \ldots, u_{m-3}$, võttes muutujate $u_1, \ldots, u_{i-2}$ väärtuseks 1 ja muutujate $u_{i-1}, \ldots, u_{m-3}$ väärtuseks 0. Sellisel viisil laiendatud väärtustuse jaoks on $c(D)$ tõene.

Olgu $c(D)$ kehtestatav, s.t. $c(D)$ on tõene mingi väärtustuse $(\alpha_1, ..., \alpha_m, \beta_1, ..., \beta_{m-3})$ jaoks. Piisab, kui näidata, et vähemalt üks l_i peab olema tõene antud väärtustuse jaoks. Oletame vastuväiteliselt, et $[l_1 \vee ... \vee l_m](\alpha_1, ..., \alpha_m) = 0$ Siis $u_1(\beta_1) = 1$ ehk $u_1(\beta) = 1$, $u_2(\beta) = 1$,..., $u_{m-3}(\beta) = 1$, siis on aga $c(D)$ viimane disjunkt $(\overline{u}_{m-3} \vee l_{m-1} \vee l_m)$ väär. $\square$

Meetodit saab üldistada iga $k > 2$ jaoks, näitamaks, et $CNF - SAT \leqslant_m^P kCNF - SAT$. $k = 2$ puhul tõestus läbi ei lähe. Enamgi veel, osutub, et varasemas toodud algoritm $DPLL$ töötab 2CNF valemitel determineeritud polünomiaalses ajas, seega kehtib $2CNF - SAT \in \mathbf{P}$. Keele $CNF - SAT$ kohta on meil parim tulemus $CNF - SAT \in \mathbf{NP}$.

Selleks, et konstrueerida polünomiaalset taandamist $SAT \leqslant_m^P CNF - SAT$, ei sobi ükski tuntud ega ka veel tundmatu algoritm, mis teisendab valemeid normaalkujule, kuna leidub lausearvutuse valemite pere, mille liikmete minimaalne CNF on eksponentsiaalse pikkusega.

Tähistame

$$\begin{aligned} \oplus(x_1, \ldots, x_n) &= x_1 \oplus x_2 \oplus \ldots \oplus x_n \\ \overline{\oplus}(x_1, \ldots, x_n) &= \neg \oplus (x_1, \ldots, x_n). \end{aligned}$$

Lihtne on veenduda, et $\oplus(\alpha_1, \ldots, \alpha_n) = 1$ parajasti siis, kui vektori α Hammingi kaal on paarituarv ning $\overline{\oplus}(\alpha_1, \ldots, \alpha_n) = 1$ vastupidisel juhul.

Teoreem 11.2.

1) *Minimaalne konjunktiivne normaalkuju valemile* $\oplus(x_1, ..., x_n)$ *on*

$$\underset{\substack{\alpha \in \{0,1\}^n, \\ n - \|\alpha\| : \text{ paaris}}}{\&} (x_1^{\alpha_1} \vee ... \vee x_n^{\alpha_n}).$$

2) *Minimaalne konjunktiivne normaalkuju valemile* $\overline{\oplus}(x_1, ..., x_n)$ *on*

$$\underset{\substack{\alpha \in \{0,1\}^n, \\ n - \|\alpha\| : \text{ paaritu}}}{\&} (x_1^{\alpha_1} \vee ... \vee x_n^{\alpha_n}).$$

Tõestus. Tõestame induktsiooniga n järgi.
Baas: n=2

1) $\oplus(x_1, x_2) \equiv x_1 \oplus x_2 \equiv (x_1 \vee x_2) \,\&\, (\overline{x}_1 \vee \overline{x}_2)$.

2) $\overline{\oplus}(x_1, x_2) \equiv x_1 \sim x_2 \equiv (x_1 \vee \overline{x}_2) \mathbin{\&} (\overline{x}_1 \vee x_2)$. Mõlemad juhud on lausearvutusest tuntud samasused.

Samm:

1) Vaatleme valemit

$$\oplus(x_1, ..., x_n) \equiv \oplus(x_1, ..., x_{n-1}) \oplus x_n \equiv$$

$$(\oplus(x_1, ..., x_{n-1}) \vee x_n) \mathbin{\&} (\overline{\oplus}(x_1, ..., x_{n-1}) \vee \overline{x}_n) \equiv$$

Rakendame induktsiooni hüpoteesi:

$$\equiv \left(\left(\mathop{\&}_{\substack{\alpha \in \{0,1\}^{n-1}, \\ n-1-\|\alpha\|:\text{paaris}}} (x_1^{\alpha_1} \vee ... \vee x_{n-1}^{\alpha_{n-1}}) \right) \vee x_n \right) \mathbin{\&}$$

$$\mathbin{\&} \left(\left(\mathop{\&}_{\substack{\alpha \in \{0,1\}^{n-1}, \\ n-1-\|\alpha\|:\text{paaritu}}} (x_1^{\alpha_1} \vee ... \vee x_{n-1}^{\alpha_{n-1}}) \right) \vee \overline{x}_n \right) \equiv$$

$$\mathop{\&}_{\substack{\alpha \in \{0,1\}^{n-1}, \\ n-\|\alpha\|:\text{paaris}}} (x_1^{\alpha_1} \vee ... \vee x_{n-1}^{\alpha_{n-1}} \vee x_n^1) \mathbin{\&} \mathop{\&}_{\substack{\alpha \in \{0,1\}^{n-1}, \\ n-\|\alpha\|:\text{paaritu}}} (x_1^{\alpha_1} \vee ... \vee x_{n-1}^{\alpha_{n-1}} \vee x_n^0)$$

Siit on näha, et esimeses pooles on paaris ning teises paaritu arv nulle. Ühe lisamisel esimese poole astendajate vektorile nullide arv ei muutu, nulli lisamine teise poolde muudab ka seal nullide arvu paarisarvuks. Haaratud on kõikvõimalikud vektorid pikkusega n, milles on paarisarv nulle. seega

$$\oplus(x_1, ..., x_n) \equiv \mathop{\&}_{\substack{n-\|\alpha\| \\ :\text{paaris}}} (x_1^{\alpha_1} \vee ... \vee x_n^{\alpha_n})$$

2) Analoogiliselt.

Näitame, et ülaltoodud konjunktiivsed normaalkujud on minimaalsed. Tõestame väite funktsiooni $\oplus(x_1, ..., x_n)$ jaoks, funktsiooni $\overline{\oplus}(x_1, ..., x_n)$ tõestus on analoogiline.

Iga disjunkt sisaldab n muutujat. Oletame üldisust kitsendamata, et mingis disjunktis puudub viimane muutuja ehk disjunkt kujul $(x_1^{\alpha_1} \vee ... \vee x_{n-1}^{\alpha_{n-1}})$ kuulub funktsiooni $\oplus(x_1, ..., x_n)$ konjunktiivse normaalkuju disjunktide hulka. See keelab ära väärtustused $(\overline{\alpha}_1, ..., \overline{\alpha}_{n-1}, 1)$ ja $(\overline{\alpha}_1, ..., \overline{\alpha}_{n-1}, 0)$. Üks nendest peab aga sisaldama paarituarvu ühtesid ja peab olema lubatud. Valemist ei saa ka ühtegi disjunkti eemaldada, kuna kõik on täielikud disjunktid. Seega, kui suvaline disjunkt eemaldada, saame vale tulemuse. $\square$

Järeldus 11.3.

1) *Minimaalne disjunktiivne normaalkuju valemile* $\oplus(x_1,...,x_n)$ *on*

$$\bigvee_{\substack{\alpha\in\{0,1\}^n,\\ \|\alpha\|:paaritu}} (x_1^{\alpha_1}\,\&\,...\,\&\,x_n^{\alpha_n}).$$

2) *Minimaalne disjunktiivne normaalkuju valemile* $\overline{\oplus}(x_1,...,x_n)$ *on*

$$\bigvee_{\substack{\alpha\in\{0,1\}^n,\\ \|\alpha\|:paaris}} (x_1^{\alpha_1}\,\&\,...\,\&\,x_n^{\alpha_n}).$$

Tõestus. Rakendame teoreemi 11.2 valemitele DeMorgani reegleid:

$$\oplus(x_1,...,x_n)\equiv\neg\neg\oplus(x_1,...,x_n)\equiv\neg\overline{\oplus}(x_1,...,x_n)\equiv$$

$$\neg\mathop{\&}_{\substack{\alpha\in\{0,1\}^n,\\ n-\|\alpha\|:\text{paaritu}}}(x_1^{\alpha_1}\vee...\vee x_n^{\alpha_n})\equiv\bigvee_{\substack{\alpha\in\{0,1\}^n,\\ n-\|\alpha\|:\text{paaritu}}}(\overline{x_1^{\alpha_1}}\,\&\,...\,\&\,\overline{x_n^{\alpha_n}})\equiv$$

$$\bigvee_{\substack{\alpha\in\{0,1\}^n,\\ n-\|\alpha\|:\text{paaritu}}}(x_1^{\overline{\alpha}_1}\,\&\,...\,\&\,x_n^{\overline{\alpha}_n})\equiv\bigvee_{\substack{\beta\in\{0,1\}^n,\\ \|\beta\|:\text{paaritu}}}(x_1^{\beta_1}\,\&\,...\,\&\,x_n^{\beta_n})$$

Funktsiooni $\overline{\oplus}(x_1,...,x_n)$ disjunktiivse normaalkuju tõestus on analoogiline. □

Järeldus 11.4. *Valemite* $\oplus(x_1,...,x_n)$ *ja* $\overline{\oplus}(x_1,...,x_n)$ *minimaalne CNF ja minimaalne DNF sisaldavad* $n\cdot 2^{n-1}$ *literaali.*

Teoreem 11.5. $SAT \leqslant_m^P$ *CNF* $- SAT$.

Tõestus. Olgu $F(x_1,...,x_n)$ lausearvutuse valem. Seame igale alamvalemile F' vastavusse lausemuutuja $u_{F'}$ järgmiselt:

1) Kui $F'=x$, siis $u_{F'}=x$.

2) Kui F' on mittetriviaalne alamvalem, siis võtame muutujaks $u_{F'}$ uue muutuja, mis erineb muutujatest $x_1,...,x_n$ ja valemi F teiste alamvalemite muutujatest.

Iga F alamvalemi F' jaoks defineerime konjunktiivsel normaalkujul valemi $C_{F'}$ järgmiselt:

1) Kui $F' = x$, siis $C_{F'} = 1$.

2) Kui $F' = \neg F''$, siis $C_{F'}$ on valemi $(u_{F'} \sim \neg u_{F''})$ konjunktiivne normaalkuju.

3) Kui $F' = F_1$ **op** F_2, kus **op** on binaarne tehe, siis $C_{F'}$ on valemi $(u_{F'} \sim (u_{F_1} \textbf{ op } u_{F_2}))$ konjunktiivne normaalkuju.

Olgu
$$K_F = u_F \;\&\; (\mathop{\&}_{F'-alamvalem} C_{F'}).$$

Iga lausearvutuse valemi F jaoks on K_F ja F samaaegselt kehtestatavad. Tõepoolest, kui $\alpha = (\alpha_1, ..., \alpha_n)$ on valemi F tõene väärtustus, siis võtame uute muutujate väärtusteks vastavate alamvalemite väärtused väärtustusel α. Vastavalt K_F konstruktsioonile on sellise väärtustuse jaoks kõik disjunktid tõesed ning ka muutuja u_F on tõene. Kui aga meil on kehtestav väärtustus valemi K_F jaoks, siis peab muutuja u_F olema tõene (ühikdisjunkt) ja ülejäänud muutujate väärtused olema tõese valemi F alamvalemite tõeväärtused kuni muutujate $x_1, \ldots, x_n$ väärtusteni välja. Viimased moodustavadki tõese väärtustuse valemile F.

Valem kujul $u_{F'} \sim (u_{F_1} \textbf{ op } u_{F_2})$ sisaldab kolme lausemuutujat ja selle konjunktiivne normaalkuju sisaldab mitte rohkem kui 8 disjunkti, igaühes kolm literaali. Seega on $|K_F| \leqslant 24 \cdot |F|$. Valemi K_F moodustamiseks kuluv tööaeg on ilmselt võrdeline väljundi pikkusega. $\square$

Näide. $F = \underbrace{\underbrace{(\underbrace{(x \oplus y)}_{u_1} \oplus z)}_{u_2} \oplus w}_{u_3}.$

1) $u_1 \sim (x \oplus y)$,
2) $u_2 \sim (u_1 \oplus z)$,
3) $u_3 \sim (u_2 \oplus w)$,

$u_1 \sim (x \oplus y) \equiv (u_1 \vee (x \sim y)) \;\&\; (\overline{u}_1 \vee (x \oplus y)) \equiv$
$\equiv (u_1 \vee ((x \vee \overline{y}) \;\&\; (\overline{x} \vee y))) \;\&\; (\overline{u}_1 \vee ((x \vee y) \;\&\; (\overline{x} \vee \overline{y}))) \equiv$
$\equiv (u_1 \vee x \vee \overline{y}) \;\&\; (u_1 \vee \overline{x} \vee y) \;\&\; (\overline{u}_1 \vee x \vee y) \;\&\; (\overline{u}_1 \vee \overline{x} \vee \overline{y})$,
$u_2 \sim (u_1 \oplus z) \equiv (u_2 \vee u_1 \vee \overline{z}) \& (u_2 \vee \overline{u}_1 \vee z) \& (\overline{u}_2 \vee u_1 \vee z) \& (\overline{u}_2 \vee \overline{u}_1 \vee \overline{z})$,
$u_3 \sim (u_2 \oplus w) \equiv (u_3 \vee u_2 \vee \overline{w}) \& (u_3 \vee \overline{u}_2 \vee w) \& (\overline{u}_3 \vee u_2 \vee w) \& (\overline{u}_3 \vee \overline{u}_2 \vee \overline{w})$.

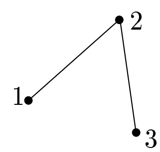
12. Klikk, sõltumatu hulk ja tippude kate

Definitsioon. $V' \subseteq V$ on graafi $G = (V, E)$ *klikk*, kui iga tipupaar u ja v hulgas V' on ühendatud servaga.

Definitsioon. $V' \subseteq V$ on graafi G *sõltumatu hulk*, kui ükski tipupaar hulgast V' ei ole seotud servaga.

Definitsioon. $V' \subseteq V$ on graafi G *tippude kate*, kui iga serva $\{x, y\} \in E$ korral $x \in V'$ või $y \in V'$

Näide. Graaf G



Klikid – $\{\}$, $\{1\}$, $\{2\}$, $\{3\}$, $\{1, 2\}$, $\{2, 3\}$.
Sõltumatud hulgad – $\{\}$, $\{1\}$, $\{2\}$, $\{3\}$, $\{1, 3\}$.
Tippude katted – $\{1, 2, 3\}$, $\{1, 2\}$, $\{1, 3\}$, $\{2, 3\}$, $\{2\}$.

Toetudes neile mõistetele defineerime kolm keelt.

CLIQUE
Antud: Graaf $G = (V, E)$, naturaalarv $k > 0$.
Omadus: Graafis G leidub klikk $\geqslant k$.

INDEPENDENT − SET
Antud: Graaf $G = (V, E)$, naturaalarv $k > 0$.
Omadus: Graafis G leidub sõltumatu hulk $\geqslant k$.

VERTEX − COVER
Antud: Graaf $G = (V, E)$, naturaalarv $k > 0$.
Omadus: Graafis G leidub tippude kate $\leqslant k$.

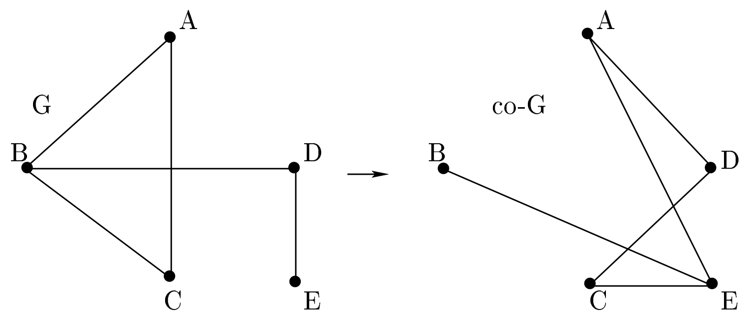
Kõik kolm esitatud probleemi on ilmselt klassist **NP**. Näitame seda keele $CLIQUE$ jaoks, ülejäänud kaks mittedetermineeritud tuvastamisalgoritmi on analoogilised.

input graaf $G = (V, E)$, naturaalarv $k > 0$.
guess $V' \subseteq V$.
check if V' on graafi G klikk.

Hulga V' mittedetermineeritud genereerimine on ilmselt polünomiaalse ajakeerukusega; me saame kasutada selleks juba tuttavat Turingi masinat *bitstring* hulga V' karakteristliku vektori mittedetermineeritud genereerimiseks. Test, kas V' on klikk nõuab $(|V'| \cdot (|V'| - 1))/2$ tipupaari jaoks kontrolli, kas vastav serv kuulub graafi G servade hulka E. Ilmselt on test teostatav polünomiaalse ajaga ka siis, kui me programmeerime selle jõumeetodil.

Definitsioon. Graafi $G = (V, E)$ täiendgraaf $coG = (V, coE)$, kus $coE = (V \times V) \backslash (E \cup \{(x, x) : x \in V\})$.

Näide.



Teoreem 12.1. *Järgmised väited on samaväärsed:*
(a) V' on graafi G tippude kate.
(b) $V \backslash V'$ on G sõltumatu hulk.
(c) $V \backslash V'$ on coG klikk.

Tõestus. (a)$\Rightarrow$(b). Kui V' on G tippude kate, siis iga graafi G serva vähemalt üks otspunkt peab kuuluma hulka V'. Seetõttu ei saa leiduda serva, mille mõlemad otspunktid on hulgast $V \backslash V'$ ja $V \backslash V'$ on graafi G sõltumatu hulk.

(b)$\Rightarrow$(c). Kui $V \backslash V'$ on graafi G sõltumatu hulk, siis suvaliste tippude $u, v \in V \backslash V'$ vahel puudub serv graafis G. Siis $\{u, v\} \in coE$ ja hulk $V \backslash V'$ on graafi coG klikk.

(c)$\Rightarrow$(a). Olgu $V \backslash V'$ graafi coG klikk. Oletame vastuväiteliselt, et V' ei ole graafi G tippude kate. Siis peab leiduma serv $\{u, v\}$ graafis G, mille mõlemad otspunktid on hulgas $V \backslash V'$. Kui $\{u, v\} \in E$, siis $\{u, v\} \notin coE$ ja $V \backslash V'$ ei ole coG klikk. Vastuolu. □

Järeldus 12.2. *Keeled $INDEPENDENT - SET$, $CLIQUE$ ja $VERTEX - COVER$ on polünomiaalselt ekvivalentsed.*

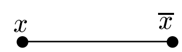
Tõestus. Tuleneb otseselt teoreemist 12.1, kuna graafi G teisendamine graafiks coG ja tippude hulga V' teisendamine hulgaks $V \backslash V'$ on ilmselt teostatavad polünomiaalse tööajaga. $\square$

Me oleme nüüdseks selgitanud, et kolm lausearvutuse probleemi klassist **NP** on polünomiaalselt ekvivalentsed: SAT, $CNF - SAT$ ja $3CNF - SAT$. Samuti on omavahel ekvivalentsed kolm graafiteooria probleemi: $VERTEX - COVER$, $INDEPENDENT - SET$ ja $CLIQUE$. Loomulik uudishimu nõuab selgust nende kahe probleemidegrupi vahekorra kohta.

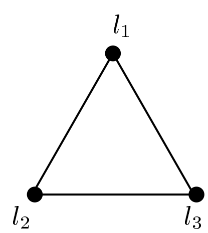
Teoreem 12.3. $3CNF - SAT \leqslant_m^P VERTEX - COVER$.

Tõestus. Olgu $F \in 3CNF$ muutujatega $X = \{x_1, ..., x_n\}$ ja $F = D_1 \,\&\, ... \,\&\, D_m$. Moodustame graafi G_F järgmiselt:

1) Igale muutujale x seame vastavusse paari:

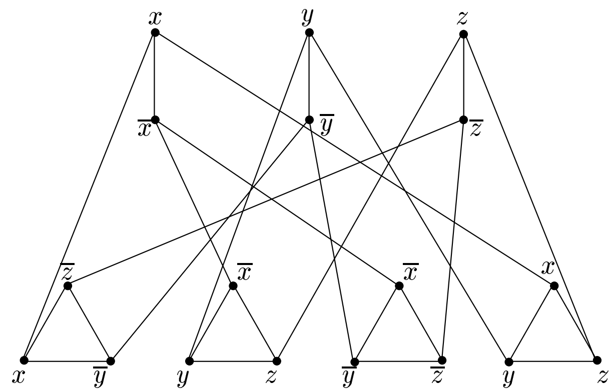


2) Igale disjunktile $D = l_1 \vee l_2 \vee l_3$ seame vastavusse kolmnurga:



3) Paaride ja kolmnurkade vahel ühendatakse sama märgendiga tipud.

Näiteks valemi $F = (x \vee \overline{y} \vee \overline{z}) \,\&\, (\overline{x} \vee y \vee z) \,\&\, (\overline{x} \vee \overline{y} \vee \overline{z}) \,\&\, (x \vee y \vee z)$ korral saame järgmise graafi:



Näitame, et F on kehtestatav siis ja ainult siis, kui graafis G_F leidub tippude kate, milles on mitte rohkem kui $2m + n$ tippu.

1) Oletame, et F on kehtestatav, siis leidub $\alpha \in \{0,1\}^n$ nii et $F(\alpha) = 1$. Valime igast paarist tipu märgendiga $x_i^{\alpha_i}$. Valime igast kolmnurgast kaks tippu, nii et kõik tipud, mille märgendid α muudab vääraks, oleks valitud. Selline valik on alati võimalik, kuna iga kolmnurga märgendid on mingi disjunkti literaalid. Vähemalt üks literaal igast disjunktist peab olema tõene väärtustusel α, sest α on kehtestav väärtustus. Seega on meil valitud $2m + n$ tippu. Veel on vaja veenduda, et tegemist on tippude kattega. Igast paarist on valitud üks tipp, mis katab paarisisese serva. Igast komnurgast on valitud kaks tippu, mis katavad kolmnurga siseservad. Suvaline kolmnurga ja paari tippude vaheline serv on kaetud paaripoolse tipuga, kui see oli valitud. Kui paaripoolne tipp ei olnud valitud, siis on selle märgend väär väärtustusel α ja seetõttu oli valitud serva kolmnurgapoolne tipp, millel on konstruktsiooni põhjal sama märgend.

2) Oletame, et graafis G_F leidub tippude kate, milles on $2m+n$ tippu ehk märgitud on üks tipp igast paarist ja kaks tippu igast kolmnurgast (muidu ei oleks kõik kolnurkade ja paaride sisemised servad kaetud). Võtame muutuja tõeseks või vääraks vastavalt sellele, kumb literaal on paaris märgitud. Kõik disjunktid peavad olema tõesed, sest kolmnurga märkimata tipust mingisse paari minev serv peab olema kaetud paari kuuluva tipu poolt.

□

Selleks, et uurida, kas keel $CLIQUE$ (ja sellega koos kõik ekvivalentsed keeled) on polünomiaalselt taandatavad keelele SAT, meenutame, et me oskame konstrueerida suvalise graafi G järgi konjunktiivsel normaalkujul valemi CF_G, mille tõesed väärtustused on graafi G klikid. Keele $CLIQUE$ väljendamiseks tuleb ära keelata need lahendid milles on vähem kui k ühte.

Olgu CF_G graafi klikkide struktuuri väljendav CNF. Teoreemide 2.2 ja 4.1 põhjal võime väita, et suvalise graafi G jaoks valemi

$$CF_G(x_1, \ldots, x_n) \,\&\, atleast(k, x_1, \ldots, x_n)$$

iga kehtestav väärtustus on niisuguse G kliki karakteristlik vektor, milles on vähemalt k tippu. Seega valem on kehtestatav siis ja ainult siis, kui graafis G leidub vähemalt k-tipuline klikk.

Paraku ei ole siin tegemist polünomiaalse taandamisega. Valemis $atleast(k, x_1, \ldots, x_n)$ on $\binom{n}{k}$ disjunkti, mis on eksponentsiaalses sõltuvuses n-st, kui $k \sim \frac{n}{2}$ (vt. ülesanne 36). Seetõttu tuleb alamgraafi tippude arvu määramiseks kasutada keerulisemat varianti.

Lähtume endiselt graafist $G = (V, E)$, $V = \{1, ..., n\}$ ja naturaalarvust $k \leqslant n$. Valemi F_G muutujate hulgaks võtame $k \times n$ maatriksi, veergudes numbritega i ja j lubame samaaegselt ühtesid ainult siis, kui serv $\{i, j\} \in E$. Lisaks nõuame, et igas reas oleks täpselt üks 1 ja igas veerus ülimalt üks 1 – see tagab alamgraafi suuruse vähemalt k.

$$F_G(x_{11}, ..., x_{kn}) = \mathop{\&}_{\{i,j\} \in co-E} \left(\mathop{\&}_{l=1}^{k} \left(\mathop{\&}_{m=1}^{k} (\overline{x}_{li} \vee \overline{x}_{mj}) \right) \right) \,\&$$

$$\& \left(\mathop{\&}_{l=1}^{k} exactly(1, x_{l1}, ..., x_{ln}) \right) \,\&\, \left(\mathop{\&}_{m=1}^{n} atmost(1, x_{1m}, ..., x_{km}) \right).$$

Teoreem 12.4. $F_G(\alpha_{11}, \ldots, \alpha_{kn}) = 1$ *parajasti siis, kui väärtustusel* α *tõeste muutujate teised indeksid moodustavad graafi* G *k-kliki.*

Tõestus. Tõestuse jätame iseseisvaks harjutuseks, vt. ülesanne 35. □

Ilmselt on valemi $F_G(x_{11}, ..., x_{kn})$ pikkus polünomiaalses sõltuvuses graafi G tippude arvust n (vt ülesanne 37) ja seetõttu realiseerib valemi F_G konstruktsioon polünomiaalse taandamise $CLIQUE \leqslant_m^P CNF - SAT$.

13. Cook-Levini teoreem

Keerukusklass **P** sisaldub klassis **NP**. Tõepoolest, kui $A \in$**P**, siis leidub deterministlik polünomiaalses ajas töötav Turingi masin, mis tuvastab keele A. Deterministlik Turingi masin on erijuhus mittedeterministlikust, seetõttu $A \in$**NP**. Mittedeterministlikku polünomiaalses ajas töötavat Turingi masinat saab modelleerida deterministliku Turingi masina abil, vaadates süstemaatiliselt läbi kõik mittedeterministlikest käskudest tekkivad lahendusteed. Üldjuhul on selliste teede arv eksponentsiaalne (tuletame meelde masinat *bitstring*); seega on ka modelleeriva Turingi masina tööaeg eksponentsiaalne. Siit ei saa aga kuidagi järeldada, et sisalduvus **P**$\subseteq$**NP** on range.

Probleem, kas **P**$\subset$**NP** või **P**=**NP** on siiani lahendamata. Kuna väga paljud olulised ülesanded kuuluvad klassi **NP** ning nende jaoks pole teada deterministlikke polünomiaalseid lahendusalgoritme, siis pakub huvi **NP**-keelte omavaheline võrdlemine (mida me kahes eelmises peatükis juba tegime). Süstemaatiliseks käsitluseks toome sisse **NP**-täieliku keele mõiste.

Definitsioon. Keel L on **NP**-*raske* (**NP**-*hard*), kui iga $L' \in$**NP** jaoks kehtib $L' \leqslant_m^P L$.

Definitsioon. L on **NP**-*täielik*, kui $L \in$**NP** ja L on **NP**-raske.

NP-täielikud keeled on polünomiaalse taandamise täpsuseni kõige raskemad keeled klassis **NP**. Järgnev teoreem annab lihtsa meetodi keelte **NP**-täielikkuse tõestamiseks. Teoreemi tõestasid sõltumatult S. Cook ([Coo71]) ja L. Levin ([Lev75]). Varem nimetati seda Cooki teoreemiks, viimasel ajal, kui L. Levini artikkel [Lev75] on saanud tuntuks ka läänemaailmas, kasutatakse nimetust Cook-Levini teoreem.

Teoreem 13.1. ***(Cook-Levini teoreem)****, Keel CNF-SAT on* **NP**-*täielik.*

Tõestus. Tuleb näidata, et suvaline keel $L \in$**NP** on polünomiaalselt taandatav keelele $CNF - SAT$. Olgu M keelt L tuvastav mittedeterministlik Turingi masin polünomiaalse keerukusega $p(n)$. Konstrueerime Turingi masina M, polünoomi $p(n)$ ja sõna x järgi konjunktiivsel normaalkujul lausearvutuse valemi $F_{M,x}$, mis on kehtestatav siis ja ainult siis, kui M aktsepteerib sõna x. Olgu $|x| = n$ ja $M(x)$ mingi arvutustee $C_1, C_2, ..., C_{p(n)}$, C_i—konfiguratsioon, $|C_i| = p(|x|)$. Kui arvutustee pikkus on väiksem kui $p(n)$ konfiguratsiooni, kordame viimast. Tähistame Turingi masina M programmi $\delta(M)$, mille käskude arv olgu $|\delta(M)|$. Olgu käsud programmis $\delta(M)$ nummerdatud arvudega $1, \ldots, \delta(M)$. Esitame $F_{M,x}$ muutujate hulga kirjelduse koos nende tähendusega.

Muutuja	Tähendus
$S(q,t)$	Konfiguratsioonis C_t on olek q
$H(h,t)$	Konfiguratsioonis C_t skaneerib lugemispea pesa h
$T(b,h,t)$	Konfiguratsioonis C_t on pesas h sümbol b
$I(j,t)$	Üleminekul $C_t \vdash C_{t+1}$ kasutatakse käsku j

Järgnevalt esitame disjunktid, mis tagavad muutujate ülaltoodud tähenduse.

1) Konfiguratsioon C_1 on Turingi masina M lähtekonfiguratsioon sisendiga $x = a_1, ..., a_n$.
$\{S(q_1, 1)\}$,
$\{T(a_1, 1, 1)\}, \{T(a_2, 2, 1)\}, ..., \{T(a_n, n, 1)\}$,
$\{T(\square, h, 1)\}$, kus $h \notin \{1, ..., n\}$.
Esimene tingimus tagab, et lähtekonfiguratsioon on olekus q_1, järgmised n tingimust tagavad, et sisend x sisaldub pesades $1...n$ ning viimase tingimuse kohaselt on ülejäänud pesades tühisümbol $\square$.

2) Igas konfiguratsioonis C_t on M täpselt ühes olekus.
$unique(S(1,t), ..., S(r,t))$.

3) Konfiguratsiooni C_t igas pesas on täpselt 1 sümbol.
$unique(T(x_0, h, t), ..., T(x_s, h, t))$.

4) Igas konfiguratsioonis C_t on lugemispea asend ühene.
$unique(H(0,t), ..., H(p(n), t))$.

5) Viimane konfiguratsioon on aktsepteeriv — $\{S(q_a, p(n))\}$.

6) Üleminekul $C_t \vdash C_{t+1}$ võib muutuda ainult skaneeritav pesa.
$\{\overline{T(b,h,t)}, \overline{T(c,h,t+1)}, H(h,t)\}$.

7) Turingi masina M käsk, mis määrab ülemineku $C_t \vdash C_{t+1}$ on ühene.
$unique(I(1,t), ..., I(|\delta(M)|, t))$.

8) Muudatused järjestikustes konfiguratsioonides $(C_t + C_{t+1})$ vastavad M programmile. Olgu iga oleku q ja sümboli b jaoks $\delta(q,b)$ käskude numbrite hulk, nii et $j \in \delta(q,b)$, kui j-nda käsu kaks esimest komponenti on q ja b. Iga t ja h jaoks lisame disjunkti:
$\{\overline{T(b,h,t)}, \overline{S(q,t)}, \overline{H(h,t)}\} \cup \{I(j,t) : j \in \delta(b,q)\}$.

Lisaks, kui j-is käsk on $\langle q, b, q', b', d \rangle$, siis lisame disjunktid:
$\{\overline{I(j,t)}, S(q', t+1)\}$,
$\{\overline{I(j,t)}, \overline{H(h,t)}, T(b', h, t+1)\}$,
$\{\overline{I(j,t)}, \overline{H(h,t)}, H(h+d, t+1)\}$.

Selgituseks punktide 6) ja 8) juurde: disjunkt kujul

$$\overline{x_1} \vee \ldots \vee \overline{x_n} \vee y_1 \vee \ldots \vee y_k$$

on samaväärne valemiga

$$(x_1 \ \& \ \ldots \ \& \ x_n) \supset (y_1 \vee \ldots \vee y_k).$$

Näitame et valem $F_{M,x}$ on kehtestatav, parajasti siis kui $x \in L$. Oletame, et $F_{M,x}$ on tõene mingi väärtustuse α korral, s.t. kõik disjunktid tingimustes 1) – 8) on tõesed väärtustuse α jaoks. Tingimus 1) ütleb, et algkonfiguratsioon on $q_1 a_1 \ldots a_n$. Tingimused 2) – 4) ning 6) – 8) määravad konfiguratsiooni C_i järgi konfiguratsiooni C_{i+1} vastavalt Turingi masina M programmile. Konfiguratsioon C_1 on üheselt kindlaks määratud tingimusega 1. Oletame et $C_1, ..., C_t$ on määratud, siis on aga lihtne määrata C_{t+1}, seega arvutustee on määratav. Tingimus 5 tagab, et $C_{p(|x|)}$ on aktsepteeriv. Vastupidi, kui arvutustee leidub, siis ta määrab väärtustuse, mis rahuldab valemit $f_{M,x}$. On lihtne veenduda, et kõik 8 tingimust on teostatavad ajalise keerukusega $O((p(n))^3)$ (vt ülesanne 40). $\square$

Cook-Levini teoreem annab lihtsa võimaluse tõestamaks, et keel L on **NP**-raske: tuleb näidata, et $CNF - SAT \leqslant_m^P L$. Tõepoolest, kuna Cook-Levini teoreemi põhjal suvalise keele $L' \in$**NP** jaoks kehtib $L' \leqslant_m^P CNF - SAT$, siis saame polünomiaalse taandamise transitiivsust (teoreem 10.1 p.3.) kasutades taandatavusest $CNF - SAT \leqslant_m^P L$ järeldada $L' \leqslant_m^P L$.

Järeldus 13.2. *Keeled SAT, $3CNF - SAT$, $VERTEX - COVER$ $INDEPENDENT - SET$, ja $CLIQUE$ on* **NP***-täielikud.*

Tõestus. Väide järeldub eelmise kahe peatüki materjalidest ja polünomiaalse taandamise transitiivsusest. $\square$

14. Rändkaupmehe ülesanne

Kaupmehel on vaja laiali vedada kaubad linnade $(l_1, ..., l_n)$ vahel. Linnade vahelised kaugused on määratud maatriksiga.

$$\text{Kauguste maatriks} — \begin{pmatrix} d_{11} & ... & d_{1n} \\ \vdots & & \vdots \\ d_{n1} & ... & d_{nn} \end{pmatrix}$$

Probleem: Leida kinnine marsruut läbi kõigi n linna nii, et iga linna läbitakse 1 kord ja summaarne kaugus oleks minimaalne. Vastav keele tuvastamise probleem oleks järgmine.

TRAVELLING − SALESMAN
Antud: kauguste maatriks D, marsruudi pikkus k.
Omadus: Leidub marsruut kogupikkusega $\leqslant k$.

Keel *TRAVELLING − SALESMAN* kuulub ilmselt keerukusklassi **NP**, tuvastav mittedeterministlik algoritm on järgmine:

input D, k.
guess järjestus $\pi = (\pi_1, \ldots, \pi_n)$.
check if $\Sigma_{i=1}^{n-1} d_{\pi_i,\pi_{i+1}} + d_{\pi_n,\pi_1} \leqslant k$.

Selleks, et tõestada keele *TRAVELLING − SALESMAN* **NP**-täielikkust, uurime kõigepealt veidi sarnast graafiteooria probleemi - Hamiltoni tsüklit.

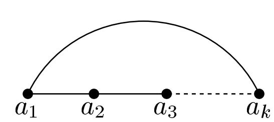
HAMILTONIAN − CIRCUIT
Antud: graaf $G = (V, E)$.
Omadus: leidub tippude järjestus $\pi = (\pi_1, \ldots, \pi_n)$ nii et $\{\pi_1, \pi_2\} \in E, \ldots, \{\pi_{n-1}, \pi_n\} \in E, \{\pi_n, \pi_1\} \in E$.

Ilmselt kuulub ka keel *HAMILTONIAN − CIRCUIT* klassi **NP**, mittedetermineeritud testalgoritm on analoogiline eelmise näitega.

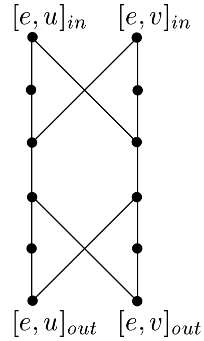
Teoreem 14.1. *VERTEX-COVER*$\leqslant_m^P$*HAMILTONIAN-CIRCUIT*.

Tõestus.
Olgu paar (G, k) tippude katte ülesanne. Konstrueerime graafi G' nii, et graafis G' leidub Hamiltoni tsükkel siis ja ainult siis, kui graafis G leidub tippude kate, mis koosneb k tipust.

a) Lisame graafi G' tsükli:

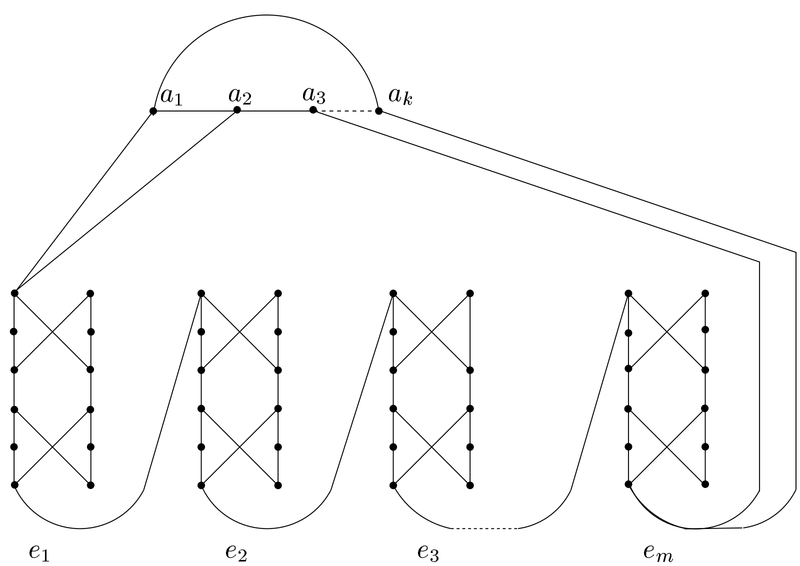


b) Iga serva $e = \{u, v\} \in E$ jaoks moodustame 12-tipulise graafi (viguri):



Seega $|V'| = k + 12|E|$.

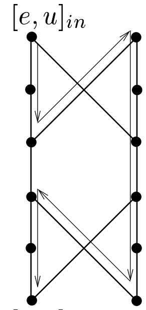
c) Iga tipu $u \in V$ jaoks olgu $e_1, ..., e_m$ servad hulgast E, mis on intsidentsed tipuga u. Lisame hulka E' järgmised servad: $\{[e_1, u]_{out}, [e_2, u]_{in}\}, \{[e_2, u]_{out}, [e_3, u]_{in}\}, ..., \{[e_{m-1}, u]_{out}, [e_m, u]_{in}\}(*)$. Lisaks ühendame veel $[e_1, u]_{in}$ ja $[e_m, u]_{out}$ iga tipuga a_i $(i = 1, ..., k)$ ringis.



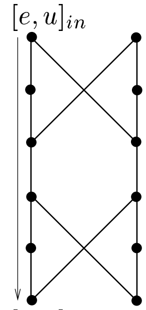
Nüüd näitame, et G sisaldab tippude katet suurusega k parajasti siis kui G' sisaldab Hamiltoni tsüklit. Oletame, et $U = \{u_1, ..., u_h\}$ on tippude kate graafis G, $h \leqslant k$. Iga tipu $u_j \in U$ jaoks defineerime tee $p_j : a_j \Rightarrow a_{j+1}$. Olgu $e_1, ..., e_m$ E servad, mis on ühenduses u_j. Tee p_j sisaldab siis kõiki servi * ja servi $\{a_j, [e_1, u_j]_{in}\}$ ning $\{[e_m, u_j]_{out}, a_{j+1}\}$.

p_j peab ühendama ka tippe $[e_i, u_j]_{in}$ ja $[e_i, u_j]_{out}$ $\forall i$ $i = 1, ..., m$, milleks on kaks moodust:

1) Kui serva e_i teine otstipp ei kuulu tippude kattesse siis:



2) Kui serva e_i teine otstipp kuulub tippude kattesse siis:



Ühendades nüüd kõik teed $p_1, ..., p_h$, saame tee $a_1 ... a_{h+1}$, mis läbib kõik tipud igas viguris. Lõpetuseks ühendatakse kõik teed tsükliks, ühendades selleks $a_{h+1}, a_{h+2}, ..., a_k$ ja a_1, mille tulemuseks on Hamiltoni tsükkel.
Teiselt poolt oletame, et eksisteerib Hamiltoni tsükkel C graafis G'. Tsükli saab jagada teedeks, millede lõpp-punktid on ringis. Iga sellise tee p jaoks leidub unikaalne tipp $u(p) \in V$, nii et p käib läbi kõigi vigurite, sest viguri läbimiseks on ainult kaks viisi, mis on näidatud eelpool. Seega on lihtne näha, et selliste tippude $u(p)$ ühend moodustab tippude katte G-s. Kuna selliseid teid C-s saab olla maksimaalselt k, on tippude katte maksimaalne suurus k. $\square$

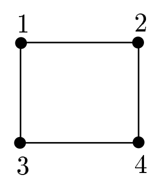
Järeldus 14.2. *Rändkaupmehe ülesanne on NP-täielik*

Tõestus. Tõestamiseks näitame, et $HAMILTONIAN-CIRCUIT \leqslant_m^p TRAVELLING-SALESMAN$. Olgu antud Hamiltoni tsükli probleem – graaf G=(V,E). Tõlgendame graafi tippude hulka V asustatud punktidena rändkaupmehe ülesande jaoks, mille kauguste maatriksi moodustame järgmiselt:

$$d_{ij} = \begin{cases} 1, \text{kui } \{i,j\} \in E, \\ |V|+1, \text{kui } \{i,j\} \notin E. \end{cases}$$

Marsruudi kogupikkuse k võtame võrdseks tippude arvuga n. Oletame, et graafis G leidub Hamiltoni tsükkel. Kuna graafi servadele vastavad kaugused on kõik võrdsed konstandiga 1, siis on Hamiltoni tsüklile vastava marsruudi pikkus n. Suvaline marsruut, mis ei ole Hamiltoni tsükkel, sisaldab serva kaugusega $|V|+1$ ja sells kogupikkus on suurem kui n. □

Näide Olgu meil antud graaf:



Moodustame maatriksi:

	1	2	3	4
1	5	1	1	5
2	1	5	5	1
3	1	5	5	1
4	5	1	1	5

Hamiltoni tsüklile (1,3,4,2) vastab kaalude summa 1+1+1+1=4, kuid valides järjestuseks (1,2,3,4), mis ei ole Hamiltoni tsükkel, saame 1+5+1+5=12.

15. Graafi tippude värvimine

Graafi $G = (V, E)$ tippude värvimisviisiks k värviga nimetatakse funktsiooni γ graafi tippude hulgast V hulka $K = \{1, \ldots, k\}$ Värvimisviisi nimetatakse korrektseks, kui ei leidu serva, mille mõlemad otstipud oleksid sama värvi. Graaf G on värvitav k värviga, kui leidub tippude korrektne värvimisviis k värviga.

$k - COLOURABILITY$
Antud: graaf G.
Omadus: graaf G on värvitav k värviga.

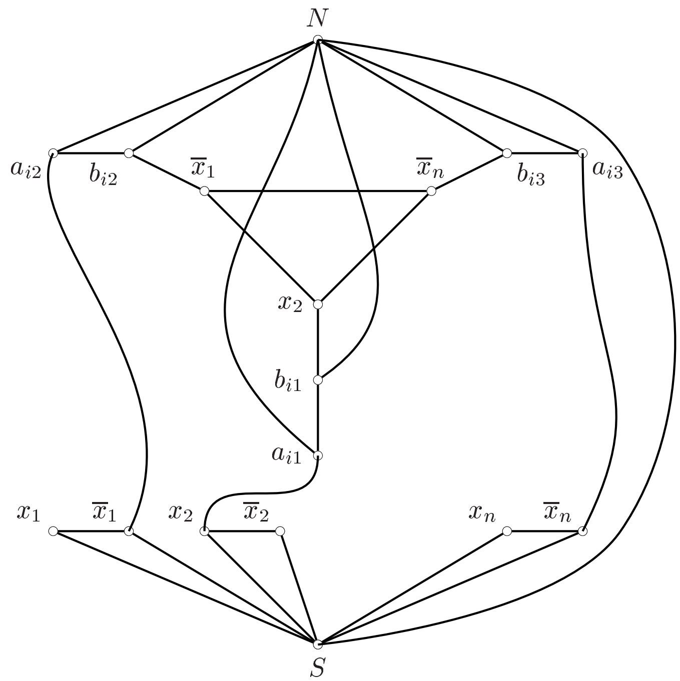
Näitame, et $k - COLOURABILITY$ $\in$**NP**. Mittedeterministlik algoritm k värviga värvitavuse testimiseks on järgmine:

input graaf $G = (V, E)$
guess tippude hulga V tükeldus k osaks.
check if kõigi servade otspunktid kuuluvad erinevatesse tükkidesse.

Teoreem 15.1. *3-COLOURABILITY on* **NP***-täielik.*

Tõestus. Teoreemi tõestuseks taandame keele *3CNF* $- SAT$ keelele $3 - COLOURABILITY$. Igale 3CNF-valemile $F(x_1, \ldots, x_n)$ seame vastavusse graafi G_F. Alguseks võtame kaks tippu N ja S ning ühendame need servaga. Lisame iga muutuja x_i jaoks servaga ühendatud tipupaari x_i ja $\overline{x}_i$ ning ühendame paaride otspunktid tipuga S. Iga disjunkti $l_{i,1} \vee l_{i,2} \vee l_{i,3}$ jaoks lisame kolmnurga tippudega $l_{i,1}, l_{i,2}, l_{i,3}$. Kolmnurga iga tipu $l_{i,j}$ ühendame ahelasse uute tippudega $b_{i,j}$ ja $a_{i,j}$. Ühendame tipu $a_{i,j}$ literaalile $l_{i,j}$ vastava tipuga paarides. Ühendame kõik tipud $b_{i,j}$ ja $a_{i,j}$ tipuga N (vt. joonist 15.1).

Vaatleme, millistel tingimustel on graaf G_F värvitav kolme värviga. Tähistame värve tähtedega **C,T** ja **F**. Üldisust kitsendamata võime värvida tipu S värviga **C** ja tipu N värviga **T**. Kuna literaalipaaride tipud on ühendatud omavahel ning tipuga S, siis on ainus võimalus värvida iga muutuja literaalid suvalisel viisil värvidega **T** ja **F**. Määrame väärtustuse α, lugedes tõeseks värviga **T** värvitud literaalid. Igale disjunktile vastab graafis G_F kolmnurk, mille tipud tuleb värvida korrektse värvingu saamiseks erinevalt. Igas kolmnurgas peab üks tipp, olgu see l_{ij},olema värvi **F**. Kuna tipud a_{ij} ja b_{ij} on ühendatud tipuga N, mis on värvi **T**, siis tuleb värvida tipp b_{ij} värviga **C** ja tipp a_{ij} värviga **F**. Vastuolu ei teki ainult sel juhul, kui tipuga a_{ij} ühendatud literaal on värvi **T**, s.t. literaal l_{ij} on tõene väärtustuse α jaoks.



Joonis 15.1: Valemi $\overline{x}_1 \vee x_2 \vee \overline{x}_n$ värvigraaf.

Seega on korrektse värvingu puhul igas disjunktis tõene literaal ning $F(\alpha) = 1$. Kui aga $F(\alpha) = 0$, siis ei ole korrektne värvimine kolme värviga võimalik. □

Järeldus 15.2. *Keel k-COLOURABILITY on* **NP**-*täielik iga* $k \geqslant 3$*jaoks.*

Tõestus. Lisame graafile G uue tipu, mille ühendame kõigi graafi G tippudega. Saadud graaf on värvitav $k+1$ värviga parajasti siis, kui G on värvitav k värviga. □

16. Kolmemõõtmeline sobitusülesanne

Olgu Z, X, Y mittelõikuvad hulgad võrdse elementide arvuga q ja $M \subseteq Z \times X \times Y$. $M' \subseteq M$ on kolmemõõtmeline sobitus, kui $|M'| = q$ ja suvalise kahe M' elemendi kõik koordinaadid on erinevad.

3 − MATCHING
Antud: Z, X, Y; $M \subseteq Z \times X \times Y$.
Omadus: Hulgas M sisaldub kolmemõõtmeline sobitus.

Mittedeterministlik algoritm keele *3 − MATCHING* tuvastamiseks on järgmine:

input $M \subseteq Z \times X \times Y$
guess $M' \subseteq M$
check if M' on kolmemõõtmeline sobitus.

Teoreem 16.1. *3CNF-SAT*$\leqslant_m^P$*3-MATCHING*

Tõestus. Moodustame valemi $F(x_1, \ldots, x_n) = \&_{i=1}^{m}(l_{i1} \vee l_{i2} \vee l_{i3})$ järgi hulgad Z, X ja Y ning $M \subseteq Z \times X \times Y$ sellised, et M sisaldab 3-sobitust siis ja ainult siis, kui F on kehtestatav.

Kõigepealt moodustame kolmikud, mis määravad väärtustuse. Iga muutuja x_i jaoks moodustame $2m$ kolmikut

$$T_i = \{(\overline{x}_i[j], a_i[j], b_i[j]) : 1 \leqslant j \leqslant m\},$$

$$F_i = \{(x_i[j], a_i[j+1], b_i[j]) : 1 \leqslant j < m\} \cup \{(x_i[m], a_i[1], b_i[m])\},$$

kus $a_i[j] \in X;\ b_i[j] \in Y;\ x_i[j], \overline{x}_i[j] \in Z : 1 \leqslant i \leqslant n,\ 1 \leqslant j \leqslant m$. Kolmikute "sisemisi" elemente $a_i[j], b_i[j]$ ei kasutata hulgale M edaspidise konstruktsiooni käigus uusi elemente lisades, seetõttu jäävad sobitust moodustades iga muutuja x_i jaoks vabaks kõik literaalid x_{ij} või kõik literaalid $\overline{x}_{ij}$. Vabad literaalid määravad väärtustuse.

Teiseks lisame hulgale M kolmikud disjunktide tõesuse kontrolliks. Selleks lisame iga disjunkti D_j jaoks hulgale X elemendid $r[j]$ ja hulgale Y elemendid $s[j]$ ning hulgale M kolmikud

$$C_j = \{(x_i[j], r[j], s[j]) : x_i \in D_j\} \cup \{(\overline{x}_i[j], r[j], s[j]) : \overline{x}_i \in D_j\}.$$

Elementide $r[j], s[j]$ haaramine sobitusse on võimalik ainult siis, kui disjunktis D_j on vähemalt üks literaal antud väärtustuse jaoks tõene.

Konstrueerides 3-sobitust, mis määrab väärtustuse ja kontrollib disjunktide tõesust selle väärtustuse jaoks, jäävad mõned elemendid kaasa haaramata. Täpsemalt, igast väärtustust määravast konstruktsioonist jäävad pärast väärtustuse fikseerimist vabaks m "tõest" literaali, kokku $m \cdot n$ elementi hulgast Z. Disjunktide kontrolli kolmikud haaravad nendest m, seega jääb vabaks $m \cdot (n-1)$ elementi. Lisame hulka X elemendid g_k ja hulka Y elemendid h_k, $1 \leqslant k \leqslant m \cdot (n-1)$ ning hulka M kolmikud

$$P = \{(x_i[j], g_k, h_k), (\overline{x}_i[j], g_k, h_k) : \\ 1 \leqslant i \leqslant n, 1 \leqslant j \leqslant m, 1 \leqslant k \leqslant m \cdot (n-1)\}.$$

Kokkuvõtteks:

$$M = \left(\bigcup_{i=1}^{n} (T_i \cup F_i) \right) \cup \left(\bigcup_{j=1}^{m} C_j \right) \cup P.$$

Iga 3-sobitus peab määrama väärtustuse, mille jaoks on kõik disjunktid tõesed ning siduma ülejäänud "vabad" literaalid kolmikutesse tippudega g_k, h_k. Kolmikute hulk M sisaldab $2mn+3m+2m^2n(n-1)$ kolmikut ja on seetõttu konstrueeritav polünomiaalse ajaga valemi F pikkusest. $\square$

17. Keerukusklass coNP

Kehtestatavad lausearvutuse valemid kirjeldavad omal (võrreldes predikaatarvutusega, piiratud) viisil maailma. Seetõttu pole juhuslik, et osa autoreid nimetab valemi tõeseid väärtustusi selle mudeliks. Samaselt tõesed valemid ei anna maailma kohta mingit informatsiooni, aga kirjeldavad loogika tõdesid. Seega pakub kindlasti huvi ka samaselt tõeste valemite keele keerukus. Selle uurimiseks toome sisse vajaliku mõistete süsteemi.

Definitsioon. Olgu **C**– keelteklass. Siis **coC**$= \{L : L^c \in$**C**$\}$.

Teoreem 17.1. coP=P.

Tõestus. Olgu L suvaline keel keerukusklassist **P** ja M deterministlik polünomiaalses ajas töötav Turingi masin, mis tunneb ära keele L. Konstrueerime M' vahetades M tabelis q_a ja q_r. M', mis on samuti determineeritud polünomiaalses ajas töötav Turingi masin, tuvastab keele L^c. □

Mittedeterministlike keelteklasside korral lõppolekute vahetamine ei toimi. Kui me vahetame keelt SAT tuvastaval mittedeterministlikul Turingi masinal lõppolekud, aktsepteerib see kõik lausearvutuse valemid, mille jaoks leidub vääraid väärtustusi. Keel SAT^c koosneb aga valemitest, mille jaoks on **kõik** väärtustused väärad.

Teoreem 17.2. coNP *on kinnine polünomiaalse taandamise suhtes.*

Tõestus. Olgu $A \leqslant_m^P B$ ja $B \in$**coNP**, siis $B^c \in$**NP**. Kasutades teoreemi 10.1 p.4 saame $A^c \leqslant_m^P B^c$. Kuna **NP** on kinnine polünomiaalse taandamise suhtes, siis $A^c \in$**NP** ja $A \in$**coNP**. □

Teoreem 17.3. *Kui L on* **NP***-täielik, siis L^c on* **coNP***-täielik.*

Tõestus. Kui L on **NP**-täielik, siis $L \in$**NP** ja $L^c \in$**coNP**. Olgu A suvaline keel klassist **coNP**. Peame näitama, et $A \leqslant_m^P L^c$. Kuna $A \in$**coNP**, siis $A^c \in$**NP** ja, kuna L on **NP**-täielik, siis $A^c \leqslant_m^P L$. Teoreemi 10.1 p.4. tõttu $A^{cc} \leqslant_m^P L^c$. Kuna $A^{cc} = A$, siis $A \leqslant_m^P L^c$. □

Teoreem 17.4. *Kui leidub* **NP**-*täielik keel* L, *nii et* $L^c \in$**NP**, *siis* **coNP**=**NP**.

Tõestus. Kuna L on NP-täielik ja $L^c \in$**NP**, siis $L^c \leqslant_m^P L$. Teoreemi 10.1 p.4 tõttu $L^{cc} \leqslant_m^P L^c$, millest järeldub $L \leqslant_m^P L^c$.

1) NP⊆coNP.
Olgu $A \in$**NP** suvaline, $A \leqslant_m^P L \leqslant_m^P L^c$. Polünomiaalse taandamise transitiivsuse tõttu $A \leqslant_m^P L^c$. Kuna **coNP** on kinnine polünomiaalse taandamise suhtes ja $L^c \in$**coNP**, siis $A \in$**coNP**.

2) coNP⊆NP.
Olgu $A \in$**coNP**, suvaline. Kuna L on **NP**-täielik, siis L^c on **coNP**-täielik. Seetõttu $A \leqslant_m^P L^c \leqslant_m^P L$. Transitiivsuse tõttu $A \leqslant_m^P L$. Kuna $L \in$**NP** ja **NP** on kinnine polünomiaalse taandamise suhtes, siis $A \in NP$.

□

Teoreem 17.5. *Kui* **coNP**≠**NP**, *siis* **P**≠**NP**.

Tõestus. Oletame vastuväiteliselt, et **P**=**NP**. Kuna **P**=**coP**, siis peaks **NP**=**coNP**, mis on aga vastuolus eeldusega. □

Iga **NP**-täieliku keele L täiend L^c on teoreemi 17.3 põhjal **coNP**-täielik. See asjaolu varustab meid ammendamatu loeteluga **coNP**-täielikest keeltest. Siiski leiduvad mõned keeled, mille jaoks **coNP** on loomulik keerukusklass. Esimene neist on käesoleva peatüki alguses märgitud samaselt tõeste valemite keel ning teine monotoonseid Boole'i funktsioone esitavate valemite keel.

TAUT
Antud: lausearvutuse valem $F(x_1, \ldots, x_n)$.
Omadus: F on samaselt tõene, s.t. iga $\alpha \in \{0,1\}^n$ jaoks $F(\alpha) = 1$.

Ilmselt kehtib $TAUT \equiv_m^P SAT^c$, kuna iga samaselt tõese valemi eitus on samaselt väär ja vastupidi. Keel SAT^c on aga **coNP**-täielik.

MON
Antud: lausearvutuse valem $F(x_1, \ldots, x_n)$.
Omadus: Boole'i funktsioon, mida esitab valem F, on monotoonne.

Teoreem 17.6. *Keel MON on* **coNP**-*täielik.*

Tõestus. 1) Näitame, et $MON \in$**coNP**. Selleks esitame polünomiaalse mittedeterministliku algoritmi keele MON^c tuvastamiseks:
input lausearvutuse valem $F(x_1, \ldots, x_n)$.
guess $\alpha, \beta \in \{0,1\}^n$, niisugused et $\alpha \prec \beta$.
check if $F(\alpha) > F(\beta)$.
Seetõttu $MON^c \in$**NP** ja $MON \in$**coNP**.

2) Näitame, et **coNP**-täielik keel SAT^c on polünomiaalselt taandatav keelele MON. Igale lausearvutuse valemile F seame vastavusse valemi $T(F)$:

$$T(F) = \begin{cases} \overline{x}_1, & \text{kui } F(1, \ldots, 1) = 1, \\ F, & \text{kui } F(1, \ldots, 1) = 0. \end{cases}$$

Näitame, et valem F on samaselt väär (s.t. $F \in SAT^c$) siis ja ainult siis, kui $T(F)$ on monotoonne.

Olgu valem F samaselt väär. Siis $F(1, \ldots, 1) = 0$ ning seetõttu $T(F) = F$. Samaselt väär valem F on aga monotoonne.

Olgu valem F kehtestatav. Kui $F(1, \ldots, 1) = 1$, siis $T(F) = \overline{x}_1$, mis ei ole monotoonne Boole'i funktsioon. Kui $F(1, \ldots, 1) = 0$, siis $T(F) = F$. Et F on kehtestatav, siis leidub $\alpha \in \{0,1\}^n$, niisugune et $F(\alpha) = 1$. Kuna $\alpha \prec^+ (1, \ldots, 1)$, aga $F(\alpha) > F(1, \ldots, 1)$, siis ei ole F monotoonne. $\square$

18. Turingi taandamine

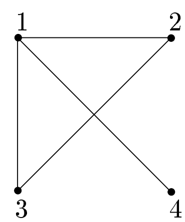
Graafiteooria rakendustes pakuvad huvi järgmised klikkidega seotud probleemid.

Antud on graaf $G = (V, E)$. Mitu tippu on graafi suurimal klikil? Leida graafi suurim klikk! Leida kõik graafi maksimaalsed klikid!

Võrreldes neid ülesandeid meie keelega $CLIQUE$ näib, nagu me püüaksime probleeme liialt lihtsustada. Ülesannetest viimane ei paku keerukusteooria seisukohast erilist huvi, kuna see on halvimal juhul eksponentsiaalse keerukusega (Moon-Moseri graafil $MM(m, l)$ on m^l maksimaalset klikki). Sõnastame kõigepealt graafi suurima kliki tippude arvu leidmise keele probleemina.

$MAXCLIQUE$
Antud: graaf G ja $k > 0$.
Omadus: kas G suurim klikk koosneb k tipust?

Näide. Vaatleme graafi



Selles graafis $\langle G, 2\rangle \in CLIQUE$, $\langle G, 2\rangle \notin MAXCLIQUE$ ning $\langle G, 3\rangle \in MAXCLIQUE$.

Kui me oskaksime polünomiaalses ajas tuvastada keelt $CLIQUE$, siis me saaksime teha sama ka keelega $MAXCLIQUE$, nagu näitab järgmine algoritm.

function $maxclique(G, k)$: $boolean$;
if $clique(G, k) = 1$ & $clique(G, k + 1) = 0$
then $return(1)$
else $return(0)$
fi

$clique(G, k)$ on siin funktsioon, mille väärtus on 1, kui graafis G leidub k-tipuline klikk ja 0 vastupidisel juhul. Sellest hoolimata ei ole näha, kuidas $\leqslant_m^P$-taandada keelt $MAXCLIQUE$ keelele $CLIQUE$. Seetõttu defineerime *Turingi taandamise*, milleks laiendame kõigepealt Turingi masinat *oraakliga*. Oraakliks on mingi keel L ning Turingi masin võib kirjutada spetsiaalsele *oraakli lindile* sõna x ja pöördub oraakli poole küsimusega $x \in L$? Sõltuvalt saadud vastusest läheb Turingi masin olekusse q_{yes} või q_{no}. Oraakli tööaega ei arvestata Turingi masina tööaja sisse. Tähistame T^L Turingi masinat T oraakliga L. Eelpool esitatud algoritm *maxclique* näeks vormistatuna oraaklialgoritmina välja järgmine.

```
maxclique(G, k)
begin
        Kirjutada oraakli lindile küsimus G, k
        if ⟨G, k⟩ ∈ CLIQUE
        then kirjuta oraakli lindile G, k + 1
                if ⟨G, k + 1⟩ ∈ CLIQUE
                then return 0
                else return 1
                fi
        fi
end
```

Definitsioon. Keel A on polünomiaalselt Turingi-taandatav keelele L (tähistame $A \leqslant_T^P L$), kui leidub polünomiaalne determineeritud Turingi masin T oraakliga L, mis tuvastab keele A.

Eelmisest näitest näeme, et $MAXCLIQUE \leqslant_T^P CLIQUE$.

Teoreem 18.1. *Kui $A \leqslant_T^P B$ ja $B \leqslant_T^P C$, siis $A \leqslant_T^P C$ (Turingi taandamine on transitiivne).*

Tõestus. Olgu M_1 ja M_2 deterministlikud polünomiaalses ajas töötavad oraakliga Turingi masinad, mis realiseerivad teoreemi eelduses näidatud taandamised, s.t. $A = L(M_1^B)$ ja $B = L(M_2^C)$. Konstrueerime masina M, mis simuleerib masinat M_1, kuid asendab M_1 oraakli poole pöördumised masina M_2 simuleerimisega M_1 oraaklilindi sisul (pöördudes oraakli poole, kui masin M_2 pöördub oraakli poole). Ilmne on, et $L(M^C) = A$ ja M töötab polünomiaalses ajas, sest M_1 ning M_2 töötavad polünomiaalses ajas ja masinat M_2 simuleeritakse arvutuse jooksul vaid polünomiaalne arv kordi ning sisenditega, mille pikkus on polünomiaalne M sisendi pikkuse suhtes. $\square$

Teoreem 18.2. *Kui $A \leqslant_m^P B$, siis $A \leqslant_T^P B$.*

Tõestus. Olgu f selline polünomiaalses ajas arvutatav funktsioon, et iga $x \in \Sigma^*$ korral $x \in A \Leftrightarrow f(x) \in B$. Konstrueerime oraakliga Turingi masina M, mis sisendi x korral arvutab sõna $f(x)$, seejärel küsib oraaklilt, kas tulemus kuulub oraaklihulka, ja aktsepteerib oraakli positiivse vastuse korral ning lükkab tagasi oraakli negatiivse vastuse korral. Siis $A = L(M^B)$. □

Teoreem 18.3. *Kui $A \in$***P***, siis iga keele B korral $A \leqslant_T^P B$.*

Tõestus. Olgu M deterministlik polünomiaalses ajas töötav Turingi masin, mille korral $L(M) = A$. Me võime masinat M lugeda oraakliga masinaks, mis ei tee oraakli poole ühtegi pöördumist. Siis suvalise keele B korral $L(M^B) = L(M) = A$. □

Kui **K** on keelteklass, siis kõiki keeli, mis on Turingi mõttes polünomiaalselt taandatavad mõnele klassi **K** kuuluvale keelele, tähistame $\mathbf{P}^{\mathbf{K}}$. Eelpooltoodud näidet arvestades võime kirjutada, et $MAXCLIQUE \in \mathbf{P}^{\mathbf{N}P}$. Keelteklass $\mathbf{P}^{\mathbf{NP}}$ on sobiv nn. ekstreemumprobleemide keerukuse kirjeldamiseks. Siiski leidub ülesandeid, mis näivad jäävat väljapoole keerukusklasse **P**, **NP** ning $\mathbf{P}^{\mathbf{NP}}$, kuid on siiski seotud mittedeterministliku polünomiaalse aktsepteerimisega. Näitena vaatleme järgmist probleemi.

$EQUIVALENT - BF$
Antud: lausearvutusvalem F ja $k \geqslant 0$.
Omadus: leidub lausearvutuse valem F', mis sisaldab k literaali esinemist ja on ekvivalentne valemiga F?

Mittedeterministlik Turingi masin keele $EQUIVALENT - BF$ tuvastamiseks on järgmine:

input lausearvutuse valem F.
guess F', milles on k literaali.
check if $F \sim F' \in TAUT$.

Esitatu on olemuselt mittedeterministlik Turingi masin, oraakliga keelest $TAUT \in$ **coNP**. See inspireerib defineerima mittedeterministlikku Turingi taandamist.

Definitsioon. Keel L on *Turingi mõttes mittedeterministlikult polünomiaalselt taandatav* keelele A, kui leidub selline mittedeterministlik polünomiaalses ajas töötav oraakliga Turingi masin M, et $L = L(M^A)$. Tähistame $L \leqslant_{\mathrm{T}}^{\mathbf{NP}} A$.

Kui **K** on mingi keelte klass, siis kõiki keeli, mis on Turingi mõttes mittedeterministlikult polünomiaalselt taandatavad mõnele klassi **K** kuuluvale keelele, tähistame $\mathbf{NP}^{\mathbf{K}}$-ga. Samamoodi võime defineerida ka teised „relativiseeritud keerukusklassid“.

Teoreem 18.4. *Suvaliste keelte A ja B korral kehtivad:*

1. $\mathbf{P}^A \subseteq \mathbf{NP}^A$.
2. $A \in \mathbf{P}^A$ *ja seega* $A \in \mathbf{NP}^A$.
3. $\mathbf{P}^A = \mathbf{P}^{A^c}$ *ja* $\mathbf{NP}^A = \mathbf{NP}^{A^c}$.
4. $\mathbf{P}^A = co\mathbf{P}^A$.
5. $\mathbf{P}^A = \mathbf{P}^B$ *parajasti siis, kui* $A \leqslant_{\mathrm{T}}^{\mathbf{P}} B$ *ja* $B \leqslant_{\mathrm{T}}^{\mathbf{P}} A$.
6. $\mathbf{P}^A$ *ja* $\mathbf{NP}^A$ *on kinnised* $\leqslant_m^P$ *suhtes.*
7. $\mathbf{NP}^A \subseteq \mathbf{PSPACE}^A$.
8. *Kui* $A \in \mathbf{P}$, *siis* $\mathbf{P}^A = \mathbf{P}$.

Tõestus.

1. Iga deterministlikku (oraakliga) Turingi masinat võime vaadata kui mittedeterministlikku.

2. Olgu M masin, mis kopeerib oma sisendilindi sisu oraaklilindile, pöördub oraakli poole ja aktsepteerib positiivse ning lükkab tagasi negatiivse vastuse korral. Siis M on polünomiaalses ajas töötav ja deterministlik ja $A = L(M^A)$.

3. Olgu M polünomiaalses ajas töötav [mitte]deterministlik oraakliga Turingi masin ja $\overline{M}$ masin, mille saame masinast M, vahetades tal ära olekud q_{yes} ja q_{no}. Siis $\bar{M}$ on polünomiaalses ajas töötav [mitte]deterministlik oraakliga Turingi masin ja $L(M^A) = L(\overline{M}^{A^c})$.

4. Kui M on polünomiaalses ajas töötav deterministlik Turingi masin ja M' on saadud masinast M, vahetades tal ära aktsepteeriva ja tagasilükkava oleku, siis $L(M^A) = L({M'}^A)^c$.

5. Näitame, et kui $A \leqslant_{\mathrm{T}}^{\mathbf{P}} B$, siis $\mathbf{P}^A \subseteq \mathbf{P}^B$. Kui $L \in \mathbf{P}^A$, siis $L \leqslant_{\mathrm{T}}^{\mathbf{P}} A$. See tähendab, et $L \leqslant_{\mathrm{T}}^{\mathbf{P}} A \leqslant_{\mathrm{T}}^{\mathbf{P}} B$ ja järelikult teoreemi **??** järgi $L \leqslant_{\mathrm{T}}^{\mathbf{P}} B$ ehk $L \in \mathbf{P}^B$. Seega kui $A \leqslant_{\mathrm{T}}^{\mathbf{P}} B$ ja $B \leqslant_{\mathrm{T}}^{\mathbf{P}} A$, siis $\mathbf{P}^A = \mathbf{P}^B$. Teisest küljest, kui $\mathbf{P}^A = \mathbf{P}^B$, siis selle teoreemi teise punkti järgi $A \in \mathbf{P}^B$ ja $B \in \mathbf{P}^A$ ehk $A \leqslant_{\mathrm{T}}^{\mathbf{P}} B$ ja $B \leqslant_{\mathrm{T}}^{\mathbf{P}} A$.

6. Olgu $B \in \mathbf{P}^A$ [vastavalt $B \in \mathbf{NP}^A$]. Olgu M_0 selline [mitte]deterministlik polünomiaalses ajas töötav oraakliga Turingi masin, et $B = L(M_0^A)$. Olgu C mingi keel, mille korral $C \leqslant_m^P B$, olgu f selline polünomiaalses ajas arvutatav funktsioon, et

$x \in C \Leftrightarrow f(x) \in B$. Konstrueerime masina M, mis sisendi x korral arvutab kõigepealt $f(x)$-i ja seejärel simuleerib tulemusel masinat M_0. Siis on M [mitte]deterministlik polünomiaalses ajas töötav oraakliga Turingi masin ja $L(M^A) = C$.

7. Olgu $B \in \mathbf{NP}^A$, olgu M selline mittedeterministlik oraakliga Turingi masin, mille tööaeg on piiratud polünoomiga $p(n)$ ja $L(M^A) = B$. Kui $x \in \Sigma^*$, siis kontrollimaks, kas $x \in B$, piisab simuleerida masina M^A iga võimalikku arvutuskäiku $p(|x|)$ sammu, kuni on leitud aktsepteeriv arvutuskäik. Pidamaks arvet, millist arvutuskäiku parajasti simuleeritakse, piisab mälust suurusjärku $O(p(|x|))$. Kuna M^A konfiguratsioonid ei saa olla suuremad kui $O(p(|x|))$, siis on M^A simuleeritav deterministliku polünomiaalse mäluga oraakliga Turingi masinaga, mis kasutab oraaklit A.

8. Ilmselt $\mathbf{P}^{\emptyset} = \mathbf{P}$, kuna meil on oraakli vastused kõik ette teada. Kui $A \in \mathbf{P}$, siis vastavalt teoreemile 18.3 $A \leqslant_{\mathrm{T}}^{\mathbf{P}} \emptyset$ ja $\emptyset \leqslant_{\mathrm{T}}^{\mathbf{P}} A$, seega $\mathbf{P}^A = \mathbf{P}^{\emptyset} = \mathbf{P}$.

□

Teoreem 18.5. *Iga keelte klassi* $\mathbf{K}$ *korral* $\mathbf{NP}^{\mathbf{P}^{\mathbf{K}}} = \mathbf{NP}^{\mathbf{K}}$.

Tõestus. Kuna $\mathbf{K} \subseteq \mathbf{P}^{\mathbf{K}}$, siis $\mathbf{NP}^{\mathbf{P}^{\mathbf{K}}} \supseteq \mathbf{NP}^{\mathbf{K}}$. Teistpidi sisalduvuse tõestamiseks olgu $B \in \mathbf{NP}^{\mathbf{P}^{\mathbf{K}}}$. Sel juhul leidub selline keel $D \in \mathbf{P}^{\mathbf{K}}$, et $B \in \mathbf{NP}^D$. Järelikult leiduvad keel $A \in \mathbf{K}$, polünomiaalses ajas töötav deterministlik oraakliga Turingi masin M_1 ja polünomiaalses ajas töötav mittedeterministlik oraakliga Turingi masin M_2 nii, et $D = L(M_1^A)$ ja $B = L(M_2^D)$. Olgu M mittedeterministlik oraakliga Turingi masin, mis töötab järgmiselt:

input x
loop
  simuleeri masina M_2 arvutust sõnal x kuni ta
    peatub või pöördub oraakli poole
  if M_2 peatub **then**
    exit loop
  else
    olgu w sõna M_2 oraaklilindil
    simuleeri masina M_1 koos oraakliga A arvutust
      sõnal w
    if masin M_1 aktsepteerib w **then**
      võta masina M_2 olekuks q_{yes}

```
            else
                võta masina M₂ olekuks q_no
            fi
        fi
    end loop
    if masin M₂ aktsepteerib then accept else reject
    end
```

Kerge on näha, et M töötab polünomiaalses ajas ja $L(M^A) = B$. Seega $B \in \mathbf{NP}^{\mathbf{K}}$. $\square$

19. Polünomiaalne hierarhia

Selleks, et üldistada eelmise peatüki tulemusi, defineerime kolm keeleklasside peret: Σ_k^P, Π_k^P, Δ_k^P.

$$\Sigma_0^P = \Pi_0^P = \Delta_0^P = P,$$

$$\Sigma_{k+1}^P = NP^{\Sigma_k^P},$$

$$\Pi_{k+1}^P = co\Sigma_{k+1}^P,$$

$$\Delta_{k+1}^P = P^{\Sigma_k^P}$$

Defineerime polünomiaalse hierarhia

$$PH = \bigcup_{k=0}^{\infty} \Sigma_k^P.$$

Teoreemi 18.4 p.3 väidab, et on ükskõik, kas kasutada oraaklina keelt L või selle täiendit L^c. Seetõttu saame laiendada keerukusklasside määranguid:

$$\Sigma_{k+1}^P = NP^{\Sigma_k^P} = NP^{\Pi_k^P},$$

$$\Delta_{k+1}^P = P^{\Sigma_k^P} = P^{\Pi_k^P}.$$

Edaspidi kasutame tõestustes oraaklina keelt klassist Σ_k^P või Π_k^P vastavalt vajadusele.

Teoreem 19.1. $\Sigma_k^P \cup \Pi_k^P \subseteq \Delta_{k+1}^P \subseteq \Sigma_{k+1}^P \cap \Pi_{k+1}^P$.

Tõestus. Olgu $L \in \Sigma_k^P \cup \Pi_k^P$. Konstrueerime Turingi masina $M(x)$: **if** $x \in L$ **then** accept **else** reject **fi**.

M on deterministlik polünomiaalses ajas töötav Turingi masin oraakliga klassist Σ_k^P või Π_k^P, mis aktsepteerib keele L. Seega $L \in \Delta_{k+1}^P$.

Kui $A \in \Delta_{k+1}^P$, siis leidub deterministlik polünomiaalses ajas töötav oraakliga Turingi masin M ja keel $B \in \Sigma_k^P$ nii, et $A = L(M^B)$. Kuna deterministlik Turingi masin on erijuht mittedeterministlikust, siis $A \in \Sigma_{k+1}^P$. Kuna $\Delta_{k+1}^P = co\Delta_{k+1}^P$ (teoreem 18.4 p.4.), siis $A^c \in \Delta_{k+1}^P$, seega $A^c \in \Sigma_{k+1}^P$ ning $A \in \Pi_{k+1}^P$. $\square$

Teoreem 19.2. *(Polünomiaalse hierarhia alternatiivne definitsioon).*

(1) $L \in \Sigma_k^P \Leftrightarrow \exists A \in \mathbf{P}$ *ja polünoom* p*, nii et*

$$x \in L \Leftrightarrow (\exists y_1)(\forall y_2)...(Q_k y_k)[\langle x, y_1, ..., y_k \rangle \in A],$$

kus $|y_i| \leqslant p(|x|)$ $1 \leqslant i \leqslant k$.

(2) $L \in \Pi_k^P \Leftrightarrow \exists A \in \mathbf{P}$ *ja polünoom* p*, nii et*

$$x \in L \Leftrightarrow (\forall y_1)(\exists y_2)...(Q_k y_k)[\langle x, y_1, ..., y_k \rangle \in A],$$

kus $|y_i| \leqslant p(|x|)$ $1 \leqslant i \leqslant k$.

Tõestus. Tõestame induktsiooniga k järgi.

Baas: $k = 0$, kvantoreid pole, midagi pole tõestada.

Samm: $\Rightarrow$ (1) $L \in \Sigma_k^P$, definitsiooni kohaselt $\exists L_1 \in \Sigma_{k-1}^P$ ja mittedeterministlik Turingi masin NT oraakliga L_1, mis tuvastab keelt L polünomiaalses ajas. Olgu $q(n)$ polünomiaalne NT^{L_1} sammude arvu tõke, siis $NT^{L_1}(x)$ arvutustee on $q(|x|)^2$. Defineerime 6 keelt:

1) $\langle x, w \rangle \in A_1 \Leftrightarrow w$ on $NT^{L_1}(x)$ arvutustee, mis lõpeb olekus q_a.

2) $\langle u, v \rangle \in A_2 \Leftrightarrow u = \langle u_1, ..., u_{h_u} \rangle$, $v = \langle v_1, ..., v_{h_v} \rangle$ iga i, j jaoks $u_i \neq v_j$.

3) $\langle x, w, u \rangle \in A_3 \Leftrightarrow u = \langle u_1, ..., u_{h_u} \rangle$ on oraaklilt küsitavate sõnade jada, mille jaoks $NT^{L_1}(x)$ arvutustee w jätkub "yes" vastuse järgi.

4) $\langle x, w, v \rangle \in A_4 \Leftrightarrow v = \langle v_1, ..., v_{h_v} \rangle$ on oraaklilt küsitavate sõnade jada, mille jaoks $NT^{L_1}(x)$ arvutustee w jätkub "no" vastuse järgi.

5) $u \in A_5 \Leftrightarrow u = \langle u_1, ..., u_{h_u} \rangle$ ja $u_i \in L_1$ iga i jaoks.

6) $v \in A_6 \Leftrightarrow v = \langle v_1, ..., v_{h_v} \rangle$ ja $v_i \notin L_1$ iga i jaoks.

Keeled $A_1, ..., A_4$ on polünomiaalses ajas tuvastatavad. Koondame keelde A keelte $A_1, \ldots, A_4$ poolt väljendatavad tingimused:

$$\langle x, w, u, v \rangle \in A \;\Leftrightarrow$$

$$\Leftrightarrow \langle x, w \rangle \in A_1 \,\&\, \langle u, v \rangle \in A_2 \,\&\, \langle x, w, u \rangle \in A_3 \,\&\, \langle x, w, v \rangle \in A_4.$$

Siis

$$x \in L \Leftrightarrow (\exists \langle w, u, v \rangle)[\langle x, w, u, v \rangle \in A \,\&\, u \in A_5 \,\&\, v \in A_6].$$

Induktsiooni hüpoteesi kohaselt leidub keel $B_1 \in \mathbf{P}$ ja polünoom r_1, nii et

$$u \in A_5 \;\Leftrightarrow\; (\exists z_1)(\forall z_2)...(Q_{k-1}z_{k-1})[\langle u, z_1, z_2, ..., z_{k-1}\rangle \in B_1],$$

kus $|z_i| \leqslant r_1(|u|)$ $(1 \leqslant i \leqslant k-1)$. Sarnaselt leidub keel $B_2 \in \mathbf{P}$ ja polünoom r_2, nii et

$$v \in A_6 \;\Leftrightarrow\; (\forall w_2)(\exists w_3)...(Q_k w_k)[\langle v, w_2, w_3, ..., w_k\rangle \in B_2],$$

kus $|w_i| \leqslant r_2(|v|)$ $(2 \leqslant i \leqslant k)$. Kasutades kvantorite sulgude ette toomise reegleid, saame

$$x \in L \Leftrightarrow$$

$$\begin{aligned}
\Leftrightarrow\quad & (\exists\langle w, u, v\rangle)(\exists z_1)(\forall z_2)(\forall w_2)(\exists z_3)(\exists w_3)...(Q_{k-1}z_{k-1})(Q_k w_k) \\
& [\langle x, w, u, v\rangle \in A \;\&\; \langle u, z_1, ..., z_{k-1}\rangle \in B_1 \;\&\; \langle v, w_2, ..., w_k\rangle \in B_2] \\
\Leftrightarrow\quad & (\exists\langle w, u, v, z_1\rangle)(\forall\langle z_2, w_2\rangle)(\exists\langle z_3, w_3\rangle)...(Q_{k-1}\langle z_{k-1}, w_{k-1}\rangle)(Q_k w_k) \\
& [\langle x, w, u, v\rangle \in A \;\&\; \langle u, z_1, ..., z_{k-1}\rangle \in B_1 \;\&\; \langle v, w_2, ..., w_k\rangle \in B_2].
\end{aligned}$$

Teoreemi väide (2) tõestatakse analoogiliselt.

$\Leftarrow$ Oletame, et leidub keel $A \in \mathbf{P}$ ja polünoom p, nii et

$$x \in L \;\Leftrightarrow\; (\exists y_1)(\forall y_2)...(Q_k y_k)[\langle x, y_1, ..., y_k\rangle \in A],$$

kus $|y_i| \leqslant p(|x|)$ iga $1 \leqslant i \leqslant k$. Defineerime keele C:

$$C = \{\langle x, y_1\rangle : (\forall y_2)...(Q_k y_k)[\langle x, y_1, ..., y_k\rangle \in A]\},$$

siis $x \in L \Leftrightarrow (\exists y_1)[\langle x, y_1\rangle \in C]$. Hüpoteesi kohaselt $C \in \Pi^P_{k-1}$. Konstrueerime mittedeterministliku Turingi masina oraakliga C, mis tuvastab keele L:

input x;
Guess $y_1 : |y_1| \leqslant p(|x|)$
if $\langle x, y_1\rangle \in C$ **then accept**

$\square$

20. Polünomiaalse hierarhia omadused

Olgu $f(x_1, \ldots, x_n)$ Boole'i funktsioon ja f_{x_i} ning $f_{\overline{x}_i}$ $n-1$ muutuja funktsioonid, mis on saadud funktsioonist f muutuja x_i väärtuse fikseerimisel konstantidega 1 ja 0. Tähistame

$$(\exists x_i) f(x_1, \ldots, x_n) \equiv f_{x_i} \vee f_{\overline{x}_i},$$

$$(\forall x_i) f(x_1, \ldots, x_n) \equiv f_{x_i} \,\&\, f_{\overline{x}_i}.$$

Kvantorite tähendus on täpselt sama, mis predikaatarvutuses, sest Boole'i funktsioon on olemuselt predikaat kahe elemendilises indiviidide piirkonnas. Kui me kasutame kvantorploki mõjupiirkonnas olevate Boole'i funktsioonide esitamiseks lausearvutuse valemeid, on tegemist *kvantoritega Boole'i valemitega* ehk lühemalt *kvantorvalemiga*.

Kvantorvalem on *kinnine*, kui kõik muutujad on seotud kvantoritega. Kinnine kvantorvalem esitab 0 muutuja Boole'i funktsiooni, s.t. see on konstantne Boole'i funktsioon 0 või 1. Defineerime kaks keelt, mis on seotud kvantorvalemi mõistega.

QBF (Quantified Boolean Formula).

Antud: kvantorvalem $(Q_1 x_1) \ldots (Q_n x_n) F(x_1, \ldots, x_n)$.

Omadus: $(Q_1 x_1) \ldots (Q_n x_n) F(x_1, \ldots, x_n) \equiv 1$.

Boole'i valemi $F(x_1, \ldots, x_n)$ muutujate hulga $X = \{x_1, \ldots, x_n\}$ tükeldus $X_1, \ldots, X_k$ on hulga X alamhulkade süsteem, mille jaoks on täidetud tingimused:

1) $X_i \neq \emptyset, 1 \leqslant i \leqslant k$,

2) $X_i \cap X_j = \emptyset, 1 \leqslant i < j \leqslant k$,

3) $\cup_{i=1}^{k} X_i = X$.

Tükeldades muutujate hulga X k plokiks saame defineerida keele:

kQBF

Antud: kvantorvalem $(Q_1 X_1)(Q_2 X_2)...(Q_k X_k) F(x_1, ..., x_n)$, milles $X_1, ..., X_k$ on muutujate hulga $\{x_1, ..., x_n\}$ tükeldus ja $Q_1, ..., Q_k$ on vahelduv kvantorjada.

Omadus: $(Q_1 X_1)(Q_2 X_2)...(Q_k X_k) F(x_1, ..., x_n) \equiv 1$.

Näide $F = ((x \vee y) \to (\overline{x} \,\&\, y \,\&\, z \vee u))$. Koostame tõeväärtustabeli

x	y	z	u	$x \vee y$	$\to$	$\overline{x} \,\&\, y \,\&\, z$	$\vee u$
0	0	0	0	0	1	0	0
0	0	0	1	0	1	0	1
0	0	1	0	0	1	0	0
0	0	1	1	0	1	0	1
0	1	0	0	1	0	0	0
0	1	0	1	1	1	0	1
0	1	1	0	1	1	1	1
0	1	1	1	1	1	1	1
1	0	0	0	1	0	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	0	0	0
1	0	1	1	1	1	0	1
1	1	0	0	1	0	0	0
1	1	0	1	1	1	0	1
1	1	1	0	1	0	0	0
1	1	1	1	1	1	0	1

$G_1 = (\exists x)(\forall y)(\exists z, u)F(x, y, z, u) = 1,$
$G_2 = (\exists x)(\forall y, z)(\exists u)F(x, y, z, u) = 1,$
$G_3 = (\exists x)(\forall y, z, u)F(x, y, z, u) = 0.$

Teoreem 20.1. *$kQBF$, mis algab olemasolu kvantoriga on Σ_k^P-täielik (ja seega $kQBF^c$ on Π_k^P-täielik).*

Tõestus. Olgu $\vec{\alpha}_i \in \{0,1\}^{|X_i|}$ muutujate hulga X_i väärtustus $(1 \leqslant i \leqslant k)$. Siis keele $kQBF$ semantika kohaselt

$$(\exists X_1)(\forall X_2)...(Q_k X_k)F(X_1, ..., X_k) \in kQBF$$

siis ja ainult siis, kui

$$(\exists \vec{\alpha}_1)(\forall \vec{\alpha}_2) \dots (Q_k \vec{\alpha}_k)[\langle F(X_1, ..., X_k), \vec{\alpha}_1, \dots, \vec{\alpha}_k \rangle \in \\ \in BOOLEANVALUE].$$

Varasemast teame, et $BOOLEANVALUE \in \mathbf{P}$. Siis saame rakendada teoreemi 19.2 p.1. mille järgi kuulub keel $kQBF$ klassi Σ_k^P.

Olgu NT^{L_1} Turingi masin oraakliga $L_1 \in \Pi_{k-1}^P$. Teoreemi 19.2 tõestusest on näha, et iga NT^{L_1} poolt polünomiaalses ajas tuvastatav keel L on polünomiaalselt taandatav keelele $kQBF$, kuna ta teisendab $NT^{L_1}(x)$ arvutused keele $kQBF$ valemiteks. Keel $kQBF$ on seega Σ_k^P-täielik. $\square$

Teoreem 20.2. *Iga $k \geqslant 1$, keele $A \in \Sigma_k^P$ ja polünoomi q jaoks, keel $B = \{x : \exists y[\langle x, y\rangle \in A \,\&\, |y| \leqslant q(|x|)]\}$ kuulub klassi Σ_k^P.*

Tõestus. Kuna keel $A \in \Sigma_k^P$, siis teoreemi 19.2 pohjal leidub keel $L \in \Pi_{k-1}^P$ nii, et $\langle x, y\rangle \in A \Leftrightarrow (\exists z)[\langle x, y, z\rangle \in L]$. Sellest järeldub,et $x \in B \Leftrightarrow (\exists\langle y, z >)[\langle x, y, z\rangle \in L]$ ja $B \in \Sigma_k^P$. □

Teoreem 20.3. *Iga $k \geqslant 1$, keele $A \in \Pi_k^P$ ja polünoomi q jaoks, keel $B = \{x : \forall y[\langle x, y\rangle \in A \,\&\, |y| \leqslant q(|x|)]\}$ kuulub klassi Π_k^P.*

Tõestus. Kuna keel $A \in \Pi_k^P$, siis teoreemi 19.2 põhjal leidub keel $L \in \Sigma_{k-1}^P$ nii, et $\langle x, y\rangle \in A \Leftrightarrow (\forall z)[\langle x, y, z\rangle \in L]$. Sellest järeldub, et $x \in B \Leftrightarrow (\forall\langle y, z\rangle)[\langle x, y, z\rangle \in L]$ ja $B \in \Pi_k^P$. □

Teoreem 20.4. *Kui $\Sigma_k^P = \Pi_k^P$ mingi $k \geqslant 1$ jaoks, siis iga $m \geqslant k$ jaoks kehtib $\Sigma_m^P = \Pi_m^P = \Sigma_k^P$.*

Tõestus. Tõestame induktsiooniga m järgi.

Baas: $m = k$. $\Sigma_k^P = \Pi_k^P = \Sigma_k^P$.

Samm: Oletame, et $m > k$ ja $m - 1$ jaoks $\Sigma_{m-1}^P = \Pi_{m-1}^P = \Sigma_k^P$. Võtame keele $A \in \Sigma_m^P$, teoreemi 19.2 järgi $\exists B \in \Pi_{m-1}^P$ ja polünoom $p(n)$, selliselt et $x \in A \Leftrightarrow \exists y[\langle x, y\rangle \in B]$, kus $|y| \leqslant p(|x|)$. Induktsiooni hüpoteesi kohaselt $\Pi_{m-1}^P = \Pi_k^P$ ning eelduse põhjal $\Pi_k^P = \Sigma_k^P$. Seega $B \in \Sigma_k^P$. Nüüd saame teoreemi 20.2 kohaselt, et $A \in \Sigma_k^P$. □

21. Pehouseki teoreem

Olgu $f : \{0,1\}^n \to \{0,1\}$ Boole'i funktsioon ja

$$(Q_1 x_1)...(Q_n x_n) f(x_1, ..., x_n)$$

selle kinnine kvantorvorm, kus $Q_i \in \{\exists, \forall\}$. Olgu q_f funktsiooni f tõeste kvantorjadade hulk, s.t.

$$q_f = \{Q_1, \dots, Q_n : (Q_1 x_1) \dots (Q_n x_n)[f(x_1, \dots, x_n) \equiv 1]\}.$$

Järgmise teoreemi püstitas hüpoteesina Dan Pehousek uudisgrupis comp.sci Kohe järgnes tõestus Ranan Fraerilt. Ilmselt on tegemist juhuga, kus hüpoteesi püstitamine on raskem kui selle tõestamine. Tegelikult ei oska keegi leida nimetatud tulemusele mõistlikku rakendust, kuid tulemus on intrigeeriv. Isegi D. Knuth viitab sellele oma monograafia "The Art of Computer Programming"neljanda köite käsikirjas.

Teoreem 21.1. *Iga muutujate järjestuse $x_1, ..., x_n$ jaoks $\#q_f = \#f$, kus $\#q_f$ on hulga q_f võimsus ja $\#f$ funktsiooni f tõeste väärtustuste arv.*

Tõestus. Tõestame induktsiooniga n järgi.
Baas: $n = 1$. moodustame tõeväärtustabeli kõigi üheargumendiliste funktsioonide jaoks:

X	f_0	f_1	f_2	f_3
0	0	0	1	1
1	0	1	0	1

Siit on näha et

$\forall x f_0(x) = 0$ $\exists x f_0(x) = 0$ $\#q_{f_0} = 0$ $\#f_0 = 0$	$\forall x f_1(x) = 0$ $\exists x f_1(x) = 1$ $\#q_{f_1} = 1$ $\#f_1 = 1$
$\forall x f_2(x) = 0$ $\exists x f_2(x) = 1$ $\#q_{f_2} = 1$ $\#f_2 = 1$	$\forall x f_3(x) = 1$ $\exists x f_3(x) = 1$ $\#q_{f_3} = 2$ $\#f_3 = 2$

Samm: Oletame, et iga $n-1$ muutuja Boole'i funktsiooni jaoks teoreem kehtib. Olgu $f(x_1, ..., x_n)$ n muutujaga Boole'i funktsioon. Boole-Shannoni lahutuse järgi $f(x_1, ..., x_n) = (x_1 \& f_{x_1}) \vee (\overline{x}_1 \& f_{\overline{x}_1})$. Siis $\#f(x_1, ..., x_n) = \#f_{x_1} + \#f_{\overline{x}_1}$.

Olgu $Q_2, \dots, Q_n$ suvaline kvantorjada pikkusega $n-1$. Vaatleme ühe muutuja Boole'i funktsiooni $(Q_2 x_2) \dots (Q_n x_n) f(x_1, x_2, \dots, x_n)$.

Tähistame $f_1 = (Q_2x_2)\dots(Q_nx_n)f(1,x_2,\dots,x_n)$ ning $f_0 = (Q_2x_2)\dots(Q_nx_n)f(0,x_2,\dots,x_n)$. Rakendades Boole-Shannoni lahutust ainsa vaba muutuja x_1 järgi saame

$$(Q_2x_2)\dots(Q_nx_n)f(x_1,x_2,\dots,x_n) = (x_1\&f_1) \vee (\overline{x}_1\&f_0).$$

Uurime, milliste kvantori Q_1 valikute puhul on valem

$$(Q_1x_1)(Q_2x_2)\dots(Q_nx_n)f(x_1,x_2,\dots,x_n)$$

tõene. Käsitleda tuleb nelja võimalust:

1) $f_1 = 0$ ja $f_0 = 0$. Boole-Shannoni lahutusest on näha, et mõlemad valikud ($x_1 \leftarrow 0$ ja $x_1 \leftarrow 1$) annavad tulemuseks nulli. Seega ei saa niisugust kvantorjada jätkata nii, et

$$(Q_1x_1)(Q_2x_2)\dots(Q_nx_n)f(x_1,x_2,\dots,x_n)$$

oleks tõene.

2) $f_1 = 1$ ja $f_0 = 0$. Valides $x_1 \leftarrow 1$ muutub Boole-Shannoni lahutuse esimene term tõeseks. Valides $x_1 \leftarrow 0$ on mõlemad termid väärad. Kvantorjada saab jätkata olemasolu kvantoriga.

3) $f_1 = 0$ ja $f_0 = 1$. Analoogiline juhuga 2)

4) $f_1 = 1$ ja $f_0 = 1$. Valides $x_1 \leftarrow 1$ on tõene esimene term, valides $x_1 \leftarrow 0$ on tõene teine term. Seega iga x_1 väärtuse jaoks on $(Q_2x_2)\dots(Q_nx_n)f(x_1,x_2,\dots,x_n)$ tõene ning me saame kvantorjada jätkata nii olemasolu kui üldisuse kvantoriga.

Kokku saame $\#q_f = \#q_{f_{x_1}} + \#q_{f_{\overline{x}_1}}$. Induktsiooni hüpoteesi tõttu $\#q_{f_{x_1}} = \#f_{x_1}$ ja $\#q_{f_{\overline{x}_1}} = \#f_{\overline{x}_1}$. Arvestades, et

$$\#f(x_1,...,x_n) = \#f_{x_1} + \#f_{\overline{x}_1}$$

saame teoreemi väite. $\square$

Funktsiooni f tõeste kvantorjadade hulka q_f võib käsitleda Boole'i funktsioonina, kui kodeerida kvantorid hulga $\{0,1\}$ elementideks: $\exists \to 1, \forall \to 0$. Koodist kvantori saamiseks kasutame tähistust $Q^1 = \exists$ ja $Q^0 = \forall$. Boole'i funktsioone saab alati esitada lausearvutuse valemite abil. Olgu $F(x_1,\dots,x_n)$ lausearvutuse valem, mis esitab Boole'i funktsiooni f ning $Q_F(q_1,\dots,q_n)$ esitagu funktsiooni f tõeste kvantorjadade Boole'i funktsiooni q_f ülaltoodud kodeeringus. Valemi Q_F teadmine valemi F jaoks võimaldaks lahendada keele QBF liikmelisuse probleemi polünomiaalse ajaga valemi Q_F pikkusest, kuna on teada, et $BOOLEANVALUE \in \mathbf{P}$. Kuna keel QBF on täielik klassis **PSPACE**, siis ei saa tõsiselt loota, et õnnestuks konstrueerida polünomiaalne algoritm, mis arvutaks suvalise valemi F järgi valemi Q_F. Siiski, osutub et mõne valemite alamklassi jaoks on see võimalik.

Teoreem 21.2. *Iga Boole'i funktsiooni $f \in \{0,1\}^n \to \{0,1\}$ jaoks on tõeste kvantorjadade funktsioon q_f monotoonne.*

Tõestus. Olgu β selline et $q_f(\beta) = 1$ ja $\beta_i = 0$. Siis

$$(Q^{\beta_1}x_1)...(Q^{\beta_{i-1}}x_{i-1})(\forall x_i)(Q^{\beta_{i+1}}x_{i+1})...(Q^{\beta_n}x_n)f(x_1,...,x_n) = 1.$$

Tähistame $\alpha = (\beta_1, \ldots, \beta_{i-1}, 1, \beta_{i+1}, \ldots, \beta_n)$, saame seega $\alpha \prec \beta$. Kvantorjadale α vastav valem

$$(Q^{\beta_1}x_1)...(Q^{\beta_{i-1}}x_{i-1})(\exists x_i)(Q^{\beta_{i+1}}x_{i+1})...(Q^{\beta_n}x_n)f(x_1,...,x_n)$$

peab samuti olema tõene kvantorite semantika tõttu. Siis ka $q_f(\alpha) = 1$ ning funktsioon q_f on monotoonne. □

Teoreem 21.3. *Kui f on monotoonne, siis $q_f \equiv f$.*

Tõestus. Olgu $f(x_1, \ldots, x_n)$ monotoonne. Kui $f \equiv 0$, siis funktsioonil f ei ole ühtegi tõest väärtustust ega ka ühtegi tõest kvantorjada. Juhu $f \not\equiv 0$ tõestame induktsiooniga väärtustuste järjestuse $\prec$ järgi.

Baas: Kuna $f(1, \ldots, 1) = 1$, siis

$$(\exists x_1) \ldots (\exists x_n) f(x_1, \ldots, x_n) = 1.$$

See on aga kvantorjada, mis vastab tõesele väärtustusele $(1, \ldots, 1)$.

Samm: Olgu $\alpha = (\alpha_1, \ldots, \alpha_n)$ väärtustus, mille jaoks väide kehtib, s.t. $f(\alpha) = 1$ ja

$$(Q^{\alpha_1}x_1) \ldots (Q^{\alpha_n}x_n) f(x_1, \ldots, x_n) = 1.$$

Olgu β niisugune väärtustus, et $f(\beta) = 1$ ja $\beta \prec \alpha$, s.t. mingi i jaoks $(1 \leqslant i \leqslant n)$:

$$\alpha = (\alpha_1, \ldots, \alpha_{i-1}, 1, \alpha_{i+1}, \ldots, \alpha_n),$$

$$\beta = (\alpha_1, \ldots, \alpha_{i-1}, 0, \alpha_{i+1}, \ldots, \alpha_n).$$

Kuna $f(\alpha) = 1$ ja $f(\beta) = 1$, siis

$$(Q^{\alpha_1}x_1) \ldots, (Q^{\alpha_{i-1}}x_{i-1}), (Q^{\alpha_{i+1}}x_{i+1})(Q^{\alpha_n}x_n) f(x_1, \ldots, x_n) = 1$$

iga x_i jaoks, s.t.
$(Q^{\beta_1}x_1)\dots(Q^{\beta_n}x_n)f(x_1,\dots,x_n)=1.$ □

Selleks, et rakendada esitatud tulemusi keerukusteoorias, defineerime järgmise keele:

MONOTONE − QBF.
Antud: kvantoritega Boole'i valem

$$(Q_1x_1)\dots(Q_nx_n)F(x_1,\dots,x_n),$$

kus valem F on CNF, mis ei sisalda eitatud muutujaid.
Omadus: $(Q_1x_1)\dots(Q_nx_n)F(x_1,\dots,x_n)\equiv 1$.

Järeldus 21.4. *MONOTONE − QBF* $\in\mathbf{P}$.

Tõestus. Kodeerime kvantorjada $Q_1,\dots,Q_n$ väärtustuseks $q=(q_1,\dots,q_n)$ ja aktsepteerime, kui $F(q)=1$. Kuna ainult positiivseid literaale sisaldav CNF on monotoonne, siis teoreem 21.3 garanteerib tulemuse õigsuse. □

22. Interaktiivsed protokollid

Definitsioon Graafid $G_0 = (V, E_0)$ ja $G_1 = (V, E_1)$ on isomorfsed ($G_0 \sim G_1$), kui leidub permutatsioon π hulgal V, et $\pi(E_0) = E_1$. Siin $\pi(E_0) = \{\{\pi(x), \pi(y)\} : \{x, y\} \in E_0\}$.

Defineerime kaks graafide isomorfismiga seotud keelt.

GI (Graph Isomorphism)
Antud: graafid G_0 ja G_1.
Omadus: Graafid G_0 ja G_1 on isomorfsed.

SGI (Subgraph Isomorphism)
Antud: graafid G_0 ja G_1.
Omadus: Leidub graafi G_1 alamgraaf G', mis on isomorfne graafiga G_0.

Mõlemad keeled - *GI* ja *SGI* on klassist **NP**. Esitame mittedeterministliku tuvastamisalgoritmi keele *SGI* jaoks.

input graafid $G_0 = (V_0, E_0), G_1 = (V_1, E_1)$.
guess $V' \subseteq V_1$ ($|V'| = |V_0|$), vastavus $\pi : V_0 \to V'$
check if $G' = \pi(G_0)$

Keele *GI* kohta ei ole teada, et see oleks **NP**-täielik, ega ka selle kuulumist klassi **P**. Küll aga on keele *SGI* positsioon lihtsalt tõestatav.

Teoreem 22.1. *Keel SGI on* **NP***-täielik.*

Tõestus. Taandame keele *CLIQUE* keelele *SGI*. Tähistame l-tipulist täielikku graafi K_l. Ilmselt sisaldab graaf G_1 l-tipulist klikki parajasti siis, kui graafis G_1 on graafiga K_l isomorfne alamgraaf.

□

Graafide isomorfismi probleem võimaldab tagasi tulla keerukusklasside **NP** ja **coNP** juurde. Nende klasside põhimõtteline erinevus on selles, et keele $L \in$**NP** ja sõna x jaoks kehtib: kui $x \in L$, siis leidub polünomiaalses ajas testitav sertifikaat selle tõendamiseks. **NP**-probleemi keerukus seisneb sertifikaadi leidmises, mitte selle testimises. Keele $L \in$**coNP** liikmelisusel taolist sertifikaati ei ole (kui **NP**$\neq$**coNP**).

Näitena meenutame keelt SAT. Lausearvutuse valem $F(x_1, \ldots, x_n)$ kuulub keelde SAT, kui **leidub** väärtustus $\alpha = (\alpha_1, \ldots, \alpha_n)$ nii, et $F(\alpha_1, \ldots, \alpha_n) = 1$. Väärtustus α on siin sertifikaadiks. Kui aga $F(x_1, \ldots, x_n) \notin SAT$, siis taolist polünnomiaalses ajas testitavat sertifikaati ei ole teada.

Kujutleme kahte müstifitseeritud persooni: kuningas Artur – polünomiaalse arvutusvõimsusega umbusklik *verifitseerija* ning võlur Merlin – piiramatu arvutusvõimsusega *tõestaja*. Oletame, et Artur käsib Merlinil teha kindlaks, kas valem F on kehtestatav ning esitada ka veenev tõestus. Kui vastus on jaatav, on Merlini ülesanne lihtne – ta esitab tõestusena tõese väärtustuse α (mille leidmiseks piisab eksponentsiaalsest arvutusvõimsusest). Artur saab kontrollida polünomiaalse aja jooksul vastuse õigsust, arvutades $F(\alpha)$ väärtuse. Kui aga vastus on eitav, siis Artur ei ole võimeline kontrollima,et $F(\alpha) = 0$ kõigi 2^n väärtustuse jaoks.

Alates aastast 1989 on siiski teada võimalus kontrolliks: *interaktiivsed protokollid* (vt. [Bab90]). Kogu teooria on liiga komplitseeritus käesolevas õpikus esitamiseks, kuid idee selgitamiseks anname interaktiivse protokolli graafide mitteisomorfismi kontrolliks.

Artur: annab Merlinile kaks graafi $G_0 = (V, E_0)$ ja $G_1 = (V, E_1)$.
Merlin: teatab **NO** (s.t. graafid ei ole isomorfsed).
Artur: valib juhusliku $k \in \{0, 1\}$, genereerib juhusliku tippude hulga V permutatsiooni π, arvutab graafi $G = \pi(G_k)$ ja saadab selle Merlinile, nõudes vastuseks graafi numbrit k, millest lähtudes ta arvutas graafi G.
Merlin: if $G \sim G_0$ **then** $l := 0$ **else** $l := 1$ **fi**. Saadab Arturile arvu l.
Artur if $k = l$ **then** *accept* **else** *reject* **fi**.

Kui graafid ei ole isomorfsed, siis saam Merlin kontrollida, kumba graafi järgi leidis Artur graafi G ja Artur aktsepteerib tõenäosusega 1. Kui aga Merlin pettis ja graafid on isomorfsed, siis on Merlinil võimatu välja arvutada õiget vastust. Sel juhul lükkab Artur tagasi tõenäosusega $1/2$ ja aktsepteerib (vale otsus!) tõenäosusega $1/2$. Viimast tõenäosust saab teha kuitahes väikeseks, korrates protokolli. Tõepoolest, korrates protokolli n korda on tõenäosus selleks, et Merlin edastab Arturile õige k võrdne $1/2^n$.

Analoogilised interaktiivsed protokollid on teada kuni keerukusklassini **PSPACE**, kuid need on siintoodust oluliselt keerulisemad.

23. Ülesanded

Ülesanne 1. Leida lausearvutuse valemi

$$F = (x \oplus (y \oplus z)) \supset (\overline{x} \mathbin{\&} \overline{z})$$

jaoks funktsioonid F_x, $F_{\overline{x}}$, F_y, $F_{\overline{y}}$, F_z ja $F_{\overline{z}}$.

Ülesanne 2. Testida valemi F (ülesandest 1) kehtestatavust tabelimeetodil ja algoritmi SAT abil.

Ülesanne 3. Leida valemi F (ülesandest 1) disjunktiivne ja konjunktiivne normaalkuju.

Ülesanne 4. Antud on graaf $G = (V, E)$, kus $V = \{1, 2, 3, 4, 5, 6\}$ ja $E = \{\{1,3\}, \{1,5\}, \{2,4\}, \{2,6\}, \{3,4\}, \{3,5\}, \{3,6\}, \{4,5\}, \{4,6\}, \{5,6\}\}$. Leida graafi G klikkide struktuuri kirjeldavad DNF ja CNF.

Ülesanne 5. Tõestada teoreem 2.2.

Ülesanne 6. Kontrollida valemi

$$F_1 = (x \vee \overline{y}) \mathbin{\&} (y \vee \overline{z}) \mathbin{\&} (z \vee \overline{w}) \mathbin{\&} (w \vee \overline{u}) \mathbin{\&} (u \vee \overline{x})$$

kehtestatavust algoritmi DPLL abil.

Ülesanne 7. Tõestada teoreem 4.2.

Ülesanne 8. Tõestada lemma 4.3.

Ülesanne 9. Arvutada elimineerimismeetodi abil valemi F_1 (ülesandest 6) tõeste väärtustuste arv.

Ülesanne 10. Ortogonaliseerida algoritmi *Ort* abil disjunktipaar:

$$(x \vee \overline{y} \vee z \vee \overline{u}),$$

$$(\overline{u} \vee \overline{v} \vee w).$$

Ülesanne 11. Leida algoritmi *Ortogonalize* abil valemi F_1 (ülesandest 6) ortogonaalne konjunktiivne normaalkuju ja tõeste väärtustuste arv.

Ülesanne 12. Leida algoritmi SATO abil valemi F (ülesandest 1) ortogonaalne DNF ja tõeste väärtustuste arv.

Ülesanne 13. Leida algoritmi DPLLO abil valemi F_1 (ülesandest 6) ortogonaalne CNF ja tõeste väärtustuste arv.

Ülesanne 14. Arvutada algoritmi #DPLL abil valemi F_1 (ülesandest 6) tõeste väärtustuste arv.

Ülesanne 15. Tõestada teoreem 8.3.

Ülesanne 16. Tõestada järeldus 8.6.

Ülesanne 17. Koostada Moon-Moseri graafi $MM(3,4)$ klikkide struktuuri kirjeldav CNF.

Ülesanne 18. Koostada Moon-Moseri graafi M_4 klikkide struktuuri kirjeldav CNF CF_{M_4} ja rakendada sellele lahendite loendamise elimineerimismeetodit ja algoritme DPLLO ja #DPLL.

Ülesanne 19. Joonistada kõik 20 monotoonset kolme muutujaga Boole'i funktsiooni, värvides kolme moodustajaga Boole'i võrel funktsiooni 1-tipud punaseks.

Ülesanne 20. Joonistada kõik 20 kolme muutujaga antiahelat, värvides kolme moodustajaga Boole'i võrel funktsiooni 1-tipud punaseks.

Ülesanne 21. Kirjutada üles nelja muutujaga monotoonseid Boole'i funktsioone kirjeldav valem $M_4(x_0, \ldots, x_{15})$.

Ülesanne 22. Kirjutada üles nelja muutujaga antiahelaid kirjeldav valem $A_4(x_0, \ldots, x_{15})$.

Ülesanne 23. Aritmetiseerida lausearvutuse valem F (ülesandest 1), kasutades Kisielewiczi teoreemi tõestuses(teoreem 9.3) esitatud meetodit.

Ülesanne 24. Arvutada Kisielewiczi teoreemis kasutatud valemi abil arvu 27357 kaheksas kahendkoht (parempoolseim bitt on number 0). Kontrollida tulemust, teisendades arv kahendsüsteemi.

Ülesanne 25. Arvutada algoritmi #DPLL abil monotoonseid kolme muutuja funktsioone kirjeldava valemi M_3 tõeste väärtustuste arv.

Ülesanne 26. Teisendada CNF

$$F_2 = (x \vee y \vee u \vee v \vee z \vee w) \;\&\; (\overline{x} \vee \overline{z}) \;\&\; (\overline{x} \vee u \vee z \vee \overline{w}) \;\&\; (z)$$

ekvivalentseks 3CNF-ks.

Ülesanne 27. Leida valemi $x \oplus y \oplus u \oplus v$ minimaalne DNF ja CNF.

Ülesanne 28. Leida valemi F (ülesandest 1) jaoks samaaegselt kehtestatav konjunktiivne normaalkuju, kasutades polünomiaalse taandamise $SAT \leqslant_m^P CNF - SAT$ (teoreem 11.5) algoritmi.

Ülesanne 29. Leida graafi G (ülesandest 4) kõik klikid, sõltumatud hulgad ja tippude katted.

Ülesanne 30. Leida DNF , mis väljendab antud graafi sõltumatute hulkade struktuuri. Tõestada selle õigsust (analoog teoreemile 2.1).

Ülesanne 31. Leida CNF , mis väljendab antud graafi sõltumatute hulkade struktuuri. Tõestada selle õigsust (analoog teoreemile 2.2).

Ülesanne 32. Leida DNF , mis väljendab antud graafi tippude katete struktuuri. Tõestada selle õigsust (analoog teoreemile 2.1).

Ülesanne 33. Leida CNF , mis väljendab antud graafi tippude katete struktuuri. Tõestada selle õigsust (analoog teoreemile 2.2).

Ülesanne 34. Antud on 3CNF $F_3 = (\overline{x} \vee \overline{y} \vee \overline{z}) \& (x \vee \overline{y} \vee u) \& (y \vee z \vee \overline{u})$. Konstrueerida sellele vastav tippude katte graaf vastavalt taandamisele $3CNF - SAT \leqslant_m^P VERTEX - COVER$ teoreemist 12.3.

Ülesanne 35. Tõestada teoreem 12.4.

Ülesanne 36. Tõestada, et $\binom{n}{[n/2]}$ on eksponentsiaalses sõltuvuses arvust n. Soovitus: kasutage Stirlingi valemit $n! \sim \sqrt{2\pi n}\frac{n^n}{e^n}$.

Ülesanne 37. Tõestada, et valemi $F_G(x_{11}, \ldots, x_{kn})$ (teoreem 12.4) pikkus on polünomiaalses sõltuvuses graafi G tippude arvust.

Ülesanne 38. Rakendada teoreemi 12.4 konstruktsiooni graafile

$$G_1 = (\{1, 2, 3\}, \{\{1, 2\}, \{1, 3\}\}),$$

võttes $k = 2$.

Ülesanne 39. Tõestada, et disjunkt kujul

$$\overline{x_1} \vee \ldots \vee \overline{x_n} \vee y_1 \vee \ldots \vee y_k$$

on samaväärne valemiga

$$(x_1 \;\&\; \ldots \;\&\; x_n) \supset (y_1 \vee \ldots \vee y_k).$$

Ülesanne 40. Tõestada, et Cook-Levini teoreemis 13.1 punktides 1-8 konstrueeritud CNF pikkus on $O(p(n)^3)$.

Ülesanne 41. Konstrueerida graafi G_1 ja tippude katte konstandi $k = 2$ jaoks Hamiltoni tsükli graaf vastavalt teoreemi 14.1 tõestuses antud konstruktsioonile.

Ülesanne 42. Tõestada, et $2CNF{-}SAT$ ja $2{-}COLOURABILITY$ on polünomiaalselt ekvivalentsed.

Ülesanne 43. Konstrueerida valemi $(x \vee y \vee z) \;\&\; (x \vee \overline{y} \vee z)$ järgi kolmemõõtmeline sobitusülesanne vastavalt teoreemi 16.1 konstruktsioonile.

Ülesanne 44. Iteratsioonivaba regulaaravaldis (IRA) tähestikus $A = \{0, 1\}$ defineeritakse järgmiselt:

1° 0 ja 1 on IRA.

2° Kui a ja b on IRA-d, siis on IRA ka $(a \vee b)$ ja (ab).

Olgu r IRA. Keel $L(r)$ defineeritakse järgmiselt:

1° $L(0) = \{0\}$.

2° $L(1) = \{1\}$.

3° Kui a ja b on IRA, siis $L(a \vee b) = L(a) \cup L(b)$.

4° Kui a ja b on IRA, siis $L(ab) = \{xy : x \in L(a) \,\&\, y \in L(b)\}$.

Defineerime keele:

$nIRA$

Antud: IRA r ja naturaalarv n.

Omadus: $L(r) \cap A^n = A^n$, s.t. kas regulaaravaldis r genereerib kõik tähestiku A sõnad pikkusega n?

Tõestada, et keel $nIRA$ on **coNP**-täielik.

Soovitus: taandada keel $TAUT$ keelele $nIRA$.

Ülesanne 45. Arvutada kvantoritega Boole'i valemi

$$(\forall x)(\exists y)(\forall z)[x \supset (\overline{y} \oplus z)]$$

väärtus.

Ülesanne 46. Leida valemi

$$(x \vee y) \,\&\, (y \vee z) \,\&\, (z \vee u)$$

kõik tõesed kvantorjadad.

Kirjandus

[Bab90] L. Babai. E-mail and the unexpected power of interaction. In *Proceedings of the 5th Structure in Complexity Theory Conference*, pages 30–44. IEEE Computer Society Press, 1990.

[BC94] D. P. Bovet and P. Crescenzi. *Introduction to the Theory of Complexity.* Prentice Hall, 1994.

[BDG95] J. Balcazar, J. Diaz, and Gabarró. *Structural Complexity I, II.* Springer-Verlag, 1995.

[Chu40] R. Church. Numerical analysis of certain free distributive structures. *Duke Math. J.*, 6(3):732–734, 1940.

[Chu65] R. Church. Enumeration by rank of the elements of the free distributive lattice with seven generators. *Not. Amer. Math. So.*, 12:724, 1965.

[Coo71] S. Cook. The complexity of theorem-proving procedures. In *Proceedings of the 3rd ACM Symposium on Theory of Computing*, pages 151–158. ACM Press, May 1971.

[Ded97] R. Dedekind. Über zerlegungen von zahlen durch ihre grossten gemeinsamen teiler. *Festschrift der Technischen Hochschule zu Braunschweig bei Gelegenheit*, 69:1–40, 1897.

[DLL62] M Davis, G Logemann, and D Loveland. A machine program for theorem proving. *Comm. Assoc. Comput. Mach.*, 5(7):394–397, 1962.

[DP60] M. Davis and H. Putnam. A computing procedure for quantification theory. *J. Assoc. Comput. Mach.*, 7:201–215, 1960.

[Dub91] O. Dubois. Counting the number of solutions for instances of satisfiability. *Theoretical Computer Science*, 81:49–64, 1991.

[Gil54] E.N. Gilbert. Lattice theoretic properties of frontal switching functions. *J. Math. Phys.*, 33(1):57–67, 1954.

[GJ79] M. Garey and D. Johnson. *Computers and Intractability: a Guide to the Theory of NP-Completeness.* W. H. Freeman and Company, 1979.

[Han66] G. Hansel. Sur le nombre des fonctions boolennes monotones de n variables. *C.R. Acad. Sci Paris*, 262(20):1088–1090, 1966.

[HO02] L. A. Hemaspaandra and M.Õgihara. *The Complexity Theory Companion*. Springer-Verlag, 2002.

[Iwa89] K. Iwama. Cnf satisfiability test by counting and polynomial average time. *SIAM J. Comput.*, 18(2):385–391, 1989.

[Kis88] A Kisielewicz. A solution of dedekind's problem on the number of isotone boolean functions. *J. reine angew. Math.*, 386:139–144, 1988.

[Kle69] D. Kleitman. On dedekind's problem: The number of monotone boolean functions. *Proc. Amer. Math. Soc.*, 21:677–682, 1969.

[Kor65] V. K. Korobkov. On monotone functions in boolean algebra (in russian). *Problemy kibernetiki*, 13:5–28, 1965.

[Kor81] A. D. Korshunov. On the number of monotone boolean functions (in russian). *Problemy kibernetiki*, 38:5–108, 1981.

[Kul64] I Kull. *Matemaatiline loogika*. Eesti Riiklik Kirjastus, 1964.

[Lev75] L. Levin. Universal sequential search prolems. *Problems of Information Transmission*, 9(3):265–266, 1975. March 1975 translation into English of Russian article originally published in 1973.

[MM65] J.W. Moon and L. Moser. On cliques in graphs. *Israel J. Math.*, 3:23–28, 1965.

[Pap94] C. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[Pra96] V. Praust. *Keerukusteooria alused*. Ülikoolide Informaatikakeskus, 1996.

[Sip05] M.Šipser. *Introduction to the Theory of Computation*. Thomson Course Technology, 2005.

[TIT01] M. Tombak, A. Isotamm, and T. Tamme. On logical method for counting dedekind numbers. In *LNCS*, volume 2138, pages 424–427. Springer-Verlag, 2001.

[Tom93] M. Tombak. One more exponential algorithm for establishing satisfiability of propositional formula. In *Proceedings of the Third Symposium on Programming Languages and Software Tools*, pages 142–146, Tartu, 1993.

[War46] M. Ward. Note on the order of free distributive lattices. *Bull. Amer. Math. Soc.*, 52(5):423, 1946.

[Wie91] D. Wiedemann. A computation of the eighth dedekind number. *Order*, 8:5–6, 1991.

[Yam53] K. A. Yamamoto. A note on the order of free distributive lattices. Technical report, Kanazawa University, 1953.

[Yam54] K.A. Yamamoto. Logarithmic order of free distributive lattice. *J. Math. Soc. Japan*, 6(3-4):343–353, 1954.

[Yap87] C.K. Yap. *Theory of Complexity Classes*. Oxford University Press, 1987. to appear, available at ftp://cs.nyu.edu/pub/local/yap/complexity-bk/.

Indeks