Latvijas Universitāte

Datorikas fakultāte

Adamāra matricu atbalsta bibliotēka

Kvalifikācijas darbs

Autors:
Artūrs Bačkurs
St. apl. nr.: ab08364

Vadītājs:
Juris Smotrovs
Dr. dat.

Rīga, 2010

Anotācija

Darbā “Adamāra matricu atbalsta bibliotēka” tiek realizēta funkciju bibliotēka, kas ļauj konstruēt dažādu izmēru Adamāra matricas. Tiek atbalstītas 9 būtiski dažādas Adamāra matricu konstrukcijas metodes. Bibliotēka tika veidota C++ programmēšanas valodā.

Atslēgas vārdi: Adamāra matricas, Adamāra matricu konstrukcijas, C++

Abstract

In the work “Hadamard matrix support library” a function library is implemented that supports the construction of Hadamard matrices of various sizes. The library Supports 9 notably different Hadamard matrix construction methods. The library is implemented in the C++ programming language.

Key words: Hadamard matrices, Hadamard matrices’ constructions, C++

Saturs

Ievads

Adamāra matricas (Nosauktas Žaka Salomona Adamāra (Jacques Salomon Hadamard) vārdā.) ir svarīgs kombinatorisko dizainu paveids. Adamāra matricas to specifisko īpašību dēļ tiek izmantotas kļūdu labošanas algoritmos, signālu apstrādē, kriptogrāfijā, spektroskopijā.

Darbs tika veikts ar mērķi izveidot funkciju bibliotēku, kas ļautu konstruēt Adamāra matricas. Lai sasniegtu šo mērķi, tika veikti šādi uzdevumi:

- tika iepazītas galvenās Adamāra matricu konstrukcijas metodes,
- tika izstrādāta programmatūras prasību specifikācija,
- tika izstrādāts programmatūras projektējuma apraksts,
- tika izstrādāts programmprodukts,
- izstrādātais programmprodukts tika pakļauts testēšanai,
- tika izstrādāta pavadošā dokumentācija.

Funkciju bibliotēka tika rakstīta C++ programmēšanas valodā. Darbs tika veidots tā, lai to viegli varētu papildināt ar jaunām Adamāra konstrukcijas metodēm. Kods tika saprātīgi komentēts, lai to varētu viegli mainīt vajadzības gadījumā un atkārtoti izmantot.

1 Pamatdefinīcijas un teorēmas

1.1 Definīcijas

1. Iesākumfails (header file) ir fails, kurā parasti tiek definētas funkciju un klašu saskarnes, bet ne vienmēr šīs funkcijas un klases ir implementētas šajā failā.
2. Par matricu sauc taisnstūrī izvietotus skaitļus. Matricas A i-tās rindas j-tais elements ir $A_{i,j}$.
3. Par divu matricu A un B summu sauc matricu $C = A + B$, tādu, ka $C_{i,j} = A_{i,j} + B_{i,j}$.
4. Par matricas A reizinājumu ar skalāru c sauc matricu $B = cA$, tādu, ka $B_{i,j} = cA_{i,j}$.
5. Par matricas A transponēto matricu sauc matricu $B = A^T$, tādu, ka $B_{i,j} = A_{j,i}$.
6. Par matricas A un B reizinājumu sauc matricu $C = AB$, tādu, ka $C_{i,j} = A_{i,0}B_{0,j} + A_{i,1}B_{1,j} + A_{i,2}B_{2,j} + ... + A_{i,r}B_{1,r}$.
7. Par kvadrātveida matricu A un B Kronekera reizinājumu sauc kvadrātveida matricu $C = A \otimes B = \begin{bmatrix} A_{1,1}B & \cdots & A_{1,n}B \\ \vdots & \ddots & \vdots \\ A_{n,1}B & \cdots & A_{n,n}B \end{bmatrix}$.
8. Matricu I_n sauc par vienības matricu, ja tā ir kvadrātveida matrica ar izmēru n, kurai uz diagonāles ir 1, bet visur citur 0.
9. Matricu H sauc par Adamāra matricu, ja tā ir kvadrātveida, kuras elementi ir tikai 1 un -1, un tai izpildās $HH^T = nI_n$, kur n ir matricas H izmērs.
10. Adamāra matricu H sauc par antisimetrisku, ja tai izpildās $H_{i,j} = -H_{j,i}$ pie nosacījuma $i \neq j$.
11. Galīgs lauks F ir galīga kopa ar divām darbībām + un $\cdot$ tādām, ka izpildās:
    (a) Katriem diviem elementiem a un b no F izpildās, ka $a + b$ un $a \cdot b$ arī pieder F.
    (b) Katriem trim elementiem a un b, un c no F izpildās $(a+b)+c = a+(b+c)$ un $(a \cdot b) \cdot c = a \cdot (b \cdot c)$.
    (c) Katriem diviem elementiem a un b no F izpildās $a + b = b + a$ un $a \cdot b = b \cdot a$.
    (d) Laukā F eksistē tādi divi elementi 0 un 1, ka katram a no lauka F izpildās $a + 0 = a$ un $a \cdot 1 = a$.
    (e) Katram elementam a no lauka F eksistē tāds elements b, ka $a + b = 0$ un katram elementam $a \neq 0$ no lauka F eksistē tāds elements b, ka $a \cdot b = 1$.
    (f) Katriem trim elementiem a, b, c no lauka F izpildās $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$.
12. Par Jakobstāla (Jacobsthal) matricu sauc tādu kvadrātveida matricu A ar izmēru q, kurai $A_{i,j} = \chi(f_i - f_j)$, kur f_i ir galīga lauka F ar q elementiem i-tais elements un $\chi(f_i)$ pieņem vērtību 0, ja $f_i = 0$, vērtību 1, ja $f_i = k^2$ kādam nenulles elementam k no lauka F, vērtību -1 visos citos gadījumos. Funkciju χ sauc par kvadrātisko raksturu (Skat [6]).
13. Polinomu P pār lauku F (Ar koeficientiem no lauka F.) sauc par nereducējamu, ja to nevar izteikt kā divu citu polinomu pār šo pašu lauku reizinājumu, kuru pakāpes mazākas par polinoma P pakāpi.

1.2 Teorēmas

1. Adamāra matricas izmērs var būt 1 vai 2, vai skaitļa 4 daudzkārtnis [5]. Nav zināms (Tā ir neatrisināta problēma.), vai pretējais apgalvojuma virziens arī ir patiess, t.i., vai katram skaitļa 4 daudzkārtnim eksistē Adamāra matrica ar šādu izmēru.

2. Jebkurš galīgs lauks F ir izomorfisks (n-1)-ās kārtas polinomu kopai ar koeficientiem no kāda pirmskaitļa p atlikumu lauka K ar darbībām pēc nereducējama polinoma laukā K [6]. Skaitļus n un p sauc attiecīgi par lauka F kārtu un harakteristiku. Šo lauku apzīmē ar $GF(p^n)$.

2 Programmatūras prasību specifikācija

2.1 Ievads

2.1.1 Nolūks

Programmatūrs prasību specifikācija ir paredzēta izstrādājamās Adamāra atbalsta bibliotēkas prasību specifikācijai. Šis dokuments domāts bibliotēkas pasūtītājiem un izstrādātājiem, lai specificētu prasības attiecībā pret izstrādājamo programmproduktu.

2.1.2 Darbības sfēra

Šajā dokumentā tiek specificētas prasības Adamāra matricu atbalsta bibliotēkai "hadlib". Programmproduktam jāļauj lietotājam izvēlēties kādu no 9 dažādām Adamāra matricas konstrukcijas metodēm [5, 241 lpp] un konstruēt šai metodei atbilstošu matricu. Programmproduktam jābūt funkciju bibliotēkai, kuru lietotājs varēs iekļaut citos programmproduktos.

2.1.3 Saistība ar citiem dokumentiem

Programmatūras prasību specifikāciju ir ieteicams lietot ar kādu grāmatu par Adamāra matricām, piemēram, [5]. Šis dokuments tika izstrādāts saskaņā ar standartu LVS 68:1996 "Programmatūras prasību specifikācijas ceļvedis".

2.2 Vispārējs apraksts

2.2.1 Produkta perspektīva

Produktam jābūt izstrādātam kā funkciju bibliotēka Adamāra matricu konstruēšanai. Produkts ir paredzēts lietošanai kā citu produktu sastāvdaļa, kuru darbībai nepieciešamas Adamāra matricas.

2.2.2 Produkta funkcijas

Programmproduktam jārealizē visas funkcijas, kas nepieciešamas Adamāra matricu konstrukcijai. Tai skaitā tās ir funkcijas darbībām ar galīgā lauka elementiem (Saskaitīšana, reizināšana u.c.), funkcijas darbībām ar matricām (Saskaitīšana, reizināšana u.c.) un funkcijas, kas izmanto matricu darbības, lai konstruētu Adamāra matricas. Lietotājam jābūt pieejamām visām uzskaitītajām funkcijām.

2.2.3 Lietotāja raksturiezīmes

Programmprodukts paredzēts programmētājiem, kuri zin Adamāra matricu īpašības un kam vajadzīgas šīs matricas kādiem pielietojumiem. Programmprodukta saskarne ir ar C++ iesākumfailiem.

2.2.4 Pieņēmumi un atkarības

Tā kā programmproduktu ir paredzēts izmantot tikai zinātniskiem mērķiem, tad laiks un izmantotā atmiņa tiks ierobežota tikai asimptotiski attiecībā pret konstruējamās Adamāra matricas izmēru.

2.3 Funkcionālās prasības

2.3.1 Galīgā lauka ģenerēšana

Ievads Funkcijai jāģenerē laukā realizējamām darbībām nepieciešamā informācija. Tiek izmantota brīvi pieejama funkciju bibliotēka[1] (Turpmāk tekstā - NTL.), lai veiktu darbības ar polinomiem pār pirmskaitļu atlikumiem.
Ievade Tiek padota lauka harakteristika un kārta.
Apstrāde Izmantojot NTL, jātiek ģenerētam nereducējamam polinomam.
Izvade Nav.

2.3.2 Darbības galīgajā laukā

Ievads Funkcijām jārealizē galīgā lauka elementu saskaitīšanas, atņemšanas un veselā pakāpē kāpināšanas darbības.
Ievade Tiek padoti divi galīgā lauka elementi (Saskaitīšanas un atņemšanas darbībām.) vai viens elements un vesels skaitlis (Kāpināšanas darbībai.).
Apstrāde Izmantojot NTL, jātiek izpildītām nepieciešamām darbībām pār doto galīgo lauku.
Izvade Funkcijai jāatgriež galīgā lauka elements - attiecīgās darbības rezultāts.

2.3.3 Galīgā lauka elementa ģenerēšana no elementa numura

Ievads Funkcijām jāģenerē galīgā lauka elements tā, ka diviem dažādiem skaitļiem no fiksētas kopas (Atkarībā no galīgā lauka parametriem.) tiek izdoti dažādi galīgā lauka elementi.
Ievade Tiek padots vesels skaitlis - elementa numurs.
Apstrāde Izmantojot NTL, jātiek ģenerētam galīgā lauka elementam, kas atbilst padotajam numuram. Šī funkcija jārealizē vairākām dažādām Adamāra matricu konstruēšanā izmantojamajām numerācijām.
Izvade Funkcijai jāatgriež uzģenerēto galīgā lauka elementu.

2.3.4 Galīgā lauka elementa numura ģenerēšana

Ievads Funkcijai jāģenerē vesels skaitlis no galīgā lauka elementa, tāds, ka diviem dažādiem elementiem tiek ģenerēti dažādi skaitļi.
Ievade Tiek padots galīgā lauka elements.
Apstrāde Izmantojot NTL, jātiek ģenerētam veselam skaitlim, izmantojot galīgā lauka elementa pierakstu kā polinomu.
Izvade Funkcijai jāatgriež uzģenerētais skaitlis - elementa numurs.

2.3.5 Galīgā lauka elementa kvadrātiskā rakstura aprēķins

Ievads Funkcijai jāaprēķina, vai padotais galīgā lauka elements ir kāda cita elementa kvadrāts.
Ievade Tiek padots galīgā lauka elements.
Apstrāde Lai noskaidrotu, vai dotais elements ir cita elementa kvadrāts, jātiek aprēķinātai dotā elementa pakāpei, kura ir atkarīga no elementu skaita galīgajā laukā.
Izvade Funkcijai jāatgriež 0, ja dotais elements bija 0, jāatgriež 1, ja aprēķinātā pakāpe ir lauka vienības elements, jāatgriež -1 citos gadījumos.

2.3.6 Vesela skaitļa pakāpes aprēķins

Ievads Funkcijai jāaprēķina vesela skaitļa pakāpe.
Ievade Tiek padots skaitlis un pakāpe, kurā šis skaitlis jākāpina.
Apstrāde Jātiek aprēķinātai dotā skaitļa pakāpei.
Izvade Jātiek atgrieztai dotā skaitļa pakāpei.

2.3.7 Pārbaude, vai skaitlis ir nepāra pirmskaitlis

Ievads Funkcijai jāatgriež, vai dotais skaitlis ir pirmskaitlis.
Ievade Tiek padots vesels skaitlis
Apstrāde Jātiek veiktai pārbaudei, vai skaitlis ir nepāra pirmskaitlis. Šajā pārbaudē drīkst pārliecināties, vai dotais skaitlis dalās ar visiem potenciālajiem dalītājiem.
Izvade Jātiek atgrieztai bool vērtībai true, ja dotais skaitlis ir nepāra pirmskaitlis un false - pretējā gadījumā.

2.3.8 Kvadrātveida matricas ģenerēšana

Ievads Funkcijai jāģenerē kvadrātveida matrica.
Ievade Tiek padots vesels skaitlis - matricas izmērs.
Apstrāde Funkcijai jāģenerē kvadrātveida matrica ar doto izmēru.
Izvade Nav.

2.3.9 Matricas elementu vērtības nomaiņa

Ievads Funkcijai jāļauj nomainīt padotajā rindā un kolonnā esošā elementa vērtība.
Ievade Tiek padoti trīs veseli skaitļi - rindas un kolonnas numuri, un jaunā elementa vērtība.
Apstrāde Jātiek nomainītam elementam attiecīgajā rindā un kolonnā.
Izvade Nav.

2.3.10 Matricas elementa vērtības nolase

Ievads Funkcijai jāatgriež matricas elementa vērtība attiecīgajā rindā un kolonnā.
Ievade Tiek padoti divi skaitļi - rindas un kolonnas numuri.
Apstrāde Jātiek nolasītai elementa vērtībai attiecīgajā rindā un kolonnā.
Izvade Jātiek atgrieztai nolasītajai elementa vērtībai.

2.3.11 Matricas izmēra nolase

Ievads Funkcijai jāatgriež matricas izmērs.
Ievade Nav.
Apstrāde Jātiek nolasītam matricas izmēram.
Izvade Jātiek atgrieztam nolasītajam matricas izmēram.

2.3.12 Darbības ar matricām

Ievads Funkcijām jāveic darbības ar matricām, kuras nepieciešamas Adamāra matricu konstrukcijās.
Ievade Atkarībā no veicamās darbības, funkcijām var būt dažādi ievaddati. Funkcijai var tikt padota viena matrica (Transponētās matricas atgriešanai, matricas pirmās rindas un kolonnas dzēšanai, matricas diagonāles elementu vērtību uzstādīšanai par 0.), divas matricas (Matricu Kronekera reizinājuma aprēķinam, matricu reizinājuma aprēķinam, matricu summas aprēķinam), matrica un vesels skaitlis (Matricas ar skalāru lielumu reizinājuma aprēķinam.), matrica un divi veseli skaitļi (Jaunas pirmās rindas un pirmās kolonnas pievienošanai matricai un aizpildei ar dotām vērtībām.), četras matricas (Pirmo divu matricu Kronekera reizinājuma summas ar otro divu matricu Kronekera reizinājumu aprēķinam.).
Apstrāde Funkcijām jāaprēķina matricu darbību rezultāts. Tā kā matricu pamatdarbību apraksti tika doti definīcijās, tad šeit netiks dots detalizēts katras matricu darbības apstrādes apraksts.
Izvade Jātiek atgrieztai matricai - matricu darbības rezultātam.

2.3.13 Jaunu matricu veidošana

Ievads Funkcijām jāveic matricu veidošana, kuras nepieciešamas Adamāra matricu konstrukcijās.
Ievade Atkarība no veicamās konstrukcijas, funkcijām var būt dažādi ievaddati. Funkcijai var tikt padoti divi veseli skaitļi - galīgā lauka kārta un harakteristika (Jakobstāla matricas konstrukcijai.), viens vesels skaitlis (Vienības matricas konstrukcijai, matricas, kurai visi elementi ir vieninieki, konstrukcijai.).
Apstrāde Funkcijām jākonstruē matrica atkarībā no uzdotajiem parametriem. Jakobstāla matrica tiek konstruēta, izveidojot galīgo lauku (Izmantojot padoto lauka kārtu un harakteristiku.) un aizpildot Jakobstāla matricas elementus ar galīgā lauka elementu kvadrātiskajiem raksturiem (Skat. definīcijas.)
Izvade Jātiek atgrieztai matricai - konstrukcijas rezultātam.

2.3.14 Darbības ar skaitļiem formā $2^r \left(p_1^{k_1}+1\right)\left(p_2^{k_2}+1\right)...\left(p_n^{k_n}+1\right)$, kur $4 \vdots p_i^{k_i}+1$

Ievads Skaitļi šajā formā tiks saukti par formas ξ skaitļiem. Lai varētu konstruēt Adamāra matricas, nepieciešams uzglabāt skaitļus formā ξ. Funkcijām jāspēj noteikt, vai dotais skaitļa sadalījums reizinātājos atbilst formai ξ un jāvar aprēķināt šī sadalījuma vērtību.
Ievade Nav.
Apstrāde Funkcijai jāvar noteikt, vai dotais sadalījums atbilst formai ξ (Dotā sadalījuma esošie pirmskaitļi ir tiešām pirmskaitļi, un pirmskaitļu pakāpes ir vienādas ar 3 pēc moduļa 4.) kā arī jāspēj aprēķināt sadalījuma vērtību.
Izvade Ja dotais skaitlis atbilst formai ξ, tad funkcijai, kas nosaka sadalījuma korektumu, jāatgriež bool vērtība true, pretējā gadījumā - false. Funkcijai, kurai jāatgriež sadalījuma vērtību, jāatgriež vesels skaitlis - sadalījuma vērtība.

2.3.15 Darbības ar matricām, kas nepieciešamas Adamāra matricu konstrukcijās

Ievads Adamāra matricu konstrukcijās atbilstoši grāmatā [5] minētajām konstrukcijām, matricu veidošanas process bieži tiek sadalīts vairākos apakšgadījumos vai vienas Adamāra matricas konstrukcija tiek izmantota citas konstrukcijas pamatā. Tāpēc ir nepieciešams veidot funkcijas, kuras tiek izmantotas, lai risinātu šos apakšgadījumus.

Ievade Adamāra matricu konstrukciju apakšgadījumus risinošo funkciju ievaddatiem ir jābūt tikai trīs veidu - skaitļa sadalījums reizinātājos formā ξ (Skat. 2.3.14 funkcionālo prasību.), jau uzkonstruēta Adamāra matrica, naturāls skaitlis, kas visos gadījumos raksturo galīgā lauka harakteristiku vai kārtu.
Apstrāde Visu konstrukciju visu apakšgadījumu risinošajām funkcijām jākonstruē Adamāra matricas tieši tā kā tas ir darīts [5] aprakstītajos konstruktīvajos pierādījumos. Drīkst veidot pēc nepieciešamības lielu un saprātīgu skaitu šādu apakšgadījumu risinošo funkciju.
Izvade Katrai funkcijai jāatgriež Adamāra matricu - attiecīgā apakšgadījuma matricu.

2.3.16 9 Adamāra matricu konstrukciju realizējošās funkcijas

Ievads Izmantojot iepriekšējās funkcionālās prasības funkcijas, programmproduktā ir jārealizē visas grāmatā [5] minētās konstrukcijas. Katrai no 9 konstrukcijām ir jāveido sava funkcija.
Ievade Adamāra matricu konstrukciju risinošo funkciju ievaddatiem ir jābūt tikai trīs veidu - skaitļa sadalījums reizinātājos formā ξ (Skat. 2.3.14 funkcionālo prasību.), jau uzkonstruēta Adamāra matrica vai šādu matricu masīvs, naturāls skaitlis, kas visos gadījumos raksturo galīgā lauka harakteristiku vai kārtu.
Apstrāde Visu konstrukciju risinošajām funkcijām jākonstruē Adamāra matricas tieši tā kā tas ir darīts [5] aprakstītajos konstruktīvajos pierādījumos. Tādām funkcijām jābūt tieši 9 - katrai konstrukcijai viena.
Izvade Katrai funkcijai jāatgriež Adamāra matricu - attiecīgās konstrukcijas atbilstošo matricu.

2.3.17 Funkcijas, kas nodrošina ārējo saskarni

Ievads Lietotāja programmai jāvar izsaukt visas 9 iespējamās Adamāra matricu konstrukciju metodes pieļaujamos argumentu diapazonos. Katram ievaddatu komplektam katrai konstrukcijas metodei jāpārliecinās, ka ievaddati ir korekti.
Ievade Saskarnes nodrošinošajām un ievaddatu pareizības pārbaudes funkcijām ievaddatiem ir jābūt tikai trīs veidu - skaitļa sadalījums reizinātājos formā ξ (Skat. 2.3.14 funkcionālo prasību.), jau uzkonstruēta Adamāra matrica vai šādu matricu masīvs, naturāls skaitlis, kas visos gadījumos raksturo galīgā lauka harakteristiku vai kārtu.
Apstrāde Jāpārliecinās, ka ievaddati visos gadījumos ir korekti (Galīgā lauka harakteristika ir pirmskaitlis, harakteristika un kārta ir pozitīvs skaitlis, sadalījums reizinātājos formā ξ ir korekts, padotās Adamāra matricas ir tiešām Adamāra matricas.), jāizsauc iepriekšējās funkcionālās prasības funkcijas, lai konstruētu attiecīgo Adamāra matricu.
Izvade Katrai funkcijai jāatgriež ievaddatiem atbilstošu Adamāra matricu, ja ievaddati ir korekti, un matricu ar izmēru 0, ja ievaddati ir nekorekti.

2.4 Nefunkcionālās prasības

2.4.1 Veiktspējas prasības

Bibliotēkai ir jāspēj konstruēt visas katrai formai atbilstošās matricas, kuru izmēri nepārsniedz 1000. Visas matricas jāvar konstruēt laikā un telpā $O\left(n^k\right)$, kur k - kāds fiksēts skaitlis un n - konstruējamās matricas izmērs, t.i., matricas jāvar konstruēt polinomiālā laikā.

2.4.2 Ierobežojumi

1. Programmproduktam ir jākompilējas ar Microsoft Visual C++ kompilatoru.
2. Programmprodukta kodam jāatbilst ISO/IEC 14882:1998 standartam.

2.4.3 Ārējās saskarnes prasības

Programmprodukta kodam jābūt organizētam C++ koda iesākumfailos. Ārējai saskarnei jāatbilst šādiem nosacījumiem

1. Visai programmprodukta funkcionalitātei, kas varētu būt nepieciešama, lai konstruētu Adamāra matricas, jābūt pieejamai vārdu telpā "hadlib".
2. Bibliotēkā ir jārealizē klase "Hadamard", kurā ir jābūt pieejamām visām 9 Adamāra matricu konstrukcijām kā klases metodēm. Klasē jāglabā matrica, kurai ir jābūt aizpildītai ar Adamāra matricas lauku vērtībām pēc Adamāra matricas konstrukcijas metodes izsaukuma.

3 Programmas projektējuma apraksts

3.1 Ievads

3.1.1 Dokumenta nolūks

Programmas projektējuma apraksta nolūks ir parādīt, kā programmatūras prasību specifikācijā iekļautās prasības tiks realizētas izvēlētajā programmatūras realizācijas vidē. Dokuments domāts programmprodukta izstrādātājiem kā palīgs analīzē, plānošanā, implementēšanā un lēmumu pieņemšanā.

3.1.2 Darbības sfēra

Šajā dokumentā tiek specificētas prasības Adamāra atbalsta bibliotēkai “hadamard”. Programmproduktam jāļauj lietotājam izvēlēties kādu no 9 dažādām Adamāra matricas konstrukcijas metodēm [5, 241 lpp] un konstruēt šai metodei atbilstošu matricu. Programmproduktam jābūt funkciju bibliotēkai, kuru lietotājs varēs iekļaut citos programmproduktos.

3.1.3 Definīcijas

Visu lietoto terminu definīcijas dotas šī dokumenta pirmajā nodaļā.

3.1.4 Saistība ar citiem dokumentiem

Programmas projektējuma apraksts lietojams kopā programmatūras prasību specifikāciju. Šo dokumentu ieteicams lietot kopā ar kādu grāmatu par Adamāra matricām, piemēram, [5]. Šis dokuments tika izstrādāts saskaņā Latvija Valsts standartu LVS 72:1996 "Ieteicamā prakse programmatūras projektējuma aprakstīšanai".

3.2 Dekompozīcijas apraksts

3.2.1 Ievads

Programmproduktu ir plānots būvēt no šādām entītijām (moduļiem):

- entītijas `galois`, kurā tiks realizēta galīgā lauka klase un funkcijas darbam ar galīgā lauka elementiem.
- entītijas `matrix`, kurā tiks realizēta matricas klase un funkcijas darbam ar matricām.
- entītijas `forms`, kurā tiks realizētas visas nepieciešamās funkcijas Adamāra matricu konstrukcijām.
- entītijas `hadamard`, kurā tiks realizēta saskarne - Adamāra matricu klase ar metodēm Adamāra matricu konstruēšanai.

Programmprodukta paredzētā lielā apmēra dēļ (vairāk kā 50 funkcijas) tika nolemts entītiju dekompozīcijā un detalizētajā projektējumā aprakstīt tikai pašas svarīgākās funkcijas - moduļa "forms" funkcijas, kuras ģenerēs Adamāra matricas. Netiks detalizēti aprakstītas moduļu "galois", "matrix", "hadamard" funkcijas, kuras neveic programmprodukta galveno funkcionalitāti. Ja funkcijas aprakstā nav norādīta Adamāra matricas konstrukcijas metode, tad to metodi var atrast [5].

3.2.2 Moduļu dekompozīcijas projektējums

3.2.2.1 Modulis `galois`

Identificējums	Nolūks
`GaloisField`	Klase galīgā lauka informācijas (Nereducējamo polinomu, harakteristiku, identitātes elemenu, nulles elementu, harakteristiku, kārtu, elementu skaitu.) uzglabāšanai.
`GaloisField::GaloisField`	Konstruktors lauka informācijas uzstādīšanai.
`GaloisField::sum`	Funkcija divu elementu saskaitīšanai.
`GaloisField::subtraction`	Funkcija divu elementu atņemšanai.
`GaloisField::elementPower`	Funkcija elementa pakāpes aprēķināšanai.
`GaloisField::elementFromNumberSymmetric`	Funkcija elementa aprēķināšai no vesela skaitļa.
`GaloisField::elementFromNumber`	Funkcija elementa aprēķināšai no vesela skaitļa.
`GaloisField::numberFromElement`	Funkcija galīgā lauka elementa numura aprēķināšanai.
`GaloisField::elementCharacter`	Funkcija lauka elementa kvadrātiskā rakstura aprēķināšanai.
`pow`	Funkcija vesela skaitļa pakāpes aprēķinam.
`isOddPrime`	Funkcija pārbaudei, vai skaitlis ir pirmskaitlis.

3.2.2.2 Modulis `matrix`

Identificējums	Nolūks
`Matrix`	Klase matricas informācijas (Lauka elementu, matricas izmēra.) uzglabāšanai
`Matrix::Matrix`	Konstruktors lauku informācijas uzstādīšanai. (Dinamiskās atmiņas atbrīvošana matricas elementiem, matricas elementa uzglabāšanai.)
`Matrix::getSize`	Funkcija matricas izmēra atgriešanai.
`Matrix::set`	Funkcija matricas elementa vērtības uzstādīšanai.
`Matrix::get`	Funkcija matricas elementa vērtības nolasīšanai.
`kroneckerProduct`	Funkcija divu matricu Kronekera reizinājuma izpildīšanai.
`transpose`	Funkcija transponētās matricas aprēķinam.
`multiplicateMatrix`	Funkcija matricu reizinājuma aprēķinam.
`multiplicateScalar`	Funkcija matricas un vesela skaitļa reizinājuma aprēķinam.
`multiplicateScalarStatic`	Funkcija matricas pareizināšanai ar veselu skaitli.
`generateJacobsthalMatrix`	Funkcija Jakobstāla matricas aprēķinam.
`expandJacobsthalMatrix`	Funkcija Jaunas rindas un kolonnas pievienošanai Jakobstāla matricai.
`generateIdentityMatrix`	Funkcija vienības matricas aprēķinam.
`sumMatrix`	Funkcija matricu summas aprēķinam.
`sumKroneckerPlusKronecker`	Funkcija pirmās un otrās padotās matricas Kronekera reizinājuma summas ar trešās un ceturtās padotās matricas Kronekera reizinājumu aprēķinam.
`cutFirstRowColumn`	Funkcija pirmās matricas rindas un kolonnas dzēšanai.
`generateOneMatrix`	Funkcija matricas, kura aizpildīta tikai ar vieniniekiem, aprēķinam.
`fill0Diagonal`	Funkcija jaunas matricas aprēķinam, kurai uz diagonāles ir 0, bet visur citur tā ir vienāda ar padoto matricu.

3.2.2.3 Modulis `forms`

Identificējums	Nolūks
`Factorization`	Klase skaitļa sadalījuma reizinātājos (Skaitļa 2 pakāpes, pirmskaitļu masīva, pirmskaitļu pakāpju masīva.) uzglabāšanai.
`Factorization::isValid`	Funkcija pārbaudei, vai skaitļa sadalījums reizinātājos ir korekts. (Pirmskaitļu masīvā ir pirmskaitļi. Pirmskaitļi attiecīgajā pakāpē ir kongruenti ar 3 pēc moduļa 4.)
`Factorization::value`	Funkcija sadalījuma reizinātājos vērtības aprēķinam.
`symmetricHadPower2`	Funkcija Adamāra matricas konstrukcijai ar izmēru 2^n, izmantojot Kronekera reizinājumu.
`skewSymmetricHadPower2`	Adamāra matricas konstrukcija, izmantojot [2].
`skewSymmetricHadMod3`	Adamāra matricas konstrukcija, izmantojot Jakobstāla matricu.
`symmetricHadMod3`	Adamāra matricas konstrukcija, izmantojot `Lemmu 14.1.4` [5].
`skewSymmetricExpand`	Adamāra matricas konstrukcija, izmantojot `Lemmu 14.1.5` [5].
`symmetricHadFact`	Adamāra matricas konstrukcija, izmantojot `Lemmu 14.1.4` [5].
`skewSymmetricHadFact`	Adamāra matricas konstrukcija, izmantojot `Lemmu 14.1.6` [5].
`antiCommutativeHad`	Adamāra matricas konstrukcija, izmantojot `Lemmu 14.1.7` [5].
`commutativeHad`	Adamāra matricas konstrukcija, izmantojot `Lemmu 14.1.7` [5].
`form7HadMod1`	Adamāra matricas konstrukcija, izmantojot `Teorēmu 14.1.5` [5].
`form7HadMod3`	Adamāra matricas konstrukcija, izmantojot funkciju `form4`.
`form8HadMod1`	Adamāra matricas konstrukcija, izmantojot `Teorēmu 14.1.6` [5].
`form8HadMod3`	Adamāra matricas konstrukcija, izmantojot `Teorēmu 14.1.6` [5].

Identificējums	Nolūks
`form1`	Funkcija Adamāra matricas konstrukcijai ar izmēru 2^n, izmantojot funkciju `symmetricHadPower2`.
`form2`	Adamāra matricas konstrukcija, izmantojot funkciju `symmetricHadMod3`.
`form3`	Adamāra matricas konstrukcija, izmantojot `Teorēmu 14.1.2` [5].
`form4`	Adamāra matricas konstrukcija, izmantojot `Teorēmu 14.1.3` [5].
`form6`	Adamāra matricas konstrukcija, izmantojot `Teorēmu 14.1.4` [5].
`form7`	Adamāra matricas konstrukcija, izmantojot `Teorēmu 14.1.5` [5].
`form8`	Adamāra matricas konstrukcija, izmantojot `Teorēmu 14.1.8` [5].
`form9`	Adamāra matricas konstrukcija, izmantojot TTP-starpībkopas no [3].
`form10`	Adamāra matricas konstrukcija, veicot Kronekera reizinājumu visām padotajām matricām.

3.2.2.4 Modulis `hadamard`

Visas funkcijas `form1`, `form2`, `form3`, `form4`, `form6`, `form7`, `form8`, `form9`, `form10` pārbauda, vai lietotāja ievaddati ir korekti. Ja ievaddati ir korekti, tad tiek izveidota Adamāra matrica ar attiecīgo izmēru. Pretējā gadījumā tiek izveidota matrica ar izmēru 0. Lai pārliecinātos, vai tika veiksmīgi izveidota Adamāra matrica, pietiek pārliecināties, ka matricas izmērs ir vismaz 1.

Identificējums	Nolūks
`Hadamard`	Klase matricas glabāšanai un saskarnes ar lietotāju nodrošināšanai.
`Hadamard::Hadamard`	Klases konstruktors, kas izveido matricu ar izmēru 0.
`Hadamard::isHadamard`	Funkcija, kas ļauj pārbaudīt, vai klasē glabātā matrica ir Adamāra matrica.
`Hadamard::getSize`	Funkcija, kas atgriež matricas, kas tiek glabāta klasē, izmēru.
`Hadamard::printMatrix`	Funkcija, kas izvada matricu standarta izvadā (`cout`).
`Hadamard::printMatrixToFile`	Funkcija, kas izvada matricu datnē, kuras nosaukums tiek padots kā parametrs.
`Hadamard::form1`	Funkcija, kas pārbauda, vai lietotāja padotie ievaddati ir korekti (Skaitļa 2 pakāpe ir nenegatīvs skaitlis.) un izsauc funkciju attiecīgās Adamāra matricas konstrukcijai. Funkcija konstruē Adamāra matricu ar tipu 1 no [5].
`Hadamard::form2`	Funkcija, kas pārbauda, vai lietotāja padotie ievaddati ir korekti (Pirmskaitlis tiešām ir nepāra pirmskaitlis, pakāpe ir pozitīva un pirmskaitļa pakāpe ir kongruenta ar 3 pēc moduļa 4.) un izsauc funkciju attiecīgās Adamāra matricas konstrukcijai. Funkcija konstruē Adamāra matricu ar tipu 2 no [5].
`Hadamard::form3`	Funkcija, kas pārbauda, vai lietotāja padotie ievaddati ir korekti (Padotā matrica ir Adamāra matrica, tās izmērs ir vismaz 2, padotais pirmskaitlis ir nepāra pirmskaitlis un pakāpe ir pozitīvs skaitlis.) un izsauc funkciju attiecīgās Adamāra matricas konstrukcijai. Funkcija konstruē Adamāra matricu ar tipu 3 no [5].
`Hadamard::form4`	Funkcija, kas pārbauda, vai lietotāja padotie ievaddati ir korekti (Skaitļa sadalījums reizinātajos ir korekts.) un izsauc funkciju attiecīgās Adamāra matricas konstrukcijai. Funkcija konstruē Adamāra matricu ar tipu 4 no [5].

Identificējums	Nolūks
`Hadamard::form6`	Funkcija, kas pārbauda, vai lietotāja padotie ievaddati ir korekti (Skaitļa sadalījumi reizinātajos ir korekti un pirmajam sadalījumam atbilstošais skaitlis ir par 4 mazāks kā otrajam.) un izsauc funkciju attiecīgās Adamāra matricas konstrukcijai. Funkcija konstruē Adamāra matricu ar tipu 6 no [5].
`Hadamard::form7`	Funkcija, kas pārbauda, vai lietotāja padotie ievaddati ir korekti (Padotās abas matricas ir Adamāra matricas, to izmēri ir vismaz 2, pirmskaitlis ir nepāra pirmskaitlis un pakāpe ir pozitīva.) un izsauc funkciju attiecīgās Adamāra matricas konstrukcijai. Funkcija konstruē Adamāra matricu ar tipu 7 no [5].
`Hadamard::form8`	Funkcija, kas pārbauda, vai lietotāja padotie ievaddati ir korekti (Padotās abas matricas ir Adamāra matricas, to izmēri ir vismaz 2, abi pirmskaitļi ir nepāra pirmskaitļi un to pakāpes ir pozitīvas.) un izsauc funkciju attiecīgās Adamāra matricas konstrukcijai. Funkcija konstruē Adamāra matricu ar tipu 8 no [5].
`Hadamard::form9`	Funkcija, kas pārbauda, vai lietotāja padotie ievaddati ir korekti (Padotie abi pirmskaitļi tiešām ir pirmskaitļi, to pakāpes ir pozitīvas, pirmā pirmskaitļa pakāpe ir par 2 mazāka par otrā pirmskaitla pakāpi.) un izsauc funkciju attiecīgās Adamāra matricas konstrukcijai. Funkcija konstruē Adamāra matricu ar tipu 9 no [5].
`Hadamard::form10`	Funkcija, kas pārbauda, vai lietotāja padotie ievaddati ir korekti (Visas padotās matricas ir Adamāra matricas.) un izsauc funkciju attiecīgās Adamāra matricas konstrukcijai. Funkcija konstruē Adamāra matricu ar tipu 10 no [5].

3.2.3 Funkciju dekompozīcija

3.2.3.1 skewSymmetricHadPower2

Tips Funkcija
Nolūks Ģenerēt antisimetrisku Adamāra matricu, kuras izmērs ir skaitļa 2 pakāpe.
Funkcija Izmantojot Kronekera reizinājumu un simetrisku Adamāra matricu ar izmēru 2, tiek konstruēta antisimetriska Adamāra matrica tā, kā tas aprakstīts [2].

3.2.3.2 symmetricHadPower2

Tips Funkcija
Nolūks Ģenerēt simetrisku Adamāra matricu, kuras izmērs ir skaitļa 2 pakāpe.
Funkcija Izmantojot Kronekera reizinājumu un simetrisku Adamāra matricu ar izmēru 2, tiek konstruētas Adamāra matricas ar izmēru skaitļa 2 pakāpes.

3.2.3.3 skewSymmetricHadMod3

Tips Funkcija

Nolūks Ģenerēt antisimetrisku Adamāra matricu, kuras izmērs ir $4 \dot{:} p^k + 1$.
Funkcija Izmantojot Jakobstāla matricu (Tiek ģenerēta no padotajiem skaitļiem p un k.), tiek ģenerēta vajadzīgā antisimetriskā matrica.

3.2.3.4 symmetricHadMod3

Tips Funkcija

Nolūks Ģenerēt simetrisku Adamāra matricu, kuras izmērs ir $4 \dot{:} p^k + 1$.
Funkcija Izmantojot funkcijas `skewSymmetricHadMod3` ģenerēto antisimetrisko matricu, tiek izveidota simetriska matrica ar tādu pašu izmēru. Lai iegūtu vajadzīgo matricu, iegūtā antisimetriskā matrica tiek pareizināta ar vajadzīgu transformācijas matricu.

3.2.3.5 skewSymmetricExpand

Tips Funkcija
Nolūks Izmantojot padoto antisimetrisko matricu, kuras izmērs ir n, ģenerēt antisimetrisku matricu ar izmēru $n(p^k + 1)$, kur $4 \dot{:} p^k + 1$.
Funkcija Izmantojot `skewSymmetricHadMod3` uzģenerēto antisimetrisko matricu ar izmēru $p^k + 1$ un padoto antisimetrisko matricu, tiek iegūta vajadzīgā matrica.

3.2.3.6 symmetricHadFact

Tips Funkcija
Nolūks Ģenerēt simetrisku Adamāra matricu ar izmēru, kas atbilst padotajam skaitļa sadalījumam reizinātājos `fact`.
Funkcija Izmantojot funkcijas `symmetricHadMod3` un `symmetricHadPower2`, tiek ģenerēta Adamāra matricu ar izmēru, kas atbilst `fact`.

3.2.3.7 skewSymmetricHadFact

Tips Funkcija
Nolūks Ģenerēt antisimetrisku Adamāra matricu ar izmēru, kas atbilst padotajam skaitļa sadalījumam reizinātājos `fact`.
Funkcija Izmantojot funkcijas `skewSymmetricHadPower2` un `skewSymmetricExpand`, tiek ģenerēta Adamāra matricu ar izmēru, kas atbilst `fact`.

3.2.3.8 antiCommutativeHad

Tips Funkcija
Nolūks Ģenerēt tādu matricu A, kurai izpildās $AB^T = -BA^T$, kur B - padotā matrica.
Funkcija Izmantojot matricu pamatdarbības, tiek ģenerēta vajadzīgā matrica.

3.2.3.9 commutativeHad

Tips Funkcija
Nolūks Ģenerēt tādu matricu A, kurai izpildās $AB^T = BA^T$, kur B - padotā matrica.
Funkcija Izmantojot matricu pamatdarbības, tiek ģenerēta vajadzīgā matrica.

3.2.3.10 form7HadMod1

Tips Funkcija
Nolūks Ģenerēt Adamāra matricu ar izmēru $n_1 n_2 (p^k + 1) p^k$, kur n_1 un n_2 ir divu padoto Adamāra matricu izmēri un p, k ir padotie galīgā lauka parametri tādi, ka $4 \vdots p^k + 3$.
Funkcija Izmantojot funkciju `antiCommutativeHad`, tiek ģenerēta vajadzīgā Adamāra matrica.

3.2.3.11 form7HadMod3

Tips Funkcija
Nolūks Ģenerēt Adamāra matricu ar izmēru $n_1 n_2 (p^k + 1) p^k$, kur n_1 un n_2 ir divu padoto Adamāra matricu izmēri un p, k ir padotie galīgā lauka parametri tādi, ka $4 \vdots p^k + 1$.
Funkcija Izmantojot funkciju `form4`, tiek ģenerēta vajadzīgā Adamāra matrica.

3.2.3.12 form8HadMod1

Tips Funkcija
Nolūks Ģenerēt Adamāra matricu ar izmēru $n_1 n_2 (p_1^{k_1} + 1) p_2^{k_2}$, kur n_1 un n_2 ir divu padoto Adamāra matricu izmēri un p_1, p_2, k_1, k_2 ir padotie galīgo lauku parametri tādi, ka $4 \vdots p_1^{k_1} + 3$ un $4 \vdots p_2^{k_2} + 3$ un $p_2^{k_2} - p_1^{k_1} = 4$.
Funkcija Izmantojot funkcijas `antiCommutativeHad`, `commutativeHad`, tiek ģenerēta vajadzīgā Adamāra matrica.

3.2.3.13 form8HadMod3

Tips Funkcija
Nolūks Ģenerēt Adamāra matricu ar izmēru $n_1 n_2 (p_1^{k_1} + 1) p_2^{k_2}$, kur n_1 un n_2 ir divu padoto Adamāra matricu izmēri un p_1, p_2, k_1, k_2 ir padotie galīgo lauku parametri tādi, ka $4 \vdots p_1^{k_1} + 1$ un $4 \vdots p_2^{k_2} + 1$ un $p_2^{k_2} - p_1^{k_1} = 4$.
Funkcija Izmantojot funkciju `form6`, tiek ģenerēta vajadzīgā Adamāra matrica.

3.2.3.14 form1

Tips Funkcija
Nolūks Ģenerēt Adamāra matricu, kuras izmērs ir skaitļa 2 pakāpe.
Funkcija Izmantojot funkciju `skewSymmetricHadPower2`, tiek ģenerēta vajadzīgā Adamāra matrica.

3.2.3.15 form2

Tips Funkcija

Nolūks Ģenerēt Adamāra matricu ar izmēru $4 \dot{:} p^k + 1$, kur p un k ir padotie galīgā lauka parametri.
Funkcija Izmantojot funkciju `symmetricHadMod3`, tiek ģenerēta vajadzīgā Adamāra matrica.

3.2.3.16 form3

Tips Funkcija
Nolūks Ģenerēt Adamāra matricu ar izmēru $n(p^k + 1)$, kur n ir padotās Adamāra matricas izmērs un p, k ir padotie galīgā lauka parametri.

Funkcija Ja $4 \dot{:} p^k + 1$, tad, izmantojot funkciju `skewSymmetricHadMod3`, tiek ģenerēta vajadzīgā Adamāra matrica. Ja $4 \dot{:} p^k + 3$, tad Adamāra matrica tiek konstruēta, izmantojot matricu pamatdarbības.

3.2.3.17 form4

Tips Funkcija
Nolūks Ģenerēt Adamāra matricu ar izmēru $n(n-1)$, kur n ir skaitlis, kas atbilst padotajam skaitļa sadalījumam reizinātājos `fact`.
Funkcija Izmantojot funkciju `skewSymmetricHadFact`, tiek ģenerēta Adamāra matrica H, kurai tiek piereizināta matrica H', kas tika iegūta no matricas H veicot dažas matricu pamatdarbības kā tas aprakstīts [5].

3.2.3.18 form6

Tips Funkcija
Nolūks Ģenerēt Adamāra matricu ar izmēru $n(n+3)$, kur n ir skaitlis, kas atbilst padotajam skaitļa sadalījumam reizinātājos `fact1` un skaitlim n+4 atbilst padotais skaitļa sadalījums pirmreizinātājos `fact2`.
Funkcija Izmantojot funkcijas `symmetricHadFact` un `skewSymmetricHadFact` attiecīgi, tiek iegūtas Adamāra matricas H_1 un H_2 atbilstoši sadalījumiem pirmreizinātājos `fact2` un `fact1`. Izmantojot dažas matricu pamatdarbības, tiek iegūta vajadzīga Adamāra matrica.

3.2.3.19 form7

Tips Funkcija
Nolūks Ģenerēt Adamāra matricu ar izmēru $n_1 n_2 (p^k + 1) p^k$, kur n_1 un n_2 ir divu padoto Adamāra matricu izmēri un p, k ir padotie galīgā lauka parametri.

Funkcija Ja $4 \dot{:} p^k + 3$, tad tiek izmantota funkcija `form7HadMod1` Adamāra matricas konstruēšanai. Ja $4 \dot{:} p^k + 1$, tad tiek izmantota funkcija `form7HadMod3` Adamāra matricas konstruēšanai.

3.2.3.20 form8

Tips Funkcija
Nolūks Ģenerēt Adamāra matricu ar izmēru $n_1 n_2 (p_1^{k_1}+1) p_2^{k_2}$, kur n_1 un n_2 ir divu padoto Adamāra matricu izmēri un p_1, p_2, k_1, k_2 ir padotie galīgo lauku parametri tādi, ka $p_2^{k_2} - p_1^{k_1} = 4$.

Funkcija Ja $4 \dot{:} p_1^{k_1} + 3$, tad tiek izmantota funkcija `form8HadMod1` Adamāra matricas konstruēšanai.

Ja $4 \dot{:} p_1^{k_1} + 1$, tad tiek izmantota funkcija `form8HadMod3` Adamāra matricas konstruēšanai.

3.2.3.21 form9

Tips Funkcija
Nolūks Ģenerēt Adamāra matricu ar izmēru $q(q+2)+1$, kur $q = p_1^{k_1}$ un $q+2 = p_2^{k_2}$, kur p_1, k_1, p_2, k_2 ir padotie galīgo lauku parametri.
Funkcija Izmantojot [3] un [5], tiek aizpildīta Adamāra matricas pirmā rinda. Funkcija izmanto galīgā lauka lauka elementus un, atkarībā no galīgā lauka elementa kvadrātiskā rakstura, aizpilda Adamāra matricas pirmo rindiņu. Pārējās Adamāra matricas rindiņas tiek aizpildītas, cikliski pārbīdot pirmo matricas rindiņu.

3.2.3.22 form10

Tips Funkcija
Nolūks Ģenerēt Adamāra matricu ar $n_1 n_2 ... n_k$, kur n_i ir i-tās padotās Adamāra matricas izmērs.
Funkcija Sareizinot visas padotās matricas izmantojot Kronekera reizinājumu, tiek iegūta vajadzīgā Adamāra matrica.

3.2.3.23 Factorization

Tips klase
Nolūks Realizēt datu struktūru skaitļa sadalījuma reizinātājos uzglabāšanai.
Funkcija Uzglabāt skaitļa sadalījuma reizinātājos (Skaitļa 2 pakāpes, pirmskaitļu masīva, pirmskaitļu pakāpju masīva.) uzglabāšanai. Kā arī ļaut pārliecināties par sadalījuma reizinātājos korektumu un aprēķināt sadalījuma vērtību, izmantojot funkcijas `isValid` un `value`.

3.2.3.24 isValid

Tips funkcija
Nolūks Ļaut pārliecināties par skaitļa sadalījuma reizinātājos korektumu.
Funkcija Tiek pārbaudīts, vai pirmskaitļu masīva elementi ir pirmskaitļi. Pirmskaitļi attiecīgajā pakāpē ir kongruenti ar 3 pēc moduļa 4.

3.2.3.25 value

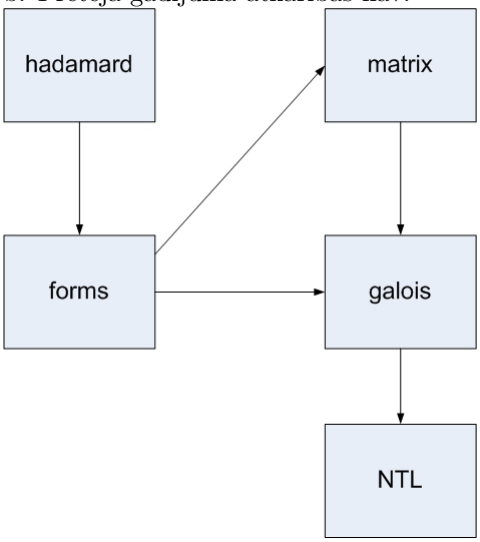
Tips funkcija
Nolūks Ļaut iegūt skaitļa sadalījuma reizinātājos vērtību.
Funkcija Tiek izrēķināta sadalījuma vērtība, sareizinot skaitļa 2 pakāpi ar par 1 palielinātu pirmskaitļu pakāpju reizinājumu.

3.3 Atkarību projektējums

Programmprodukta funkciju lielā skaita dēļ (Vairāk kā 50), atkarību projektējumā grafiski tiks parādītas tikai moduļu un darbā izmantotās NTL savstarpējās atkarības. Funkciju atkarību projektējums tiks noformēts tabulas veidā. Tabulas pirmajā kolonnā norādīts funkcijas nosaukums, kurai otrajā kolonnā tajā pašā rindā ir dots funkciju saraksts, kuras šī funkcija izsauc.

3.3.1 Moduļu atkarības

Ja ir norādīta moduļa a atkarība no moduļa b (Zīmējumā ir novilka bultiņa no moduļa a uz moduli b.), tad eksistē funkcija vai datu struktūra modulī a, kas izmanto funkciju vai datu struktūru no moduļa b. Pretējā gadījumā atkarības nav.



3.3.2 Funkciju atkarības

Funkcija	Atkarības
GaloisField::GaloisField	pow, GaloisField::elementFromNumber
GaloisField::sum	Nav
GaloisField::subtraction	Nav
GaloisField::elementPower	GaloisField::elementFromNumber
GaloisField::elementFromNumberSymmetric	Nav
GaloisField::elementFromNumber	Nav
GaloisField::numberFromElement	Nav
GaloisField::elementCharacter	GaloisField::elementPower
pow	Nav
isOddPrime	Nav
Matrix::Matrix	Nav
Matrix::getSize	Nav
Matrix::set	Nav
Matrix::get	Nav
Matrix::~Matrix	Nav
kroneckerProduct	Matrix::getSize, Matrix::get, Matrix::set
transpose	Matrix::getSize, Matrix::get, Matrix::set
multiplicateMatrix	Matrix::getSize, Matrix::get, Matrix::set
multiplicateScalar	Matrix::getSize, Matrix::get, Matrix::set
multiplicateScalarStatic	Matrix::getSize, Matrix::get, Matrix::set
generateJacobsthalMatrix	pow, GaloisField::elementFromNumberSymmetric, GaloisField::subtraction, GaloisField::elementCharacter, Matrix::set
expandJacobsthalMatrix	Matrix::getSize, Matrix::get, Matrix::set
generateIdentityMatrix	Matrix::set
sumMatrix	Matrix::getSize, Matrix::get, Matrix::set
sumKroneckerPlusKronecker	kroneckerProduct, sumMatrix
cutFirstRowColumn	Matrix::getSize, Matrix::get, Matrix::set
generateOneMatrix	Matrix::set
fill0Diagonal	Matrix::getSize, Matrix::get, Matrix::set
Factorization::isValid	pow, isOddPrime
Factorization::value	pow
symmetricHadPower2	kroneckerProduct, Matrix::set
skewSymmetricHadPower2	kroneckerProduct, Matrix::set, generateIdentityMatrix, sumMatrix
skewSymmetricHadMod3	generateJacobsthalMatrix, Matrix::getSize, expandJacobsthalMatrix, generateIdentityMatrix, sumMatrix
symmetricHadMod3	skewSymmetricHadMod3, Matrix::getSize, Matrix::set, multiplicateMatrix
skewSymmetricExpand	Matrix::getSize, generateIdentityMatrix, multiplicateScalar, sumMatrix, skewSymmetricHadMod3, symmetricHadMod3, sumKroneckerPlusKronecker
symmetricHadFact	symmetricHadPower2, symmetricHadMod3, kroneckerProduct

Funkcija	Atkarības
skewSymmetricHadFact	skewSymmetricHadPower2, skewSymmetricExpand
antiCommutativeHad	Matrix::getSize, Matrix::set, generateIdentityMatrix, kroneckerProduct, multiplicateMatrix
commutativeHad	Matrix::getSize, Matrix::set, generateIdentityMatrix, kroneckerProduct, multiplicateMatrix
form7HadMod1	antiCommutativeHad, generateJacobsthalMatrix, pow, generateIdentityMatrix, sumKroneckerPlusKronecker, generateOneMatrix, kroneckerProduct, expandJacobsthalMatrix
form7HadMod3	form4, kroneckerProduct
form8HadMod1	antiCommutativeHad, commutativeHad, generateJacobsthalMatrix, pow, generateIdentityMatrix, generateOneMatrix, sumKroneckerPlusKronecker, multiplicateScalarStatic, sumMatrix, kroneckerProduct, expandJacobsthalMatrix
form8HadMod3	form6, kroneckerProduct
form1	skewSymmetricHadPower2
form2	symmetricHadMod3
form3	pow, skewSymmetricHadMod3, kroneckerProduct, Matrix::getSize, Matrix::set, generateIdentityMatrix, kroneckerProduct, multiplicateMatrix, generateJacobsthalMatrix, expandJacobsthalMatrix, sumKroneckerPlusKronecker
form4	skewSymmetricHadFact, Matrix::getSize, Matrix::set, Matrix::get, fill0Diagonal, cutFirstRowColumn, generateIdentityMatrix, generateOneMatrix, sumKroneckerPlusKronecker
form6	symmetricHadFact, Matrix::getSize, Matrix::set, Matrix::get, skewSymmetricHadFact, multiplicateScalarStatic, fill0Diagonal, cutFirstRowColumn, generateIdentityMatrix, generateOneMatrix, sumMatrix, sumKroneckerPlusKronecker
form7	pow, form7HadMod1, form7HadMod3
form8	pow, form8HadMod1, form8HadMod3
form9	Matrix::getSize, Matrix::set, Matrix::get, pow, GaloisField::elementFromNumber, GaloisField::elementCharacter, GaloisField::sum
form10	kroneckerProduct, Matrix::set
Hadamard::Hadamard	Nav
Hadamard::~Hadamard	Nav
Hadamard::isHadamard	Matrix::getSize, transpose, multiplicateMatrix
Hadamard::getSize	Matrix::getSize
Hadamard::printMatrix	Matrix::getSize, Matrix::get
Hadamard::printMatrixToFile	Matrix::getSize, Matrix::get

Funkcija	Atkarības
Hadamard::form1	form1
Hadamard::form2	pow, isOddPrime, form2
Hadamard::form3	Hadamard::isHadamard, Hadamard::getSize, isOddPrime, form3
Hadamard::form4	form4, Factorization::isValid
Hadamard::form6	form6, Factorization::isValid
Hadamard::form7	Hadamard::isHadamard, Hadamard::getSize, isOddPrime, form7
Hadamard::form8	pow, Hadamard::isHadamard, Hadamard::getSize, isOddPrime, form8
Hadamard::form9	pow, isOddPrime, form9
Hadamard::form10	Hadamard::isHadamard, form10

3.4 Detalizētais projektējums

Tā kā programmprodukts ir domāts zinātniskiem mērķiem, tad vienīgā prasība pret ātrdarbību ir tas, lai visu funkciju laika sarežģītība būtu $O(n^k)$ kādam fiksētam k. Programmprodukta lielā izmēra dēļ tika nolemts veidot detalizētu projektējumu tikai tām entītijām, kas aprakstītas dokumenta sadaļā "Entītiju dekompozīcija". Tika izvēlēts funkciju saskarņu projektējumu un detalizēto projektējumu veidot vienotu.

3.4.1 `symmetricHadPower2`

Ievade Funkcijas ievaddati:

1. Vesels skaitlis `power` - skaitļa 2 pakāpe, kurā jākāpina 2, lai iegūtu iegūstamās Adamāra matricas izmēru.

Apstrāde Lai iegūtu simetrisku Adamāra matricu ar izmēru 2^{power}, vienības matricai ar izmēru 1, tiek piereizināta ar Kronekera reizinājuma palīdzību power reizes matrica $\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$.
Izvade Tiek izvadīta iegūtā simetriskā Adamāra matrica.

3.4.2 `skewSymmetricHadPower2`

Ievade Funkcijas ievaddati:

1. Vesels nenegatīvs skaitlis `power` - skaitļa 2 pakāpe, kurā jākāpina 2, lai iegūtu iegūstamās Adamāra matricas izmēru.

Apstrāde Lai iegūtu antisimetrisku Adamāra matricu ar izmēru 2^{power}, tiek power reizes darbināts cikls, kurā katrā iterācija no simetriskas un antisimetriskas Adamāra matricas ar izmēriem 2^i tiek iegūta simetriska un antisimetriska Adamāra matrica izmēriem 2^{i+1}.
Izvade Tiek izvadīta iegūtā antisimetriskā Adamāra matrica.

3.4.3 `skewSymmetricHadMod3`

Ievade Funkcijas ievaddati:

1. Nepāra pirmskaitlis `prime`, kas ir nepieciešams galīgā lauka konstruēšanai kā lauka harakteristika.

2. Pozitīvs vesels skaitlis `power`, kas nepieciešams galīgā lauka konstruēšanai kā lauka kārta.

Apstrāde Lai iegūtu antisimetrisku Adamāra matricu ar izmēru $4 \dot{:} prime^{power} + 1$, tiek aprēķināta Jakobstāla matrica pār lauku $GF(prime^{power})$, kurai pievieno rindu aizpildītu ar 1, kolonnu aizpildītu ar -1 un aizpilda diagonāles elementus ar 1.
Izvade Tiek izvadīta iegūtā antisimetriskā Adamāra matrica.

3.4.4 `skewSymmetricExpand`

Ievade Funkcijas ievaddati:

1. Antisimetriska Adamāra matrica `skewSymmetricHad`.
2. Nepāra pirmskaitlis `prime`, kas ir nepieciešams galīgā lauka konstruēšanai kā lauka harakteristika.
3. Pozitīvs vesels skaitlis `power`, kas nepieciešams galīgā lauka konstruēšanai kā lauka kārta.

Apstrāde Lai iegūtu antisimetrisku Adamāra matricu ar izmēru $(prime^{power} + 1)n$, kur n ir dotās Adamāra matricas izmērs, tiek aprēķināta simetriska un antisimetriska Adamāra matrica (Tiek izmantotas funkcijas `skewSymmetricHadMod3` un `symmetricHadMod3`.), kurām tiek pielietots Kronekera reizinājums attiecīgi ar matricām `skewSymmetricHad`, kurai uz diagonāles ir 0 un vienības matricu ar izmēru n un rezultāts tiek saskaitīts.
Izvade Tiek izvadīta iegūtā antisimetriskā Adamāra matrica.

3.4.5 `symmetricHadMod3`

Ievade Funkcijas ievaddati:

1. Nepāra pirmskaitlis `prime`, kas ir nepieciešams galīgā lauka konstruēšanai kā lauka harakteristika.
2. Pozitīvs vesels skaitlis `power`, kas nepieciešams galīgā lauka konstruēšanai kā lauka kārta.

Apstrāde Lai iegūtu simetrisku Adamāra matricu ar izmēru $4 \dot{:} prime^{power} + 1$, tiek aprēķināta antisimetriska Adamāra matrica, izmantojot funkciju `skewSymmetricHadMod3`, kurai tiek piereizināta tāda matrica, lai iegūtu simetrisku Adamāra matricu.
Izvade Tiek izvadīta iegūtā simetriskā Adamāra matrica.

3.4.6 `symmetricHadFact`

Ievade Funkcijas ievaddati:

1. Skaitļa sadalījums reizinātājos `fact`.

Apstrāde Lai iegūtu simetrisku Adamāra matricu, kuras izmērs atbilst `fact`, tiek konstruēta simetriska Adamāra matrica izmantojot `symmetricHadPower2`, kurai pielietojot Kronekera reizinājumu, tiek piereizinātas ar `symmetricHadMod3` ģenerētās simetriskās Adamāra matricas, atbilstošās `fact`.
Izvade Tiek izvadīta iegūtā simetriskā Adamāra matrica.

3.4.7 `skewSymmetricHadFact`

Ievade Funkcijas ievaddati:

1. Skaitļa sadalījums reizinātājos `fact`.

Apstrāde Lai iegūtu simetrisku Adamāra matricu, kuras izmērs atbilst `fact`, tiek konstruēta simetriska Adamāra matrica izmantojot `symmetricHadPower2`, kurai pielietojot `skewSymmetricExpand`, tiek ģenerēta Adamāra matrica atbilstoša `fact`.
Izvade Tiek izvadīta iegūtā antisimetriskā Adamāra matrica.

3.4.8 `antiCommutativeHad`

Ievade Funkcijas ievaddati:

1. Adamāra matrica hadMatrix.

Apstrāde Lai iegūtu Adamāra matricu B tādu, ka $hadMatrix \cdot B^T = -B \cdot hadMatrix^T$, tiek ģenerēta bloku matrica ar blokiem $\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$ uz galvenās diagonāles (Visur citur 0.), kura tiek piereizināta dotajai matricai kreisajā pusē.
Izvade Tiek izvadīta iegūtā Adamāra matrica.

3.4.9 `commutativeHad`

Ievade Funkcijas ievaddati:

1. Adamāra matrica hadMatrix.

Apstrāde Lai iegūtu Adamāra matricu B tādu, ka $hadMatrix \cdot B^T = B \cdot hadMatrix^T$, tiek ģenerēta bloku matrica ar blokiem $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ uz galvenās diagonāles (Visur citur 0.), kura tiek piereizināta dotajai matricai no kreisās puses.
Izvade Tiek izvadīta iegūtā Adamāra matrica.

3.4.10 `form7HadMod1`

Ievade Funkcijas ievaddati:

1. Adamāra matricas `hadMatrix1` un `hadMatrix2`.
2. Nepāra pirmskaitlis `prime`, kas ir nepieciešams galīgā lauka konstruēšanai kā lauka harakteristika.
3. Pozitīvs vesels skaitlis `power`, kas nepieciešams galīgā lauka konstruēšanai kā lauka kārta.

Apstrāde Lai iegūtu Adamāra matricu ar izmēru $n_1 n_2 (p^k + 1) p^k$, kur n_1 un n_2 ir divu padoto Adamāra matricu izmēri un $p = prime$, $k = power$, tiek pielietota funkcija `antiCommutativeHad` abām padotajām matricām, tiek iegūta Jakobstāla matrica un, veicot matricu pamatdarbības kā tas aprakstīts [5], tiek iegūta vajadzīga Adamāra matrica.
Izvade Tiek izvadīta iegūtā Adamāra matrica.

3.4.11 `form7HadMod3`

Ievade Funkcijas ievaddati:

1. Adamāra matricas `hadMatrix1` un `hadMatrix2`.
2. Nepāra pirmskaitlis `prime`, kas ir nepieciešams galīgā lauka konstruēšanai kā lauka harakteristika.
3. Pozitīvs vesels skaitlis `power`, kas nepieciešams galīgā lauka konstruēšanai kā lauka kārta.

Apstrāde Lai iegūtu Adamāra matricu ar izmēru $n_1 n_2 (p^k+1)p^k$, kur n_1 un n_2 ir divu padoto Adamāra matricu izmēri un $p = prime$, $k = power$, tiek pielietota funkcija `form4`, lai iegūtu Adamāra matricu ar izmēru $(p^k + 1)p^k$, kura tiek piereizināta ar Kronekera reizinājuma palīdzību abām dotajām Adamāra matricām.
Izvade Tiek izvadīta iegūtā Adamāra matrica.

3.4.12 `form8HadMod1`

Ievade Funkcijas ievaddati:

1. Adamāra matricas `hadMatrix1` un `hadMatrix2`.
2. Nepāra pirmskaitļi `prime1` un `prime2`, kas ir nepieciešami galīgo lauku konstruēšanai kā lauku harakteristikas.
3. Pozitīvi veseli skaitļi `power1` un `power2`, kas nepieciešami galīgā lauka konstruēšanai kā lauka kārtas.

Apstrāde Lai iegūtu Adamāra matricu ar izmēru $n_1 n_2 (p_2^{k_2} + 1)p_1^{k_1}$, kur n_1 un n_2 ir divu padoto Adamāra matricu izmēri un $p_2 = prime2$, $k_2 = power2$, $p_1 = prime1$, $k_1 = power1$ un izpildās $p_1^{k_1} - p_2^{k_2} = 4$, tiek pielietota funkcija `antiCommutativeHad` abām padotajām Adamāra matricām un funkcija `commutativeHad` matricai `hadMatrix1,` kā arī tiek ģenerētas Jakobstāla matricas pār laukiem $GF(p_1^{k_1})$ un $GF(p_2^{k_2})$. Pēc tam tiek veiktas elementāras darbības ar iegūtajām matricām kā tas aprakstīts [5], lai iegūtu vajadzīgo Adamāra matricu.
Izvade Tiek izvadīta iegūtā Adamāra matrica.

3.4.13 `form8HadMod3`

Ievade Funkcijas ievaddati:

1. Adamāra matricas `hadMatrix1` un `hadMatrix2`.
2. Nepāra pirmskaitļi `prime1` un `prime2`, kas ir nepieciešami galīgo lauku konstruēšanai kā lauku harakteristikas.
3. Pozitīvi veseli skaitļi `power1` un `power2`, kas nepieciešami galīgā lauka konstruēšanai kā lauka kārtas.

Apstrāde Lai iegūtu Adamāra matricu ar izmēru $n_1 n_2 (p_2^{k_2} + 1)p_1^{k_1}$, kur n_1 un n_2 ir divu padoto Adamāra matricu izmēri un $p_2 = prime2$, $k_2 = power2$, $p_1 = prime1$, $k_1 = power1$ un izpildās $p_1^{k_1} - p_2^{k_2} = 4$, tiek pielietota funkcija `form6`, lai iegūtu Adamāra matricu ar izmēru $(p_2^{k_2} + 1)p_1^{k_1}$, kura tiek piereizināta matricām `hadMatrix1` un `hadMatrix2` izmantojot Kronekera reizinājumu, lai iegūtu vajadzīgo Adamāra matricu.
Izvade Tiek izvadīta iegūtā Adamāra matrica.

3.4.14 `form1`

Ievade Funkcijas ievaddati:

1. Vesels skaitlis `power` - skaitļa 2 pakāpe, kurā jākāpina 2, lai iegūtu iegūstamās Adamāra matricas izmēru.

Apstrāde Lai iegūtu Adamāra matricu ar izmēru 2^{power}, tiek izsaukta funkcija `skewSymmetricHadPower2`.
Izvade Tiek izvadīta iegūtā Adamāra matrica.

3.4.15 `form2`

Ievade Funkcijas ievaddati:

1. Nepāra pirmskaitlis `prime`, kas ir nepieciešams galīgā lauka konstruēšanai kā lauka harakteristika.
2. Pozitīvs vesels skaitlis `power`, kas nepieciešams galīgā lauka konstruēšanai kā lauka kārta.

Apstrāde Lai iegūtu Adamāra matricu ar izmēru $prime^{power}+1$, tiek izsaukta funkcija `symmetricHadMod3`.
Izvade Tiek izvadīta iegūtā Adamāra matrica.

3.4.16 `form3`

Ievade Funkcijas ievaddati:

1. Antisimetriska Adamāra matrica `hadMatrix`.
2. Nepāra pirmskaitlis `prime`, kas ir nepieciešams galīgā lauka konstruēšanai kā lauka harakteristika.
3. Pozitīvs vesels skaitlis `power`, kas nepieciešams galīgā lauka konstruēšanai kā lauka kārta.

Apstrāde Gadījumā, ja $4 \vdots prime^{power} + 1$, tiek iegūta šāda izmēra Adamāra matrica izmantojot funkcijas `skewSymmetricHadMod3`, kura tiek piereizināta matricai `hadMatrix,` izmantojot Kronekera reizinājumu. Gadījumā, ja $4 \vdots prime^{power} + 3$, tiek pielietotas elementāras matricu darbības kā tas aprakstīts [5], lai iegūtu vajadzīgo matricu.
Izvade Tiek izvadīta iegūtā Adamāra matrica.

3.4.17 `form4`

Ievade Funkcijas ievaddati:

1. Skaitļa sadalījums reizinātājos `fact`.

Apstrāde Izmantojot funkciju `skewSymmetricHadFact`, tiek iegūta antisimetriska Adamāra matrica ar izmēru n, kur n ir skaitlis, kas atbilst sadalījumam fact. Tad tiek veiktas elementāras matricu darbības kā tas aprakstīts [5], lai iegūtu Adamāra matricu ar izmēru $n(n-1)$.
Izvade Tiek izvadīta iegūtā Adamāra matrica.

3.4.18 `form6`

Ievade Funkcijas ievaddati:

1. Skaitļa sadalījumi reizinātājos `fact1` un `fact2`.

Apstrāde Lai iegūtu Adamāra matricu ar izmēru $n(n+3)$, kur skaitlis n atbilst `fact1` un $n+4$ atbilst `fact2`, tiek pielietota funkcija `symmetricHadFact` sadalījumam `fact2`, lai iegūtu simetrisku Adamāra matricu, un tiek pielietota funkcija `skewSymmetricHadFact` sadalījumam `fact1`, lai iegūtu antisimetrisku Adamāra matricu. Tad tiek veiktas elementāras matricu darbības kā tas aprakstīts [5], lai iegūtu vajadzīgo Adamāra matricu.
Izvade Tiek izvadīta iegūtā Adamāra matrica.

3.4.19 `form7`

Ievade Funkcijas ievaddati:

1. Adamāra matricas `hadMatrix1` un `hadMatrix2`.
2. Nepāra pirmskaitlis `prime`, kas ir nepieciešams galīgā lauka konstruēšanai kā lauka harakteristika.
3. Pozitīvs vesels skaitlis `power`, kas nepieciešams galīgā lauka konstruēšanai kā lauka kārta.

Apstrāde Lai iegūtu Adamāra matricu ar izmēru $n_1 n_2(p^k+1)p^k$, kur n_1 un n_2 ir divu padoto Adamāra matricu izmēri un $p = prime$, $k = power$, tiek pielietota funkcija `form7HadMod1` gadījumā, ja $4 \dot{:} p^k + 3$ un funkcija `form7HadMod3` gadījumā, ja $4 \dot{:} p^k + 1$.
Izvade Tiek izvadīta iegūtā Adamāra matrica.

3.4.20 `form8`

Ievade Funkcijas ievaddati:

1. Adamāra matricas `hadMatrix1` un `hadMatrix2`.
2. Nepāra pirmskaitļi `prime1` un `prime2`, kas ir nepieciešami galīgo lauku konstruēšanai kā lauku harakteristikas.
3. Pozitīvi veseli skaitļi `power1` un `power2`, kas ir nepieciešami galīgā lauka konstruēšanai kā lauka kārtas.

Apstrāde Lai iegūtu Adamāra matricu ar izmēru $n_1 n_2(p_2^{k_2}+1)p_1^{k_1}$, kur n_1 un n_2 ir divu padoto Adamāra matricu izmēri un $p_2 = prime2$, $k_2 = power2$, $p_1 = prime1$, $k_1 = power1$ un izpildās $p_1^{k_1} - p_2^{k_2} = 4$, tiek pielietota funkcija `form8HadMod1` gadījumam, ja $4 \dot{:} p_1^{k_1} + 3$ un funkcija `form8HadMod3` gadījumam, ja $4 \dot{:} p_1^{k_1} + 1$.
Izvade Tiek izvadīta iegūtā Adamāra matrica.

3.4.21 `form9`

Ievade Funkcijas ievaddati:

1. Nepāra pirmskaitļi `prime1` un `prime2`, kas ir nepieciešami galīgo lauku konstruēšanai kā lauku harakteristikas.

2. Pozitīvi veseli skaitļi `power1` un `power2`, kas ir nepieciešami galīgā lauka konstruēšanai kā lauka kārtas.

Apstrāde Lai iegūtu Adamāra matricu ar izmēru $q(q+2)+1$, kur $q = p_1^{k_1}$ un $q+2 = p_2^{k_2}$, kur $p_2 = prime2$, $k_2 = power2$, $p_1 = prime1$, $k_1 = power1$, tiek izveidoti galīgie lauki $GF(p_1^{k_1})$ un $GF(p_2^{k_2})$, kas tiek izmantoti, lai izveidotu kvadrātveida matricu ar izmēriem $q(q+2)$, kuras rindas raksturo atbilst TTP-starpībkopai no [3]. Matricas katrā rindā ir izvietota viena TTP-starpībkopa. Iegūtajai matricai tiek pievienota jauna rinda un kolonna ar elementu vērtībām 1.
Izvade Tiek izvadīta iegūtā Adamāra matrica.

3.4.22 `form10`

Ievade Funkcijas ievaddati:

1. Pozitīvs vesels skaitlis `matrixCount` - padoto Adamāra matricu skaits.

2. Adamāra matricu masīvs matrices.

Apstrāde Lai iegūtu Adamāra matricu ar izmēru $n_1 n_2 ... n_k$, kur n_i ir i-tās padotās matricas izmērs, tiek izmantots Kronekera reizinājums, lai aprēķinātu visu padoto matricu Kronekera reizinājumu.
Izvade Tiek izvadīta iegūtā Adamāra matrica.

4 Testēšanas dokumentācija

Šajā dokumentā ir aprakstīts Adamāra matricu atbalsta bibliotēkas vienībtestēšanas process. Vienībtestēšana tika veikta, iekļaujot bibliotēku C++ programmā un pielietojot testpiemērus. Tā kā izvaddati (Adamāra matrica.) var būt ļoti liela izmēra (Bibliotēka ļauj konstruēt Adamāra matricas ar izmēru 1000 un lielākas.), tad vienībtestēšanas izvaddatu pareizības pārbaudei tika izmantota funkcija `isHadamard`, kura ļauj pārliecināties, vai izvaddati ir korekti (Vai izvadītā matrica ir Adamāra matrica.). Tika veikta arī šīs funkcijas testēšana mazām matricām, par kurām var pārliecināties, ka tās ir Adamāra matricas veicot aprēķinus uz papīra. Ierobežotā laika dēļ tika testētas tikai moduļa hadamard funkcijas, kuras tiek izsauktas Adamāra matricu konstruēšanai. Veicot šo funkciju testēšanu, tiek netieši pārbaudīta visu programmprodukta funkciju pareizība, jo tās izsauc visas citas funkcijas. Testpiemēri tika veidoti tā, lai aptvertu maksimālu funkciju izpildes scenāriju kopu.

Katrā testu kopā funkcijām `form1`, `form2`, `form3`, `form4`, `form6`, `form7`, `form8`, `form9`, `form10` ir aprakstīts, kādai parametru kopai jāģenerē Adamāra matrica.

4.1 Testu kopas

4.1.1 Testu kopa `Hadamard` metodes `form1` testēšanai

power	Sagaidāmais rezultāts
-1	Nepareizi ievaddati
0	Adamāra matrica ar izmēru 1
1	Adamāra matrica ar izmēru 2
5	Adamāra matrica ar izmēru 32
10	Adamāra matrica ar izmēru 1024

4.1.2 Testu kopa `Hadamard` metodes `form2` testēšanai

prime	power	Sagaidāmais rezultāts
6	2	Nepareizi ievaddati
-1	4	Nepareizi ievaddati
2	3	Nepareizi ievaddati
3	2	Nepareizi ievaddati
3	3	Adamāra matrica ar izmēru 28
3	5	Adamāra matrica ar izmēru 244
5	3	Nepareizi ievaddati
11	1	Adamāra matrica ar izmēru 12

4.1.3 Testu kopa `Hadamard` metodes `form3` testēšanai

hadMatrix	prime	power	Sagaidāmais rezultāts
Adamāra matrica ar izmēru 2	6	2	Nepareizi ievaddati
Adamāra matrica ar izmēru 2	-1	4	Nepareizi ievaddati
Adamāra matrica ar izmēru 4	2	3	Nepareizi ievaddati
Adamāra matrica ar izmēru 4	3	2	Adamāra matrica ar izmēru 40
Adamāra matrica ar izmēru 8	3	3	Adamāra matrica ar izmēru 224
Adamāra matrica ar izmēru 2	3	5	Adamāra matrica ar izmēru 488
Adamāra matrica ar izmēru 1	5	3	Nepareizi ievaddati
Adamāra matrica ar izmēru 2	11	1	Adamāra matrica ar izmēru 24

4.1.4 Testu kopa `Hadamard` metodes `form4` testēšanai

fact	Sagaidāmais rezultāts
$2^0(3^1+1)$	Adamāra matrica ar izmēru 12
$2^0(3^3+1)$	Adamāra matrica ar izmēru 756
$2^1(5^2+1)$	Nepareizi ievaddati
$2^0(3^1+1)(7^1+1)$	Adamāra matrica ar izmēru 992
$2^{-1}(3^3+1)$	Nepareizi ievaddati
$2^5(6^3+1)$	Nepareizi ievaddati

4.1.5 Testu kopa `Hadamard` metodes `form6` testēšanai

fact1	fact2	Sagaidāmais rezultāts
$2^0(3^1+1)$	$2^0(7^1+1)$	Adamāra matrica ar izmēru 28
$2^0(3^3+1)$	$2^0(11^1+1)$	Nepareizi ievaddati
$2^0(3^1+1)(3^1+1)(3^1+1)$	$2^0(67^1+1)$	Adamāra matrica ar izmēru 4288
$2^0(3^1+1)(7^1+1)$	$2^1(17^1+1)$	Adamāra matrica ar izmēru 1120
$2^{-1}(3^3+1)$	$2^5(6^3+1)$	Nepareizi ievaddati
$2^2(3^3+1)$	2^5	Nepareizi ievaddati

4.1.6 Testu kopa `Hadamard` metodes `form7` testēšanai

hadMatrix1	hadMatrix2	prime	power	Sagaidāmais rezultāts
Adamāra matrica ar izmēru 2	Adamāra matrica ar izmēru 2	3	2	Adamāra matrica ar izmēru 360
Adamāra matrica ar izmēru 1	Adamāra matrica ar izmēru 2	5	1	Nepareizi ievaddati
Adamāra matrica ar izmēru 2	Adamāra matrica ar izmēru 4	6	1	Nepareizi ievaddati
Adamāra matrica ar izmēru 2	Adamāra matrica ar izmēru 4	3	-1	Nepareizi ievaddati
Adamāra matrica ar izmēru 4	Adamāra matrica ar izmēru 2	11	1	Adamāra matrica ar izmēru 1056
Adamāra matrica ar izmēru 4	Adamāra matrica ar izmēru 2	3	2	Adamāra matrica ar izmēru 720

4.1.7 Testu kopa `Hadamard` metodes `form8` testēšanai

hadMatrix1	hadMatrix2	prime1	power1	prime2	power2	Sagaidāmais rezultāts
Adamāra matrica ar izmēru 2	Adamāra matrica ar izmēru 2	13	1	3	2	Adamāra matrica ar izmēru 520
Adamāra matrica ar izmēru 2	Adamāra matrica ar izmēru 2	17	1	13	1	Adamāra matrica ar izmēru 952
Adamāra matrica ar izmēru 2	Adamāra matrica ar izmēru 4	3	2	5	1	Adamāra matrica ar izmēru 432
Adamāra matrica ar izmēru 4	Adamāra matrica ar izmēru 2	7	1	5	1	Nepareizi ievaddati
Adamāra matrica ar izmēru 4	Adamāra matrica ar izmēru 1	11	1	7	1	Nepareizi ievaddati
Adamāra matrica ar izmēru 4	Adamāra matrica ar izmēru 2	7	1	3	1	Adamāra matrica ar izmēru 224

4.1.8 Testu kopa `Hadamard` metodes `form9` testēšanai

prime1	power1	prime2	power2	Sagaidāmais rezultāts
7	1	3	2	Adamāra matrica ar izmēru 64
15	1	17	1	Nepareizi ievaddati
2	2	2	1	Nepareizi ievaddati
5	1	7	1	Nepareizi ievaddati
5	2	3	3	Adamāra matrica ar izmēru 676
23	1	5	2	Adamāra matrica ar izmēru 576

4.1.9 Testu kopa `Hadamard` metodes `form10` testēšanai

matrixCount	hadMatrices	Sagaidāmais rezultāts
-1	Neviena matrica	Nepareizi ievaddati
0	Neviena matrica	Nepareizi ievaddati
1	Adamāra matrica ar izmēru 2	Adamāra matrica ar izmēru 2
2	2 Adamāra matricas ar izmēriem 2 un 28	Adamāra matrica ar izmēru 56
4	4 Adamāra matricas ar izmēriem 2, 4, 8, 12	Adamāra matrica ar izmēru 768
10	10 Adamāra matricas ar izmēriem 2	Adamāra matrica ar izmēru 1024

4.1.10 Testu kopa `Hadamard` metodes `isHadamard` testēšanai

matrix	Sagaidāmais rezultāts
$\begin{bmatrix} 1 \end{bmatrix}$	Ir Adamāra matrica
$\begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}$	Ir Adamāra matrica
$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	Nav Adamāra matrica
$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}$	Ir Adamāra matrica
$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & -1 & -1 \\ -1 & -1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 \\ -1 & 1 & -1 & 1 & -1 \end{bmatrix}$	Nav Adamāra matrica

4.2 Testēšanas žurnāls

Programmatūra tika testēta `Microsoft Visual Studio 2008` vidē. Testēšanas mērķiem tika izveidota testēšanas programma, kas izsauca "hadlib" funkcijas Adamāra matricu konstrukcijām. Pēc matricas konstrukcijas, tika izsaukta funkcija `isHadamard`, lai pārbaudītu, vai konstruētā matrica ir Adamāra matrica. Šeit iekļauts tikai funkcijas `form6` testēšanas žurnāla fragments:

fact1	fact2	Sagaidāmais rezultāts	Rezultāts
	...		
$2^0(3^3+1)$	$2^0(11^1+1)$	Nepareizi ievaddati	`V`
$2^0(3^1+1)(3^1+1)(3^1+1)$	$2^0(67^1+1)$	Adamāra matrica ar izmēru 4288	`X`
$2^0(3^1+1)(7^1+1)$	$2^1(17^1+1)$	Adamāra matrica ar izmēru 1120	`V`
	...		

4.3 Problēmu ziņojumi

Programmatūras testēšanā tika atrasta tikai viena problēma, kura tika labota.

4.3.1 form6

Problēma radās funkcijas `Factorization::value,` kuru izsauca funkcija `form6,` nepareizas darbības rezultātā.

Problēmas apraksts Izsaucot funkciju `form6` ar skaitļu 64 un 68 sadalījumu reizinātājos kā $2^0(3^1+1)(3^1+1)(3^1+1)$ un $2^0(67^1+1)$ attiecīgi, funkcija form6 nepareizi konstatēja, ka ievaddati ir nekorekti.

Kļūdas cēlonis Kļūdas cēlonis bija funkcijas `Factorization::value` nekorekta darbība, kas nepareizi aprēķināja sadalījuma reizinātājos vērtību. Funkcija aprēķināja pirmskaitļu pakāpju reizinājumu, bet tai vajadzēja aprēķināt par viens palielinātu pirmskaitļu pakāpju reizinājumu. (Skaitļa $2^0(3^1+1)(3^1+1)(3^1+1)$ vietā tika aprēķināts skaitlis $2^0 \cdot 3^1 \cdot 3^1 \cdot 3^1$.

Kļūdas labojums Kļūda tika labota, pie pirmskaitļu pakāpēm pieskaitot 1 pirms reizināšanas.

5 Programmatūras pirmkoda fragmenti

Programmatūras lielā izmēra dēļ nebija iespējams iekļaut visu pirmkodu šajā dokumentā. Tika nolemts dokumentā iekļaut:

- moduļa `forms` funkciju `skewSymmetricHadPower2`,
- moduļa `forms` funkciju `form10`,
- moduļa `forms` funkciju `symmetricHadMod3`,
- moduļa `hadamard` funkciju `form3`,
- moduli `galois`,
- bibliotēkas lietošanas piemēru Adamāra matricas konstruēšanai.

5.1 Funkcija `skewSymmetricHadPower2`

```
// Returns skew symmetric Hadamard matrix of order 2^'power'.
// Matrix if computed using doubling technique from http://rangevoting.org/SkewHad.html .
// 'power' -- power of 2.
Matrix *skewSymmetricHadPower2(int power)
{
    Matrix *symmetricBase=new Matrix(2);
    symmetricBase->set(0,0,1);
    symmetricBase->set(0,1,1);
    symmetricBase->set(1,0,1);
    symmetricBase->set(1,1,-1);

    Matrix *symmetric=new Matrix(1);
    symmetric->set(0,0,1);
    Matrix *skewSymmetric=new Matrix(1);
    skewSymmetric->set(0,0,1);

    Matrix *identity=generateIdentityMatrix(2);
    Matrix *skewIdentity=new Matrix(2);
    skewIdentity->set(0,0,0);
    skewIdentity->set(0,1,1);
    skewIdentity->set(1,0,-1);
    skewIdentity->set(1,1,0);

    // Invariant of cycle:
    // 'symmetric' contains symmetric Hadamard matrix of order 2^('pow'+1) and
    // 'skewSymmetric' contains skew symmetric Hadamard matrix  of order 2^('pow'+1).
    for (int pow=0; pow<power; pow++)
    {
        Matrix *skewComponent1=kroneckerProduct(identity,skewSymmetric);
```

```
        Matrix *skewComponent2=kroneckerProduct(skewIdentity,symmetric);
        Matrix *nextSkewSymmetric=sumMatrix(skewComponent1,skewComponent2);
        delete skewComponent1; delete skewComponent2; delete skewSymmetric;
        skewSymmetric=nextSkewSymmetric;

        Matrix *nextSymmetric=kroneckerProduct(symmetricBase,symmetric);
        delete symmetric;
        symmetric=nextSymmetric;
    }

    delete symmetricBase; delete symmetric; delete identity; delete skewIdentity;
    return skewSymmetric;
}
```

5.2 Funkcija `form10`

```
// Returns Hadamard matrix 'matrices[0]' X 'matrices[1]' X 'matrices[2]' X 'matrices[3]' X ...
// where 'matrices[i]' is Hadamard matrix and X is kronecker product.
Matrix *form10(int matrixCount, Matrix **matrices)
{
    Matrix *result=new Matrix(1);
    result->set(0,0,1);

    // Using kronecker product for construction of desired matrix
    // 'matrices[0]' X 'matrices[1]' X ... X 'matrices[matrixCount-1]'.
    for (int matrixNumber=0; matrixNumber<matrixCount; matrixNumber++)
    {
        Matrix *kroneckerProd=kroneckerProduct(result,matrices[matrixNumber]);
        delete result;
        result=kroneckerProd;
    }

    return result;
}
```

5.3 Funkcija `symmetricHadMod3`

```
// Lemma 14.1.4.
// Returns symmetric Hadamard matrix of order 'prime'^'power'+1==0(mod 4).
// 'prime' -- prime number.
// 'power' -- power of 'prime'.
Matrix *symmetricHadMod3(int prime, int power)
{
```

```
    // Uses function 'skewSymmetricHadMod3' for matrix construction of
    // order 'prime'^'power'+1.
    Matrix *skewSymmetric=skewSymmetricHadMod3(prime,power);
    int size=skewSymmetric->getSize();

    // 'makeSymmetric' -- matrix such that 'makeSymmetric' * 'skewSymmetric' is
    // a symmetric Hadamard matrix of order 'prime'^'power'+1.
    Matrix *makeSymmetric=new Matrix(size);

    for (int row=0; row<size; row++)
        for (int column=0; column<size; column++)
        {
            if ((row==0)&&(column==0))
            {
                makeSymmetric->set(row,column,-1);
            } else
            if ((row==1)&&(column==1))
            {
                makeSymmetric->set(row,column,1);
            } else
            if (row+column==size+1)
            {
                makeSymmetric->set(row,column,1);
            } else
            {
                makeSymmetric->set(row,column,0);
            }
        }

    Matrix *result=multiplicateMatrix(makeSymmetric,skewSymmetric);
    delete skewSymmetric; delete makeSymmetric;
    return result;
}
```

5.4 Funkcija `form3`

```
// 'hadMatrix' -- Hadamard matrix.
// 'prime' -- prime number.
// 'power' -- power of 'prime'.
// Constructs Hadamard matrix of order 'hadMatrix.getSize()'*('prime'^'power'+1)
int Hadamard::form3(const Hadamard &hadMatrix, int prime, int power)
{
    delete matrix;
    int size=hadMatrix.getSize();
    bool isHad=hadMatrix.isHadamard();
```

```
    // Checks whether a given Hadamard matrix is actually a Hadamard matrix.
    // Checks whether an order of given Hadamard matrix is at least 2.
    // Checks whether 'power' is a positive number.
    // Checks whether 'prime' is an odd prime number.
    if (isHad && (size>=2) && (power>0) && (core::isOddPrime(prime)) )
    {
        matrix=core::form3(hadMatrix.matrix,prime,power);
        return 1;
    } else
    {
        matrix=new core::Matrix(0);
        return -1;
    }
}
```

5.5 Modulis `galois`

```
// hadlib by Arturs Backurs Copyright 2010
//
// galois.cpp
//
//    This file is part of hadlib.
//
//    hadlib is free software: you can redistribute it and/or modify
//    it under the terms of the GNU General Public License as published by
//    the Free Software Foundation, either version 3 of the License, or
//    (at your option) any later version.
//
//    hadlib is distributed in the hope that it will be useful,
//    but WITHOUT ANY WARRANTY; without even the implied warranty of
//    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
//    GNU General Public License for more details.
//
//    You should have received a copy of the GNU General Public License
//    along with hadlib.  If not, see <http://www.gnu.org/licenses/>.
//

namespace hadlib
{
    namespace core
    {

        // Constructor for 'GaloisField' which sets values of
        // 'irreduciblePoly, 'identityElement', 'zeroElement',
        // 'prime', 'power', 'order'.
```

```
// 'fieldPrime' -- prime describing an order of field.
// 'fieldPower' -- power of 'fieldPrime' describing an order of field.
GaloisField::GaloisField(int fieldPrime, int fieldPower)
{
    prime=fieldPrime;
    power=fieldPower;

    NTL::ZZ_p::init(NTL::to_ZZ(fieldPrime));
    order=pow(fieldPrime, fieldPower);

    // Using NTL library for construction of an irreducible polynomial.
    NTL::BuildIrred(irreduciblePoly, fieldPower);

    identityElement=elementFromNumber(1);
    zeroElement=elementFromNumber(0);
}

// Sums two Galois field's elements returning their sum.
// 'element1' and 'element2' -- field's elements to be summed.
GaloisFieldElement GaloisField::sum(GaloisFieldElement element1,
                                    GaloisFieldElement element2)
{
    NTL::ZZ_p::init(NTL::to_ZZ(prime));
    return (element1+element2);
}

// Subtracts two Galois field's elements returning their subtraction.
// 'element1' and 'element2' -- field's elements to be subtracted.
GaloisFieldElement GaloisField::subtraction(GaloisFieldElement element1,
                                            GaloisFieldElement element2)
{
    NTL::ZZ_p::init(NTL::to_ZZ(prime));
    return (element1-element2);
}

// Calculates power of given Galois field's element.
// 'element' -- Galois field's element and base of exponentiation.
// 'toPower' -- power of 'element'.
GaloisFieldElement GaloisField::elementPower(GaloisFieldElement element, int toPower)
{
    NTL::ZZ_p::init(NTL::to_ZZ(prime));
    GaloisFieldElement result=elementFromNumber(1), power2=element;

    // Calculating power using binary representation of a 'toPower'
    // thus achieving logarithmic time complexity.
    while (toPower>0)
```

```
    {
        if (toPower%2==1)
            result=(result*power2)%irreduciblePoly;
        toPower/=2;
        power2=(power2*power2)%irreduciblePoly;
    }
    result.normalize();
    return result;
}

// Calculates Galois field's element from given integer number such
// that 'elementFromNumberSymmetric(0)'=='zeroElement' and if a!=0 then
// 'elementFromNumberSymmetric(a)'+'elementFromNumberSymmetric(b)'='zeroElement'
// where b is such an integer number that if bi and ai -- i-th numbers
// from representation of b and a in 'prime'-ary system then ai+bi=='prime'.
// 'number' -- integer to be evaluated as a Galois field's element.
GaloisFieldElement GaloisField::elementFromNumberSymmetric(int number)
{
    NTL::ZZ_p::init(NTL::to_ZZ(prime));
    GaloisFieldElement result;
    if (number==0)
    {
        // Returning field's zero element.
        return result;
    } else
    {
        if (number<=(order-1)/2)
            number--;
    }

    for (int deg=0; deg<power; deg++)
    {
        int coeff=number%prime;
        if (coeff<(prime-1)/2)
            coeff=coeff+prime-(prime-1)/2; else
            coeff=coeff-(prime-1)/2;

        // Setting polynomial's (Representing Galois field's element.) coefficient.
        NTL::SetCoeff(result, deg, coeff);
        number/=prime;
    }

    return result;
}

// Calculates Galois field's element from given integer number.
```

```
// 'number' -- integer to be evaluated as a Galois field's element.
GaloisFieldElement GaloisField::elementFromNumber(int number)
{
    NTL::ZZ_p::init(NTL::to_ZZ(prime));
    GaloisFieldElement result;
    int deg=0;
    while (number>0)
    {

        // Setting polynomial's (Representing Galois field's element.) coefficient.
        NTL::SetCoeff(result,deg,number%prime);
        deg++;
        number/=prime;
    }
    return result;
}

// Inverse function of 'elementFromNumber'.
// 'element' -- Galois field's element to be evaluated as an integer number.
int GaloisField::numberFromElement(GaloisFieldElement element)
{

    // Using an polynomial's (Corresponding to a Galois field's element 'element'.)
    // coefficients calculating an unique number represeting an 'element'.

    NTL::ZZ_p::init(NTL::to_ZZ(prime));
    int length=element.rep.length();
    int sum=0;
    int primePower=1;
    for (int deg=0; deg<length; deg++)
    {

        // Using a coefficient from polynomial's 'deg'-th therm for evaluation of 'sum'.
        sum+=primePower*to_int(rep(element.rep[deg]));
        primePower*=prime;
    }
    return sum;
}

// Returns character for a given Galois field's element.
// Returns 0 if a given element is zero element of Galois field.
// Returns 1 if a given element is square of some Glaois field's nonzero element.
// In other case returns -1.
// 'element' -- Galois field's element for which character is required.
int GaloisField::elementCharacter(GaloisFieldElement element)
{
```

```
    NTL::ZZ_p::init(NTL::to_ZZ(prime));
    if (element==zeroElement)
    {

        // Given element is a zero element.
        return 0;
    }
    GaloisFieldElement elPower=elementPower(element,(order-1)/2);

    if (elPower==identityElement)
    {

        // Given element is a square of some nonzero field's element.
        return 1;
    } else
    {

        // Given element is not a square of some field's element.
        return -1;
    }
}

// Auxilary function for calculation of 'base'^'power'.
// 'base' -- base for exponentiation.
// 'power' -- power of 'base'.
int pow(int base, int power)
{
    int result=1;
    for (int deg=0; deg<power; deg++)
    {
        result*=base;
    }
    return result;
}

// Auxilary function for checking whether a given number is an odd prime.
// 'number' -- number to be checked whether it is prime or not.
bool isOddPrime(int number)
{
    int divCount=0;

    // Counting number of divisors for integer 'number'.
    for (int divisor=1; divisor<=number; divisor++)
    {
        if (number%divisor==0)
            divCount++;
```

```
            }

            // Returning 'true' only in case if divisor count==2
            // and given number is not a even prime.
            return (divCount==2)&&(number!=2);
        }

    } // namespace core
} // namespace hadlib
```

5.6 bibliotēkas lietošanas piemēru Adamāra matricas konstruēšanai

```
// examples.h

#include "hadamard.h"
using namespace std;

int main()
{
    // form3 example
    hadlib::Hadamard h1, h; // h1 and h - Hadamard matrices.
    h1.form1(1); // Constructs Hadamard matrix of order 2
    h.form3(h1,11,1); // Constructs Hadamrd matrix of order (11^1+1)*2=24.

    h.printMatrixToFile("Hadamard.txt"); // Prints Hadamard matrix to file "Hadamard.txt"
    h.printMatrix(); // Prints Hadamard matrix to standart output (cout).
    cout<<"Matrix size : "<<h.getSize()<<endl; // Outputs matrix order.

    system("pause");
    return 0;
}
```

6 Projekta organizācija

Adamāra matricu atbalsta bibliotēka "hadlib" tika izstrādāta ņemot par pamatu spirāles dzīves cikla modeli. Šis dzīves cikla modelis tika izvēlēts, jo programmprodukta izstrādes sākumā nebija skaidri zināms, kādas Adamāra konstrukcijas metodes tiks realizētas un kāda būs precīza gatava programmprodukta arhitektūra. Programmprodukts tika izstrādāts divos ciklos. Pirmajā ciklā tika izstrādāts sākotnējais priekšstats par programmprodukta prasībām, tika veikts sākotnējs programmprodukta projektējums. Tad programmprodukts tika implementēts un veikta vienībtestēšana. Kad programmprodukts realizēja visas Adamāra matricas konstrukcijas, tika uzsākts otrs modeļa cikls. Tajā tika izstrādāta programmas prasību specifikācija, programmatūras projektējuma apraksts. Tad tika implementēta programmprodukta gala versija. Gatavais programmprodukts tika testēts.

Nepieciešamos teorētiskos materiālus programmprodukta realizācijai autoram sagādāja darba vadītājs. Tālākā programmprodukta izstrāde notika patstāvīgi, kontaktējoties ar darba vadītāju. Programmproduktu pilnībā izstrādāja autors, kurš veica dokumentēšanu, implementēšanu, testēšanu. Jāatzīmē, ka lielā programmprodukta apjoma dēļ, darbībām galīgajos laukos tika izmantota [1], jo galīgo lauku darbību realizācija ir vismaz vēl viena kvalifikācija darba apjoms.

Programmprodukts tika izstrādās `Microsoft Visual Studio` vidē. Dokumentācija tika rakstīta LyX vidē lietojot LaTeX.

7 Konfigurāciju pārvaldība

Programmprodukta mazā apjoma un tikai viena izstrādātāja dēļ, konfigurāciju pārvaldībai netika izmantota speciāla programmatūra. Konfigurāciju pārvaldība notika, veidojot programmprodukta kopiju atsevišķā mapē. Katrai jaunai programmprodukta versijai tika saglabāta jauna programmprodukta versijas kopija. Analoģiski notika dokumentācijas versiju uzglabāšana. Vienmēr tika uzglabāta programmprodukta dokumentācijas iepriekšējā versija.

8 Kvalitātes nodrošināšana

Programmprodukta kvalitāte tika nodrošināta, veicot šādas darbības:

- Programmprodukta dokumentācija tika izstrādāta vadoties pēc atbilstošajiem Latvijas Valsts standartiem,
- Programmprodukta kods pilnībā tika komentēts angļu valodā,
- Visas programmprodukta funkcijas tika pakļautas vienībtestēšanai un tika pārbaudīts, vai tās atbilst programmatūras prasību specifikācijas funkcionālajām prasībām.
- Tika pārbaudīta programmatūras koda atbilstība standartam ISO/IEC 14882:1998 .

9 Darbietilpības novērtējums

Programmprodukta darbietilpība tika novērtēta izmantojot COCOMO metodi. Lai varētu izmantot COCOMO metodi, nepieciešams aprēķināt paredzamo koda rindiņu skaitu. Lai to izdarītu, nepieciešams aprēķināt programmprodukta funkcijpunktu skaitu. Bibliotēkai ir 9 dažādi ievadi (Katrai no 9 dažādām konstrukcijas metodēm pa vienam ievadam.) kā arī 11 dažādi izvadi (Katrai no 9 dažādām Adamāra matricas konstrukcijas metodei kā arī 2 izvadi - Adamāra matricas izvade uz standarta izvada un Adamāra matrica izvade datnē.). Tā kā visi izvadi ir vienkārši, tad iegūst $9 \cdot 3 + 11 \cdot 4 = 71$ funkcijpunktus. Tā kā viena funkcijpunkta realizēšanai vidēji vajag 29 koda rindiņas, tad paredzamais koda rindiņu skaits ir $29 \cdot 71 = 2059$.

Izmantojot iegūto koda rindiņu skaits un [4] pieejamo darbietilpības aprēķināšanu, tiek iegūta 3.78 mēnešu darbietilpība. Lietojot [4] tika izmantots iegūtais koda rindiņu skaits (SLOC) 2059 kā arī izvēlēta `Organic` izstrādes vide, jo programmu izstrādāja viens cilvēks mājas apstākļos. Pārējos parametri tika izvēlēti, balstoties uz iepriekšējo programmizstrādes pieredzi.

Reāli projekts tika izstrādāts nedaudz vairāk kā 3 personmēnešu laikā. Programmprodukta kods satur 1248 koda rindiņas neskaitot tukšās rindiņas un komentārus. To, ka aprēķināto 2059 koda rindiņu vietā iznāca 1248 koda rindiņas autors izskaidro ar to, ka reāli vienu funkcijpunktu varēja realizēt mazākā skaitā koda rindiņu pieņemto 29 koda rindiņu vietā.

10 Rezultāti

Kvalifikācijas darba rezultātā tika izstrādāta strādājoša Adamāra matricu atbalsta bibliotēka. Ar šajā bibliotēkā realizētajām funkcijām var konstruēt 9 dažādu veidu Adamāra matricas. Autors cer, ka turpmāk bibliotēka tiks papildināta ar jaunām Adamāra matricu konstrukcijas matodēm.

11 Secinājumi

Programmprodukta izstrādes laikā autors secināja, ka programmproduktu zinātniskiem mērķiem ir ļoti grūti izstrādāt pašam. Piemēram, lai izstrādātu "hadlib" bibliotēku, bija nepieciešamība veikt darbības galīgajos laukos. Tā kā šo darbību realizācija ir ļoti laikietilpīga un grūti programmējama, tad nācās izmantot brīvi pieejamu [1]. Autors nonāca arī pie secinājuma, ka brīvi pieejama zinātniskiem mērķiem domāta programmatūra bieži ir slikti dokumentēta. Tā, piemēram, ilgi nācās strādāt ar [1], līdz tika iegūta nepieciešamā galīgo lauku funkcionalitāte. Tāpēc lielu uzmanību autors pievērsa sava darba saskarnes ar lietotāja programmatūru vienkāršībai.

Pateicības

Autors izsaka pateicību darba vadītājam Jurim Smotrovam darba tēmas ieteikšanu un teorētisko materiālu nodrošinājumu. Autors izsaka pateicību arī Madaram Virzam par padomiem dokumentācijas noformēšanā.

Izmantotā literatūra un resursi

[1] *NTL: A Library for doing Number Theory* [tiešsaiste]. [atsauce 12.02.2010]. Pieejams: `http://www.shoup.net/ntl/`

[2] *Skew Hadamard matrices* [tiešsaiste]. [atsauce 25.02.2010]. Pieejams: `http://rangevoting.org/SkewHad.html`

[3] *Twin Prime Paper* [tiešsaiste]. [atsauce 27.02.2010]. Pieejams: `http://www.maths.nuigalway.ie/~padraig/Docs/TwinPrimePaper.pdf`

[4] *The Little COCOMO Calculator* [tiešsaiste]. [atsauce 27.02.2010]. Pieejams:`http://sunset.usc.edu/research/COCOMOII/cocomo81_pgm/cocomo81.html`

[5] **Marshall Hall**, *Combinatorial Theory.* John Wiley & Sons, 1986. 440 pp.

[6] **Ireland, M., Rosen, K.** *A Classical Introduction to Modern Number Theory (2nd ed.) [p. 238-263].* Springer, 1998. 389 pp.

Kvalifikācijas darbs "**Adamāra matricu atbalsta bibliotēka**" izstrādāts Latvijas Universitātes Datorikas fakultātē.

Ar savu parakstu apliecinu, ka darbs izstrādāts patstāvīgi, izmantoti tikai tajā norādītie informācijas avoti un iesniegtā darba elektroniskā kopija atbilst izdrukai.

Autors: **Artūrs Bačkurs** _____ _____**.2010**

Rekomendēju darbu aizstāvēšanai
Darba vadītājs: **Dr.Dat. Juris Smotrovs** _____ _____**.2010**

Recenzents: dipl.fiz. Brants Juris

Darbs iesniegts
Kvalifikācijas darbu pārbaudījumu komisijas sekretārs: _____ _____