KAUNO TECHNOLOGIJOS UNIVERSITETAS

ELEKTROS IR ELEKTRONIKOS FAKULTETAS

Šarūnas Stašaitis

ELEKTROS TINKLO SKAIČIAVIMŲ GREITINIMAS GRAFINIU PROCESORIUMI

Baigiamasis magistro projektas

Vadovas
dr. Vytautas Šiožinys

KAUNAS, 2017

KAUNO TECHNOLOGIJOS UNIVERSITETAS

ELEKTROS IR ELEKTRONIKOS FAKULTETAS
ELEKTROS ENERGETIKOS SISTEMŲ KATEDRA

ELEKTROS TINKLO SKAIČIAVIMŲ GREITINIMAS GRAFINIU PROCESORIUMI

Baigiamasis magistro projektas
Elektros energetikos inžinerija 621H63003

Vadovas
dr. Vytautas Šiožinys

Recenzentas
Prof. dr. Saulius Gudžius

Projektą atliko
Šarūnas Stašaitis

KAUNAS, 2017

KAUNO TECHNOLOGIJOS UNIVERSITETAS

ELEKTROS IR ELEKTRONIKOS FAKULTETAS

(Fakultetas)

Šarūnas Stašaitis

(Studento vardas, pavardė)

Elektros energetikos inžinerija 621H63003

(Studijų programos pavadinimas, kodas)

Baigiamojo projekto „Elektros tinklo skaičiavimų greitinimas grafiniu procesoriumi“

AKADEMINIO SĄŽININGUMO DEKLARACIJA

2017 m. birželio 5 d.

Kaunas

Patvirtinu, kad mano Šarūno Stašaičio baigiamasis projektas tema „Elektros tinklo skaičiavimų greitinimas grafiniu procesoriumi“ yra parašytas visiškai savarankiškai, o visi pateikti duomenys ar tyrimų rezultatai yra teisingi ir gauti sąžiningai. Šiame darbe nei viena dalis nėra plagijuota nuo jokių spausdintinių ar internetinių šaltinių, visos kitų šaltinių tiesioginės ir netiesioginės citatos nurodytos literatūros nuorodose. Įstatymų nenumatytų piniginių sumų už šį darbą niekam nesu mokėjęs.

Aš suprantu, kad išaiškėjus nesąžiningumo faktui, man bus taikomos nuobaudos, remiantis Kauno technologijos universitete galiojančia tvarka.

(vardą ir pavardę įrašyti ranka) *(parašas)*

Stašaitis, Šarūnas. Elektros Tinklo Skaičiavimų Greitinimas Grafiniu Procesoriumi. Magistro baigiamasis projektas / vadovas dr. Vytautas Šiožinys; Kauno technologijos universitetas, Elektros ir elektronikos fakultetas, Elektros energetikos sistemų katedra.

Mokslo kryptis ir sritis: Elektros ir elektronikos inžinerija, Technologiniai mokslai

Reikšminiai žodžiai: *skaičiavimai vaizdo plokšte, bendro naudojimo GPU.*

Kaunas, 2017. 39 p.

SANTRAUKA

Elektros tinklų skaičiavimus patogu išreikšti matricų algebra. Tradiciškai kompiuteriniai skaičiavimai atliekami centriniuose procesoriuose. Centriniai procesoriai gerai pritaikyti atlikti nuoseklius veiksmus, modernūs centriniai procesoriai tipiškai lygiagrečiai gali vykdyti nuo 4 iki 16 gijų. Grafiniai procesoriai originaliai buvo sukurti naudoti kompiuteriniuose žaidimuose, bet šiuo metu yra plačiai taikomi atlikti įvairius skaičiavimus. Tam naudojamos aplikacijų programavimo sąsajos CUDA (sukurta Nvidia), ir OpenCL (originaliai sukurta Apple, šiuo metu palaikoma Khronos grupės). Pastarasis palaikomas ne vien vaizdo plokštėse. Grafiniai procesoriai vienu meto gali leisti daug (2000 ir daugiau) lygiagrečių gijų ir yra labiau specializuoti nei centriniai procesoriai veiksmų su slankiojančio kablelio skaičiais atlikimui. Tai leidžia žymiai greičiau atlikti skaičiavimus, ypač kai reikia apdoroti didelius duomenų kiekius.

Stašaitis, Šarūnas. Acceleration of Power Grid Calculations with Graphic Processor: Master‘s thesis / supervisor assoc. prof. Vytautas Šiožinys. Kaunas University of Technology, Faculty of Electrical and Electronics Engineering, department of Electric Power Systems

Research area and field: Electrical and Electronics Engineering, Technological Sciences

Key words: *calculations on GPU, general purpose GPU*

Kaunas, 2017. 39 p.

SUMMARY

Electrical network calculations are convenient to express using matrix algebra. Traditionally calculations are performed using CPUs, modern CPUs typically have 4 to 16 threads. GPUs were originally created for use in video games, currently are used to accelerate various calculation tasks. To accomplish that APIs like CUDA (create by Nvidia) and OpenCL (originally created by Apple, currently maintained by the Khronos group). The later may be used in more devices than just GPUs. GPUs can simultaneously run many (200 or more) concurrent threads and are specialized towards floating point calculations as compared to CPU. That allows to perform calculations much faster, especially when large volumes of data need to be processed.

Turinys

SANTRUMPŲ IR ŽENKLŲ AIŠKINAMASIS ŽODYNAS

GPU – grafinis procesorius

CPU – centrinis procesorius

FPGA – programuojamas integruotas grandynas

CUDA – aplikacijų programavimo sąsaja skirta bendros paskirties GPU programavimui Nvidia vaizdo plokštėse

OpenCL – aplikacijų programavimo sąsaja skirta programuoti GPU, CPU ir FPGA it kita.

SPMD – (ang. single program, multiple data), programos stilius, kurioje mažas kodo kiekis pritaikomas dideliam duomenų kiekiui, žr. SIMD

WLP – paralelizmo lygis

RAM – kompiuterio sisteminė operatyvioji atmintis

VRAM – vaizdo plokštės operatyvioji atmintis

ALU – (ang. Arithmetic logic unit) kompiuterio dalis, kuri vykdo aritmetines ir logines operacijas

SIMD – (ang. Single instruction, multiple data), viena instrukcija, daug duomenų

SSE – (ang. Streaming SIMD Extensions), x86 architektūros plėtinys

AVX – (ang. Advanced Vector Extensions), x86 architektūros plėtinys

DRAM – dinaminė atsitiktinės prieigos atmintis

P – realioji galia

Q – menamoji galia

V, E – įtampa

Y – laidis

Z – varža

J – jakobiano matricos

δ – fazės kampas

LU – žemutinė aukštutinė matrica, taip, kad LU=A

A – pirmojo tipo incidencijų matrica

B – antrojo tipo incidencijų matrica

z – šakų varžų matrica

Z_k – kontūrų varžų matrica

e – mazgų įtampų šaltinių matrica

E_k – kontūrų įtampų matrica

I_k – kontūrų srovių matrica

i – mazgų srovių matrica

Y_m – mazgų laidžių matrica

U_m – mazgų įtampų matrica

ĮVADAS

Elektros tinklų skaičiavimus patogu išreikšti matricų algebra. Tradiciškai kompiuteriniai skaičiavimai atliekami centriniuose procesoriuose. Centriniai procesoriai gerai pritaikyti atlikti nuoseklius veiksmus, modernūs centriniai procesoriai tipiškai lygiagrečiai gali vykdyti nuo 4 iki 16 gijų. Grafiniai procesoriai originaliai buvo sukurti naudoti kompiuteriniuose žaidimuose, bet šiuo metu yra plačiai taikomi atlikti įvairius skaičiavimus. Tam naudojamos aplikacijų programavimo sąsajos CUDA (sukurta Nvidia), ir OpenCL (originaliai sukurta Apple, šiuo metu palaikoma Khronos grupės). Pastarasis palaikomas ne vien vaizdo plokštėse. Grafiniai procesoriai vienu meto gali leisti daug (2000 ir daugiau) lygiagrečių gijų ir yra labiau specializuoti nei centriniai procesoriai veiksmų su slankiojančio kablelio skaičiais atlikimui. Tai leidžia žymiai greičiau atlikti skaičiavimus, ypač kai reikia apdoroti didelius duomenų kiekius.

Darbo tikslas yra ištirti elektros tinklų skaičiavimų pagreitėjimą, kai skaičiavimams atlikti naudojamos vaizdo plokštės.

LITERATŪROS APŽVALGA

Straipsniuose dažniausiai nagrinėjamos problemos yra LU dekompozicijos radimas ir efektyvūs tiesinių matricinių lygčių sprendimo būdai. Straipsniuose dažnai yra neatkreipiamas dėmesys į konkrečių vaizdo plokščių pajėgumus atliekant veiksmus naudojant 64 bitų slankiojančio kablelio skaičius [1][2][3].

Taikant OpenCL egzistuojantiems algoritmams, ne visi aptarti sprendimo LU ar tiesinių lygčių sprendimo būdai yra praktiški, dėl keliamų specifinių reikalavimų skaičiavimuose naudojamoms matricoms [4].

Literatūros teigiami pagreitėjimai naudojant skaičiavimų greitinimą vaizdo plokšte, priklausomai nuo algoritmo ar uždavinio dydžio, siekia nuo iki 10 kartų. Prieinamuose palyginamuose palyginimai atliekami naudojant serverių įrangą arba naudojant 32 bitų slankiojančio kablelio skaičius [3][5].

Lietuvoje GPU lygiagrečiųjų skaičiavimų architektūros sprendimų pritaikomumą tiria atliekant duomenų apdorojimą finansų rinkose [6], vaizdų paveikslėlyje radimo uždaviniams [7] bei resursų apskaitos sistemos sukūrimui [8].

UŽDAVINIAI

1. Ištirti OpenCL taikytinumą atliekant skaičiavimus 64 bitų slankiojančio kablelio kompleksiniais skaičiais vidutinės klasės vaizdo plokštėse.

2. Ištirti OpenCL spendimų integravimo į esamą programinę įrangą EA-PSM pritaikant galios srautų ir trumpųjų jungimų skaičiavimams.
3. Patikrinti galios srautų skaičiavimų kokybę naudojant IEEE standarto 44 mazgų testinį uždavinį
4. Įvertinti realų skaičiavimų spartos pokytį.

1. CENTRINIŲ IR GRAFINIŲ PROCESORIŲ SKIRTUMAI

Vienas pagrindinių centrinių procesorių prioritetų yra vienos gijos veiki greičio didinimas, bet pastaruoju metu šis progresas, žymiai sulėtėjo. Centrinių procesorių gamintojams tapo nebepraktiška didinti dažnių, dėl neproporcingai didėjančių galios reikalavimų. Tai uždeda ribą praktiškiems nuosekliems skaičiavimams. Dėl to buvo pradėti gaminti kelių branduolių procesoriai. Tuo tarpu grafiniai procesorių užduotis yra atlikti daug lygiagrečių skaičiavimų, todėl šie koncentruoti į branduolių kiekio didinimą. Iš to išeina, kad algoritmų kūrimas grafiniams procesoriams (GPU) ir centriniams procesoriams (CPU) žymiai skiriasi: CPU vienu metu vykdys iki keliasdešimt gijų, GPU – kelis tūkstančius [1].

1. GPU sukuria ir vienu metu leidžia tūkstančius maskuoti duomenų transliavimo magistrale vėlinimą. Reikalingas didelis duomenų apdorojimo ir prieigos prie atminties santykis atminties magistralės įsotinimo vengimui.
2. Atminties pralaidumo tausojimui, kai kelios gretimos gijos skaito atmintį, jos vykdo globalų nuskaitymą, nuskaitymai dėl optimizavimo yra apjungiami.
3. Darbo atmintis gijų grupėje susideda iš programiškai valdomos laikinosios atminties („shared memory“ CUDA ir „local memory“ OpenCL). Ši greita atmintis, riboto dydžio, atmintis paskirstyta tarp gijų blokų.
4. Atminties turi ribotą kiekį prieigos taškų kiekį, tinkamas gijų eiliškumas padeda išvengti prieigos konfliktų.
5. GPU programavimas bendru atveju yra viena programa, daug duomenų stiliaus (SPMD)

GPU architektūroje gijų blokai („blocks“) dalinasi duomenimis ir gali sinchronizuotis. Blokai yra grupuojami į grupes („warps“). Gijų grupė yra vykdoma procesoriaus. Grupės gali būti vykdomos pakaitomi, vienu metu, principiškai panašiai kaip centrinis procesorius vykdo gijas [9].

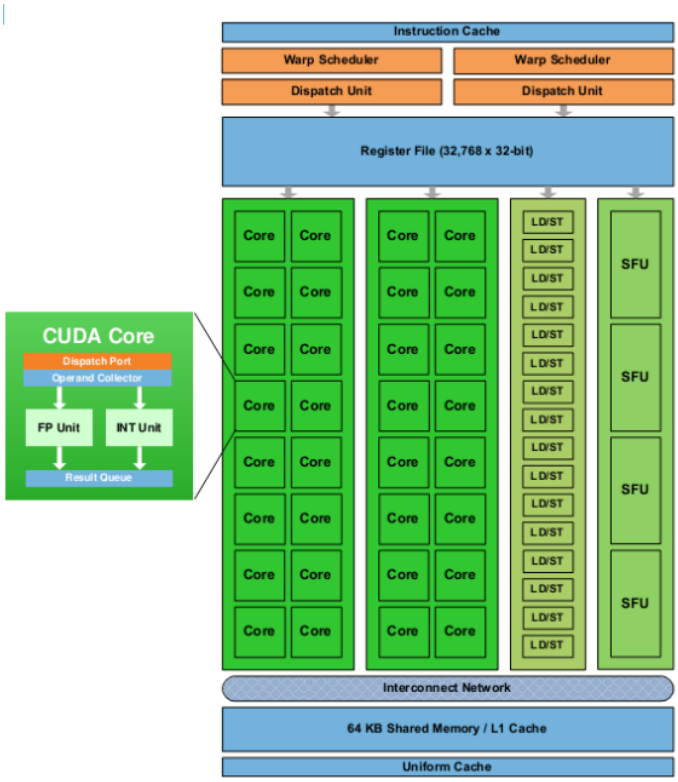
Kiekvienam kerneliui apskaičiuojamas maksimalus grupės lygio paralelizmas, WLP_{max} (warp level parallelism), kas yra maksimalus grupių kiekis, kurį galima paskirti vykdyti GPU, neperžiangiant aparatinių resursų, kaip bendros atminties, ar kernelio naudojamų registrų ir kita.

Paralelizmo lygis gali kisti užduoties metu. Paralelizmo lygis (WLP) apribotas kodo segmentui vadinsis WLP_{local}. Vidutinis WLP prieinamas visam kerneliui bus $WLP_{avb.}$ Grupės lygmenyje, GPU bando mažinti atminties vėlinimą naudodamas duomenų lygio paralelizmą. Duomenų lygio paralelizmas pasiekiamas vienu metu naudojant kelis atminties bankus. Gijos lygmenyje našumą gali gerinti instrukcijų lygio paralelizmu, taip slepiant tarpgrupinius uždelsimus, kuriuos gali sukelti skaitymas iš globalios atminties ir registrų priklausomybės. Prieinamas instrukcijų lygio paralelizmas skirsis skirtinguose kodo segmentuose.

Kernelio atsparumas vėlinimui kiekvienoje darbo fazėje priklauso nuo tarpgijinio instrukcijų paralelizmo, gijų organizacijos smulkumo (grupės dydžio), magistralės pločio. Atminties uždelsimai klasifikuojami į sukeltus dėl riboto atminties greičio ir grynai sukeltus atminties vėlinimo. Vėlinimo sukelti delsimai gali būti automatiškai paslėpti tarpgijinio paralelizmo [10].

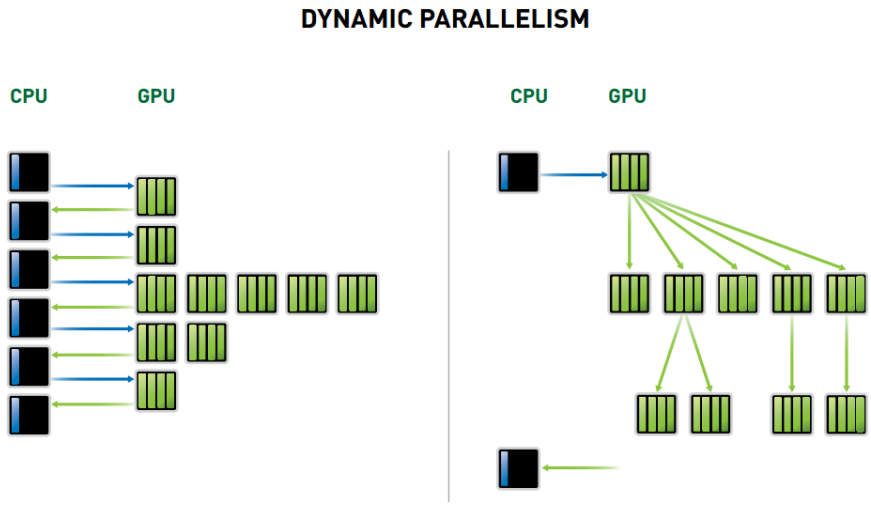
2. GPU ARCHITEKTŪRA

Atliekant didelių matricų algebros veiksmus reikalingi dideli kompiuterių resursai. Dažniausiai programos skaičiavimų algoritmuose pilnai išnaudoja kompiuterio centrinio procesoriaus CPU resursus. Vystantis kompiuterių technologijoms CPU procesorių skaičius didinamas bei spartinamas darbinis dažnis. Skaičiavimo procesai atliekami lygiagrečiai. Sudėtingi grafikos vaizdai apdorojami vaizdo plokštėje (GPU). Nvidia firmos GPU vaizdo plokščių, skirtų 3D vaizdo uždavinių apdorojimui, pirmoji karta prasidėjo 1999 m. Nuo 2006 metų Nvidia pristato CUDA (angl. Compute Unified Device Architecture) architektūros sprendimus, kurie leidžia programinės įrangos vystytojams ir inžinieriams išnaudoti GPU resursus. CUDA platforma leidžia tiesiogiai valdyti GPU instrukcijų rinkinius ir lygiagrečius skaičiavimo elementus. Programinis kodas lengvai keičiamas programavimo kalbomis C, C++, Fortran. Esamos bibliotekos yra suderinamos su matematikos programomis bei programavimo kalbomis: Python, Perl, Fortran, Java, Ruby, Haskell, R, Matlab. 2006 metais Nvidia išleidžia Tesla GPU, nuo 2009 metų Fermi, o nuo 2012 metų Kepler architektūros vaizdo grafikos apdorojimo plokšte. Pagrindinis Kepler architektūros pranašumas yra lygiagretūs dinamiški skaičiavimai 2.1. Pav. bei kontrolė, atskirų GPU procesorių valdymas vienu metu 2.4. Pav., papildomas GPU procesorių gijų valdymo modulis, Nvidia GPUDirect modulis – leidžiantis vieno GPU kompiuterio ar GPU skirtingų serverių, esančių tinkle, tiesiogiai keistis duomenimis tiesiogiai nesikreipiant į CPU atminties sistemą [11].

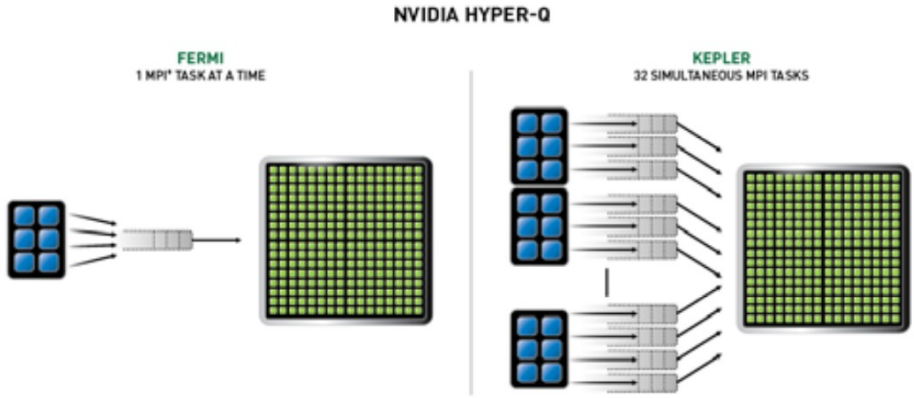


2.1. Pav.: Fermi architektūra [12]

2.2. Pav.: Kepler architektūra [13]



2.3. Pav.: Lygiagrečių dinamiškų savybių palyginimas [13]



2.4. Pav.: HYPER-Q savybės palyginimas [13]

3. OPENCL ATMINTIES VALDYMAS IR ARCHITEKTŪRA

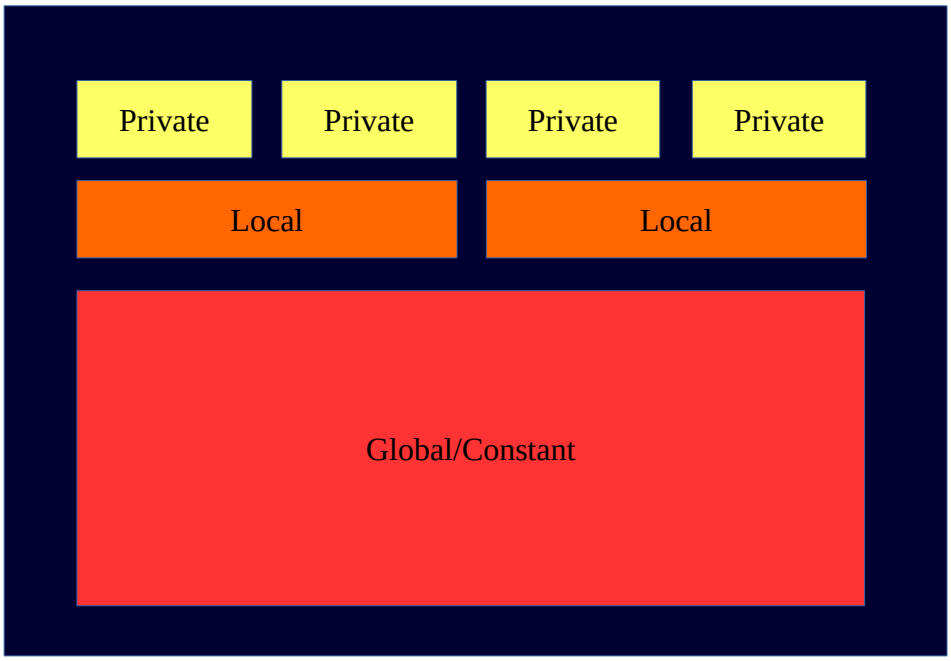
OpenCL naudojami 4 atminties tipai, vizualiai pavaizduoti 3.1. Pav. [14]:

1. Globali (Global) – prie jos gali prieiti ir skaityti/rašyti visi darbo vienetai, prie jos gali prieiti ir valdantysis (host) CPU kodas.
2. Nekintama (Constant) – darbo vienetams tai yra tik skaityti skirtas atminties regionas. Valdantysis kodas gali ir skaityti ir rašyti.
3. Lokali (Local) – atminties prieinama vienai darbo grupei. Naudojama bendriems darbo grupės kintamiesiems ir komunikacijai tarp darbo vienetų.
4. Privati (Private) – prieinama skaityti/rašyti vienam darbo vienetui.

Paprasta OpenCL programa vadinama kerneliu. Kernelis yra instrukcija naudojama apdoroti didelį duomenų kiekį, nedalomas duomenų kiekis reikalingas kerneliui vadinamas darbo vienetu. Prieš apdirbant duomenis juos reikia perduoti iš valdančio kodo į OpenCL prietaisą. Dažniausiai tai reiškia duomenų kopijavimą ir pagrindinės operatyviosios atminties (RAM) į vaizdo plokštės operatyviąją atmintį (VRAM) [15].

OpenCL kodas yra kilnojamas, kas reiškia, kad parašyta programa gali veikti ant skirtingų prietaisų, pavyzdžiui CPU, GPU, FPGA, bet skirtingoms architektūroms reikalingas skirtingas kodo optimizavimas. Kai kurie skirtumai:

Vektorių ALU (ang.: Arithmetic Locgic Unit) resursai CPU (SSE/AVX) (ang.: Streaming SIMD Extensions/Advanced Vector Extensions), tam, kad būtų galima naudoti sutrumpintą SSE kodo generaciją, taip padidinant veikimo greitį, reikalauja vektorinių tipų naudojimo, kaip float4 ar kiti.



3.1. Pav.: OpenCL atminties tipai

GPU Vektorių ALU įranga yra lankstesnė, gali efektyviau naudoti slankiojančio kablelio procesorius. Kodas naudojantis keturių vienetų ilgio vektorius dažniausiai sugeneruoja geros kokybės kodą ir CPU ir GPU.

AMD OpenCL CPU implementacijoje darbo vienetai iš tos pačios darbo grupės vykdomi vienas po kito tame pačiame fiziniame branduolyje. Dažna GPU algoritmų strategija optimaliam prieigos prie atminties suvienijimui yra darbo vienetams iš vienos darbo grupės prieiti prie vienos spartinančiosios atminties eilutės. GPU šie darbo vienetai yra vykdomi lygiagrečiai ir sugeneruoja suvienytą prieigą prie atminties. CPU pirmas darbo vienetas vykdomas iki galo (arba iki barjero), po to leidžiami sekantys. Įprastai, jei duomenų rinkinys, naudojamas vieno darbo vieneto, telpa į CPU spartinančiąją atmintį, ši strategija yra efektyvi. Pirmas darbo vienetas į spartinančiąja (L2 ar L3) atmintį įrašo duomenis, kuriuos vėliau gali naudoti sekantys darbo vienetai. Dideliems duomenų kiekiams, netelpantiems į L2 ir L3 ši strategija nėra tokia efektyvi. Kiekvienas darbo vienetas per naują perkelia atminties eilutes, kurias buvo perkėlę prieš veikę darbo vienetai, bet kurios buvo išmestos iš spartinančiosios prieš jomis pasinaudojant [16].

CPU neturi įrangos sukurtos būtent tam, kad spartintų prieigą prie vietinės atminties. CPU lokali atminties priskiriama prie to paties galimo įrašyti į spartinančiąją atmintį DRAM kaip ir globali atmintis, todėl kintamųjų aprašymas kaip lokalių negerina greitaveikos [14].

CPU palaiko nedaug fizinių gijų, dažniausiai nuo 2 iki 8. Mažas darbo grupių kiekis mažina CPU ciklus išleidžiamus gijų valdymui. GPU turi dedikuotą aparatinę įrangą gijų valdymui.

4. GALIOS SRAUTŲ UŽDAVINYS

Galios srautų skaičiavimai yra vienas tankiausiai pasitaikančių uždavinių elektros tinklų analizėje. Tam dažnai naudojamas Niutono-Rafsono metodas, naudojamas nuo septinto dešimtmečio [17]. Galios srautų uždavinyje ieškomos sistemos taškų įtampos, galios srautai linijose prie duotos apkrovos, generacijos ir tinklo konfigūracijos. Galios srautų uždavinys sprendžiamas netiesine lygčių sistema:

$$0 = \Delta P_i = P_i^{inj} - V_i \sum_{j=1}^{N_{bus}} V_j Y_{IJ} \cos\left(\delta_i - \delta_j - \phi_{ij}\right) \tag{1}$$

$$0 = \Delta Q_i = Q_i^{inj} - V_i \sum_{j=1}^{N_{bus}} V_j Y_{IJ} \sin\left(\delta_i - \delta_j - \phi_{ij}\right) \tag{2}$$

$i = 1, \ldots, N_{bus}$

Čia P_i^{inj} ir Q_i^{inj} aktyvioji ir reaktyvioji galia mazge i, V_i efektyvioji įtampos vertė prie šynos i, δ_i - įtampos fazoriaus kampas prie šynos i. Y_{ij} ir ϕ_{ij} yra ij-tieji laidžių matricos elementai. N_{bus} – bendras elektros tinklo mazgų skaičius minus vienas.

Sistema dažniausiai sprendžiama Niutono-Rafsono metodu. Metodas pradedamas spėjant pradines įtampų vertes ir kampus tinkle ir iteraciniu būdų tikslina rezultatą [18], mažinant ΔP_i ir ΔQ_i kuo arčiau nulio. Tada prijungtoji prie mazgų galia ir galia suskaičiuota iš įtampos parametrų yra lygios. Įtampos ir fazės kampai atnaujinami kiekvienos iteracijos k metu suskaičiuojami iš:

$$\begin{bmatrix} J_1^{(k)} & J_2^{(k)} \\ J_3^{(k)} & J_4^{(k)} \end{bmatrix} \begin{bmatrix} \Delta\delta_1^{(k)} \\ \Delta\delta_2^{(k)} \\ \vdots \\ \Delta\delta_{N_{bus}}^{(k)} \\ \Delta V_1^{(k)} \\ \Delta V_2^{(k)} \\ \vdots \\ \Delta V_{N_{bus}}^{(k)} \end{bmatrix} = - \begin{bmatrix} \Delta P_1^{(k)} \\ \Delta P_2^{(k)} \\ \vdots \\ \Delta P_{N_{bus}}^{(k)} \\ \Delta Q_1^{(k)} \\ \Delta Q_2^{(k)} \\ \vdots \\ \Delta Q_{N_{bus}}^{(k)} \end{bmatrix} \tag{3}$$

Jakobiano matricos diferencijuojant galio srauto lygtį ir yra:

$$\begin{bmatrix} J_1^{(k)} & J_2^{(k)} \\ J_3^{(k)} & J_4^{(k)} \end{bmatrix} = \begin{bmatrix} \frac{\%\partial \Delta P}{\%\partial \delta} & \frac{\%\partial \Delta P}{\%\partial V} \\ \frac{\%\partial \Delta Q}{\%\partial \delta} & \frac{\%\partial \Delta Q}{\%\partial V} \end{bmatrix}_{\delta=\delta^{(k)}, V=V^{(k)}} \tag{4}$$

$$\delta = \left[\delta_1 \delta_2 \cdots \delta_{N_{bus}}\right], V = \left[V_1 V_2 \cdots V_{N_{bus}}\right] \tag{5}$$

Submatricos J_1, J_2, J_3 it J_4 yra dalinės galios srautų lygties išvestinės pagal fazę ir efektinę vertę. Išsprendus sistemą įtampų efektinės vertės ir fazės atnaujinamos [3]:

$$\delta_i^{(k+1)} = \delta_i^{(k)} + \Delta\delta_i^{(k)} \tag{6}$$

$$V_i^{(k+1)} = V_i^{(k)} + \Delta V_i^{(k)} \tag{7}$$

Čia $\delta_i^{(k)}$ ir $V_i^{(k)}$ yra uždavinio sprendiniai mazgui *i* iš iteracijos *k*. $\delta_i^{(k)}$ ir $V_i^{(k)}$ yra skirtumai apskaičiuoti iš (3).

Niutono-Rafsono metodas paverčia netiesinę lygčių sistemą į tiesinių lygčių sistemų seką. Tiesinės lygties sprendimas yra skaičiavimais brangiausia procedūros dalis. Svarbiausias yra šios dalies optimizavimas.

5. TRIFAZIO TRUMPOJO JUNGIMO SKAIČIAVIMAS

Trifazis trumpasis jungimas tyrime skaičiuojamas pagal IEC 60909 standartą [19], kuris skirtas momentinėms trumpojo jungimo srovėms skaičiuoti. Algoritme trumpojo jungimo vietoje jungiamas ekvivalentinis įtampos šaltinis, tinklas maitinamas stambesnio iš perdavimo tinklo. Algoritme numatomi du režimai, maksimalus ir minimalus. Abejų režimų kompiuteriniu skaičiavimo sudėtingumas yra ekvivalentus, toliau tolimesniuose palyginimuose bus analizuojamas tik maksimalus režimas. Algoritmo pradžioje pagal tinklo schemą sudaromos pradinės tinklo duomenų matricos, *A*, *B*, *e*, *z*, *y*.

Srovės tinklo šakose apskaičiuojamos pagal (8):

$$\begin{aligned} Z_k &= B^T \cdot z \cdot B \\ E_k &= -B^T \cdot e \\ I_k &= Z_k^{-1} \cdot e \\ i &= B \cdot I_k \end{aligned} \tag{8}$$

Įtampos tinklo mazguose apskaičiuojamos pagal (9):

$$\begin{aligned} Y_m &= A^T \cdot y \cdot A \\ I_m &= -A^T \cdot y \cdot e \\ U_m &= Y_m^{-1} I_m \end{aligned} \tag{9}$$

6. TIESINIŲ LYGČIŲ SISTEMŲ SPRENDIMO METODAI

Tiesinės lygčių sistemos sprendimas dažniausiai yra skaičiavimams brangiausia daugumos analizės būdų dalis [20]. Tiesinė sistema dažniausiai aprašoma:

$$\mathrm{Ax} = \mathrm{B} \tag{10}$$

$$\begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{n,n} \\ a_{2,1} & a_{2,2} & a_{2,3} & \cdots & a_{n,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & a_{n,3} & \cdots & a_{n,n} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \tag{11}$$

Čia *A* yra *N*×*N* dydžio koeficientų matrica, b yra *N*×*1* vektorius, toliau vadinamas RHS, *x* yra *N*×*1* nežinomas vektorius. Vienas iš būdų rasti *x,* yra apskaičiuoti matricos koeficientų matricos inversiją ir ją padaugini iš RHS vektoriaus:

$$\mathrm{x} = \mathrm{A}^{-1}\mathrm{b} \tag{12}$$

Matricos inversijos radimas yra skaičiavimais brangi operacija, užima papildomą atmintį, todėl buvo ieškoma būdų išspręsti lygtį neskaičiuojant matricos inversijos.

Metodai tiesinės lygčių sistemos sprendimui krenta į dvi pagrindines kategorijas: tiesioginius metodus ir netiesioginius metodus.

6.1. Tiesioginis LU Metodas

Tiesioginiai metodai išsprendžia sistemą per 1 iteraciją. Tiesioginiai metodai teoriškai gali surasti tikslų sprendinį atlikę baigtinį operacijų skaičių.

Dažniausiai pasitaikantis tiesioginio sprendimo metodas paremtas Gauso eliminacija. Pirmas Gauso eliminacijos žingsnis yra rasti pirmą nežinomąjį pagal kitus pirmos lygties nežinomuosius. Tada pirmasis nežinomasis pašalinamas iš likusios lygties, iš ko gaunama $N - 1$ lygčių it $x_2 \ldots x_N$ nežinomųjų.

$$\begin{aligned} a_{1,1}x_1 + \; & a_{1,2}x_2 + \; a_{1,3}x_3 + \; \cdots \; a_{1,N}x_N = b_1 \\ & a^{(1)}_{2,2}x_2 + \; a^{(1)}_{2,3}x_3 + \; \cdots \; a^{(1)}_{1,N}x_N = b^{(1)}_2 \\ & a^{(1)}_{1,2}x_2 + \; a^{(1)}_{1,3}x_3 + \; \cdots \; a^{(1)}_{1,N}x_N = b^{(1)}_2 \\ & \vdots \qquad\quad \vdots \qquad\quad \vdots \qquad\quad \vdots \\ & a^{(1)}_{N,2}x_2 + \; a^{(1)}_{N,3}x_3 + \; \cdots \; a^{(1)}_{N,N}x_N = b^{(1)}_N \end{aligned} \tag{13}$$

Indeksas (1) rodo tiesinės sistemos koeficientus po pirmo žingsnio. Kitame žingsnyje pašalinamas antrasis nežinomasis iš $N - 1$ lygčių. Procesas tęsiamas kol liekat tik paskutinis nežinomasis:

$$\begin{aligned} a_{1,1}x_1 + \; & a_{1,2}x_2 + \; a_{1,3}x_3 + \; \cdots \; a_{1,N}x_N = b_1 \\ & a^{(1)}_{2,2}x_2 + \; a^{(1)}_{2,3}x_3 + \; \cdots \; a^{(1)}_{1,N}x_N = b^{(1)}_2 \\ & \qquad a^{(2)}_{1,3}x_3 + \; \cdots \; a^{(2)}_{1,N}x_N = b^{(2)}_2 \\ & \qquad\qquad\qquad\qquad \vdots \\ & \qquad\qquad\qquad\qquad a^{(N-1)}_{N,N}x_N = b^{(N-1)}_N \end{aligned} \tag{14}$$

Ši lygtis su vienu nežinomuoju duoda paskutinio nežinomojo vertę. Kai randamas šis kintamasis, jo vertė įrašoma į prieš esančią lygtį ir randama x_{N-1}. Šis procesas tęsiamas kol randami visi nežinomieji.

LU algoritmas yra Gauso eliminacija paremtas bendros paskirties algoritmas tiesinių lygčių sprendimui. LU faktorizacijos metu koeficientų matrica išskirstoma į apatinę trikampę matricą (L) ir viršutinę trikampę matricą (U) [2]:

$$A = LU \tag{15}$$

$$\begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & a_{2,3} & \cdots & a_{2,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & a_{n,3} & \cdots & a_{n,n} \end{bmatrix} = \begin{bmatrix} l_{1,1} & 0 & 0 & \cdots & 0 \\ l_{2,1} & l_{2,2} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ l_{n,1} & l_{n,2} & l_{n,3} & \cdots & l_{n,n} \end{bmatrix} \begin{bmatrix} u_{1,1} & u_{1,2} & u_{1,3} & \cdots & u_{1,n} \\ 0 & u_{2,2} & u_{2,3} & \cdots & u_{2,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & u_{n,n} \end{bmatrix} \tag{16}$$

Tada tiesinė sistema tampa:

$$LUx = b \tag{17}$$

Vektorius gaunamas sudauginus x ir U vadinamas z:

$$Ux = z \tag{18}$$

Tada:

$$Lz = b \tag{19}$$

Šios lygties sprendimas paprastas. Pirmoje lygtyje yra tik pirmasis nežinomasis, antroje lygtyje yra pirmasis ir antrasis nežinomieji, o k-tojoje lygtyje yra pirmi k nežinomųjų. Lygtys sprendžiamos paeiliui ir paeiliui randami nežinomieji. Šis procesas panašus į Gauso eliminaciją, bet kadangi koeficientų matrica yra apatinės trikampės formos, nežinomieji yra randami beeliminuojant kintamuosius iš apatinių lygčių. Kai vektorius z yra žinomas, iš (18) apskaičiuojamas x. Ši sistema sprendžiama analogiškai iš kito galo.

Yra keli būdai apskaičiuoti koeficientų matricoms L ir U. Pagrindiniai yra Krauto faktorizacija, Dolitlio faktorizacija ir Čoleskio faktorizacija. Pagrindinė visų šių metodų idėja yra surašyti santykius tarp L ir U matricų verčių sekančiu būdu [21]:

$$\begin{aligned} l_{1,1}\,u_{1,1} &= a_{1,1} \\ l_{2,1}\,u_{1,1} &= a_{2,1} \\ l_{3,1}\,u_{1,1} &= a_{3,1} \\ &\vdots \end{aligned} \tag{20}$$

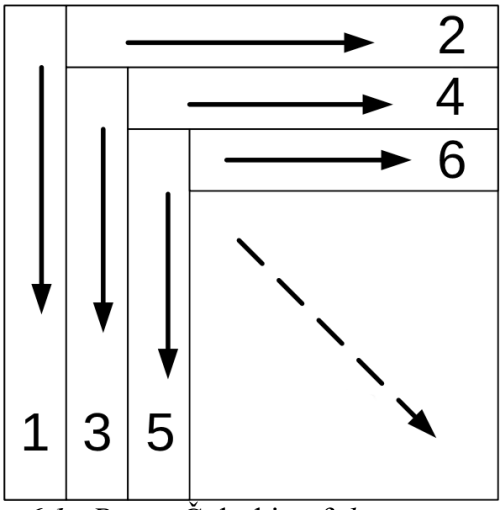
Krauto faktorizacijos metode priimama, kad elementai $u_{i,i}$ lygūs vienetui. Tada, pagal (20), elementai nuo $l_{1,1}$ iki $l_{N,1}$ būtų atitinkamai lygūs elementams nuo $a_{1,1}$ iki $a_{N,1}$ – pirmas matricos L stulpelis yra žinomas. Antrame žingsnyje apskaičiuojama:

$$\begin{aligned} l_{1,1}\,u_{1,2} &= a_{1,2} \\ l_{1,1}\,u_{1,3} &= a_{1,3} \\ l_{1,1}\,u_{1,4} &= a_{1,4} \\ &\vdots \end{aligned} \tag{21}$$

Kadangi $l_{1,1}$ jau žinomas, galima suskaičiuoti pirmąją U eilutę. Procesas kartojamas kol randamos L ir U matricos, grafiškai faktorizacijos tvarka atvaizduota 6.1 Pav..

Pseudokodas:

```
for j = 2 -> N do
    u1,i = ai,1/l1,1
end for
for k = 2 to N do
    for i = k -> N do
        li,k = li,k – (Σ(j=1 -> k-1) li,j × uj,k)
    end for
    for i = k + 1 -> N do
        uk,i = (uk,i – (Σ(j=1 -> k-1) lk,j × uj,i))/lk,k
    end for
end for
```



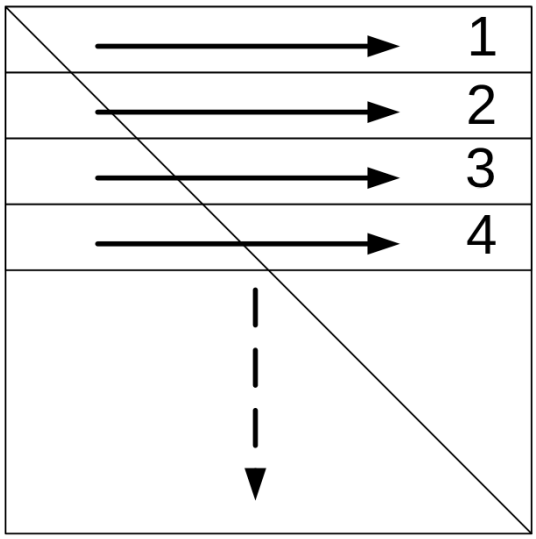
6.1 Pav.: Čoleskio f*aktorizacijos tėkmė*

L stulpeliai skaičiuojami nelyginių žingsnių metu, U eilutės apskaičiuojamos lyginių žingsnių metu. Sekančiam žingsniui reikalingi prieš esančio žingsnio rezultatai, todėl šio metodo paralelizacija yra ribota, tolesniuose proceso žingsniuose paralelizacijos galimybės vis mažesnės.

Dolitlio metode priimama, kad visos elementų $l_{i,i}$ reikšmės yra lygios vienetui. Vieno žingsnio metu randama viena faktorizacijos matricos eilutė. L ir U elementams, apskaičiuojamiems vieno žingsnio metu, reikalingi to paties žingsnio rezultatai. Tam reikalinga daug tarpgijinės komunikacijos, kas kels vėlinimą dėl atminties, faktorizacijos tvarka grafiškai atvaizduota 6.2 Pav..

Jo pseudokodas [21]:

```
for k = 2 -> N do
    for i = 1 -> k - 1 do
        lk,i = (lk,i - (Σ(j=1 -> k-1) lk,j*uj,i))/ui,i
    end for
    for i = k -> N do
        uk,i = uk,i - (Σ(j=1 -> k-1) lk,j*uj,i)
    end for
end for
```



6.2 Pav.: Dolitlio faktorizacijos tėkmė

Čoleskio faktorizacija yra specialus LU faktorizacijos metodas, iš kurio gaunama LL^{t}. Jei koeficientų matrica yra simetriška teigiama, elementų $l_{i,i}$ reikšmės yra realios, kitu atveju jos yra kompleksinės. Čoleskio būdas elektros sistemų analizei naudojamas retai, nes matricos su kuriomis dirbama dažniausiai nebūna simetrinės ir neteigiamos.

6.2. Netiesioginis metodas

Netiesioginiai (iteraciniai) metodai pradeda aproksimuojant sprendimą ir iteruojant jį tikslina. Apytikris sprendimas gali konverguoti į tikslų sprendimą per baigtinį arba begalinį iteracijų kiekį. Skaičiavimus galima nutraukti pasiekus norimą tikslumą. Šiuo atveju bus kalbama apie metodą Krylovo pagrindu: jungtinio gradiento (CG) metodą. CG yra tinkamas lygiagrečių skaičiavimų platformoms, nes jo naudojamos operacijos našiai įgyvendinamos šiose platformose. CG metodas garantuotai konverguoja tik, kai koeficientų matrica simetriška teigiama. Kitas būdas, bi-jungtinio gradiento algoritmas (BiCG) [22][23], tinkamas simetrinėms ir nesimetrinėms sistemoms [24].

CG metodas originaliai buvo sukurtas suprastinti kvadratiniai lygčiai suprastinti:

$$f(x)=\frac{1}{2}x^{T}Ax-b^{T}x+c \tag{22}$$

Čia A simetrinė N dydžio matrica, b ir x yra $N\times 1$ dydžio vektorius, c yra skaliaras.

Funkcijos $f(x)$ gradientas:

$$f'(x)=\frac{1}{2}A^{T}x+\frac{1}{2}Ax-b \tag{23}$$

Kadangi A yra simetriška matrica, $A^{T}=A$:

$$f'(x) = Ax - b \tag{24}$$

Bet kokiam vektoriui *p* lygtis gali būti pertvarkyta į:

$$f(p) = f(x) + \frac{1}{2}(p-x)^T A(p-x) \tag{25}$$

Stačiausio nusileidimo metodas pradedamas pasirenkant bet kokį pradinį sprendinį $x^{(0)}$, ir juda į priešingą gradiento vektoriui pusę kiekvienos iteracijos metu, kol sprendinys priartėja prie atsakymo x. $f'(x)$ rodo didžiausia $f(x)$ kitimo linkme, todėl taip f mažės greičiausiai. k-tajai iteracijai stačiausio nusileidimo metodas judės $-f'(x^{(k)})$ linkme

$$-f'(x^{(k)}) = b - Ax^{(k)} = r^{(k)} \tag{26}$$

Vektorius $r^{(k)}$ yra likutinė k-tosios iteracijos vertė. Greičiausio nusileidimo metodas juda likutinės vertės linkme. (k+1)-oji iteracija:

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} r^{(k)} \tag{27}$$

Čia $\alpha^{(k)}$ k-tosios iteracijos žingsnio dydis. f minimizavimui α atžvilgiu, išvestinė $\frac{d}{d\alpha} f(x^{(k)})$ prilyginama nulius, tada:

$$\frac{d}{d\alpha} f(x^{(k)}) = f'(x^{(k)})^T \frac{d}{d\alpha}(x^{(k)}) = f'(x^{(k)})^T r^{(k-1)} = 0 \tag{28}$$

Jei $f'(x^{(k)})^T r^{(k-1)} = 0$, tada likutinis ir gradiento vektoriai yra statmeni, iš to:

$$r^{(k)T} r^{(k-1)} = 0 \tag{29}$$

Tam, kad gauti a formulę naudojamos sekančios formulės:

$$\begin{gathered} r^{(k)T} r^{(k-1)} = 0 \\ (b - Ax^{(k)})^T r^{(k-1)} = 0 \\ (b - A(x^{(k-1)} + \alpha^{(k)} r^{(k-1)}))^T r^{(k-1)} = 0 \\ (b - Ax^{(k-1)}) r^{(k-1)} - \alpha^{(k)} (Ar^{(k-1)})^T r^{(k-1)} = 0 \\ (b - Ax^{(k)})^T r^{(k-1)} = \alpha^{(k)} (Ar^{(k-1)})^T r^{(k-1)} \\ r^{(k-1)T} r^{(k-1)} = a^{(k)} r^{(k-1)T} (Ar^{(k-1)}) \\ \alpha^{(k)} = \frac{r^{(k-1)T} r^{(k-1)}}{r^{(k-1)T} (Ar^{(k-1)})} \end{gathered} \tag{30}$$

Pseudokodu:

```
Pasirenkamas pradinis x(0), k=0
while nekonverguoja do
    r(k) = b - Ax(k)
    α(k) = (r(k-1)T r(k-1))/r((k-1)T Ar(k-1))
    x(k+1) = x(k) + α(k)r(k)
    if ||r(k)|| ≤ ε end, k++
```

end while

Iteracijų kiekis gali būti sumažintas, jei iteracijų kryptys yra sustatmeninamos

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} d^{(k)} \tag{31}$$

$$\forall \, i, j = 1 \ldots n, d_i^T d_j = 0 \tag{32}$$

Kadangi (32) reikalauja statmenumo, (30) negali būti naudojama, todėl paieškos kryptis parenkama statmena A. Vektoriai d_i ir d_j statmeni A, jei:

$$d_i^T A d_j = 0 \tag{33}$$

Šie du A statmeni vektoriai vadinami jungtiniais vektoriais. Vektoriai randami pradedant nuo bet kokio tiesiškai nepriklausomų vektorių rinkinio $u_0, u_1, \ldots, u_{N-1}$. Vektorius d_i sudaromas, kaip u_i ir buvusių jungtinių krypčių kartotinių suma:

$$\begin{aligned} d_0 &= u_0 \\ d_i &= u_i + \sum_{k=0}^{i-1} \beta_{ok} d_i^T A d_j \end{aligned} \tag{34}$$

β_{ik} radimui naudojama A statumo charakteristika:

$$\begin{aligned} 0 = d_i^T A d_j &= u_i^T A u_j + \sum_{k=0}^{i-1} \beta_{ik} d_i^T A d_j \\ \beta_{ij} &= \frac{-u_i^T A d_j}{d_i^T A d_j} \end{aligned} \tag{35}$$

Kai žinomas β_{ik}, kai $k = 0..i$, iš (34) galima rasti d_i.

CG metodas tinkamas simetrinėms teigiamoms sistemoms. Nesimetrinėms sistemoms dažnai naudojamas BiCG metodas. BiCG metode sukuriamos dvi statmenos sekos paremtos A ir A^T. Liekamųjų verčių sekos:

$$r^{(i)} = r^{(i-1)} - \alpha^{(i)} A p^{(i)} \quad , \quad \tilde{r}^{(i)} = \tilde{r}^{(i-1)} - \alpha^{(i)} A \tilde{p}^{(i)} \tag{36}$$

paieškos dimensijų sekos:

$$p^{(i)} = r^{(i-1)} + \beta^{(i-1)} p^{(i-1)} \quad , \quad \tilde{p}^{(i)} = \tilde{r}^{(i-1)} + \beta^{(i-1)} \tilde{p}^{(i-1)} \tag{37}$$

BiCG pseudokodas:

```
Pasirenkamas pradinis x(0), k = 1
r(0) = b - Ax(0)
Pasirenkama r̄ (pvz. r̄ = r(0))
do
    ρk-1 = r̄ᵀr(k-1)
    β(k-1) = (ρ(k-1)/ρ(k-2))/(α(k-1)/ω(k-1))
    p(k) = r(k-1) + β(k-1)(p(k-1) - ω(k-1)v(k-1))
```

$\tilde{\tilde{p}} = M^{-1}p^{(k)}$

$v^{(k)} = A\tilde{\tilde{p}}$

$\alpha(k) = \rho^{(k-1)}/\bar{r}^{T}v^{(k)}$

$s = r^{(k-1)} - \alpha^{(k)}v^{(k)}$

if $||s|| \leq \varepsilon_1$ {**break,** $x^{(k)} = x^{(k-1)} + \alpha^{(k)}\tilde{\tilde{p}}$}

$\tilde{\tilde{s}} = M^{-1}s$

$t = A\tilde{\tilde{s}}$

$\omega^{(k)} = t^{T}s \;/\; t^{T}t$

$x^{(k)} = x^{(k-1)} + \alpha^{(k)}\tilde{\tilde{p}} + \omega^{(k)}\tilde{\tilde{s}}$

$r^{(k)} = s - \omega^{(k)}t$

while $||r^{(k)}|| \leq \varepsilon_2$

6.3. Čebyšovo metodas

Pagrindinė išankstinio parengimo idėja yra koeficientų matricos inversijos aproksimacija naudojant matricos daugianarius. Pats paprasčiausias daugianario išankstinis parengimas yra atliekamas naudojantis inversijos aproksimacija, kuri naudoja Neumano eilutes.

$$A^{-1} = \sum_{k=1}^{\infty} N^{k} \tag{38}$$

N ir A yra tokio pačio dydžio matricos. Neumano eilutės sprendinys yra lygus tiksliam matricos A vaizdui, jeigu A = I – N ir matricos N spektrinis spindulys yra mažesnis už vienetą. Atitinkama išankstinio parengimo matrica gali būti gaunama sumuojant tik baigtinius Neumano eilutės skaičius:

$$M_r = \sum_{k=1}^{r} N^{k} \tag{39}$$

kur r yra polinomo skaičius naudojamas skaičiuojant daugianario išankstinį parengimą.

Čebyšovo algoritmas yra iteracinis metodas sukurtas skaliarinio dydžio inversijos aproksimacijai. Atliktos studijos rodo, kad Čebyšovo metodas naudojant matricas gali būti naudojamas kaip daugianario alternatyvus išankstinio parengimo būdas. Šiame metode Čebyšovo daugianariai yra skaičiuojami rekursiškai naudojant linijinės sistemos koeficientų matricą kaip argumentą. Didinant iteracijų skaičių Čebyšovo daugianarių linijinė kombinacija konverguoja į koeficientų matricos inversiją [5].

Pirmas žingsnis norint Čebyšovo metodu atlikti išankstinį parengimą yra Z matricos sudarymas. Diagonaliosios A matricos tikrines vertės yra perstumiamos į [-1, 1] intervalą:

$$z = \frac{2}{\beta - \alpha} A D^{-1} - 1 \tag{40}$$

čia α ir β yra mažiausia ir didžiausia pakeistos A matricos tikrinės vertės.

β galima apytiksliai nustatyti naudojant laipsnio metodu, o α yra priskiriamas dydis priklausantis nuo tipinės Čebyšovo išankstinio parengimo elgsenos. Rekomenduojama priskirti $\alpha=\frac{\beta}{5}$ jeigu $r<5$ ir $\alpha=\frac{\beta}{\frac{r}{2}\text{x}\,5}$ jeigu $r\geq 3$, kur $[x]$ yra funkcija, kuri gražina didžiausią sveikąjį skaičių kuris yra mažesnis už x [4]. Jeigu koeficientų matrica turi žinomas spektrines charakteristikas, tada laipsninio metodo skaičiavimas gali būti praleistas ir naudojama apytikrė β reikšmė. Čebyšovo daugianariai yra skaičiuojami pagal šias lygtis:

$$T_0=I, T_1=Z \tag{41}$$

$$T_k=2ZT_{k-1}+c_kT_k \tag{42}$$

$$M_0=c_0 I \tag{43}$$

$$M_k=M_{k-1}+c_kT_k \tag{44}$$

Kuriose T_k žymi k-tąjį daugianarį. Pradinė M vertė yra nurodoma (43), kuri veda prie Čebyšovo pirminio parengimo matricos apibrėžimo. Pirminio parengimo matrica M yra linijinė kombinacija sudaryta iš daugianarių apskaičiuotų (41) ir (43) lygtyse. Šios linijos kombinacijos koeficientai yra apskaičiuoja pagal žemiau esančias formules:

$$c_k=\frac{1}{\sqrt{\alpha\beta}}(-q)^k \tag{45}$$

$$q=\frac{1-\sqrt{\frac{\alpha}{\beta}}}{1+\sqrt{\frac{\alpha}{\beta}}} \tag{46}$$

Jeigu r Čebyšovo daugianariai yra naudojami skaičiuojant Čebyšovo pirminį parengimą, gaunama pirminio parengimo matrica M_r, kurios išraiška pateikta (43). Matrica apskaičiuota tokiu metodu yra lygi Čebyšovo matricos inversijos aproksimacijai.

$$M_r=\frac{c_0}{2}I+\sum_{k=1}^{r}c_kT_k\approx A^{-1} \tag{47}$$

Skaliarinis dydis r yra naudojamas kaip viršutinė sumos riba, kuri nusako Čebyšovo iteracijų skaičių. Nuo šios ribos parinkimo priklauso sprendinio tikslumas, aprašyto metodo efektyvumas ir kompiuterinių resursų bei atminties reikalavimai.

7. GRAFINIO IR CENTRINIO PROCESORIAUS SKAIČIAVIMŲ SPARTOS ANALIZĖ

Testams naudojami 2 kompiuteriai, kurių parametrai pateikti 1 lentelėje.

1. Lentelė: Tyrimui naudoti kompiuteriai

	PC1	**PC2**
CPU	AMD FX-8350	Intel i5-6600k
RAM	DDR3 1600MHz, 8 GB	DDR3 1333MHz, 32 GB
GPU	AMD RX 480 4GB	AMD R7 370 2GB Intel HD Graphics 530 (integruota)

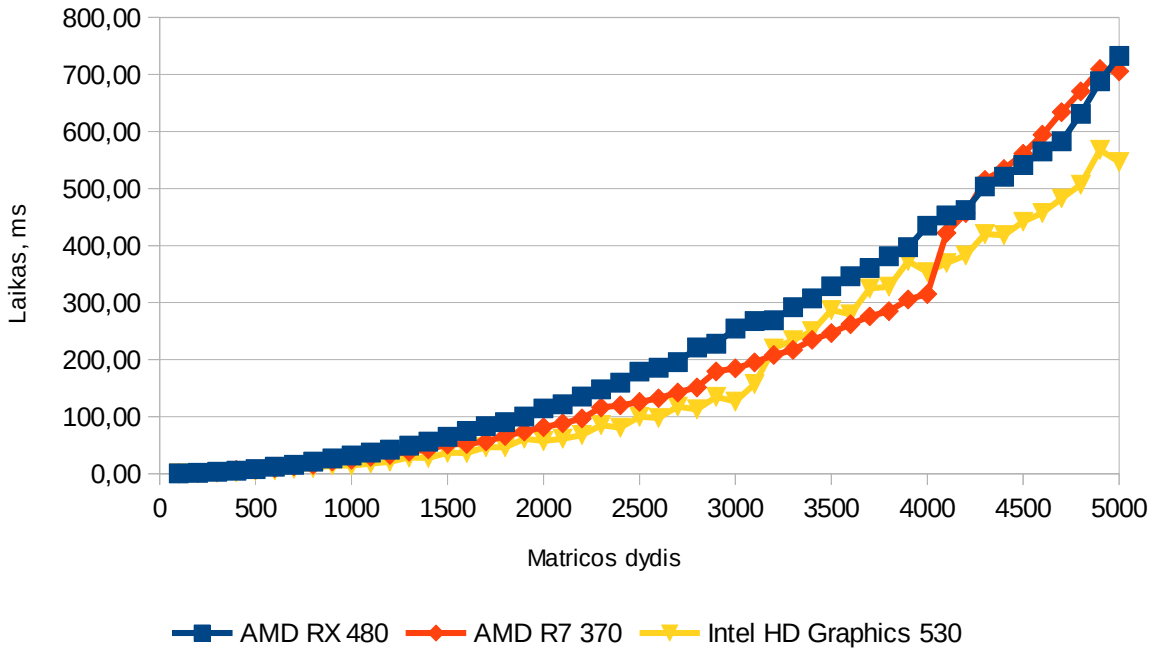
Pirmame kompiuteryje naudojamas 8 branduolių AMD Vishera architektūros procesorius, šioje architektūroje 2 branduoliai dalinasi vienu FPU, procesorius dirba 4,0 ~ 4,2 GHz dažniu. Šio kompiuterio vaizdo plokštėje naudojama AMD Polaris architektūra, GPU veikia 1306 MHz dažniu, naudojama atmintis yra GDDR5 tipo, veikia 1750 MHz dažniu.

Antrame kompiuteryje naudojamas 4 branduolių Intel Skylake architektūros procesorius veikiantis 3,5 ~ 3,9 GHz dažniu. Naudota diskreti vaizdo plokštė yra AMD Pitcairn Pro architektūros, veikia 1000 MHz dažniu, naudoja GDDR5 tipo 1400 MHz atmintį. Integruota vaizdo plokštė veikia 1150 MHz dažniu, naudoja sisteminę atmintį.

7.1. Kopijavimo testas

Šiuo testu tiriamas matricos perdavimo ir grąžinimo iš vaizdo plokštės laikas. Naudojama kvadratinė atsitiktinai sugeneruoja matrica.

Kopijos padarymo trukmės priklausomybė pavaizduota 7.1. Pav. Vidutinis perdavimo greitis: 0,54 GB/s, teorinis perdavimo greitis 8 GB/s (PCI Express x16). Vienu metu buvo atliekamas tik viena kopijavimo komanda. Tai yra neefektyvu, nes OpenCL tvarkyklė negali apjungti veiksmų, kas yra reikalinga efektyviam vaizdo plokštės veikimui. Kadangi kopijavimą galima atlikti asinchroniškai, daug efektyviau yra vienas po kito užduoti kelis veiksmus, tada gali efektyviai valdyti prieinamus kompiuterio resursus. Efektyvumą be to galima padidinti ir skaičiavimų metu tarpinius rezultatus laikant vaizdo plokštės atmintyje.

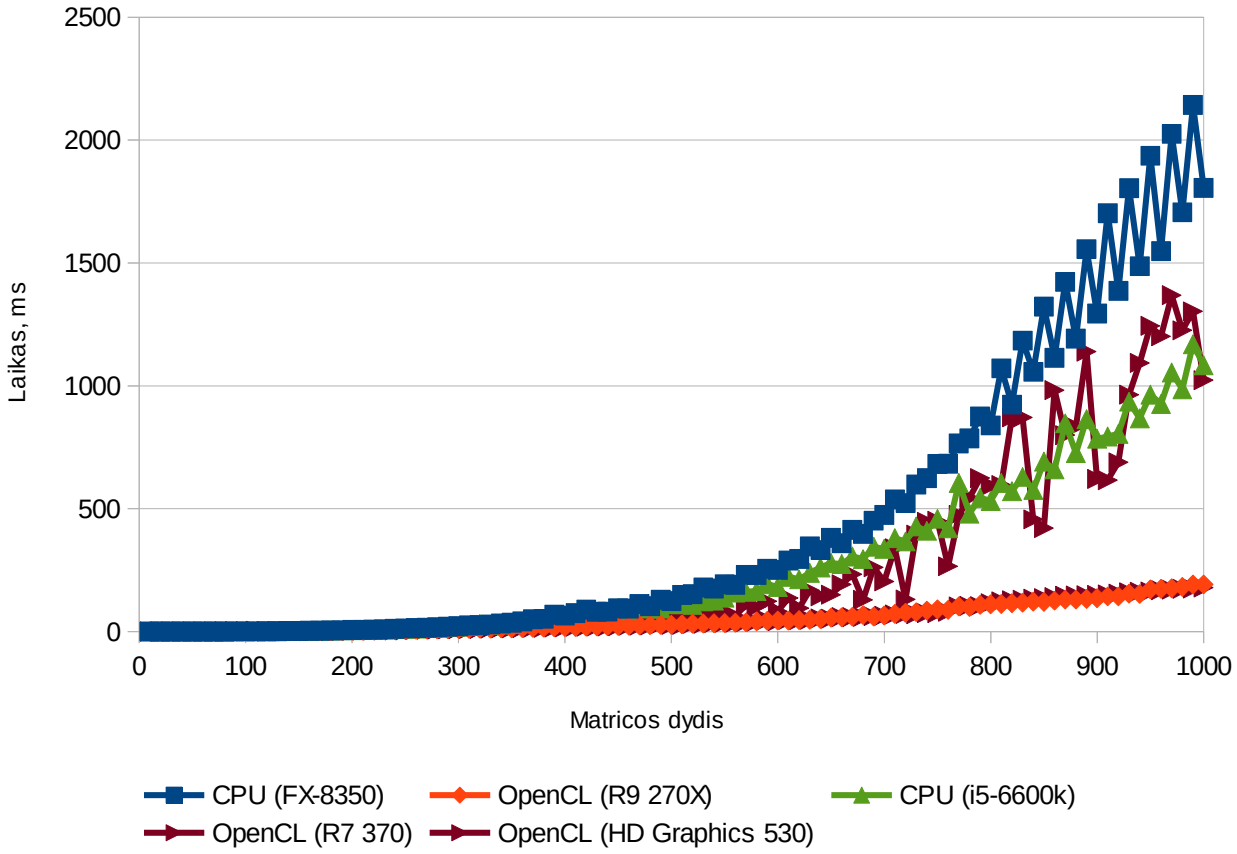


7.1. Pav.: Kopijavimo testas

7.2. Daugybos testas

Šiame teste kvadratu keliama atsitiktinai sugeneruota kvadratinė matrica

Matricų daugyba yra vienas lengviausių uždavinių, kuriuos galima atlikti vaizdo plokšte, kiekvieną rezultato matricos langelį galima apskaičiuoti nepriklausomai nuo kito, tai leidžia efektyviai išnaudoti vaizdo plokštės resursus. Veiksmo atlikimo laiko priklausomybė nuo matricos dydžio pateikta 7.2. Pav. Lėčiausias GPU rezultatas dauginant 1000 x 1000 dydžio matricas: 192,14 ms. Greičiausiai CPU rezultatas: 1272,04 ms. Pagreitėjimas: 6,6 ~ 10,1 karto.



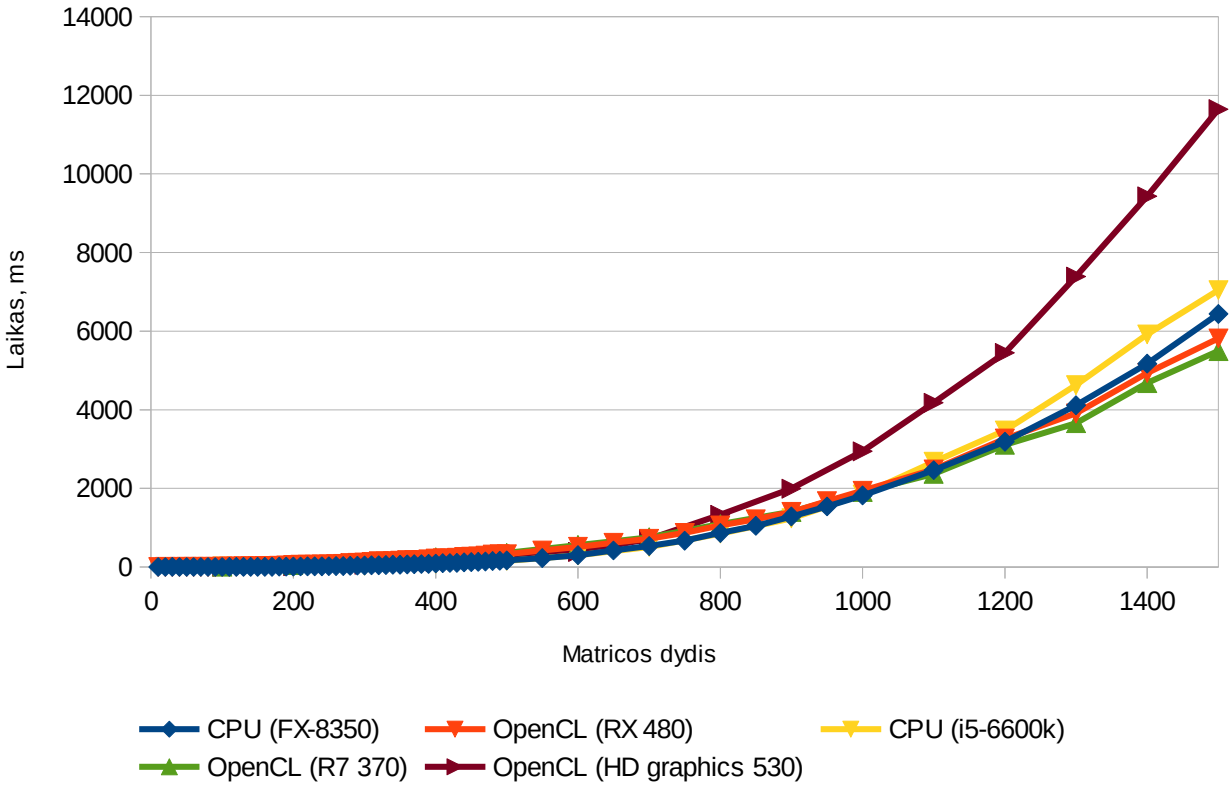
7.2. Pav.: Daugybos testas

7.3. Inversijos testas

Šiame teste inversija atliekama LU pagrindu, LU dalis atliekama procesoriuje

Šiame vaizdo plokštės ir centrinio procesoriau rezultatai yra panašūs. Pirma inversijos dalis (LU dekompozicija) atliekama procesoriuje, kas užima didelę laiko dalį, antra inversijos dalis GPU teste atliekama vaizdo plokštėje. Šis daug silpniau paralelizuojamas, paralelizuojama kiekviena atsakymo eilutė, o ne langelis, kaip buvo galima daryti daugyboje, todėl vaizdo plokštė įgauna pranašumą tik prie didesnių matricų.

1500 x 1500 matricos inversijos laikas CPU: 6438,03 ms, laikas GPU: 5816,43 ms. Pagreitėjimas: 1,11 karto. Vaizdo plokštė pralenkia centrinį procesorių kai matricos dydis 1100 x 1100.



7.3. Pav.: Inversijos testas

7.4. Elektrotechninių skaičiavimų greičio įvertinimas

Elektros tinklo galios srautų ir trumpųjų jungimų skaičiavimuose naudojamos 4 testinės schemos:

- 173 mazgų ir 242 jungčių, toliau vadinama 173 mazgų schema
- 345 mazgų ir 504 jungčių, toliau vadinama 345 mazgų schema
- 689 mazgų ir 1008 jungčių, toliau vadinama 689 mazgų schema
- 1377 mazgų ir 2016 jungčių. Toliau vadinama 1377 mazgų schema

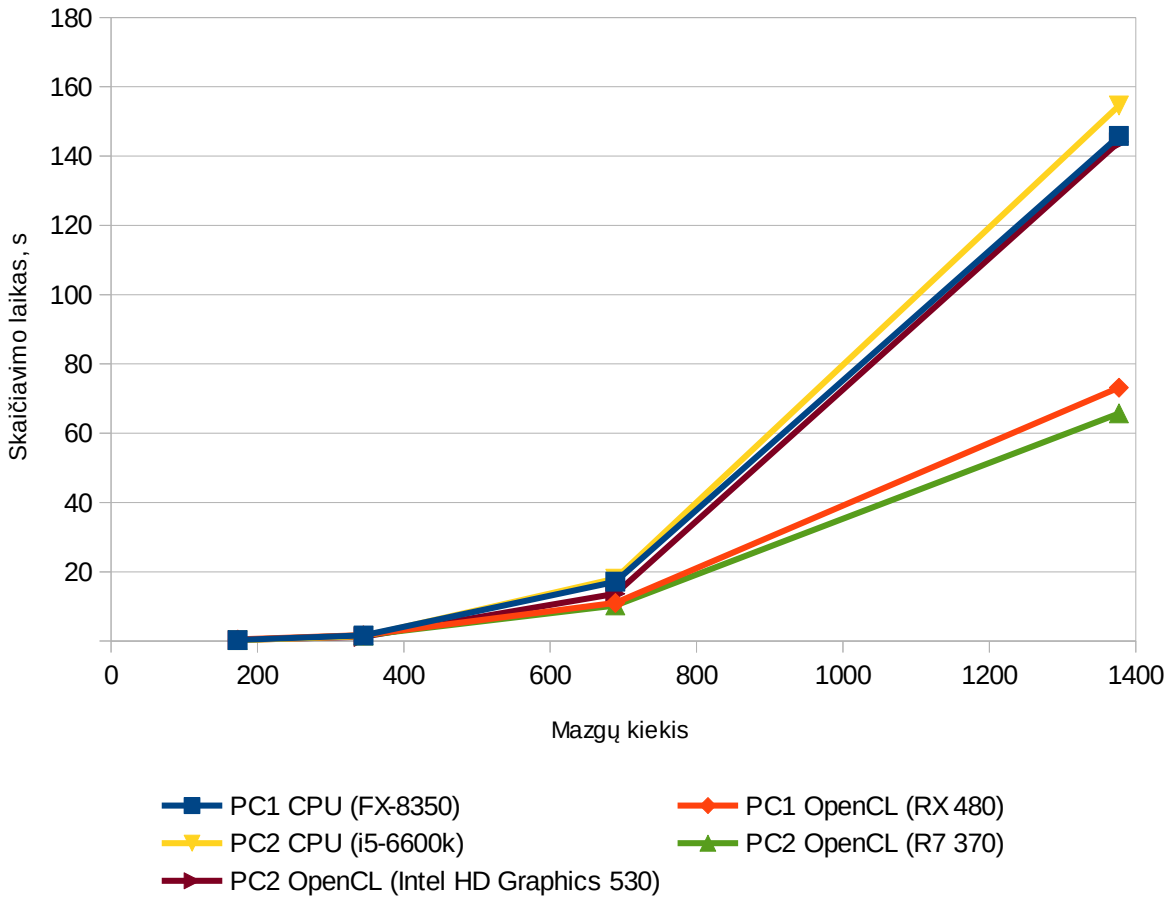
Tyrimu siekiama įvertinti skaičiavimo pagreitėjimą įprastuose elektros tinklo analizės uždaviniuose, 7.6 skyriuje vertinamas galios srautų uždavinio pagreitėjimas, 7.7 vertinamas trifazio trumpojo jungimo skaičiavimo pagreitėjimas.

7.5. Skaičiavimo metodo tikslumo vertinimas

Skaičiavimu tikslumas vertinamas IEEE standarto 44 šynų schema. Schema aprašomas 1 Priede. Skaičiavimo rezultatai pateikiami 3 Priede. Kaip matoma 3 Priedo lentelėse, rezultatai gauti skaičiuojant centriniu ir grafiniu procesoriumi yra vienodi, tikslumo atžvilgiu centriniai ir grafiniai procesoriai laikomi ekvivalenčiais.

7.6. Galios srautų uždavinys

Atlikti skaičiavimai 7.4 skyriuje aprašytose schemose, įvertintas skaičiavimo pagreitėjimas, rezultatai pateikti 7.4. Pav. ir 2 lentelėje.



7.4. Pav.: Galios srautų uždavinio skaičiavimo laiko priklausomybė nuo schemos dydžio

2. Lentelė: Galios srautų uždavinio užbaigimo laikai

Mazgų skaičius	**Skaičiavimo laikas, s**				
	CPU (FX-8350)	**CPU (i5-6600k)**	**OpenCL (RX 480)**	**OpenCL (R7 370)**	**OpenCL (HD Graphics 530)**
1377	145,79	73,13	154,59	65,72	144,05
689	17,14	11,01	18,04	10,27	13,63

Mazgų skaičius	Skaičiavimo laikas, s				
	CPU (FX-8350)	CPU (i5-6600k)	OpenCL (RX 480)	OpenCL (R7 370)	OpenCL (HD Graphics 530)
345	1,61	1,68	1,43	1,63	1,22
173	0,31	0,44	0,25	0,31	0,32

Vidutinis laikas CPU 1377 mazgų schemoje: 150,19 s. Vidutinis laikas GPU (diskretūs) 1377 mazgų schemoje: 69,43 s. Pagreitėjimas skaičiuojant naudojant GPU spartinimą siekia 2,16 karto lyginant su skaičiavimu naudojant vien centrinį procesorių.

Kaip matoma iš 2 lentelės ir 7.4. Pav. pagreitėjimas tuo geresnis, kuo didesnė yra schema. Mažesnėse schemose (345 ir 173 mazgų) skaičiavimo laikas panašus, skiriasi ne daugiau 27% (lyginant i5-6600k centrinį procesorių HD Graphics 530 grafinį procesorių), priklausomai nuo vaizdo plokštės. Tokio dydžio schemose santykinai daug laiko užima vaizdo plokštės valdymas ir duomenų transliavimas. Naudojant integruotą Intel HD Graphics 530 grafinį procesorių, kuris yra integruotas į Intel i5-6600k procesorių žymaus pagreitėjimo didesnėse schemose nėra, taip yra dėl šios vaizdo plokštės lėtumo.

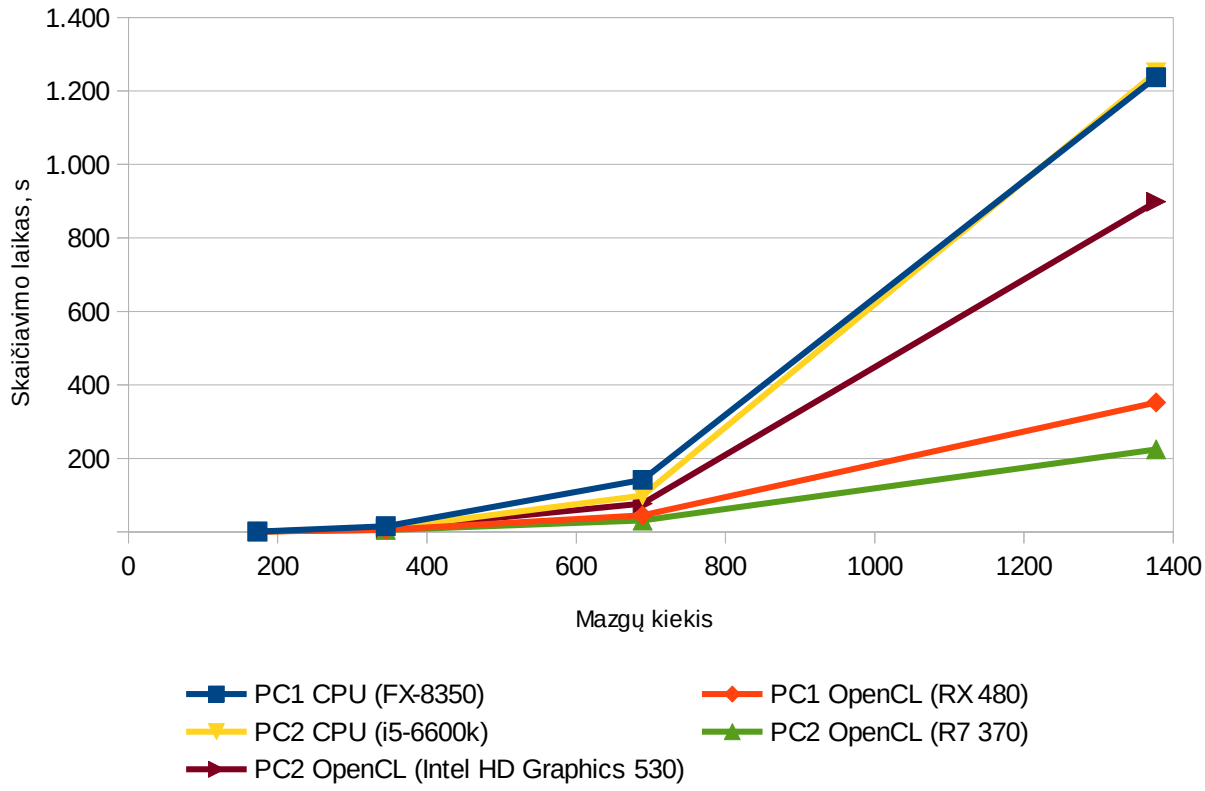
7.7. Trifazių trumpųjų jungimų skaičiavimas

Trifazio trumpojo jungimo pagreitėjimo tyrimas atliekamas naudojant tas pačias schemas kaip ir galios srautų uždavinyje, vertinamas skaičiavimo pagreitėjimas. Rezultatai pateikiami 3 lentelėje ir 7.5. Pav.

3. Lentelė: Trifazio trumpojo jungimo suskaičiavimo viename mazge laikai

Mazgų skaičius	Skaičiavimo laikas, ms				
	CPU (FX-8350)	CPU (i5-6600k)	OpenCL (RX 480)	OpenCL (R7 370)	OpenCL (HD Graphics 530)
1377	1237,44	1250,77	352,51	224,42	898,62
689	141,68	99,03	44,64	30,99	77,13
345	15,55	8,88	6,10	5,24	9,18
173	1,19	1,00	1,01	1,27	0,93

Vidutinis laikas CPU 1377 mazgų schemoje: 1244,11 s. Vidutinis laikas GPU (diskretūs) 1377 mazgų schemoje: 274,97 s. Pagreitėjimas skaičiavimus atliekant pasitelkiant vaizdo plokštę siekia 4,52 karto lyginant su skaičiavimu vien centriniu procesoriumi.



7.5. Pav.: Trifazio trumpojo jungimo skaičiavimo laiko priklausomybė nuo schemos dydžio

Trumpųjų jungimų skaičiavime pagreitėjimas žymiai didesnis, suskaičiuojama vidutiniškai per 4,51 karto mažesnį laiką. Šiame teste ir integruota vaizdo plokštė pagreitino skaičiavimą 1,39 karto, bet vis vien žymiai atsilieka nuo diskrečių vaizdo plokščių.

IŠVADOS

1. Vidutinės klasės vaizdo plokštės tinkamos 64 bitų slankiojančio kablelio skaičiavimams atlikti, tyrime nustatytas iki 10 kartų pagreitėjimas lyginant su centriniai procesoriais Tyrimu nustatyta, kad ilgiausiai trunkanti skaičiavimo operacija buvo matricos inversija, naudota tiesinei lygčių sistemai spręsti. Didžiausią įtaką skaičiavimų greičiui turėjo LU dekompozicija, kuri atliekama centriniu procesoriumi.
2. OpenCL sprendimai integruoti sėkmingai, optimaliam aparatinės įrangos išnaudojimui reikia optimizuoti algoritmus vengiant matricų inversijos
3. Skaičiavimų tikslumo pakitimas praktiškai nenustatytas
4. Atliktas skaičiavimų spartos tyrimas:
   a) Skaičiavimų atlikimas grafiniais procesoriai žymiai pagreitina uždavinius, kuriuose naudojama matricų algebra. Galios srautų uždavinys pagreitintas 2,16 karto, trifazio trumpojo jungimo skaičiavimas pagreitintas 5,42 karto.

b) Didesnėse schemose pasiekti didesni skaičiavimo greičiai, dėl didesnio lygiagrečių veiksmų skaičiaus lyginant su nuoseklių veiksmų skaičiumi.
c) Tyrimu nustatyta, kad būtina optimizuoti ne tik matricų algebrą, bet pakeisti galios srautų ir trumpųjų jungimų skaičiavimo algoritmus, pakeičiant matricų inversijas į matricų daugybą.

LITERATŪROS ŠALTINIAI

1. M. Garland, Sparse Matrix Computations on Manycore GPU's, Design Automation Conference, 2008. DAC 2008. 45th ACM/IEEE, 2008, ISBN: 978-1-60558-115-6
2. N. Galoppo, N. K. Govindaraju, M. Henson, D. Manocha, LU-GPU: Efficient Algorithms for Solving Dense Linear Systems on Graphics Hardware, Proceedings of the 2005 ACM/IEEE SC|05 Conference, 2005
3. H. Jiang, D. Chen, Y. Li, R. Zheng, A Fine-Grained Parallel Power Flow Method for Large Scale Grid Based on Lightweight GPU Threads, C2016 IEEE 22nd International Conference on Parallel and Distributed Systems, 2016, ISBN: 978-1-5090-4457-3 (ICPADS
4. X. Li, F. Li, Estimation of the largest eigenvalue in Chebyshev preconditioner for parallel conjugate gradient method-based power flow computation, IET Generation, Transmission & Distribution, 2015, p 123-130
5. X. Li, F. Li, GPU-based power flow analysis with Chebyshev preconditioner and conjugate gradient method, University of Tennessee, 2014
6. Aukšto dažnio prekybos sistemų modeliavimas finansų biržose naudojant GPU lygiagrečiųjų skaičiavimų architektūrą bei genetinius algoritmus. Magistro baigiamasis darbas, atliko: Justinas Lipnickas, vadovas dr. Aistis Raudys. Vilniaus universitetas, Matematikos ir informatikos fakultetas, Informatikos katedra. Vilnius, 2012 m.
7. Greitas ir tikslus objekto parametrų nustatymas mašininės regos sistemose. T. Kazakevičius, Vilniaus Gedimino technikos universitetas. 14-osios Lietuvos jaunųjų mokslininkų konferencijos „Mokslas – Lietuvos ateitis“ 2011 metų teminės konferencijos straipsnių rinkinys. ISBN 978-9955-28-835-0.
8. Resursų apskaitos sistemos prototipo projektas. VP1 - 3.1 - ŠMM - 08 - K - 01 – 012, „ Virtualizavimo , vizualizavimo ir saugose e.paslaugų technologijų kūrimas ir tyrimai“. 2014.
9. Sara S. Baghsorkhi Matthieu Delahaye Sanjay J. Patel William D. Gropp Wen-mei W. Hwu, An Adaptive Performance Modeling Tool for GPU Architectures, University of Illinois, 2010
10. AMD Accelerated Parallel Processing OpenCL Programming Guide. [žiūrėta 2017m. Gegužės 25d.] Prieiga per internetą: http://developer.amd.com/wordpress/media/2013/07/AMD_Accelerated_Parallel_Processing_OpenCL_Programming_Guide-rev-2.7.pdf

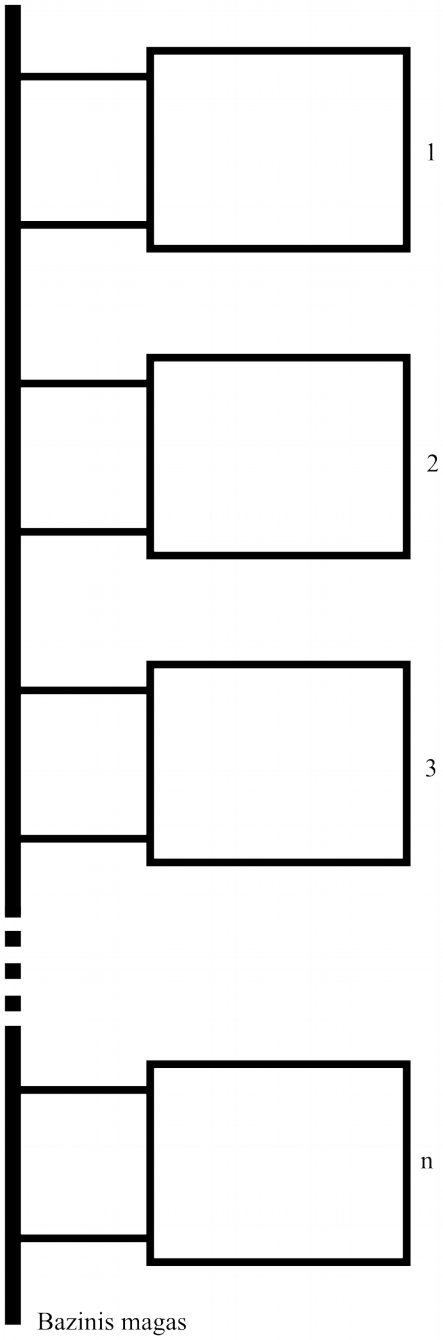
11. C. McClanahan. Georgia Tech College of Coumputing, History and Evolution of GPU Architecture. Georgia Tech, 2010
12. NVIDIA’s Next Generation CUDATCompute Architecture Fermi, 2009, [žiūrėta 2017m. Gegužės 25d.] Prieiga per internetą: http://www.nvidia.com/content/pdf/fermi_white_papers/nvidia_fermi_compute_architecture_whitepaper.pdf
13. KEPLER ARCHITECTURE, [žiūrėta 2017m. Gegužės 25d.] Prieiga per internetą: http://www.nvidia.com/object/nvidia-kepler.html
14. M. Segal, K. Akeley, The OpenGL: A Specification. [žiūrėta 2017m. Gegužės 25d.] Prieiga per internetą: https://www.opengl.org/registry/doc/glspec44.core.pdf
15. J. Tompson, K. Schlachter, An Introduction to the OpenCL Programming Model, NYU: Media Research Lab, 2012
16. S.J. Pennycooka, S.D. Hammondb, S.A. Wright, J.A. Herdmanc, I. Miller, S.A. Jarvis, An investigation of the performance portability of OpenCL, Journal of Parallel and Distributed Computing (Volume 73, Issue 11), 2012, p. 1439-1450
17. W. F. Tinney, C. E. Hart, Power Flow Solution by Newton's Method, IEEE Transaction on power apparatus and systems, 1967, p. 1449-1460
18. A. J. Flueck, H. D. Chiang, Solving the Nonlinear Power Flow Equations with an Inexact Newton Method Using GMRES, IEEE Transactions on Power Systems (Volume: 13, Issue: 2, May 1998), 1998, p. 267-273
19. IEC 60909-0:2001 Short-circuit currents in three-phase a.c. systems - Part 0: Calculation of currents, 2001
20. J. A. Meijerink, H. A. van der Vorst, An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix, Math. Comp, American Mathematical Society, 1977, p 148-162
21. T. C. Oppe, D. R. Kincaid, Parallel LU-Factorization Algorithms for Dense Matrices, Supercomputing, Springer Berlin/Heidelberg, 1988, p 576-594
22. I. C. Decker, D.M. Facao, E. Kaszkurewicz, Conjugate Gradient Methods for Power System Dynamic Simulation on Parallel Computers, IEEE Transactions on Power Systems, 1996, p 1218-1227
23. H. Dag, A. Semlyen, A New Preconditioned Conjugate Gradient Power Flow, IEEE TRANSACTIONS ON POWER SYSTEMS, VOL. 18, NO. 4, 2003, p 1248-1255
24. Y. Saad, Iterative Methods for Sparse Linear Systems Second Edition, Society for Industrial and Applied Mathematics, 2003. 520 p. ISBN: 978-0-89871-800-3

25. M. Scarpino, OpenCL in Action, Manning Publications Co., 2012. 458p. ISBN: 9781617290176
26. CUDA Toolkit Documentation v8.0. [žiūrėta 2017m. Gegužės 25d.] Prieiga per internetą: http://docs.nvidia.com/cuda/#axzz4Wlfz8NuZ
27. K. Anupindi, A. Skjellum, P. Coddington, G. Fox, Parallel Differential- Algebraic Equation Solvers for Power System Transient Stability Analysis, Syracuse University, 1993, ISBN: 0-8186-4980-1
28. JavaCL: OpenCL bindings for Java. [žiūrėta 2017m. Gegužės 25d.] Prieiga per internetą: http://javacl.googlecode.com
29. S. Xiao, W. Feng, Inter-Block GPU Communication via Fast Barrier Synchronization, 2010 IEEE International Symposium on. IEEE, 2010

PRIEDAI

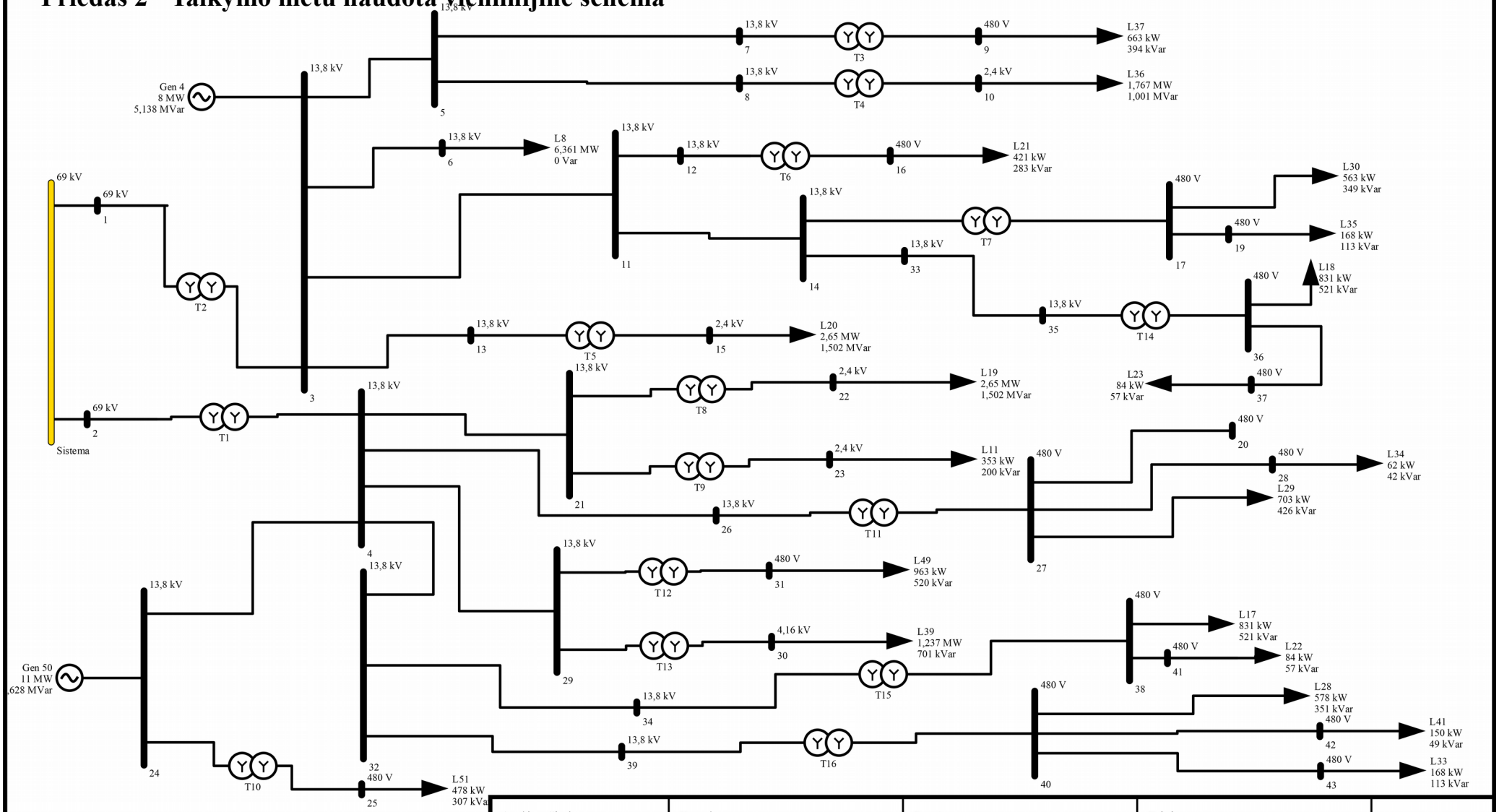
Priedas 1 Skaičiuojamosios schemos sudarymas

Skaičiuojamoji schema sudaroma lygiagrečiai sujungiant, reikiamą kiekį subschemų. Taip sudaroma reikiamo dydžio schema, skaičiavimų rezultatai atitinkamose pilnos schemos dalyse vienodi ir lengvai validuojami. Skaičiavimuose atskiros schemos dalys neekvivalentinamos. Detali subschema pateikta Priede 2.



P1.1. Pav.: Subschemų jungimo logika

Priedas 2 Taikymo metu naudota vienlinijinė schema



Atsakinga žinyba	Konsultantas	Rengė	Tvirtino	
KTU	dr. V. Šiožinys	Š. Stašaitis	dr. V. Šiožinys	

Elektros energetikos sistemų katedra	Documento tipas: Brėžinys	Document statusas: Tvirtinamas	
	Antraštė: Tyrimuose naudojamos schemos fragmentas	**20170531-01-MB-E**	
		Laida: A / Data: 2017-05-31 / Kalba: Lt / Lapas: 1/1	

Priedas 3 Galios srautų skaičiavimų rezultatai

4. Lentelė: Galios srautų uždavinio rezultatai mazguose skaičiuojant centriniu procesoriumi

Pavadinimas	Įtampos lygis	Ux	Ux, %	δ	Generacija	Vartojimas
Sistema	69 kV	69 kV	100 %	0	0 VA	0 VA
1	69 kV	68,959 kV	99,94 %	-0,099	0 VA	0 VA
2	69 kV	69,016 kV	100,023 %	0,049	0 VA	0 VA
3	13,8 kV	13,815 kV	100,111 %	-1,803	9,508 MVA	0 VA
4	13,8 kV	13,779 kV	99,851 %	0,873	0 VA	0 VA
5	13,8 kV	13,809 kV	100,068 %	-1,807	0 VA	0 VA
6	13,8 kV	13,809 kV	100,063 %	-1,837	0 VA	6,361 MVA
7	13,8 kV	13,808 kV	100,057 %	-1,808	0 VA	0 VA
8	13,8 kV	13,807 kV	100,047 %	-1,809	0 VA	0 VA
9	480 V	466,13 V	97,11 %	-3,788	0 VA	771,236 kVA
10	2,4 kV	2,336 kV	97,329 %	-4,056	0 VA	2,031 MVA
11	13,8 kV	13,815 kV	100,11 %	-1,804	0 VA	0 VA
12	13,8 kV	13,813 kV	100,091 %	-1,805	0 VA	0 VA
13	13,8 kV	13,801 kV	100,01 %	-1,808	0 VA	0 VA
14	13,8 kV	13,81 kV	100,073 %	-1,807	0 VA	0 VA
15	2,4 kV	2,339 kV	97,474 %	-3,978	0 VA	3,046 MVA
16	480 V	467,035 V	97,299 %	-3,424	0 VA	507,277 kVA
17	480 V	469,857 V	97,887 %	-3,274	0 VA	662,397 kVA
19	480 V	469,412 V	97,794 %	-3,273	0 VA	202,467 kVA
20	480 V	468,169 V	97,535 %	-0,674	0 VA	0 VA
21	13,8 kV	13,773 kV	99,801 %	0,869	0 VA	0 VA
22	2,4 kV	2,332 kV	97,176 %	-1,313	0 VA	3,046 MVA
23	2,4 kV	2,373 kV	98,86 %	0,191	0 VA	405,72 kVA
24	13,8 kV	13,812 kV	100,085 %	0,989	11,934 MVA	0 VA
25	480 V	473,206 V	98,585 %	0,054	0 VA	568,096 kVA
26	13,8 kV	13,777 kV	99,832 %	0,872	0 VA	0 VA
27	480 V	468,169 V	97,535 %	-0,674	0 VA	822,001 kVA
28	480 V	468,003 V	97,501 %	-0,674	0 VA	74,887 kVA
29	13,8 kV	13,776 kV	99,826 %	0,871	0 VA	0 VA
30	4,16 kV	4,024 kV	96,736 %	-1,486	0 VA	1,422 MVA
31	480 V	467,117 V	97,316 %	-0,967	0 VA	1,094 MVA
32	13,8 kV	13,774 kV	99,809 %	0,87	0 VA	0 VA
33	13,8 kV	13,808 kV	100,058 %	-1,808	0 VA	0 VA
34	13,8 kV	13,773 kV	99,803 %	0,87	0 VA	0 VA
35	13,8 kV	13,807 kV	100,051 %	-1,808	0 VA	0 VA
36	480 V	467,035 V	97,299 %	-3,656	0 VA	980,817 kVA
37	480 V	466,811 V	97,252 %	-3,656	0 VA	101,514 kVA
38	480 V	462,989 V	96,456 %	-1,32	0 VA	980,817 kVA
39	13,8 kV	13,766 kV	99,751 %	0,866	0 VA	0 VA
40	480 V	466,494 V	97,186 %	-0,967	0 VA	676,229 kVA
41	480 V	462,761 V	96,408 %	-1,32	0 VA	101,514 kVA
42	480 V	466,191 V	97,123 %	-0,976	0 VA	157,801 kVA
43	480 V	466,042 V	97,092 %	-0,966	0 VA	202,467 kVA

5. Lentelė: Galios srautų uždavinio rezultatai jungtyse skaičiuojant centriniu procesoriumi

Pavadinimas	Aktyvioji varža	Reaktyvioji varža	I	φ	ΔU	ΔU, %	Pij	Qij	ΔS
3 - 6	14,47 mΩ	17,52 mΩ	265,865 A	-1,059	10,464 V	0,076 %	6,361 MW	-82,619 kVar	4,772 kVA
3 - 5	22,472 mΩ	18,663 mΩ	118,795 A	-32,581	6,011 V	0,044 %	2,442 MW	1,455 MVar	1,199 kVA
5 - 8	15,045 mΩ	12,379 mΩ	86,03 A	-32,296	2,903 V	0,021 %	1,773 MW	1,044 MVar	397,249 VA
5 - 7	21,329 mΩ	17,71 mΩ	32,77 A	-33,331	1,574 V	0,011 %	668,135 kW	409,823 kVar	68,776 VA
3 - 11	100 nΩ	1,904 mΩ	103,948 A	-35,051	0,343 V	0,002 %	2,08 MW	1,364 MVar	1,759 VA
11 - 12	52,371 mΩ	43,611 mΩ	21,529 A	-36,333	2,541 V	0,018 %	424,408 kW	291,994 kVar	72,829 VA
3 - 13	51,419 mΩ	35,993 mΩ	128,884 A	-32,255	14,011 V	0,102 %	2,659 MW	1,563 MVar	3,094 kVA
11 - 14	27,233 mΩ	22,662 mΩ	82,427 A	-34,718	5,058 V	0,037 %	1,656 MW	1,072 MVar	685,357 VA
17 - 19	879 µΩ	564 µΩ	245,966 A	-36,25	0,445 V	0,093 %	167,922 kW	108,953 kVar	189,516 VA
4 - 21	20,758 mΩ	17,33 mΩ	147,763 A	-30,325	6,921 V	0,05 %	3,017 MW	1,827 MVar	1,734 kVA
4 - 24	23,234 mΩ	46,277 mΩ	476,412 A	158,345	42,729 V	0,31 %	-10,503 MW	-4,356 MVar	35,205 kVA
4 - 26	29,899 mΩ	24,948 mΩ	38,381 A	-31,922	2,589 V	0,019 %	770,017 kW	496,156 kVar	141,499 VA
27 - 28	879 µΩ	564 µΩ	92,172 A	-34,599	0,167 V	0,035 %	62,019 kW	41,713 kVar	26,581 VA
27 - 20	924 µΩ	891 µΩ	84,917 µA	89,326	0 V	0 %	0 W	-0,069 Var	0,069 VA
4 - 29	14,283 mΩ	11,998 mΩ	108,094 A	-29,895	3,492 V	0,025 %	2,217 MW	1,32 MVar	617,206 VA
4 - 32	28,566 mΩ	23,805 mΩ	91,026 A	-31,869	5,863 V	0,042 %	1,827 MW	1,175 MVar	887,331 VA
32 - 34	7,237 mΩ	6,094 mΩ	46,735 A	-33,211	0,766 V	0,006 %	923,43 kW	624,779 kVar	51,337 VA
33 - 14	20,948 mΩ	17,33 mΩ	45,907 A	145,165	2,162 V	0,016 %	-920,509 kW	-598,412 kVar	141,442 VA
33 - 35	8,76 mΩ	7,427 mΩ	45,907 A	-34,835	0,913 V	0,007 %	920,509 kW	598,412 kVar	56,877 VA
36 - 37	879 µΩ	564 µΩ	123,813 A	-36,766	0,224 V	0,047 %	83,892 kW	54,71 kVar	47,993 VA
32 - 39	80,747 mΩ	67,225 mΩ	44,319 A	-30,456	8,065 V	0,058 %	903,167 kW	549,7 kVar	582,821 VA
38 - 41	879 µΩ	564 µΩ	126,078 A	-35,113	0,228 V	0,048 %	84,024 kW	56,233 kVar	49,767 VA
40 - 42	790 µΩ	482 µΩ	195,008 A	-18,707	0,313 V	0,065 %	150,072 kW	48,01 kVar	105,542 VA
40 - 43	879 µΩ	564 µΩ	249,999 A	-34,62	0,452 V	0,094 %	168,145 kW	111,939 kVar	195,784 VA
T1 apvija1	1,49 Ω	25,348 Ω	22,719 A	-169,38	499,624 V	0,724 %	-2,67 MW	498,236 kVar	18,169 kVA
T1 apvija2	59,611 mΩ	1,014 Ω	113,613 A	-169,334	99,935 V	0,724 %	-2,671 MW	480,82 kVar	19,606 kVA
T3 apvija1	1,959 Ω	10,774 Ω	32,771 A	-33,335	310,784 V	2,252 %	668,066 kW	409,826 kVar	17,583 kVA
T3 apvija2	2,37 mΩ	13,034 mΩ	942,23 A	-33,341	10,81 V	2,252 %	664,911 kW	392,557 kVar	17,642 kVA
T4 apvija1	435,91 mΩ	4,358 Ω	86,032 A	-32,298	326,346 V	2,365 %	1,773 MW	1,044 MVar	48,571 kVA
T4 apvija2	13,184 mΩ	131,822 mΩ	494,693 A	-32,3	56,757 V	2,365 %	1,768 MW	995,52 kVar	48,629 kVA

Pavadinimas	Aktyvioji varža	Reaktyvioji varža	I	φ	ΔU	ΔU, %	Pij	Qij	ΔS
T5 apvija1	231,954 mΩ	2,783 Ω	128,885 A	-32,256	311,764 V	2,259 %	2,656 MW	1,561 MVar	69,538 kVA
T5 apvija2	7,016 mΩ	84,188 mΩ	741,101 A	-32,258	54,22 V	2,259 %	2,65 MW	1,492 MVar	69,597 kVA
T6 apvija1	2,864 Ω	14,317 Ω	21,53 A	-36,338	272,245 V	1,973 %	424,335 kW	291,994 kVar	10,095 kVA
T6 apvija2	3,464 mΩ	17,321 mΩ	619,055 A	-36,346	9,47 V	1,973 %	422,343 kW	282,126 kVar	10,154 kVA
T7 apvija1	1,11 Ω	7,215 Ω	36,523 A	-34,578	230,907 V	1,673 %	734,571 kW	472,873 kVar	14,549 kVA
T7 apvija2	1,343 mΩ	8,729 mΩ	1,05 kA	-34,583	8,032 V	1,673 %	732,35 kW	458,524 kVar	14,608 kVA
T8 apvija1	231,954 mΩ	2,783 Ω	130,589 A	-30,448	315,884 V	2,289 %	2,661 MW	1,619 MVar	71,391 kVA
T8 apvija2	7,016 mΩ	84,188 mΩ	750,896 A	-30,45	54,937 V	2,289 %	2,655 MW	1,548 MVar	71,449 kVA
T9 apvija1	1,062 Ω	6,902 Ω	17,178 A	-29,393	103,884 V	0,753 %	353,938 kW	206,509 kVar	3,032 kVA
T9 apvija2	32,113 mΩ	208,744 mΩ	98,784 A	-29,403	18,068 V	0,753 %	353,468 kW	203,542 kVar	3,09 kVA
T10 apvija1	1,217 Ω	7,198 Ω	24,111 A	-32,665	152,437 V	1,105 %	480,123 kW	319,653 kVar	6,308 kVA
T10 apvija2	1,473 mΩ	8,708 mΩ	693,247 A	-32,673	5,302 V	1,105 %	479,061 kW	313,465 kVar	6,367 kVA
T11 apvija1	1,11 Ω	7,215 Ω	38,382 A	-31,925	242,663 V	1,758 %	769,885 kW	496,105 kVar	16,074 kVA
T11 apvija2	1,343 mΩ	8,729 mΩ	1,104 kA	-31,93	8,441 V	1,758 %	767,432 kW	480,249 kVar	16,133 kVA
T12 apvija1	1,127 Ω	6,763 Ω	46,908 A	-29,024	278,512 V	2,018 %	970,327 kW	557,859 kVar	22,571 kVA
T12 apvija2	1,364 mΩ	8,182 mΩ	1,349 kA	-29,028	9,688 V	2,018 %	966,607 kW	535,626 kVar	22,63 kVA
T14 apvija1	1,11 Ω	7,215 Ω	45,909 A	-34,838	290,246 V	2,103 %	920,453 kW	598,425 kVar	23,021 kVA
T14 apvija2	1,343 mΩ	8,729 mΩ	1,32 kA	-34,842	10,096 V	2,103 %	916,944 kW	575,702 kVar	23,081 kVA
T15 apvija1	1,303 Ω	8,47 Ω	46,736 A	-33,214	346,865 V	2,514 %	923,383 kW	624,799 kVar	28,021 kVA
T15 apvija2	1,577 mΩ	10,247 mΩ	1,344 kA	-33,218	12,065 V	2,514 %	919,113 kW	597,134 kVar	28,08 kVA
T16 apvija1	1,11 Ω	7,215 Ω	44,32 A	-30,458	280,204 V	2,03 %	902,691 kW	549,364 kVar	21,452 kVA
T16 apvija2	1,343 mΩ	8,729 mΩ	1,274 kA	-30,463	9,746 V	2,031 %	899,421 kW	528,191 kVar	21,511 kVA
T2 apvija1	1,49 Ω	25,348 Ω	46,754 A	5,716	1,028 kV	1,49 %	5,556 MW	-565,79 kVar	81,764 kVA
T2 apvija2	59,611 mΩ	1,014 Ω	233,761 A	5,694	205,617 V	1,49 %	5,551 MW	-646,689 kVar	83,192 kVA
T13 apvija1	821,568 mΩ	6,573 Ω	61,197 A	-30,564	351,063 V	2,544 %	1,246 MW	761,544 kVar	37,153 kVA
T13 apvija2	74,657 mΩ	597,286 mΩ	203,015 A	-30,567	105,83 V	2,544 %	1,241 MW	724,705 kVar	37,208 kVA
Sistema - 1	661,779 mΩ	1,409 Ω	46,796 A	6,205	126,117 V	0,183 %	5,56 MW	-604,537 kVar	38,99 kVA
Sistema - 2	661,779 mΩ	1,409 Ω	22,649 A	-170,379	61,159 V	0,089 %	-2,669 MW	452,375 kVar	45,872 kVA

6. Lentelė: Galios srautų uždavinio rezultatai mazguose skaičiuojant grafiniu procesoriumi

Pavadinimas	Įtampos lygis	Ux	Ux, %	δ	Generacija	Vartojimas
Sistema	69 kV	69 kV	100 %	0	0 VA	0 VA
1	69 kV	68,959 kV	99,94 %	-0,099	0 VA	0 VA
2	69 kV	69,016 kV	100,023 %	0,049	0 VA	0 VA
3	13,8 kV	13,815 kV	100,111 %	-1,803	9,508 MVA	0 VA
4	13,8 kV	13,779 kV	99,851 %	0,873	0 VA	0 VA
5	13,8 kV	13,809 kV	100,068 %	-1,807	0 VA	0 VA
6	13,8 kV	13,809 kV	100,063 %	-1,837	0 VA	6,361 MVA
7	13,8 kV	13,808 kV	100,057 %	-1,808	0 VA	0 VA
8	13,8 kV	13,807 kV	100,047 %	-1,809	0 VA	0 VA
9	480 V	466,13 V	97,11 %	-3,788	0 VA	771,236 kVA
10	2,4 kV	2,336 kV	97,329 %	-4,056	0 VA	2,031 MVA
11	13,8 kV	13,815 kV	100,11 %	-1,804	0 VA	0 VA
12	13,8 kV	13,813 kV	100,091 %	-1,805	0 VA	0 VA
13	13,8 kV	13,801 kV	100,01 %	-1,808	0 VA	0 VA
14	13,8 kV	13,81 kV	100,073 %	-1,807	0 VA	0 VA
15	2,4 kV	2,339 kV	97,474 %	-3,978	0 VA	3,046 MVA
16	480 V	467,035 V	97,299 %	-3,424	0 VA	507,277 kVA
17	480 V	469,857 V	97,887 %	-3,274	0 VA	662,397 kVA
19	480 V	469,412 V	97,794 %	-3,273	0 VA	202,467 kVA
20	480 V	468,169 V	97,535 %	-0,674	0 VA	0 VA
21	13,8 kV	13,773 kV	99,801 %	0,869	0 VA	0 VA
22	2,4 kV	2,332 kV	97,176 %	-1,313	0 VA	3,046 MVA
23	2,4 kV	2,373 kV	98,86 %	0,191	0 VA	405,72 kVA
24	13,8 kV	13,812 kV	100,085 %	0,989	11,934 MVA	0 VA
25	480 V	473,206 V	98,585 %	0,054	0 VA	568,096 kVA
26	13,8 kV	13,777 kV	99,832 %	0,872	0 VA	0 VA
27	480 V	468,169 V	97,535 %	-0,674	0 VA	822,001 kVA
28	480 V	468,003 V	97,501 %	-0,674	0 VA	74,887 kVA
29	13,8 kV	13,776 kV	99,826 %	0,871	0 VA	0 VA
30	4,16 kV	4,024 kV	96,736 %	-1,486	0 VA	1,422 MVA
31	480 V	467,117 V	97,316 %	-0,967	0 VA	1,094 MVA
32	13,8 kV	13,774 kV	99,809 %	0,87	0 VA	0 VA
33	13,8 kV	13,808 kV	100,058 %	-1,808	0 VA	0 VA
34	13,8 kV	13,773 kV	99,803 %	0,87	0 VA	0 VA
35	13,8 kV	13,807 kV	100,051 %	-1,808	0 VA	0 VA
36	480 V	467,035 V	97,299 %	-3,656	0 VA	980,817 kVA
37	480 V	466,811 V	97,252 %	-3,656	0 VA	101,514 kVA
38	480 V	462,989 V	96,456 %	-1,32	0 VA	980,817 kVA
39	13,8 kV	13,766 kV	99,751 %	0,866	0 VA	0 VA
40	480 V	466,494 V	97,186 %	-0,967	0 VA	676,229 kVA
41	480 V	462,761 V	96,408 %	-1,32	0 VA	101,514 kVA
42	480 V	466,191 V	97,123 %	-0,976	0 VA	157,801 kVA
43	480 V	466,042 V	97,092 %	-0,966	0 VA	202,467 kVA

7. Lentelė: Galios srautų uždavinio rezultatai jungtyse skaičiuojant grafiniu procesoriumi

Pavadinimas	Aktyvioji varža	Reaktyvioji varža	I	φ	ΔU	ΔU, %	Pij	Qij	ΔS
3 - 6	14,47 mΩ	17,52 mΩ	265,865 A	-1,059	10,464 V	0,076 %	6,361 MW	-82,619 kVar	4,772 kVA
3 - 5	22,472 mΩ	18,663 mΩ	118,795 A	-32,581	6,011 V	0,044 %	2,442 MW	1,455 MVar	1,199 kVA
5 - 8	15,045 mΩ	12,379 mΩ	86,03 A	-32,296	2,903 V	0,021 %	1,773 MW	1,044 MVar	397,249 VA
5 - 7	21,329 mΩ	17,71 mΩ	32,77 A	-33,331	1,574 V	0,011 %	668,135 kW	409,823 kVar	68,776 VA

Pavadinimas	Aktyvioji varža	Reaktyvioji varža	I	φ	ΔU	ΔU, %	Pij	Qij	ΔS
3 - 11	100 mΩ	1,904 mΩ	103,948 A	-35,051	0,343 V	0,002 %	2,08 MW	1,364 MVar	1,759 VA
11 - 12	52,371 mΩ	43,611 mΩ	21,529 A	-36,333	2,541 V	0,018 %	424,408 kW	291,994 kVar	72,829 VA
3 - 13	51,419 mΩ	35,993 mΩ	128,884 A	-32,255	14,011 V	0,102 %	2,659 MW	1,563 MVar	3,094 kVA
11 - 14	27,233 mΩ	22,662 mΩ	82,427 A	-34,718	5,058 V	0,037 %	1,656 MW	1,072 MVar	685,357 VA
17 - 19	879 µΩ	564 µΩ	245,966 A	-36,25	0,445 V	0,093 %	167,922 kW	108,953 kVar	189,516 VA
4 - 21	20,758 mΩ	17,33 mΩ	147,763 A	-30,325	6,921 V	0,05 %	3,017 MW	1,827 MVar	1,734 kVA
4 - 24	23,234 mΩ	46,277 mΩ	476,412 A	158,345	42,729 V	0,31 %	-10,503 MW	-4,356 MVar	35,205 kVA
4 - 26	29,899 mΩ	24,948 mΩ	38,381 A	-31,922	2,589 V	0,019 %	770,017 kW	496,156 kVar	141,499 VA
27 - 28	879 µΩ	564 µΩ	92,172 A	-34,599	0,167 V	0,035 %	62,019 kW	41,713 kVar	26,581 VA
27 - 20	924 µΩ	891 µΩ	84,917 µA	89,326	0 V	0 %	0 W	-0,069 Var	0,069 VA
4 - 29	14,283 mΩ	11,998 mΩ	108,094 A	-29,895	3,492 V	0,025 %	2,217 MW	1,32 MVar	617,206 VA
4 - 32	28,566 mΩ	23,805 mΩ	91,026 A	-31,869	5,863 V	0,042 %	1,827 MW	1,175 MVar	887,331 VA
32 - 34	7,237 mΩ	6,094 mΩ	46,735 A	-33,211	0,766 V	0,006 %	923,43 kW	624,779 kVar	51,337 VA
33 - 14	20,948 mΩ	17,33 mΩ	45,907 A	145,165	2,162 V	0,016 %	-920,509 kW	-598,412 kVar	141,442 VA
33 - 35	8,76 mΩ	7,427 mΩ	45,907 A	-34,835	0,913 V	0,007 %	920,509 kW	598,412 kVar	56,877 VA
36 - 37	879 µΩ	564 µΩ	123,813 A	-36,766	0,224 V	0,047 %	83,892 kW	54,71 kVar	47,993 VA
32 - 39	80,747 mΩ	67,225 mΩ	44,319 A	-30,456	8,065 V	0,058 %	903,167 kW	549,7 kVar	582,821 VA
38 - 41	879 µΩ	564 µΩ	126,078 A	-35,113	0,228 V	0,048 %	84,024 kW	56,233 kVar	49,767 VA
40 - 42	790 µΩ	482 µΩ	195,008 A	-18,707	0,313 V	0,065 %	150,072 kW	48,01 kVar	105,542 VA
40 - 43	879 µΩ	564 µΩ	249,999 A	-34,62	0,452 V	0,094 %	168,145 kW	111,939 kVar	195,784 VA
T1 apvija1	1,49 Ω	25,348 Ω	22,719 A	-169,38	499,624 V	0,724 %	-2,67 MW	498,236 kVar	18,169 kVA
T1 apvija2	59,611 mΩ	1,014 Ω	113,613 A	-169,334	99,935 V	0,724 %	-2,671 MW	480,82 kVar	19,606 kVA
T3 apvija1	1,959 Ω	10,774 Ω	32,771 A	-33,335	310,784 V	2,252 %	668,066 kW	409,826 kVar	17,583 kVA
T3 apvija2	2,37 mΩ	13,034 mΩ	942,23 A	-33,341	10,81 V	2,252 %	664,911 kW	392,557 kVar	17,642 kVA
T4 apvija1	435,91 mΩ	4,358 Ω	86,032 A	-32,298	326,346 V	2,365 %	1,773 MW	1,044 MVar	48,571 kVA
T4 apvija2	13,184 mΩ	131,822 mΩ	494,693 A	-32,3	56,757 V	2,365 %	1,768 MW	995,52 kVar	48,629 kVA
T5 apvija1	231,954 mΩ	2,783 Ω	128,885 A	-32,256	311,764 V	2,259 %	2,656 MW	1,561 MVar	69,538 kVA
T5 apvija2	7,016 mΩ	84,188 mΩ	741,101 A	-32,258	54,22 V	2,259 %	2,65 MW	1,492 MVar	69,597 kVA
T6 apvija1	2,864 Ω	14,317 Ω	21,53 A	-36,338	272,245 V	1,973 %	424,335 kW	291,994 kVar	10,095 kVA
T6 apvija2	3,464 mΩ	17,321 mΩ	619,055 A	-36,346	9,47 V	1,973 %	422,343 kW	282,126 kVar	10,154 kVA
T7 apvija1	1,11 Ω	7,215 Ω	36,523 A	-34,578	230,907 V	1,673 %	734,571 kW	472,873 kVar	14,549 kVA
T7 apvija2	1,343 mΩ	8,729 mΩ	1,05 kA	-34,583	8,032 V	1,673 %	732,35 kW	458,524 kVar	14,608 kVA
T8 apvija1	231,954 mΩ	2,783 Ω	130,589 A	-30,448	315,884 V	2,289 %	2,661 MW	1,619 MVar	71,391 kVA
T8 apvija2	7,016 mΩ	84,188 mΩ	750,896 A	-30,45	54,937 V	2,289 %	2,655 MW	1,548 MVar	71,449 kVA
T9 apvija1	1,062 Ω	6,902 Ω	17,178 A	-29,393	103,884 V	0,753 %	353,938 kW	206,509 kVar	3,032 kVA
T9 apvija2	32,113 mΩ	208,744 mΩ	98,784 A	-29,403	18,068 V	0,753 %	353,468 kW	203,542 kVar	3,09 kVA
T10 apvija1	1,217 Ω	7,198 Ω	24,111 A	-32,665	152,437 V	1,105 %	480,123 kW	319,653 kVar	6,308 kVA
T10 apvija2	1,473 mΩ	8,708 mΩ	693,247 A	-32,673	5,302 V	1,105 %	479,061 kW	313,465 kVar	6,367 kVA
T11 apvija1	1,11 Ω	7,215 Ω	38,382 A	-31,925	242,663 V	1,758 %	769,885 kW	496,105 kVar	16,074 kVA
T11 apvija2	1,343 mΩ	8,729 mΩ	1,104 kA	-31,93	8,441 V	1,758 %	767,432 kW	480,249 kVar	16,133 kVA
T12 apvija1	1,127 Ω	6,763 Ω	46,908 A	-29,024	278,512 V	2,018 %	970,327 kW	557,859 kVar	22,571 kVA
T12 apvija2	1,364 mΩ	8,182 mΩ	1,349 kA	-29,028	9,688 V	2,018 %	966,607 kW	535,626 kVar	22,63 kVA
T14 apvija1	1,11 Ω	7,215 Ω	45,909 A	-34,838	290,246 V	2,103 %	920,453 kW	598,425 kVar	23,021 kVA
T14 apvija2	1,343 mΩ	8,729 mΩ	1,32 kA	-34,842	10,096 V	2,103 %	916,944 kW	575,702 kVar	23,081 kVA
T15 apvija1	1,303 Ω	8,47 Ω	46,736 A	-33,214	346,865 V	2,514 %	923,383 kW	624,799 kVar	28,021 kVA
T15 apvija2	1,577 mΩ	10,247 mΩ	1,344 kA	-33,218	12,065 V	2,514 %	919,113 kW	597,134 kVar	28,08 kVA
T16 apvija1	1,11 Ω	7,215 Ω	44,32 A	-30,458	280,204 V	2,03 %	902,691 kW	549,364 kVar	21,452 kVA
T16 apvija2	1,343 mΩ	8,729 mΩ	1,274 kA	-30,463	9,746 V	2,031 %	899,421 kW	528,191 kVar	21,511 kVA
T2 apvija1	1,49 Ω	25,348 Ω	46,754 A	5,716	1,028 kV	1,49 %	5,556 MW	-565,79 kVar	81,764 kVA
T2 apvija2	59,611 mΩ	1,014 Ω	233,761 A	5,694	205,617 V	1,49 %	5,551 MW	-646,689 kVar	83,192 kVA
T13 apvija1	821,568 mΩ	6,573 Ω	61,197 A	-30,564	351,063 V	2,544 %	1,246 MW	761,544 kVar	37,153 kVA
T13 apvija2	74,657 mΩ	597,286 mΩ	203,015 A	-30,567	105,83 V	2,544 %	1,241 MW	724,705 kVar	37,208 kVA
Sistema - 1	661,779 mΩ	1,409 Ω	46,796 A	6,205	126,117 V	0,183 %	5,56 MW	-604,537 kVar	38,99 kVA
Sistema - 2	661,779 mΩ	1,409 Ω	22,649 A	-170,379	61,159 V	0,089 %	-2,669 MW	452,375 kVar	45,872 kVA