TARTU ÜLIKOOL
MATEMAATIKA-INFORMAATIKATEADUSKOND
ARVUTITEADUSE INSTITUUT
TARKVARASÜSTEEMIDE ÕPPETOOL
Infotehnoloogia eriala

KAURI KÄGO
Füüsika-keemiateaduskond

Grid-tööde haldamise vahend bioinformaatika rakenduse näitel

Bakalaureusetöö (4 ap)

Juhendajad: Jaak Vilo, PhD
Eero Vainikko, PhD

Autor: ....." ". mai 2005
Juhendaja:....." ". mai 2005
Õppetooli juhataja: ....." "..... 2005

Tartu 2005

SISUKORD

LÜHENDID JA MÕISTED

LÜHENDID

API	Application Programming Interface	Rakendusliides
ARC	Advanced Resource Connector	NorduGridi vahevara
CE	Cluster Element	Arvutuselement
DARPA	Department of Defense's Advanced Research Projects Agency	USA Kaitseministeeriumi Uuringute agentuur
DNA	Deoxyribonucleic acid	Desoksüribonukleiinhape
GGF	Global Grid Forum	Globaalne Gridi Foorum
GIIS	Grid Index Information Service	Gridi indeksiteenus
GM	Grid Manager	Gridi haldur
GRAM	Globus Resource Allocation Manager	Globuse ressursileidja
GRIS	Grid Resource Information Service	Gridi ressursiinfo teenus
GSF	GridFTP Server	Grid failiedastusserver
GSI	Grid Security Infrastructure	Gridi turvainfrastruktuur
GT	Globus Toolkit	Globuse töövahendite komplekt
HPF	High Performance Fortran	Laiendite kogum Fortran 90 jaoks
IETF	Internet Engineering Task Force	Internetiehituse töörühm
LDAP	Lightweight Directory Access Protocol	Lihtsustatud kataloogi sirvimise protokoll
LSF	Load Sharing Facility	Ressursihaldur LSF
MDS	Metacomputing Directory Service	Metaarvutuste kataloogiteenus
MPI	Message Passing Interface	Teadete edastamise liides
OGSA	Open Grid Services Architecture	Avatud Griditeenuste arhitektuur
OGSI	Open Grid Services Infrastructure	Avatud Griditeenuste infrastruktuur
P2P	Peer-to-Peer	Partnervõrk, võrdõigusvõrk
PBS	Portable Batch System	Ressursihaldur PBS
PSE	Problem Solving Environment	Probleemi lahenduskeskkond
PVM	Parallel Virtual Machine	Virtuaalne paralleelmasin
RC	Replica Catalog	Koopiate kataloog
RFC	Request for Comments	Kommentaaride nõue
RPC	Remote Procedure Call	Kaugprotseduurikutsung
RSL	Resource Specification Language	Ressursi kirjeldus keel
SDK	Software Development Kit	Arendustarkvara programmikomplekt
SE	Storage Element	Salvestuselement
SMP	Symmetric multiprocessors	sümmeetrilised multiprotsessorid
SSL	Secure Sockets Layer	Turvasoklite kith
UI	User Interface	Kasutajaliides
VO	Virtual Organization	Virtuaalne organisatsioon
WS	Web Services	Veebiteenused
WSDL	Web Services Definition Language	Veebiteenuste kirjelduskeel
WSRF	Web Services Resource Framework	Ressursiraamistik veebiteenustele
XML	eXtensible Markup Language	laiendatud märgistuskeele
xRSL	eXtended RSL	Laiendatud ressursi kirjeldus keel

MÕISTED

API	Arvuti operatsioonisüsteemiga või rakendusprogrammiga määratud reeglistik, mille alusel rakendusprogramm kasutab operatsioonisüsteemi või teise rakendusprogrammi teenuseid (süsteemifunktsioone).
Autentimine	Kasutaja väidetud identsuse kontrollimine.
Autoriseerimine	Protsess, mis annab õiguse ligi pääseda ettenähtud ressurssidele.
Klaster	Teisisõnu kobararvuti on omavahel ühendatud arvutussõlmed, mis on tsentraalselt hallatavad ja käsutatavad juhtarvustist. Sõlmed on enamasti homogeensed, nende arv ulatub mõnest kuni tuhandeteni. Klastrid on mõeldud suurte arvutusülesannete lahendamiseks, mille jaoks ühe arvuti võimsusest jääb väheseks.
Krüptimine	Andmete semantilise sisu peitmine, nende volitamata kasutamise või märkamata muutmise vältimise eesmärgil.
LDAP	Lihtsustatud kataloogisirvimise protokoll. Komplekt protokolle, mis võimaldavad ligipääsu infokataloogidele. LDAP aluseks on X.500 standard, kuid LDAP on oluliselt lihtsam. Lisaks toetab ta ka TCP/IPprotokolle, mida X.500 ei toeta. Võimaldab peaaegu igal rakendusel ja igal arvutil ligipääsu kataloogides asuvale informatsioonile (näit. avalikud võtmed, e-posti aadressid jms).
SDK	Arendustarkvara programmipakett, mis võimaldab programmeerijal luua rakendusi konkreetsele platvormile. Harilikult sisaldab arendustarkvara üht või enamat rakendusliidest, programmeerimisvahendeid ja dokumentatsiooni.
Veebiteenus	Veebiteenused võimaldavad erinevatest allikatest pärinevatel erinevatel rakendustel üksteisega suhelda ilma aeganõudva spetsiaalse programmeerimiseta ning kuna kogu suhtlus toimub XML-keeles, siis ei ole veebiteenused seotud ühegi operatsioonisüsteemi või programmikeelega.
XML	XML on andmete struktuuri märgistuskeel, mis loodi eemärgiga võtta see veebis kasutusele HTML'i asemel. HTML osutus oma fikseeritud elementide ja atribuutidega paljude ülesannete jaoks liialt piiratuks.

SISSEJUHATUS

Termin Grid tuli kasutusele 90ndate keskel, tähistamaks hajutatud arvutusinfrastruktuuri, mida kasutasid teadlased ja insenerid. Sellest hetkest alates on toimunud märgatav progress, arendades edasi sõna Grid taga peituvat tehnoloogiat.

Juhtivaks teadusharuks Grid tehnoloogiate rakendamisel on kõrgete energiate füüsika, kus suuri ja kulukaid eksperimente tehakse rahvusvahelistes teaduslaborites. Suurim neist on Genfis asuv Euroopa Elementaarosakestefüüsika Keskus (CERN). Interneti ja informatsioonitehnoloogiate kiire areng ning hindade langus võimaldavad suurt arvutusjõudlust kasutada ka teistes teadusharudes ning tootmise ja arendusega tegelevates ettevõtetes.

Üheks Gridi edasise arengu mootoriks peetakse erinevate autorite poolt bioinformaatikat, teadusharu, mis tegeleb bioloogiliste andmete analüüsiga (DNA, valgujärjestused, molekulide struktuurid, funktsioon, võrgustikud, elusorganismide mudeldamine, haiguste põhjuste väljaselgitamine jne) arvuti abil.

DNA sekveneerimisega tegeldakse mitmete raskete haiguste ennetamis- ja ravimeetodite väljatöötamiseks. Peale DNA sekveneerimist on vaja üles otsida ja kaardistada geenid ja kõik teised funktsionaalset rolli mängivad DNA omadused. Näiteks DNA piirkonnad, mis on konserveerunud kaugetel sugulastel nagu inimesel ja kanal, sisaldavad reeglina olulisi bioloogilisi rolle täitvaid signaale. Hinnanguliselt on viie aasta pärast genoomiuuringutes vaja regulaarselt kasutada mitmekümne erineva organismi täielikku DNA sekventsi. Vastavate uuringute ja eksperimentide arvutusmahud on suured – pärmi genoom koosneb 12 miljonist *aluspaarist* (nukleotiidide (A,C,G,T) paarist), inimgenoom u. 3 miljardist aluspaarist. See seletab ka bioniformaatika vajaduse Gridi järele.

Käesoleva töö eesmärk on Gridi tööde haldamise võimaluste selgitamine ja nende realiseerimine. Samuti parimaks lahenduseks hinnatud haldusvahendi töö kontrollimine reaalse katse käigus. Eksperimendi tulemuste põhjal peaks selguma, kuidas komplekteerida Gridi töid nii, et piisavalt palju arvutusi tehtaks ühe esitatud töö käigus ning liialt aega ei kulutataks töö ettevalmistamisele ja käivitamisele.

Töö on jaotatud kolmeks peatükiks, millest esimeses tuuakse välja Gridi

ajaloo olulisemad aspektid, Gridi arhitektuuri kaks ülesehitust: Foster-Kesselman-Tuecke ning Buyya-Chetty struktuurid. Teises peatükis vaadeldakse lähemalt antud uurimuses kasutatavat Eesti Gridi ning antakse ülevaade selle vahevarast. Viimases peatükis tuuakse välja võimalused Gridi tööde haldamiseks, mis võib olla tülikas, kui töid on palju. Selgitatakse kolme võimalust, kuidas ülesandeid Gridi saata, jälgida ja tulemusi kokku koguda. Üht loodud lahendust kasutatakse, et leida Gridi saadetavate tööde optimaalne mahu ja hulga suhe. Etteantavaks ülesandeks on maatriksi tekstile sobitamine ning sisendiks pärmi genoom.

1. GRID TEHNOLOOGIA

1.1 Mis on Grid?

Grid on termin, mis tähistab arvutuste ja andmete haldamise infrastruktuuri - geograafiliselt eri kohtades paiknevad arvutid, superarvutid ja spetsiaalseadmed (andmehoidlad, sensorid jms) moodustavad ühtselt vaadeldava ressursi nii, et süsteemi kasutaja ei pea mõistma, kus täpselt tema arvutus- ja andmeanalüüsi ülesandeid lahendatakse ja kuidas täpselt toimub andmete haldamine. Gridi peamised eesmärgid on suure jõudluse pakkumine ressursinõudlikele töödele ning erinevate arvutite, arvutuskeskuste, andmehoidlate ja spetsiaalseadmete ühendamine. Tehes seda nii, et nende kasutamine oleks lihtne ja mugav ka ilma sügavate teadmisteta infotehnoloogias, super- ja hajusarvutustes.

Paljude geograafiliselt hajusalt paiknevate organisatsioonide erinevate ressursside (heterogeensus) liitmisest ja lahti ühendamisest (dünaamilisus) moodustunud Grid on suuremõõtmeline [1].

Lihtsaim viis Gridi mõistmiseks on vaadelda seda kui hajussüsteemi, mida on edasi arendatud, et võimaldada ressursside efektiivsemat kasutamist. Eesmärgiks on tekitada illusioon lihtsast, kuid suurest ja võimsast virtuaalarvutist, mille moodustavad paljud omavahel ühendatud heterogeensed süsteemid, mis jagavad erinevates kombinatsioonides ressursse[2].

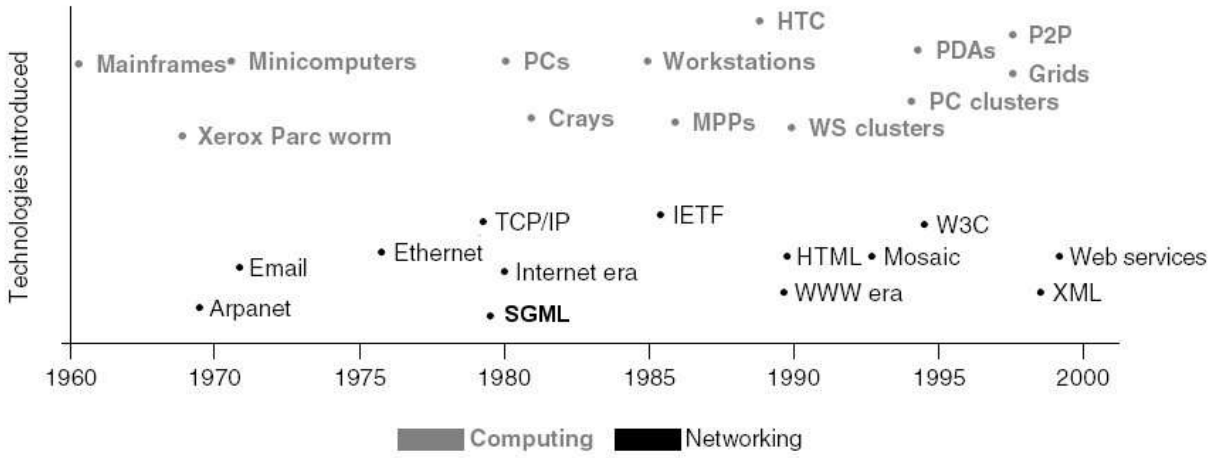
Gridist selgema ülevaate saamiseks võib välja tuua Ian Fosteri kolmepunktilise nimekirja [3]. Selle põhjal on Gridi defineeritud kui kindlate omadustega süsteemi, millel on:

- võime koordineerida ja ühendada erinevates kontrolldomeenides (lauaarvuti, tsentraalarvuti, erinevad asutused, selle allüksused) paiknevaid ressursse ja kasutajaid, mis ei ole ühtse tsentraliseeritud haldusmehhanismi osad.
- standardsed ja üldkasutatavad protokollid.
- omavahel integreeritud koostisosad (ressursid) esitatud teenusena, mille väärtus ühtsena on suurem kui selle moodustanud koostisosade summa.

1.2 Gridi ajalugu

Võrreldes Gridi elektrivõrguga, mille ajalugu ulatub kahe sajandi taha, siis Grid on seotud pigem kogu arvutikommunikatsiooni infrastruktuuri, Internetiga, mille ajalugu on lühem kui pool sajandit. Olles seega alles arenemisjärgus, liigub Grid 21 sajandil standardiseerituse suunas, nagu elektervõrk 20ndal. Joonisel 1 on välja toodud peamised tehnoloogilised edusammud, võrgu ja arvutuste juhtimise alalt, mille ilmnemine on kaasa aidanud *peer-to-peer* (P2P) võrkude ning Gridi tekkimisele.

Gridi infrastruktuuriks on Internet, mis sai alguse tagasihoidlikust USA Kaitseministeeriumi Uurimusprojektide Agentuuri (DARPA - Department of Defense's Advanced Research Projects Agency) testvõrgust 1969 aastal.



Joonis 1. [4] Võrgu ja arvutitehnoloogiate tähtsamad verstapostid aastast 1960 kuni praeguseni.

Idee rakendada kasutamata protsessoritsükleid üle võrgu tekkis juba 1970ndate algul, kui arvutid esmakordselt omavahel ühendati. Jooniselt 1 on näha, et koos tehniliste võimalustega on esile kerkinud ja ka kadunud erinevaid süsteeme. 1960ndatel teenisid kasutajate vajadusi *mainframe*-tüüpi suurarvutid, dekaad hiljem aga kahandasid suurarvutite turuosa juba DEC'i odavamad mini-arvutid. 1980ndatel vektorarvutid, nagu Cray, ja hiljem paralleelarvutid, massiliselt paralleelselt töötavate protsessoritega. Sellel kümnendil hakati ka lahendama teaduse ja tehnika *grand-challenge* [5] probleeme, mille tarbeks loodi suuremõõtmelised arvutus-infrastruktuurid.

80ndatel ja 90ndatel fokuseeriti paralleelsete arvutite tarkvara võimsate protsessorite kommunikatsiooni haldamismehhanismide ning arendus- ja rakendus-keskkonna pakkumisele paralleelsete masinate tarbeks. Paralleelne virtuaalmasin (*Parallel Virtual Machine*, PVM), teadete edastamise liides (*Message Passing Interface*, MPI), Fortran 90 laiendused (*High Performance Fortran*, HPF) ja OpenMP arendati skaleeritavate rakenduste toetamiseks. Et mõjutada jagatud ja hajutatud mälu arhitektuuri, töötati välja edukad näite rakendused. Alguses arvati, et Gridist on enim kasu paralleelsete arvutusnäidete laiendamisest tihedaltasetsevatelt klastritelt geograafiliselt hajutatud süsteemideni. Praktikas on Grid kui ühendusplatvorm lõdvaltsidestatud rakendustele, mille mõned osad, mis võivad paralleelselt töötada madala latentsusega paralleelmasinal, saab kokku linkida ühildamatute ressurssidega (nagu näiteks salvestamine, arvutamine, visualiseerimine, instrumendid) [1].

Esimeseks kaasaegseks Gridiks peetakse I-WAY (*information wide-area year*) - *Super Computing '95* raames üles seatud arvutivõrku, mis sillutas tee aktiivsele Gridi tarkvara ja infrastruktuuri arendamisele. I-WAYle rakenduste ja infrastruktuuri loomine andis olulisi kogemusi Gridi valdkonnas, mis erineb uurimisfookuselt hajusarvutustest. Hajusarvutuste arendustöö eesmärk on geograafililise eraldatuse ületamine, Gridi arendus keskendub tarkvara haldamisele ja integratsioonile.

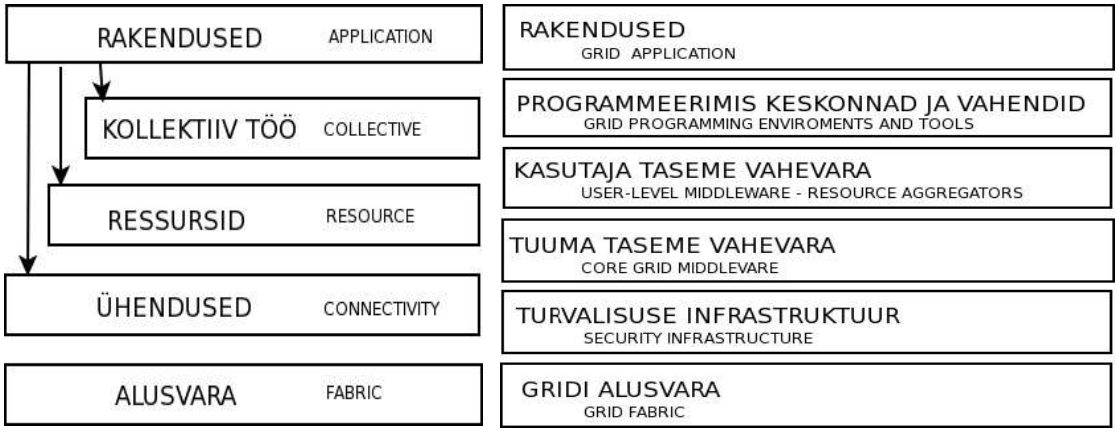
Globus[6] ja Legion[7] projektid uurisid tava-süsteemi tasemel Gridi võimaldamist. Condor[8] projekt eksperimenteeris suure läbilaskevõime ajakavastamisega, samas AppLeS[9], APST[10], Mars[11] ja Prophet[12] jälle suure jõudluse ajakavaga. Network Weather Service[13] tegeles ressursi monitooringu ja ennustamisega, Storage Resource Broker[14] ühtse juurdepääsu tagamisega heterogeensetele ressurssidele. Need ja paljud teised projektid panid aluse tänasele Gridi tarkvarale ja ideedele.

1990ndate lõpus moodustati Global Grid Forum (GGF) [15], kus varasemad uurimustööd arendatakse standarditeks, mis moodustavad tuleviku Gridi aluse. GGFis on välja arendatud Open Grid Services Architecture (OGSA) [16], mis ühendab nii Globuse kui veebi teenuste WSDL (*Web Services Description Language*) [17] püüdlused. Defineeriti ühtsed teavitamis mehhanismid näiteks teenuste leidmiseks ning registreerimiseks.

1.3 Gridi arhitektuur

1.3.1 Tasemeteks jagamine

Grid koosneb erinevatest komponentidest - põhiressurssidest kuni lõppkasutaja rakendustarkvarani, mida tihti kujutatakse kihtidena *(layers)*. Joonisel 2 on välja toodud kaks levinumat kihistruktuuri. Vasakul pool on Foster, Kesselman ja Tuecke nägemus Gridi koostisosadest [1]. Paremal oleva 6 -kihilise iterpretatsiooni [4] autorid on Rajkumar Buyya ja Madhu Chetty.



Joonis 2. Foster, Kesselman ja Tuecke 5-kihiline Gridi struktuur vasakul ning Buyya ja Chetty 6-kihiline paremal pool.

Ülaltoodud joonisel olevad struktuurid on loodud identifitseerimaks fundamentaalseid süsteemi koostisosi, spetsifitseerimaks funktsionaalsust ja eesmärke ning selgitamaks komponentide omavahelisi interaktsioone. Ülesehituse vaatenurk on seatud virtuaalsete organisatsioonide (VO) vastastikuse koostöövõime tagamisele. Koostöö tähendab eelkõige ühiseid protokolle, sest Gridi arhitektuur on peamiselt protokollide arhitektuur. See on tarkvaraline ehituskunst, kus defineeritakse VO kasutajate ja ressursi vahelised loomis-, haldamis- ning kasutamisseoste mehhanismid. Samuti on oluline API *(Application Programming Interface)* ja SDK *(Software Development Kit)* toe tekitamine, et lihtsustada rakenduste loomist.

VO moodustavad indiviidid või organisatsioonid, kes jagavad oma ressursse (tarkvara, arvuteid, andmeid). VO abil toimub juurdepääsude andmine ressurssidele. Kes, mis, millal ja kuidas saab ressursile juurdepääsu, määrab omanik.

1.3.2 Ülevaade Foster, Kesselman ja Tuecke kihtidest

Käesolevas punktis selgitatakse lähemalt joonisel 2 (vasakul pool) toodud Gridi arhitektuuri, mis on jaotatud tasemeteks. Kirjeldatavad tasemed on võrreldavad järgmises punktiks (1.3.3) välja pakutud lahendusega (6 taset).

Alusvara tase: liidesed lokaalsele alusvarale. Selles kihis paikneb Gridi tegelik ressurss: arvutus-, salvestus-, võrguressursid, kataloogid, sensorid jne. Siin realiseeritakse kõrgemate tasemete nõudeid. Alusvara tasemel on nõutavad päringute, protsessi alustamise ja monitoorimise, failide panemise-võtmise ning võrguülekande mehhanismid, koodi hoidla, andmebaas .

Ühendustase: lihtne ja turvaline kommunikatsioon. Ühenduskihis defineeritakse Gridi-spetsiifiliste võrgu transaktsioonide kommunikatsiooni ja autentimisprotokollid; andmete vahetamine alusvara kihi ressursside vahel, kasutajate ja ressusrside identiteedi kontrollimine. Autentimisega seotud lahendused on järgmised:

- *ühekordne sisselogimine (single sign-on)* - kasutaja identifitseerib ennast Gridile ainult ühe korra ja saab juurdepääsu mitmesugustele ressurssidele, mis on kirjeldatud alusvara kihis;
- *delegeerimine* - kasutaja peab saama oma programmi sellistes õigustes, et see saaks juurdepääsu ressurrsidele, mida kasutaja on autoriseerinud. Samuti peab programm saama delegeerida oma õiguste alamhulka mingile teisele programmile;
- *integratsioon lokaalsete turvalahendustega* - iga ressursipakkuja võib kasutada erinevaid turvalahendusi (näit. Kerberos, Unix). Gridi turvalahendused peavad suutma nendega koostööd teha;
- *kasutajapõhine usaldussuhe* - kasutaja, kui tal on vastavad õigused, saab kasutada ressursse erinevatelt pakkujatelt koos ilma ressursipakkujate turvalahenduste omavahelise koostööta.

Ressursitase: üksikute ressursside jagamine. Antud tase on üles ehitatud ühendustaseme kommunikatsiooni- ja audentimisprotokollidele, et defineerida

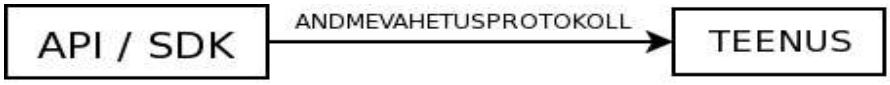
protokollid, API-d ning SDK-d turvaliseks suhtlemiseks, monitooringuks, kontrolliks, ressursimüügi halduseks ja individuaalse ressursi jagatud operatsioonide tasustamiseks.

Selle kihi suhtlusstandardid võib jaotada kaheks:

- *Informatsiooniprotokolle* kasutatakse, et saada infot ressursside struktuuri ja oleku kohta.
- *Haldusprotokolle* kasutatakse juurdepääsu saamiseks jagatud ressurssidele, täpsustades näiteks nõuded ressursile, ja operatsioonide sooritamiseks nagu protsessi loomine või juurdepääs andmetele.

Kollektiivtöö kiht: ressursside koordineerimine. Siin paiknevad reeglistikud ja teenused, API-d ning SDK-d pole seotud kindla ressursiga, vaid on pigem globaalsed, jälgides ressursside vahelisi toiminguid. Kollektiivtöö tase on rajatud ressursitaseme vähestele protokollidele, võimaldades kasutada suurt hulka teenuseid seadmata uusi piiranguid jagatavatele ressurssidele. Näiteks: kataloogiteenused, jälgimis- ja diagnostika, andmete kopeerimisteenused, "gridistatud" *(to grid-enable)* programmeerimissüsteemid, töökoormuse haldamine ja koostöö raamistik (tuntud ka kui probleemi lahendamise keskkond, PSE *(Problem Solving Environment)*), tarkvara otsingu teenused jms.

Rakendustase - Gridi arhitektuuri kõige ülemine kiht koosneb kasutaja rakendustest, mis on konstrueeritud kasutama teenuseid igast kihist. Igal tasemel on defineeritud protokollid, mis tagavad juurdepääsu vajalikele teenustele: ressursihaldus, ligipääs andmetele, ressursi leidmine jne. Kihtidel võivad olla defineeritud API-d, mille rakendused (parimal juhul realiseeritud sõltumatu SDK poolt) suhtlevad sobivate teenustega, et sooritada soovitud tegevus. Skemaatiliselt on see kujutatud joonisel 3.



Joonis 3. SDKde poolt realiseeritud API-d, mis vajadusel kasutavad Gridi protokolle, et suhelda võrguteenustega.

1.3.3 Buyya ja Chetty kuus taset

Kui Foster, Tueck ja Kesselman tõid välja 5-kihilise struktuuri, siis Buyya ja Chetty loodu on kuue kihiline (joonis 2, parempoolne). Sellist lähenemist on kasutatud NorduGrid [18] projektis.

Alumisse kihti on paigutatud ressursid, mida haldab kohalik ressursihaldur koos kohaliku poliitikaga ja mis on ühendatud, kas kohaliku või laivõrgu kaudu.

Alusvara kihi koostisosad on:

- Arvutid (PCd, tööjaamad, või SMPd (sümmeetrilised multiprotsessorid - *symmetric multiprocessors*)) koos operatsioonisüsteemiga (näit Unix, Linux, Windows).
- Erinevate operatsioonisüsteemidega klastrid.
- Ressursihaldurid nagu LSF (*Load Sharing Facility*)[19], Condor, PBS (*Portable Batch System*)[20], ja SunGE *(Grid Engine)*[21].
- Salvestusseadmed.
- Andmebaasid.
- Spetsiaalsed teadusinstrumendid (nagu raadioteleskoobid ja sensorid).

Turvalise infrastruktuuri tase pakub turvalist ja autoriseeritud juurdepääsu Gridi ressurssidele. Olulised märksõnad: ühekordne sisselogimine (*single sign-on)*, autentimine ja turvaline kommunikatsioon.

Gridi tuuma vahevara kiht võimaldab ühtset turvalist juurdepääsu ressurssidele: tööde etteandmine (*job submission*), salvestuskäitlus (*storage access*), infoserverid, ressursimüügi haldus (*trading accounting*).

Kasutajataseme vahevara kiht koosneb ressursimaakleritest (*resource broker*) või tööjaotussüsteemidest, mis vastutavad ressursside komplekteerimise eest. Ressursihaldurid tegelevad rakenduste jooksutamisega hajutatud ressurssidel, kasutades sobivat tööjaotusstrateegiat.

Programmeerimiskeskkondade ja -vahendite kihis on keeled, teegid *(libraries)*, kompilaatorid, paralleliseerimisvahendid. Kasutatakse Gridi arendusvahendeid, et "gridistada" rakendusi.

Gridi rakendused (teadus-, tehnika, kommertsrakendused, veebiportaalid) on kiht, mis varieerub koostööl põhinevatest arvutustest kuni kaug-juurdepääsuni *(remote access)* teaduslikele vahenditele ja simulatsioonidele.

2. EESTI GRID

2.1 Eesti Gridi ja NorduGridi ülevaade

2.1.1 Eesti Grid

Eesti Grid on pikaajaline projekt Eestis asuvate suuremate arvutusvõimsuste sidumiseks ühtsesse süsteemi ning tulemi integreerimiseks sarnaste rahvusvaheliste projektidega, et tagada Eesti teadusele juurdepääs jagatud ressurssidele ja globaalsetele andmebaasidele. Eesti Grid integreerub Euroopa Grid projektidesse ja arendab selle alast rahvusvahelist koostööd [22].

Esimesed masinad ühendati Eesti Gridi 2004 aasta veebruaris. Alusvarana kasutatakse klastreid, mida on ehitatud mitmetes suuremates keskustes - Keemilise ja Bioloogilise Füüsika Instituudis (KBFI), Eesti Hariduse ja Teaduse Andmesidevõrgus (EENET), Tartu Ülikoolis (TÜ) ja Tallinna Tehnikaülikoolis (TÜ). Moodustunud struktuur tagab projekti ühtlase arengu terves riigis.

Eesti Grid on tihedalt seotud NorduGrid projektiga ning kasutab vahevarana NorduGrid ARC (*Advanced Resource Connector*) Gridi tarkvara.

2.1.2 NorduGrid

Projekti NorduGrid, samuti tuntud ka kui *Nordic Testbed for Wide Area Computing and Data Handling*. NorduGridi peamine eesmärk on LHC-st *(Large Hadron Collider)* [23], maailma suurimast kiirendist, tuleva informatsiooni töötlemine. NorduGrid arhitektuur kasutab alusena Globus Toolkit'i komponente, lisades uusi teenuseid, samas muutmata Globuse vahendeid. NorduGrid ARC vahetarkvara arhitektuur püüdleb teenuste stabiilsuse ja viimistletuse poole, mis eeldab komponentide täielikku detsentraliseeritust ja funktsionaalset iseseisvust. Toetamaks sellist arhitektuuri, kasutati mitmeid innovaatilisi lähenemisi nagu näiteks detsentraliseeritud ressursihaldur. NorduGridi juurde on kirjutatud mõned kasulikud moodulid nagu Gridi haldur *(Grid Manager)*. Paljusid Globuse komponente on laiendatud või edasi arendatud [18].

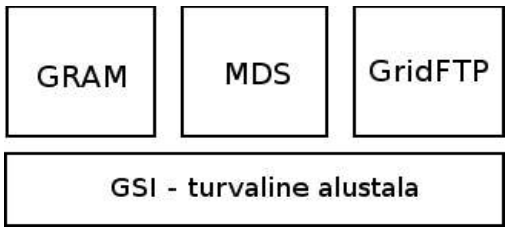
NorduGrid ARC vahevara on üles ehitatud Globus Toolkit versiooni 2 rakendusliidestele, teekidele ja teenustele.

2.2 Globus Toolkit

Globus Toolkit (GT) ei ole valmisgridi vahevara, vaid avaliku koodiga standardile vastavate rakendusliideste, teekide ja teenuste kogum, millest saab ise sobiva süsteemi kokku panna. Esimene GT versioon valmis aastal 1998, aastal 2002 teine (GT 2), kolmas 2004 ja nüüd 2005 aprilli lõpus versioon 4.0, mis baseerub uuel avatud Gridi teenuste *(Grid services)* standardil.

2.2.1 GT 2

GT 2 moodustavad 3 põhikomponenti (joonis 4): ressursihaldus, informatsiooni teenused ja andmete haldus, mis kõik kasutavad ühtset turvalisuselementi. GRAM (*Globus Resource Allocation Manager*) tegeleb ressursihaldusprotokolliga, MDS (*Metacomputing Directory Service*) implementeerib infoteenuste suhtlusstandardid, ja GridFTP andmevahetusprotokolli ning kõik eelnimetatud kasutavad ühendustasemel GSI (*Grid Security Infrastructure*) turvalisusprotokollide kogumit.



Joonis 4. GT 2 kolm põhikomponenti, mis kasutavad turvalisus komponenti.

GSI tagab turvalise autentimise ja kommunikatsiooni üle avatud võrgu. GSI põhineb avaliku võtme krüpteerimisel, kasutab X.509 sertifikaate ja *Secure Sockets Layer* (SSL) protokolli. Lisaks neile standarditele on lisatud ühekordne sisselogimine ning delegeerimine.

GRAMi kasutatakse kaugtööde saatmiseks ja oleku kontrollimiseks (monitooring). GRAM kasutab *Resource Specification Language* (RSL) töökirjeldus-keelt, et leida töö täitmiseks vajalik ressurss.

MDS on LDAP (*Lightweight Directory Access Language*) protokollil põhinev süsteemi komponentide informatsiooni katalogiseerija. Igal ressursil on eraldi ressursiinfo teenus GRIS (*Grid Resource Information Service*), mis omab infot antud arvutuselemendi, salvestuselemendi või spetsiaalseadme kohta. Samuti haldab GRIS järjekorras olevate tööde ja antud masinal lubatud kasutajate infot. GRIS saadab andmed indeksiteenusele GIIS (*Grid Index Information Services*), mida saab klient kasutada vajalike ressursside leidmiseks [24].

GridFTP on suure jõudlusega, turvaline ja usaldusväärne andmevahetus-protokoll, mis on optimiseeritud kiiretele laivõrkudele (*wide area networks*). GridFTP põhineb Interneti FTP (*File Transfer Protocol* - failiedastusprotokoll) protokollil, mille omadused ja laiendused on juba defineeritud IETFi (*Internet Engineering Task Force*) RFCs (*Request for Comments*). Lisatud on mõned Gridi jaoks olulised elemendid, et toetada GSId, osalist faili ülekannet, kolmanda osapoole (serverist - serverisse) ülekandeid, paralleelset mitmelõimelist (*multi-thread*) andmeedastust [25].

2.2.2 GT uuemad versioonid

Globus Toolkit'i arendajad olid kohandatud protokollide ja API tasemelt jõudnud standardsete protokollideni aastaks 2003. Kolmas versioon baseerus juba OGSA ja OGSI *(Open Grid Services Infrastructure)* spetsifikatsioonidel, millega koos toodi sisse Gridi teenuste mõiste *(Grid Service).* OGSI kirjeldab täiendavaid mehhanisme veebiteenustel põhinevate Gridi teenusete loomiseks ja haldamiseks. OGSI ise on WSDL ja XML-skeemi laiendus.

Hiljuti välja antud neljas Toolkit'i versioon toetub WSRF *(Web service Resource Framework)* [26] standardil, mis säilitab OGSI funktsionaalsuse. Mitmed protokollid ja standardid, mis olid kasutusel enne veebiteenusete teket on veel toetatud.

2.3 ARC komponendid

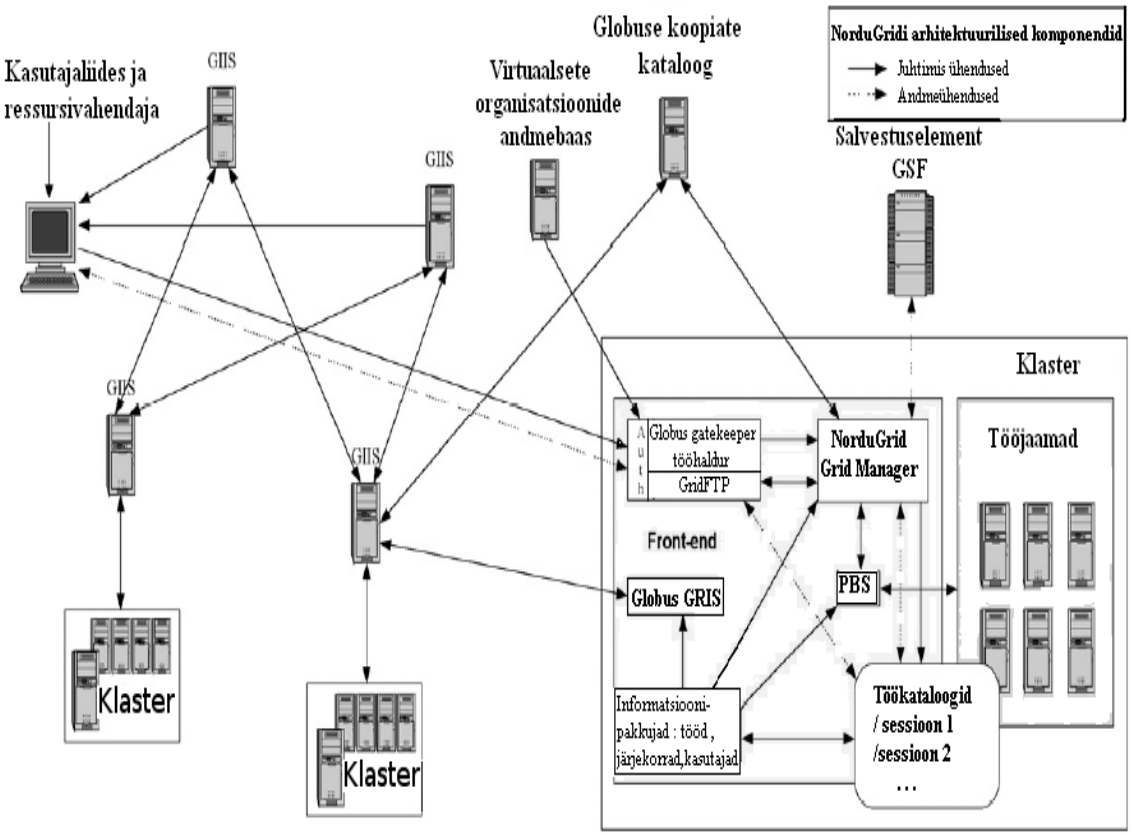
NorduGridi arhitektuur [27] planeeriti pidades silmas kasutajate ja administraatorite vajadusi, mille saab välja tuua järgnevate punktidena :

- süsteem peab olema skaleeruv,
- ressursiomanikud säilitavad täieliku kontrolli oma vara üle,
- NorduGridi vahetarkvara vajab ainult *front-end* masin,
- arvutussõlmed *(compute nodes)* ei pea olema avalikus võrgustikus,
- klastrid ei pea olema eksklusiivselt kasutatavad ainult Gridi tööde jaoks.

NorduGrid ARC sisaldab vahendeid töö saatmiseks ja haldamiseks, kasutajateringi haldamiseks, mõningaseks andmehalduseks ja monitooringuks. Järgnevalt kirjeldame süsteemi komponente, mille omavahelised suhted on toodud joonisel 5. Nendeks komponentideks on arvutus- ja salvestuselement, Gridi haldur, koopiate kataloog, kasutajaliides ning informatsioonisüsteem koos monitooringuks vajaliku Grid monitoriga.

Arvutuselement (Cluster Element ka *Computing Cluster,* CE*)*. Arvutuselement on tegelik arvutusressurss (arvuti, kobararvuti, superarvuti või mingi spetsiaalseade), millel Gridi tööd arvutatakse. Tavaliselt on Gridi keskonnas selliseks masinaks Linuxiga kobararvuti (klaster) koos mingi järjekorra haldamise süsteemiga. Kobararvuti põhimasin (front-end) on ühendatud Gridi ja tegeleb tööde vastuvõtmise ning kohalikes arvutussõlmedes (back-end nodes) käivitamisega. Põhimasina ja arvutussõlmede vahel peab olema mingisugune jagatud failisüsteem (näit. NFS, Network File System). NorduGrid vahevara on vaja installeerida ainult põhimasinale, kohalikke masinaid hallatakse läbi lokaalse tööjaotussüsteemi (*local batch system*) nagu PBS.

Salvestuselement (*Storage Element,* SE) Salvestuselement on sarnane arvutuselemendile, kuid see sisaldab endas teist Gridi põhiressurssi - salvestusmahtu. See on kättesaadav Gridis ja seal saab hoida suuremat kogust lähteandmeid või siis arvutuste tulemusi kauem kui ööpäev. Tavaliselt on SE realiseeritud GridFTP serverina. Tarkvaraks kas GT oma või GM jaoks arendatud GFS (GridFTP server).



Joonis 5. [27,lk 3] NorduGrid ARC vahevara komponendid ja nende vahelised seosed.

Gridi haldur (Grid Manager, GM*)* [28] toimib kui intelligentne eesliides (*front-end*) töö saatmisel klastrile (CE), tegeleb tööde eel- ja järeltöötlusega ning võimaldab tööde haldmist, koos metaandmete kataloogiga. Operatsioone andmetega tehakse lisatasemel – väljaspool GMi. Andmeid töödeldakse GM poolt ainult töö alustamisel ja lõpetamisel, mis eeldab kasutajalt täielikku sisend- ja väljundandmete kirjeldamist.

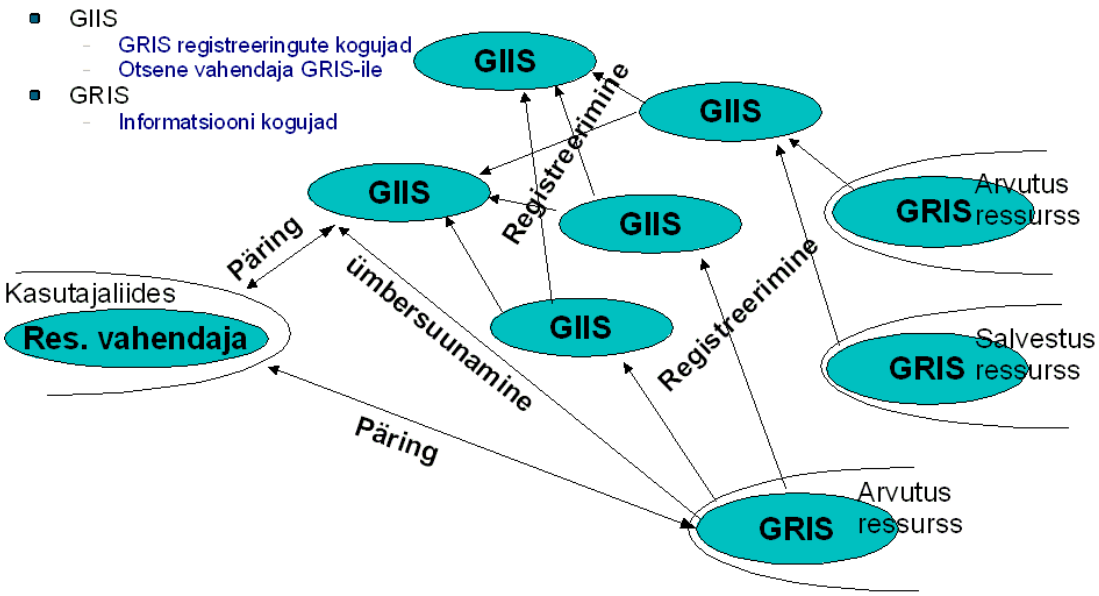
Tagamaks GridFTP liidest spetsiaalsete tööde tarbeks arendati Grid-orienteeritud GridFTP server (GSF), millel on järgnevad omadused:

- virtuaalne kataloogipuu, seadistatud iga kasutaja jaoks eraldi;
- juurdepääsu kontroll, mis põhineb kasutaja sertifikaadil;
- juurdepääs lokaalsele failisüsteemile.

Koopiate kataloogi (Replica Catalog, RC*)* kasutatakse andmeallikate registreerimiseks ja lokalieerimiseks. RC andmeid sisestab ja kasutab peamiselt GM, aga ka kasutajaliides ressursivahendamiseks.

ARC RC nagu Globus RCgi baseerub OpenLDAPil ning juurde on lisatud võimalus kasutada turvaliselt autenditud ühendusi, mis põhinevad Globus GSI mehhanismil.

Informatsioonisüsteem koosneb hajutatud andmebaaside dünamilisest kogust, mis on sidestatud arvutus- ja salvestusressurssidega, et anda informatsiooni konkreetse ressursi oleku kohta. Registrid on ühendatud kasutajaliidese või monitooringu agentidega, et leida kohalike andmebaaside kontaktinformatsiooni, otsese päringu tegemiseks.

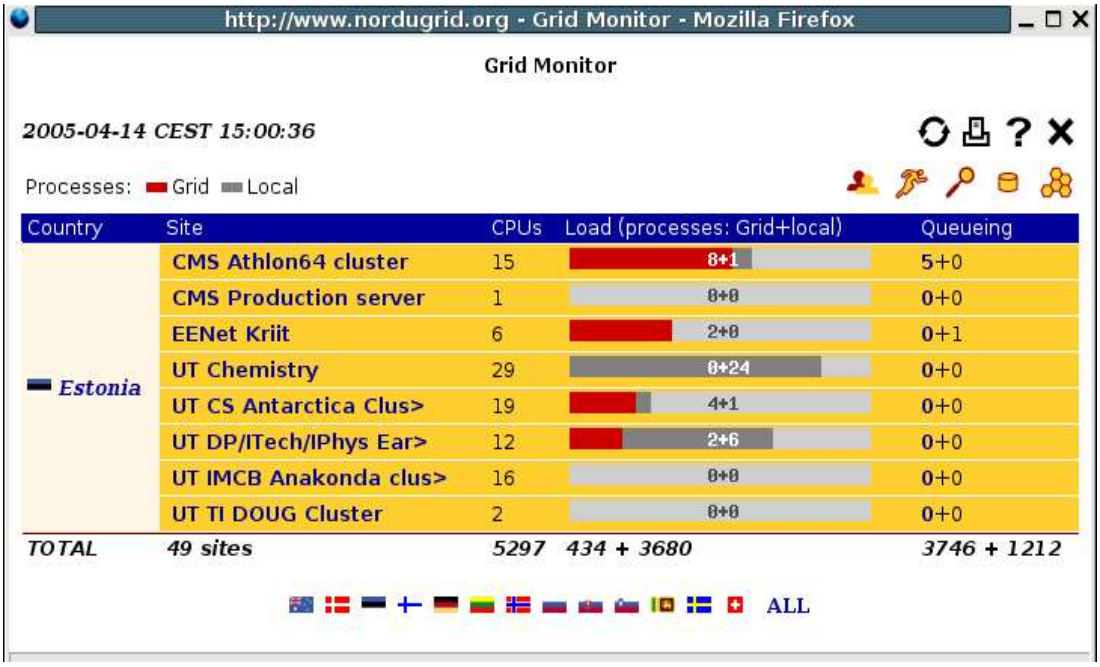


Joonis 6. NorduGridi informatsioonisüsteemi usaldatav selgroog.

NorduGridi informatsioonisüsteemi usaldatava selgroo moodustab laiendatud Globus MDS (joonis 6). Kasutataval MDSil põhineval süsteemil salvestatakse informatsioon atribuut-väärtus paaridena LDAP sissekannetena, mis on organiseeritud hierarhilisse puusse. Lokaalsed andmebaasid ja indeksiteenused (GRIS) on paigutatud mitmetasemelisse hierarhiapuuse, mis üritab järgida loomulikku geograafilist paigutust. Ressursid on asukohamaa järgi jaotatud gruppideks (virtuaalseteks

organisatsioonideks) ja registreeritud riigi indeksteenuses (GIIS), need indeksid omakorda registreeruvad NorduGridi tipp-taseme indeksteenuses (*top-level* GIIS). Vältimaks ühe punkti vigu *(single point of failure)* on NorduGridi GIISi-puu mitmejuureline, omades mitut tipp-taseme GIISi.

Monitooring - Grid Monitor. NorduGrid pakub lihtsalkasutatavat jälgimisvahendit, mis on realiseeritud veebiliidesena informatsioonisüsteemist. Monitor lubab sirvida kogu olemasolevat *(published)* teavet süsteemi kohta, võimaldades seega reaalajalist monitooringut ja olles kasutatav esmase silurina *(debugger)*.



Joonis 7. Grid monitori kuva Eesti kohta. "Total"-real on kogu NorduGridi sel hetkel aktiivsed olevad ressursid ja tööde näitajad.

Grid monitoril on iga objektide klass (klaster, kasutaja, tööd jne) esitatud lihtsasti käsitlevate viidetena (link). See on realiseeritud kasutades brauseri aknaid, kus iga aken on seotud vastava mooduliga. "*Match-it-yourself*"-otsingut kasutades on võimalik teha keerukamaid päringuid ressursside, tööde või kasutajate kohta. Joonisel 7 olev ekraanipilt Gridi monitorist on tehtud NorduGridi ühendatud Eesti ressursside

kohta ja pärineb veebiaadressilt *http://www.nordugrid.org/monitor*. Enamus tabeli elemente on klikitavad pildid, mille peale näidatakse rohkem infot valitud objekti kohta.

Kasutajaliides (User Interface -UI *)* on kõrgetasemelise funktsionaalsusega, võimaldab ressursileidmist, vahendamist, tööde Gridi saatja ja oleku päringut. UI suhtleb Gridi halduriga ning teeb päringuid informatsioonisüsteemist ja koopiate kataloogist. Kasutajaliides on kui kliendi pakett, mida saab installeerida suvalisele Linuxi masinale. Seega NorduGrid ei vaja keskset ressursivahendajat, vaid see sisaldub igas UIs.

Kasutajaliides on käsureapõhine ja sisaldab järgnevaid käske:

- ngsub - töö saatmine *(submission),*
- ngstat - näitab tööde ja klastrite olekut,
- ngcat - näitab (jooksva) ülesande väljundvooge,
- ngget - otsib välja *(retrieve)* lõpetanud töö väljundi,
- ngkill - tapab *(kill)* jooksva ülesande,
- ngclean - kustutab töö väljundi klastrist,
- ngsync - taasloob kasutajaliidese kohaliku informatsiooni jooksvate tööde kohta,
- ngcopy - kopeerib faile salvestuselementide ja koopiate kataloogide sisse, seest ja vahel,
- ngremove - kustutab failid salvestuselementidest ja koopiate kataloogidest.

Detailsemat informatsiooni nende käskude kohta leiab kasutajaliidese juhendist [29].

Töö saatmisel Gridi kasutades *ngsub* käsku on vaja see eelnevalt kirjeldada kasutades laiendatud RSLi (xRSL). Rohkem infot xRSL kohta leiab punktis 2.4.

2.4 Töökirjelduskeel xRSL

2.4.1 xRSL ülevaade

Grid infrastruktuuri korral pole klastrid, kuhu töid saadetakse, tingimata kohtvõrgus tööd saatva masinaga, vaid võivad olla suurel määral hajutatud. Selline

lähenemine nõuab korralikku töö valikute (*job options*) kirjeldust. Globus projekt arendas Gridi vahevara (*middleware toolkit*), mis kasutab tööde valikute ja ressurssihaldus süsteemi definitsioonide, koostisosade ning seoste tuvastamiseks, ressursi spetsifitseerimiskeelt RSL (*Resource Specification Language*). NorduGrid projekti raames arendati ARC lahendus Grid vahendile, mis on sobilik keerukate ülessannete, näit kõrgenergia füüsika, kirjeldamiseks. Kuna tegemist on keerulise tööga, vajab RSL täiendusi, et see lahendus töötaks.

ARC-võimalustega ressursile töö saatmiseks on vaja kasutada laiendatud Globus RSL versiooni. Laiendusega tuuakse sisse nii uued atribuudid kui ka erinevus kahe taseme vahel töö valiku spetsifikatsioonis:

1) *kasutajapoolne*-RSL on atribuutide kogum, mis on täpsustatud kasutaja poolt töö-spetsiifilises failis. Seda faili interpreteeritakse kasutajaliidese poolt ja pärast vajalikke muutmisi edastatakse Gridi haldurile.
2) *Gridi halduri poolne* RSL – Gridi haldur interpreteerib kasutajaliidese poolt etteantud atribuutide kogumit.

Kasutajal on vaja teada ainult omapoolset osa, et seda kasutades kirjeldada Gridi töid. Gridi haldur kasutab natuke erinevat esitust kui kasutajaliides. Tavaline *.xrsl fail on sisuliselt üks hulk atribuute, mis on sätitud sulgudesse vastavalt nende rakendamise prioriteetidele [30].

2.4.2 xRSL süntaks

Gridi ülessanne kirjeldatakse xRSL atribuutidega, mida võib sisestada käsurealt või koguda kokku xRSL faili, mis on mugavam. See fail koosneb üksnes atribuutide sõnede listist ja boolean operandidest & (JA, *and*) ning | (VÕI, *or*).

Tavaliselt algab xRSL töökirjeldus ampersandiga (&), märkimaks varjatud seotust kõigi atribuutide vahel:

&(atribuut1=väärtus1)(atribuut2=väärtus2)...

Kui on vaja kasutada disjunktsiooni kahe või enama atribuudi vahel võib kasutada järgmist konstruktsiooni:

(|(atribuut1=väärtus1)(atribuut2=väärtus2)...)

Avaldistes võib kasutada järgnevaid operande:

= != > < >= <=

Kommentaarid algavad (* ja lõpevad *) -ga.

(* siia võib kirjutada kommentaare *)

Mitme töö kirjeldus ühes failis realiseeritakse Globus RSL multi-nõude operandiga + , mis peaks eelnema multitöö kirjeldusele:

+(&(...))(&(...))(&(...))

Atribuudid xRSL failis võivad olla kirjutatud ühe sõnena või olla jaotatud ridadeks; tühikuid (atribuut=väärtus) vahel eiratakse.

Järgnevalt toon näite lihtsa xRSL faili sisust :

```
&
(executable="/bin/echo")        (* käivitatav programm*)
(arguments="Gridi too")         (* programmile antavad argumendid*)
(jobName="Gridi naidistoo") (* töö nimi, mis kuvatakse Gridi monitoril*)
(stdout="stdout")               (* standartse väljundi fail*)
(stderr="stderr")               (* veateadete väljundfail*)
(gmlog="gmlog")                 (* GM logifail *)
(CPUTime=5)       (* töö sooritamiseks nõutav protsessoriaeg minutites *)
```

2.5 Töö liikumine

Eelnevalt kirjeldatud komponendid, mille omavahelised seosed on näha joonisel 5, on disainitud toetama järgnevat töökäiku [27]:

1. Kasutaja valmistab ette töö kirjelduse, kasutades xRSLi. Kirjeldus võib sisaldada rakendusespetsiifilisi nõudeid nagu sisend- ja väljudandmete määratlemine, kui ka muid valikuid ressursside otsimisel. Näiteks töömasinate arhitektuur või mälumaht.
2. Kasutajaliides interpreteerib töökirjeldusfaili (.xrsl fail), otsib vastava ressursi, kasutades informatsioonisüsteemi ja RC andmeid ning edastab töö

valitud klastrile.

3. GM, mis paikneb klastri põhimasinas *(front-end),* võtab vastu töö nõude (xRSL formaadis), teeb eel- ja järeltöötluse sõltuvalt töö kirjeldusest. Peale eeltöötlust edastab GM ülesande kohalikule tööjaoturile (tavaliselt PBS). GM teeb ka sisend- ja väljundandmete manipulatsioone, kasutades RCde ja SEde abi.
4. Kasutaja võib töö staatuse jälgimiseks nõuda e-posti teateid või lihtsalt kasutada kasutajaliidest või monitori. Peale töö lõpetamist on väljundfailid kasutajale kättesaadavad 24 tunni jooksul, peale mida klastri GM nad kustutab.

3. GRIDI TÖÖDE HALDAMISE VAHENDID

Teadusarvutustes on tavaline, et tehakse palju arvutusi sama programmiga muutes ainult lähteandmeid (sisendfaile). Selline kasutusjuht on ideaalne Gridile, kus on suur hulk veel ebaefektiivselt kasutatud protsessoreid.

Ülesannete saatmine Gridi ja nende haldamine võib muutuda takistuseks, kui töid on palju. Sellele probleemile püütakse käesolevas peatükis lahendust leida, tuues välja kolm võimalust, kuidas ülesandeid Gridi saata, jälgida ja tulemusi kokku koguda. Samuti varieeritakse tööde mahu ja arvuga, et leida optimaalseim lahendus ülesannete Gridi saatmiseks. Katseandmed on välja toodud punktis 3.4.2

Selles peatükis välja toodud ülesannete Gridi saatmise meetodid on realiseeritud keeles Python [32]. Eesti Grid on osa NorduGridist ja kasutab tööks avatud koodiga vahevara ARC. Parema jälgitavuse ja arusaamise tagamiseks võib vaadata NorduGrid kasutusjuhendit [31]. Olgu öeldud, et ARC vahevara töötab Linuxi distributsioonidel.

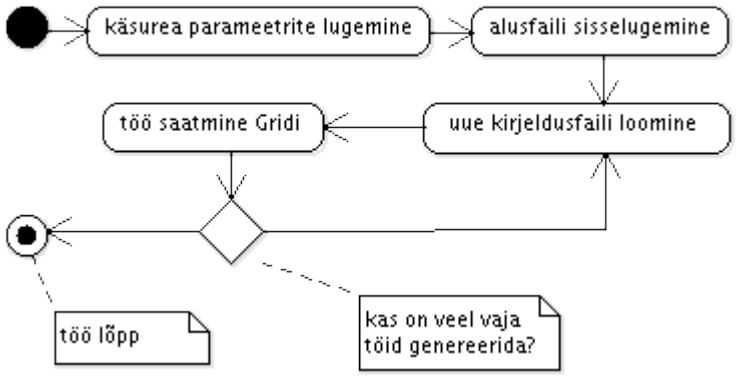
3.1 Tööde genereerimise ja Gridi saatmise skript

Antud alapunktis tuuakse välja, kuidas on realiseeritud ülesannete Gridi saatmine Nordugridi vahevaraga, samuti tööde genereerimise skripti põhimõtteline skeem (joonis 8).

Juurdepääsu Gridile tagab installeeritud ARC tarkvara (juhend aadressil [33]) ja omandatud Grid sertifikaat. Seejärel luuakse piiratud kestvusega proxy-sertifikaat käsuga *grid-proxy-init*. Sisestatakse parool, mis on sama, millega teie salajane võti on krüpteeritud.

Proxy on defineeritud kui vahendaja või volinik, kellel on õigus teise subjekti eest tegutseda. Gridis tähendab see ülesannetele teie õiguste volitamist kasutamaks seal leiduvaid ressursse.

Luuakse töökirjeldusfail. Näidisfail on välja toodud allpool (lk 26). Ülesande saatmine toimub käsuga: *ngsub -f kirjeldus_fail.xrsl*



Joonis 8. Üldine tööde genereerimise skripti skeem.

Suure hulga sarnaste tööde genereerimise ja Gridi saatmise skripti tegemiseks luuakse töökirjelduse alusfail, milles teatud kohti muutes (kasutades näiteks regulaaravaldist) genereeritakse uus töökirjeldusfail. Näiteks muudetakse töö number argumendina (ülevaate tagamiseks) ja sisendfaili (failirea) number, mis kirjeldab konkreetse töö lähteandmeid. Allpool toodud näites on <NR> ülesande n.ö järjekorra number ja <SF> lähteandmetega seotud sisendfaili muutuja.

&(executable=tegija.py)
(arguments=<NR> <SF>)
(jobName= ylesanne-<NR>)
(inputfiles = (sisend.txt ""))
(outputfiles=(valjund<NR>.txt "valjund<NR>.txt"))
(stdout=out<NR>)(join=true)(gmlog=gm)

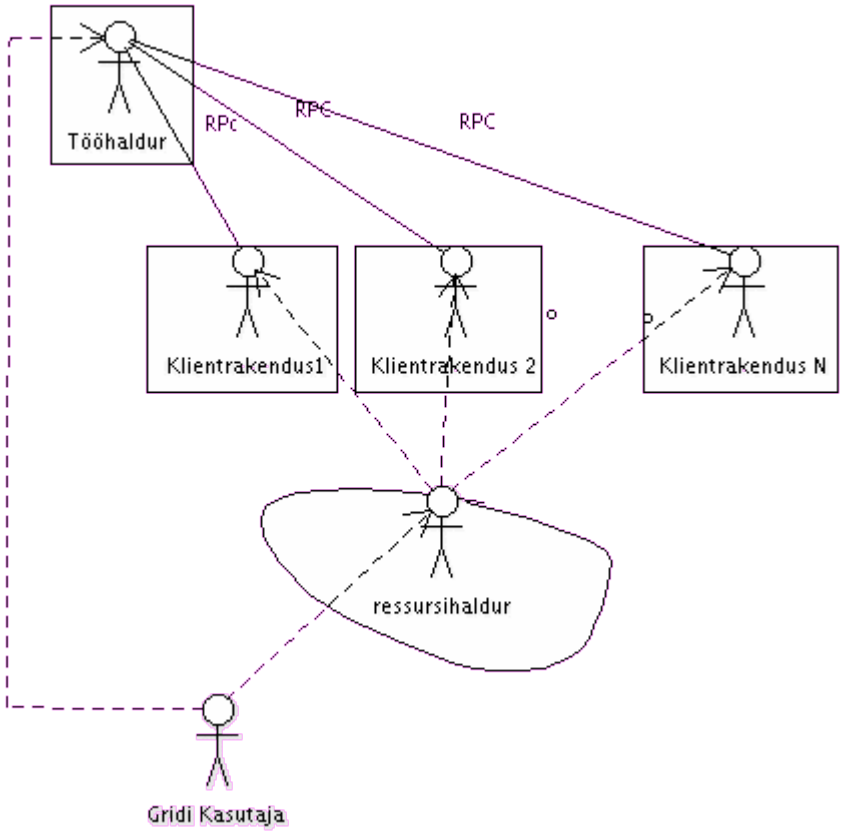
Sellise skripti tegemine on otsene, aga puudulik lahendus. Puudusteks on, et kord juba Gridi saadetud tööde kirjeldustes ei ole võimalik enam muudatusi teha. Miinuseks on ka Gridi ülesannete katkemine (võrguvead, klastri Gridi haldur ei tööta korrektselt jne). Vigade ilmnemisel tuleb töö uuesti saata, kuid seda ei ole mõistlik teha enne, kui on välja selgitatud probleemi olemus. Praktika näitab, et paljudel juhtudel on viga kasutaja enda programmis või töökirjelduses. Teadmaks, milline töö on katkenud, tuleb kontrollida tööde staatust. See kõik nõuab aega, kasutaja

tähelepanu ja suure arvu vigade korral muutub tülikaks. Antud skripti eelisteks on lihtsus ja ei nõua väga suuri teadmisi skripti loomiseks, seetõttu on ta ka lihtsates arvutustes laialt kasutusel.

Järgmistes alapunktides välja toodavad kaks võimalust kasutavad antud alapunktis kirjeldatud Gridi saatmise meetodit, kuid lihtsustavad kasutajapoolset tegevust.

3.2 Tööhalduri kasutamine

Eesmärgiks on mugava tööde Gridi saatmise liidese loomine. Tööhaldurit kasutav lähenemine on illustreeritud joonisel 9.



Joonis 9. Tööde haldamine kasutades tööhaldurit.

Gridi kasutaja ei ole otseses ühenduses Gridi arvutussõlmedega, ühendus on loodud ressursivahendaja kaudu. Arvutussõlm ja ka Gridi kasutaja on kontaktis

tööhalduriga *(sheduler),* mis peab töögraafikut tehtud ja tegemata tööde üle ning annab klientrakendustele veel tegemata ülesandeid, samuti informeerib Gridi kasutajat lõpetatud tööde seisut.

Kasutaja saab vajadusel oma tööde lähteandmeid redigeerida läbi tööhalduri ka siis, kui rakendus on Gridi saadetud või juba arvutussõlmes käivitatud. Tööhaldur peab arvet ette antud, töös olevate ja tegemata ülesannete üle. Kui vastust ei tagastata mingi mõistliku aja jooksul, siis selle ülesande staatuseks märgitakse "tegemata".

Tööhalduri ja klientrakenduse omavaheline suhtlus realiseeriti RPC *(Remote Procedure Call)* standardit XML-RPCd (*eXtensible Markup Language-Remote Procedure Call*) kasutades, millega on lihtne luua klient-server rakendusi.

Tööhaldurina ei saa kasutada GridFTP serverit, kus on fail töökirjeldustega, millest võetakse esimene seni nõudmata ülesanne. Põhjus - Gridi saadetul tööl, mis on juba käivitatud pole juurdepääsu kasutaja *proxy* sertifikaadile, seega ka GridFTP servelile. Tööd täitvas arvutis võib samuti puududa ARC vahevara.

Antud lahendus on sobilik lühikeste ülesannete lahendamiseks, mille väljundiks on paar rida teksti ekraanil. XML-RPCd pole mugav kasutada failide saatmiseks ning protokolli turvalisus on olematu, kui just ise krüptimist juurde ei kirjutata. Lahendus eeldab, et tööhaldur kui ülesandeid jagav server oleks kättesaadav (töötaks) kui klientrakendused Gridis "käima" pannakse. Tööde käivitumise aeg Gridis pole täielikult ette teada, sõltub klastri töödejärjekorra suurusest. See omakorda tähendab tööhalduri pidevat valmisolekut arvutusi jagada. Vastasel korral pole Gridi saadetud rakendustest kasu.

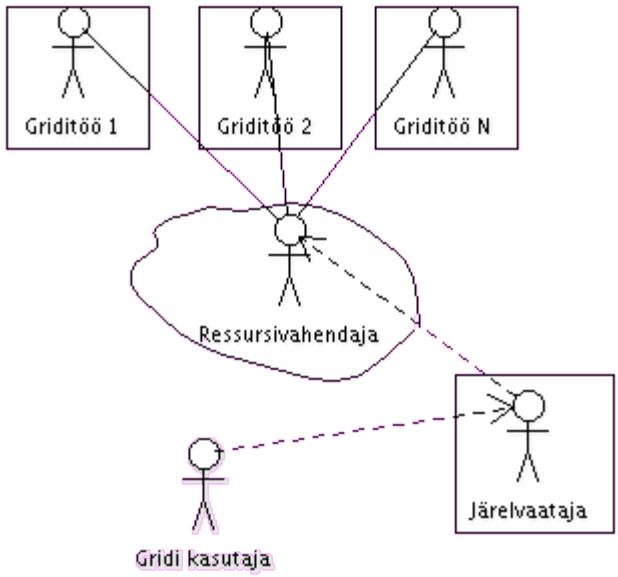
3.3 Tööde haldamine järelvaatajat kasutades

Käesolevas punktis realiseeritakse suure hulga Gridi saadetavate ülesannete haldamist lihtsustav lahendus.

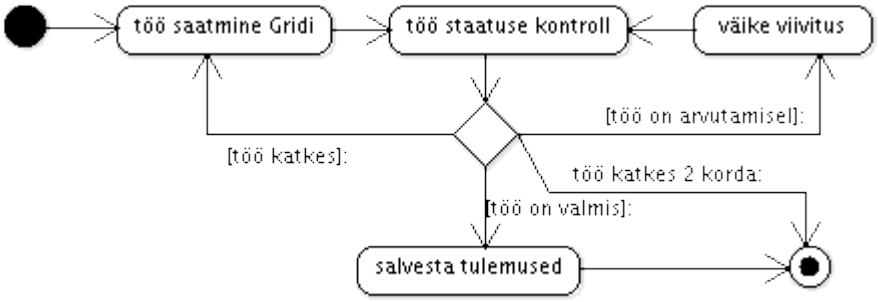
Keskendakse Gridi saadetud arvutuste staatuse jälgimisele ning tulemuste kogumisele. Kui mingi projekti käigus on vaja täita sadu ülesandeid, siis pole meeldiv igat tulemifaili eraldi kokku koguda või nende valmimist jälgida. Seda olukorda saab parandada kui hallata oma töid järelvaatajat *(watchdog)* kasutades. Järelvaatajaks on programm, mis hoolitseb selle eest, et katkenud arvutused saaks uuesti saadetud ja

tulemused meile sobivasse kohta salvestatud.

Järelvaataja seosed teiste osapooltega on välja toodud joonisel 10 ja tegutsemis- skeem *(activity diagram)* joonisel 11.



Joonis 10. Griditööde haldamine kasutades järelvaatajat.



Joonis 11. Järelvaataja programmi tegevusskeem

Järgnevalt on selgitatud joonisel 11 välja toodud järelvaataja tegevusskeem. Järelvaataja saadab töö Gridi ja hakkab peale seda tema olekut kontrollima. Staatust uuritakse kord minutis. Sagedam päringu esitamine oleks arvutuselemendi üleliigne koormamine. Kui ülesanne on katkenud (andnud veateate), siis järelvalvaja salvestab oma töökataloogi veateate, kustutab katkenud arvutuse ja saadab sama kirjeldusega

töö veelkord Gridi. Teistkordse katkemise korral on põhjuseks enamasti vigane töökirjeldus või saadetud rakendus. Teinegi veateade salvestatakse ning lõpetatakse programmi töö. Tagamaks lahenduse veakindlust salvestatakse iga töö andmed "wd"-laiendiga faili, mille abil on võimalik katkenud järelvaataja seisund taastada.

Korrektselt lõpetatud arvutuse väljund(id) talletatakse töökataloogi. See tegevus tähistab järelvalvaja töö lõppemist.

Antud lahenduse eelised eelmisega võrreldes on kasutaja väiksem sekkumine protsessi. Uurimistöös kasutati just järelvaatavat tööhaldurina, kuna peetakse nendest kolmest võimalusest parimaks lahenduseks tööeesmärkide täitmiseks.

3.4 Järelvaataja kasutamine optimaalse töömahu leidmisel

3.4.1 Ülesande püstitus minimaalse ühikulise aja leidmiseks

Gridi saadetava töö optimaalne maht on leitud, kasutades näidisülesandeks [34] kirjeldatud maatriksi tekstile sobitamise ülesannet ja sisendiks 30 pärmi transkriptsioonifaktorit kirjeldavat informatsioonimaatriksit ning pärmi genoomi 6423 geeniga.

Defineeriti keskmine aeg, mis kulub ühe pärmi transkriptsioonifaktorit väljendava informatsioonimaatriksi sobitumiste leidmiseks kõigil pärmi ülesvoolu järjestustel ühikulise ajana. Lokaalse arvuti jaoks on see leitud 30 erineva pärmi transkriptsioonifaktorit kirjeldava informatsioonimaatriksi ning pärmi genoomi 6423 geeniga arvutamiseks kulunud aja summa jagamisel informatsioonimaatriksite arvuga (s.o 30). Griditöö ühikuline aeg on leitaud Gridi saadetud ülesannete lahendamiseks kulunud aja jagamisel ühikuliste tööde arvuga. Ühikuliste tööde arvuks on ühes töös arvutatavate maatriksi sobitamiste arv korda Gridi tööde hulk.

Esmalt leiti ühikuline aeg kohalikul masinal, et võrrelda seda Gridil saadavaga. Seejärel varieeriti Gridi saadetava töö mahtu ning tööde arvu. Gridi ühikulises ajas arvestati nii selliste suurustega nagu aeg, mis kulub töö Gridi saatmiseks ja tulemuste allalaadimiseks, kui ka ajaga, mis kulub tööl klastris järekorras olemiseks. On selge, et Gridi saadetava arvutuse maht peab olema tunduvalt suurem kui aeg, mis kulub selle saatmiseks ja allalaadimiseks. Arvestada tuleks ka võimaliku järjekorraga Gridis. Teiseks peab olema saadetavate tööde hulk

suurem kui üks, kui tahetakse saavutada väiksemat ühikulist aega. Vastasel korral kasutatakse Gridi kui teist arvutit, et enda omaga midagi muud teha, saavutamata ajalist võitu.

Saadud tulemused on välja toodud punktis 3.4.2.

3.4.2 Minimaalne ühikuline aeg

Minimaalne ühikuline aeg on leitud katse käigus, kus eelnevalt nimetatud maatriksi ülesannet on saadetud Gridi erinevas mahus, varieerides arvutustes kasutatava tekstile sobitamise ülesande väljakutsumiste kordadega. Lisaks muudetakse Gridil lahendatavate tööde hulka.

Gridi arvutustulemused on välja toodud tabelis 1 ja graafiliselt kujutatud joonisel 12. Joonisel 12 on tööde maht ja hulk kantud abtsissteljele kujul *m* x *h*, kus *m* tähistab tööde mahtu ja *h* Gridi tööde arvu. Ühikulise töö lahendamiseks kulunud keskmine aeg lokaalsel masinal[1] saadi 42 sekundit.

Tabel 1. Gridis tehtud arvutuste tulemused.

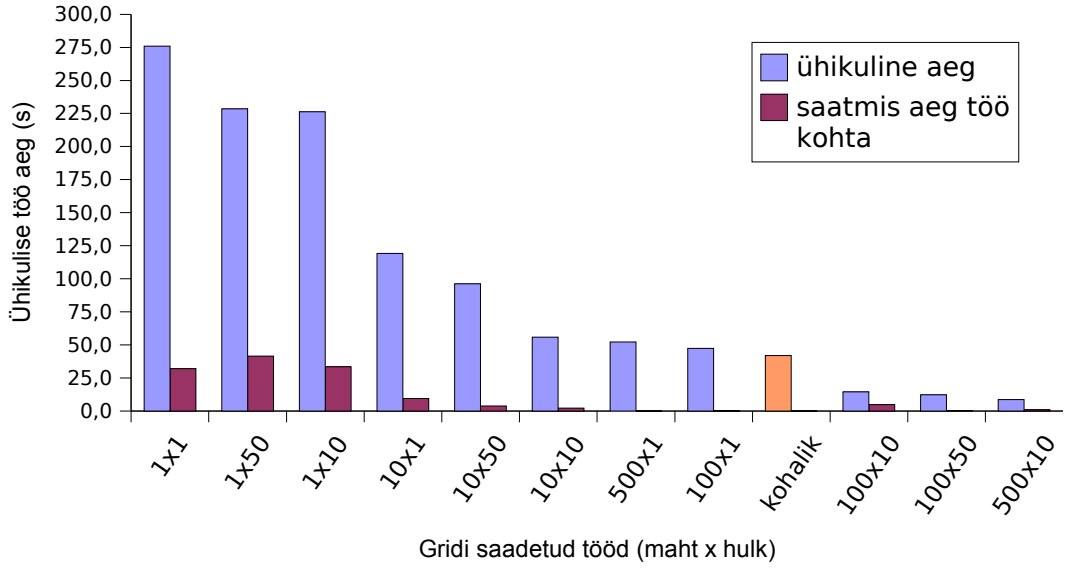
Gridi töö	*töö maht*	*tööde hulk*	*Gridi aeg(s)*	*aeg lokaalsel masinal[2] (s)*	*ühikuline aeg (s)*	*ühikulise töö saatmise aeg Gridi (s)*
1x1	1	1	276	42	276	32
1x50	1	50	11421	2100	228,4	41,5
1x10	1	10	2263	420	226,1	33,4
10x1	10	1	1191	420	119,1	9,6
10x50	10	50	48078	21000	96,2	3,9
10x10	10	10	5590	4200	55,9	2,2
500x1	500	1	26155	21000	52,3	0,1
100x1	100	1	4751	4200	47,5	0,2
100x10	100	10	14489	42000	14,5	4,8
100x50	100	50	60992	210000	12,2	0,4
500x10	500	10	43503	210000	8,7	1

Antud katseandmete korral saadi optimaalseks (parim ühikuline aeg) Gridi

1 Pentium 4 @ 2600 MHz , Ram 1024 MB
2 Hinnanguline aeg, võttes ühe ühikulise töö lahendamise ajaks 42 sekundit.

tööks 500x10, mille korral ühikuline aeg on 8.7 s.

Suhe 500x10 (500 maatriksi sobitamist tekstile ühes Gridi töös, saadetuna Gridi kümne tööna) tähendab teistsuguste ülesannete korral: 500 x 42 sekundit ehk 21000 sekundilist arvutust, mida saadetakse Gridi 10 korda.



Joonis 12. Gridis saadud ühikuliste aegade võrdlus kohaliku ajaga.

Graafiku trendi põhjal võib järeldada, et mahu ja hulga suurendamisel väheneb ühikuline aeg veelgi. Arvatavasti omab ühikuline aeg mingisugust minimaalset väärtust, mida antud töös ei leitud. Suurte tööhulkade korral sõltub ühikulise aja väärtus otseselt kasutatava Gridi protsessorite arvust – tekkivad tööjärjekorrad Gridis.

Tulemuste põhjal võib väita, et Gridi eelis tuleb selgelt esile eelkõige suurte töömahtude korral (vt tabel 1 viimaseid ridu, kus Gridi ühikuline aeg on ligi 5 korda väiksem lokaalsest).

KOKKUVÕTE

Käesoleva bakalaureusetöö eesmärgiks oli Grid tööde haldamisvahendi loomine ja selle kasutamine optimaalse töö mahu ja hulga suhte leidmiseks.

Antud töö teoreetilises osas anti kirjanduse põhjal lühiülevaade Grid tehnoloogiast ja selle ajaloost. Pikemalt käsitleti Eesti Gridiga seonduvat – Globus Toolkit ja NorduGrid ARC vahevara.

Toodi välja kolm võimalust Gridi tööde haldamiseks:

1) lihtne tööde genereerimise ja Gridi saatmise skript,
2) tööhalduri kasutamine üle XML-RPC protokolli,
3) tööde järelvaataja (*watchdog*).

Viimasena mainitud skripiti kasutati maatriksi tekstile sobitamise ülesande lahendamiseks Gridil, et leida minimaalne ühikuline aeg. Selleks varieeriti saadetavate tööde mahtu ja hulka. Saadi töö ühikuline aeg Gridi saadetuna. Suhtelised ajad, mis kulusid ühe sobitamisülesande lahendamiseks, kõrvutati sama arvutuse keskmise ajaga lokaalsel arvutil. Lisaks arvutati ühikulise töö Gridi edastamise suhtelised ajad.

Minimaalne tulemus saadi, kui ca 210 000 sekundit ühikulisi töid jaotati kümneks Gridi tööks. Saadud ühikuliseks ajaks oli 8.7 sekundit.

Kindlasti vajab käesolev töö veel lisauuringuid, kuna antud töö oli siiski rohkem ülevaatlik ja tehti esmane lihtsustatud analüüs. Peatähelepanu all võiks olla järelvaataja täiendamine, edasiarendamine ning eksperimentide tegemine suuremate mahtudega.

GRID-JOBS MANAGING TOOL WITH BIOINFORMATIC APPLICATION EXAMPLE

Bachelor thesis

Kauri Kägo

Abstract

The goal of this research was to create a Grid-jobs managing tool and to find optimum relation between job capacity and quantity.

In the theoretical part we gave a brief overview about Grid technology related to Estonian Grid – Globus Toolkit and NorduGrid ARC.

For managing Grid jobs we described three different possibilities:

1) a simple script that generates jobs and submits them to Grid,
2) to use job manager over XML-RPC protocol,
3) the best solution against manual labour – the watchdog script.

The last mentioned script was used to compute matrix matching to text calculation on Grid to find minimal unit-time. For this we varied the number and duration of jobs. The unit-times of Grid jobs were found. Relative times were compared to average time the same task took on a local computer. Submitting times of unit-jobs to Grid were calculated.

Minimal results were obtained when approximately 210 000 seconds of unit-jobs sliced into 10 Grid jobs. The best unit-time was 8.7 seconds.

Future research is needed to focus on watchdog development, since present work was more like an overview and simplified analysis. Experiments with greater capacities are also possible.

KIRJANDUS

1 Berman F., Fox G. , Hey T. *Grid Computing – Making the Global Infrastructure a Reality,* 2003. John Wiley & Sons.

2 Ferreira L., Berstis V., Armstrong J., Kendzierski M., Neukoetter A., Takagi M., Bing-Wo R., Amir A., Murakawa R., Hernandez O., Magowan J., Bieberstein N. *Introduction to Grid Computing with Globus,* 2003. http://www.redbooks.ibm.com/redbooks/pdfs/sg246895.pdf. *

3 I. Foster, *What is the Grid? A Three Point Checklist*, 2002. http://www-fp.mcs.anl.gov/~foster/Articles/WhatIsTheGrid.pdf.

4 Chetty M., Buyya R. *Weaving Computational Grids: How Analogous Are They with Electrical Grids,* 2002. http://www.buyya.com/papers/WeavingGrid.pdf.

5 NITRD. *Grand Challenges in Networking and Information Technology and Information Technology Research and Development*, 2004. http://www.hpcc.gov/pubs/200311_grand_challenges.pdf.

6 The Globus Project. http://www.globus.org.

7 Legion. http://www.cs.virginia.edu/~legion.

8 The Condor Project. http://www.cs.wisc.edu/condor.

9 Berman F., Wolski R. *The AppLeS Project: A Status Report*, 1997. http://www.cs.ucsd.edu/groups/hpcl/apples/pubs/nec97.ps.

10 APST. *AppLeS Parameter Sweep Template* http://www.nsf-middleware.org/documentation/NMI-R5/0/gridscenter/apst.

11 Ghering J., Reinefeld A. *MARS – A Framework for Minimizing the Job Execution Time in a Metacomputing Environment,* 1997. http://wwwcs.uni-paderborn.de/pc2/papers/files/11.pdf.

12 Prophet, http://dps.uibk.ac.at/projects/prophet.

13 Network Weather Service, http://nws.cs.ucsb.edu.

14 Storage Resource Broker, http://www.npaci.edu/DICE/SRB.

15 Global Grid Forum, http://www.gridforum.org.

*Kõik URL viited viimati külastatud 20. mai 2005

16 Open Grid Services Architecture, http://www.globus.org/ogsa.

17 Web Services Description Language, http://www.w3.org/TR/wsdl.

18 The NorduGrid Project, http://www.nordugrid.org.

19 Load Sharing Facility, http://www.platform.com/products/LSF.

20 Portable Batch System, http://www.openpbs.org.

21 Sun Grid Engine, http://www.sun.com/software/gridware.

22 Eesti Grid, http://grid.eenet.ee.

23 LHC – The Large Hadron Collider, http://lhc-new-homepage.web.cern.ch/lhc-new-homepage.

24 Foster I., Kesselman C. *The Grid: Blueprint for a New Computing Infrastructure,* 1999. San Francisco, Morgan Kaufmann.

25 Smirnova O., Kónya B. *Performance evaluation of the GridFTP within the NorduGrid project*, 2001. http://www.nordugrid.org/documents/gridftp_report.pdf.

26 The WS-Resource Framework, http://www.globus.org/wsrf.

27 Eerola P., Konya B., Smirnova O., Ekelöf T., Ellert M., Hansen J.R., Nielsen J.L., Wäänänen A., Konstantinov A., Ould-Saada F. *The NorduGrid architecture and tools*, 2003. http://www.nordugrid.org/documents/MOAT003.pdf.

28 Konstantinov A. *The NorduGrid Grid Manager and GridFTP Server,* 2005. http://www.nordugrid.org/documents/GM.pdf .

29 Ellert M. *The NorduGrid toolkit user interface,* 2003. http://www.nordugrid.org/documents/NorduGrid-UI.pdf.

30 *Extended Resource Specification Language Reference Manual*, 2005. http://www.nordugrid.org/documents/xrsl.pdf.

31 The NorduGrid / ARC User Guide, 2005. http://www.nordugrid.org/documents/userguide.pdf.

32 Python. http://www.python.org.

33 NorduGridi kliendi paigaldamine. http://grid.eenet.ee/failid/nordugridi_klient.htm.

34 Tasa T. *Kaalumaatriksite rakendusi bioinformaatikas,* bakalaureusetöö, 2005. Tartu.