TARTU ÜLIKOOL
Arvutiteaduse instituut
Informaatika õppekava

Tanel Joosep

Rakendus andmepäringute visualiseerimiseks

Bakalaureusetöö (9 EAP)

Juhendaja(d): Toomas Haller, PhD
Tõnis Tasa, MSc

Tartu 2017

Rakendus andmepäringute visualiseerimiseks

Lühikokkuvõte:

Käesoleva bakalaureusetöö kirjutamise käigus valmis andmepäringute visualiseerimise rakendus, mille loomisel kasutati Java 8 ning sellesse sisseehitatud lisateeke – JavaFX, Nashorn JavaScript Engine. Paremate kasutajaliidese komponentide lisamiseks kasutati ControlsFX teeki. Jõudlustestide täpsemaks hindamiseks kasutati vastavatesse rakendustesse sisse ehitatud ajamõõtmise funktsioone. Rakendus loodi ideega lihtsustada programmeerimisega igapäevaselt mitte tegelevate inimeste tööd andmete esmases analüüsis. Rakenduse eesmärk on pakkuda lõppkasutajale mugavat ja intuitiivse lähenemisega andmetöötlust. Rakendus on vabavaraline ning mõeldud kõigile kasutamiseks.

Võtmesõnad:

Andmeanalüüs, andmepäringud, prototüüp, vabavara

CERCS: P170 Arvutiteadus, arvanalüüs, süsteemid, juhtimine (automaatjuhtimisteooria)

The application for visualizing data inquiries

Abstract:

The purpose of this Bachelors thesis is to develop a desktop application that helps to visualize data inquiries and provides an intuitive overview of data. The following technologies were used: Java 8, JavaFX, Nashorn JavaScript Engine and ControlsFX library for better user interface design. The application was created with the idea of simplifying the work of people without programming skills and statistical background. The main aim of the application was to provide the end user a convenient a toll for data queries. The application is freeware and publicly available.

Keywords:

Desktop application, data inquiries, prototype.

CERCS: P170 Computer science, numerical analysis, systems, control

Sisukord

Sissejuhatus

Suur osa teadustöös tehtavatest järeldustest toetuvad andmeanalüüsile. Analüütiliste oskustega inimeste vähesus osutub tihti protsesside edenemisel takistuseks. Olemasolevad tarkvaralised lahendused ja nende kasutamine nõuavad tihti kasutajalt lisaoskusi, näiteks programmeerimist või tugevat statistilist tausta. Käesoleva rakenduse idee tuleneb Tartu Ülikooli Eesti Geenivaramu töötajatelt. Vaja on lihtsasti kasutatavat rakendust andmete esmaseks analüüsiks. Töö eesmärk ei ole võistelda olemasolevate rakendustega, vaid pakkuda kasutajasõbralikumat alternatiivi.

Käesoleva bakalaureuse töö raames valmis kirjeldatud probleemi lahendav tööriist, mille peamiseks eesmärgiks on pakkuda lõppkasutajale vahendeid andmeanalüüsiks ja kasutajasõbralikku keskkonda. Rakenduse kirjutamisel lähtuti ennekõike Tartu Ülikooli Eesti Geenivaramu bioloogide vajadustest ja soovidest, kelle igapäevane töö sisaldab hüpoteeside püsitamist ning andmehulkade põhjal bioloogilistele küsimustele vastuste otsimist.

Töö esimene osa sisaldab taustainformatsiooni. Kirjeldatakse põhjuseid järjekordse tööriista loomiseks ja probleeme, mida antud tööriist peaks lahendama. Seejärel seletatakse rakenduse poolt kasutatavaid meetodeid. Töö teises osas kirjeldatakse kõige populaarseimaid ja enim kasutusel olevaid rakendusi ja tööriistasid. Peamiselt keskendutakse kolmele tuntumale rakendusele. Need on Microsoft Excel, Google Refine ning programmeerimiskeel R. Viimane on esmajärgus mõeldud just statistiliseks andmetöötluseks. Iga konkureeriva rakenduse peatükis kirjeldatakse ka vastava rakenduse eripäraseid. Töö kolmandas osas kirjeldatakse täpsemalt seda, milliseid nõudeid ja eesmärke rakendus täidab. Töö neljandas osas kirjeldatakse autori poolt valminud rakendust. Tuuakse välja, millistele arhitektuurilistele otsustele tugineti ning kirjeldatakse üht peamist kasutuslugu. Töö viiendas osas võrreldakse rakendusi ja loodud lahendust omavahel. Võrreldakse nende omavahelisi jõudlusi, kasutades selleks jõudlusteste erinevate masinate peal. Peatüki lõpus diskuteerib autor valitud otsuste üle ning pakub välja alternatiivseid lahendusi. Kirjeldatakse rakenduse headusekriteeriumeid ning kvaliteedi tagamiseks kasutatud meetodeid. Viimaks pakub töö autor veel välja lahendusi, mida tuleks rakenduse kvaliteedi tõstmiseks teha.

1. Taust

Tänaseks päevaks on loodud suurel hulgal erinevaid rakendusi, mis lihtsutavad andmetega töötamist – LibreCalc, Apache OpenOffice, Google Drive, jpm. Rohkem rakendusi läheb vaja selleks, et saaks valida erinevate tööriistade vahel, mis sobivad konkreetse ülesande jaoks. Suurem valikuvõimalus tagab, et leitakse lahendus töö tegemiseks.

1.1 Lamefail

Lamefail (ingl *flat file*) on põhiliselt kasutusel olev failitüüp, milles olevad andmed on esitatud teksti kujul. Igal real on üks kirje, kus väljad on eraldatud kindlat tüüpi eraldajaga, näiteks komad või tabulaatorid. Lamefailis ei ole andmed kuidagi omavahel seoses. [1] See tähendab, et erinevalt andmebaasi tabelist, mis oma olemuselt on samuti fail, ei ole lamefailis mingisugust relatsioonilist seost mõne teise tabeli veeru või rea väärtustega. Samuti puuduvad lamefailis võtmeväärtused (ingl *keys*), mille abil kiiresti ja lihtsasti realt või veerult andmeid leida. [2]

Lamefaili eelis struktureeirtud failide ees on see, et kasutusel on vähem arvuti mälu. Kahe faili võrdluses, kus üks on struktureeritud ja teine struktureerimata, on faili mahtude erinevus ligi kuus korda [3]. Teisalt on lamefaile lihtsam hallata ning neid on kergem transportida. Samas, lamefaili negatiivne omadus on see, et programm peab teadma, kuidas failis andmed on paigutatud. Informatsiooni, mis on lamefailides, saab ainult lugeda, salvestada ja edastada. Joonisel 1 on toodud näide tavapärasemast korrastamata faili struktuurist.

```
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa
4.6,3.4,1.4,0.3,Iris-setosa
5.0,3.4,1.5,0.2,Iris-setosa
4.4,2.9,1.4,0.2,Iris-setosa
4.9,3.1,1.5,0.1,Iris-setosa
5.4,3.7,1.5,0.2,Iris-setosa
4.8,3.4,1.6,0.2,Iris-setosa
4.8,3.0,1.4,0.1,Iris-setosa
```

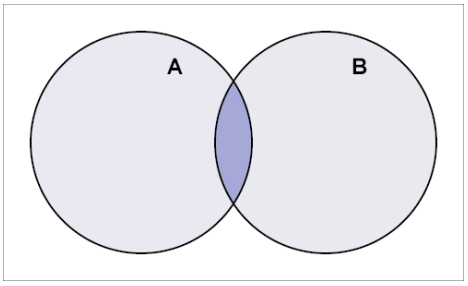
Joonis 1. Lamefaili näidis

1.2 Filtrite avaldised

Faili andmeid käsitletakse programmis kui hulkasid. Peamiselt on kasutusel andmete töötlemiseks erinevad filtri operatsioonid ehk avaldised. Filtriks nimetatakse seost kasutaja ja andmestiku vahel, mille kaudu viiakse läbi andmestikus esinevate failiosade sorteerimine. Rakendus leiab sorteeritud tulemused vastavalt kasutaja poolt sisestatud tingimustele.

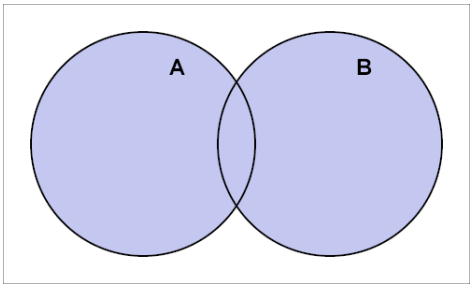
Peamised võmalused, mida käsitletav programm kasutab:

1. **AND** – tähistab hulkade ühisosa. Hulkade A ja B ühisosaks nimetatakse hulka, mille moodustavad hulga A ja hulga B elemendid [4]. Rakenduses kasutatakse ühisosa seost erinevate filtreerimiskriteeriumite loomisel. Peamiselt kasutatakse selleks, et kitsedanda otsingu- või filtreerimistulemusi. Joonis 2 illustreerib hulkade ühisosa.



Joonis 2. Hulkade ühisosa

2. **OR** – tähistab rakenduses hulkade ühendit. Hulkade A ja B ühendiks nimetatakse hulka, mille moodustavad hulga A või hulga B elemendid [4]. Rakenduses on hulkade ühend kasutusel selleks, et laiendada otsitavaid tulemusi. Joonis 3 illustreerib hulkade ühendit.



Joonis 3. Hulkade ühend

Andmestiku veergude filtreerimise jaoks on olemas peamised matemaatilised operatsioonid:

1. > – Suurem kui. Kasutatakse veergude filtreerimisel, kui otsitakse andmesitkust arvulisi tulemusi, mis on suuremad kasutaja poolt sisestatud argumendist.
2. < – Väiksem kui. Kasutatakse veergude filtreerimisel, kui otsitakse andmesitkust arvulisi tulemusi, mis on väiksemad kasutaja poolt sisestatud argumendist.
3. = – Võrdne. Kasutatakse veergude filtreerimisel, kui otsitakse andmesitkust tulemusi, mis on võrdsed kasutaja poolt sisestatud argumendist. „Võrdne“ on kasutusel ka siis, kui otsitakse mittearvulisi tulemusi.
4. ≠ – Ei ole võrdne. Kasutatakse veergude filtreerimisel, kui otsitakse andmesitkust tulemusi, mis on ei ole võrdsed kasutaja poolt sisestatud argumendist. „Ei ole võrdne“ on kasutusel ka siis, kui otsitakse mittearvulisi tulemusi.

1.3 Faili kirjeldavad statistikud

Suurtemate andmestike korral ei ole täpselt teada millisel kujul andmed failis esinevad. Kogu faili läbitöötamine manuaalselt on ajamahukas. Sellistel juhtudel on hea, kui on olemas faili formaati ja statistikuid edastavad funktsioonid. Näiteks prorgrammeermiskeeles R on olemas funktsioonid *str* ja *summary*.

Faili formaat näitab kasutajale esmast vajalikku informatsiooni:

- Mitu rida on käsitletavas failis, ehk kui palju on failis üldse kirjeid kokku
- Millist tüüpi on veerud. Rakendus keskendub peamiselt arvulistele ja mittearvulistele väärtustele. Samuti edastatake iga veeru kohta esimesed elemendid.
- Mitu erinevat veergu esineb andmestikus.
- Kuvatakse andmestikus esinevate veergude esimesed viis objekti. See parandab kasutaja seisukohast lähtudes andmestiku käsitlemist ning annab parema ülevaate, millega üldse tegemist on.

Faili statistik näitab peamiseid vajalikke parameetreid failis esinevate ridade ja veergude kohta. Iga veeru kohta edastatakse järgmist informatsiooni:

- Miinimum (ingl *minimum*) – arvuliste väärtuste korral leitakse veerust kõige väiksem element.
- Maksimum (ingl *maximum*) – arvuliste väärtuste korral leitakse veerust kõige suurem element.

- Aritmeetiline keskmine (*mean*) – Arvutliste väärtuse korral leitakse järgmise valemi alusel:

$$x = \frac{x_1 + x_2 + \ldots + x_n}{n}$$

kus x_1 ... x_n on veerus leitavad arvulised väärtused ja n on koguarv.

- Mediaan (ingl *median*) – On jaotuse keskmine liige, kus mõlemale poole jaotust jääb võrdne arv elemente. Mediaani kasutatakse selleks, et kindlaks määrata jaotuse keskpunkti. Mediaan leitakse järgmise valemi alusel:
    - Paarituarvulise n korral on mediaan rea keskmine liige, mis leitakse järgmise seose abil:

$$l = x_{\frac{N+1}{2}}$$

kus N on kogumi maht ja x kogumi element.

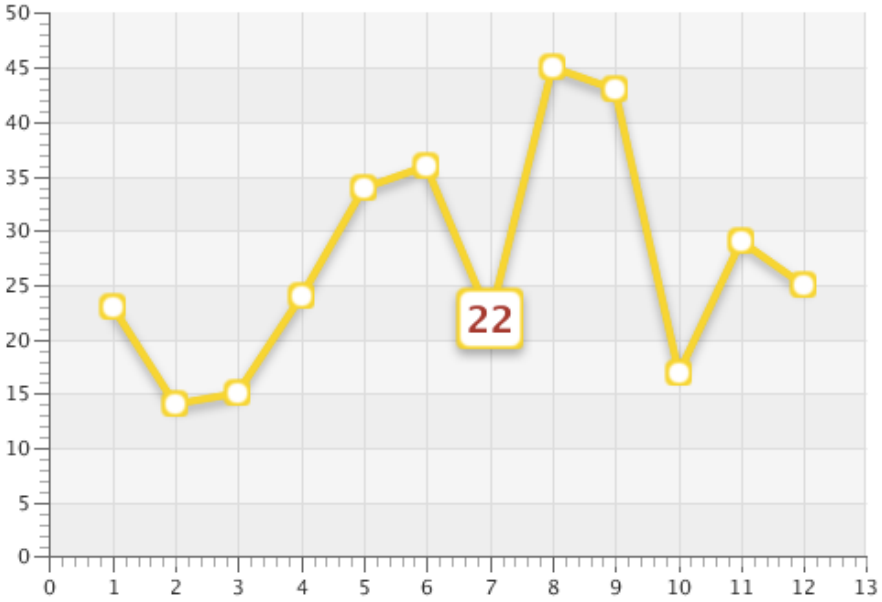
    - Paarisarvulise liikmete arvuga rea korral leitakse mediaan kahe liikme aritmeetilise keskmisega:

$$l = \frac{1}{2}(x_{\frac{N}{2}} + x_{\frac{N}{2}+1})$$

Veeru miinimum ja maksimum panevad paika kindlad piirid andmesikus esinevate arvulistele väärtustele. Keskmine ja mediaan annavad ülevaatlikuma pildi andmeveergude kohta. Tänu sellise informatsiooni olemasoluline on kasutajal lihtsam koostada päringuid.

1.4 Päringute visualiseerimine

Päring on info leidmise korraldus andmestikust. Päringu visualiseerimine otsitava info esitlemine illustreeritud kujul. Päringu pikkus võib olla üldjoontes kui tahes pikk. Päringute visualiseerimine lihtsustab päringute loomist ning minimaliseerib võimalike vigade tegemist nende koostamisel [5]. Joonisel 5 on toodud näide, kuidas päringute kohta informatsiooni saab edastada. X-teljel on kirjas mitmenda päringuga on tegu ning y-telg näitab andmestiku kogu mahtu. Sellisel kujul esitlemine võimaldab lihtsasti kontrollida päringute tulemusi.



Joonis 5. Näide päringute visualiseerimisest

1.5 Jõudlustestid

Jõudlustestid on protseduurid tarkvara jõudluse hindamiseks. Jõudlustestide eesmärk on hinnata rakenduse suhtelist sooritust. Selleks käivitatakse teatud hulgal standardseid teste ning mõõdetakse nende testide kestvust. Võrreldes kaht samal taktsagedusel töötavat protsessi ei pruugi tulemused olla alati samad, sest tegelik tulemus sõltub paljudest tingimustest, näiteks arvutil esinevate tuumade arvust, vahemälu suurusest ning lõimede arvust. Järgmine põhjus, milleks jõudlusteste sooritatakse, on rakenduste mitmekülgne koormamine. Tänu sellisele lähenemisele tulevad programmis esinevad vead kiiremini esile. Jõudlustestide peamine eelis on see, et teste saab käivitada mitmetes erinevates süsteemides, sest teste ise on lihtne koostada. [6]

Hea jõudlustest omab järgmisi omadusi [6] :

1. Lineaarsus. Kui ühe masina võimsus on kolm korda parem kui teises masinas oma, siis jõudlustesti tulemus peaks samuti kolm korda parem olema.
2. Usaldusväärsus. Kui ühes masinas jooksutatud testid on paremate tulemustega kui teises masinas, siis selline sooritus peab testide jooksul alati säilima.
3. Korduvus. Teste on võimalik korrata erinevates masinates ühte moodi.
4. Kergelt mõõdetavus. Kui testi tulemusi ei ole kerge mõõta, siis on vähetõenäoline, et keegi on huvitatud tulemusest
5. Järjekindlus. Testi tulemused kehtivad erinevates masinates.

6. Sõltumatus. Kõik jõudlustestid peavad olema kõikidest teguritest sõltumatud.

1.6 T-test

Andmeanalüüsi põhiliseks ülesandeks on parameetrite hindamine ning seoste otsimine ja seoste kindlakstegemine. Statistiline hüpotees on väide teatud parameetrite kohta, näiteks mõõtetulemuste kohta. Kontrollimiseks esitatakse alati kaks teinetest välistavad hüpoteesi, millest vaid üks saab tõene olla:

H_1 - sisukas ehk alternatiivne hüpotees.

H_0 - nullhüpotees.

Sisukas hüpotees on väide, mida soovitakse tõestada. Nullhüpotees on aga sisuka hüpoteesi vastand, mida ei ole võimalik tõestada. Hüpoteeside kontrollimise eesmärk on tõestada sisukas hüpotees nullhüpoteesi ümberlükkamise teel. Selleks arvutatakse valimi andmete põhjal vastav test statistik. Kui sisukat hüpoteesi tõestada ei õnnestu, tuleb jääda nullhüpoteesi juurde. Hüpteeside kontrollimiseks kasutatakse erinevaid teste. Üks võimalus on t-testi kasutamine. T-test on statistilise hüpoteesti test. T-testi kasutatakse normaaljaotusele alluva juhusliku suuruse keskväärtuste võrdlemiseks. Selle abil on võimalik leida, kas kahe andmehulga aritmeetilised keskmised erinevad statistiliselt olulisel määral või mitte. Testis võrreldakse pidevate tunnuste aritmeetilisi keskmisi ja sellest lähtuvadlt sobib t-test kasutamiseks juhul, kui aritmeetiline keskmine on sisuliselt arvutatav. [7]

T-testide erinevate liikidena eristatakse kahte põhivormi [7]:

1. Sõltuvate valimite t-test (ingl *paired samples t-test*), mis võimaldab võrrelda sama valimi kahe erineva tunnuse aritmeetilisi keskmisi.
2. Sõltumatute valimite t-test (ingl *independent samples t-test*), mis sobib erinevate valimite pidevate väärtustega tunnuste omavaheliseks võrdlemiseks. Näiteks poiste ja tüdrukute pikkuste keskmine võrdlemine.

Sõltuvate valimite t-testi korral peab tunnuste väärtuseid olema sama palju. Sõltumatute valimite t-testi korral tuleb tingimustes määrata, kas võrreldavatel tunnustel on sarnane või erinev variatsioon. T-testi puhul kasutatakse test statistikuna t-statistikut, mille väärtuse saab arvutada järgmiselt:

$$t = \frac{(x_1 - x_2)}{\sqrt{\frac{s_1}{n_1} + \frac{s_2}{n_2}}}$$

Valemis on x_1 ja x_2 keskmine, s_1 ja s_2 on standardhälve ning n_1 ja n_2 on valimisuurus vastavalt valimites 1 ja 2.

Eelpool toodud valemit kasutatakse määramaks kindlaks kahe vailimi keskmiste erinemise mõõt. Testi eelduseks on valimite võrdsed dispersioonide ja tunnuste normaaljaotused.

P-väärtusena defineeritakse tõenäosus arvutatud test statistiku väärtuse korral vastu võtta alternatiivne hüpotees, eeldusel et nullhüpotees kehtib [7]. Levinuim piirmäär esimest tüüpi vea tegemiseks ehk alternatiivse hüpoteesi vastuvõtmiseks kui nullhüpotees kehtib on tõenäosus 0.05 (5%) Juhul, kui leitud p-väärtus on väiksem, kui aktsepteeritav esimest tüüpi vea tegemise piirmäär, siis võetakse vastu alternatiivne hüpotees. Vastasel juhul jäädakse nullhüpoteesi juurde. [7]

T-testi korral leitakse p-väärtus kasutades antud vabadusastmete arvuga t-jaotuse tõenäosusfunktsiooni väärtust leitud t-väärtuse korral. Arvutatud p-väärtuse põhjal otsustatakse kas võrreldavate valimite keskmised on erinevad või mitte.

2. Olemasolevad rakendused

Käesolevas peatükis antakse ülevaade kolmest peamiselt kasutusel olevatest tööriistadest andmeanalüüsi valdkonnas. Nendeks on Microsoft Excel, Google OpenRefine, SAS, SPSS Statistics ning programmeerimiskeel R [9]. Peatükis kirjeldatakse nende programmide omadusi ning omavahelisi erinevusi.

2.1 Microsoft Excel

Microsoft Excel on üks tuntumaid tabelarvutus- ja töötlusprogramme. Excel on mõeldud nii algajatele, kui ka spetsialistidele. Excel võimaldab kasutajal kirjeldada erineva raskusastmega arvutusi, andmete talletamist, säilitamist ja taasesitamist. Peale tabelarvutus funktsioonide täitmist on Excelis võimalik kirjutada ka makrosid - programmilõike, mis kirjutatakse Visual Basic programmeerimiskeeles. Excelis on kasutusel erinevad andmetöötlusfunktsioonid, näiteks mediaani ja standardhälbe leidmine ning samuti eelnevalt kirjeldatud t-testi leidmise funktsioonid. [8]

2.1.1 Exceli omadused

Excel on olnud kasutusel juba 1987. aastast ning seetõttu on olemas hulganisti õpetusmaterjale. Exceli kasutamiseks pole vaja spetsiifilisi oskusi, nagu näiteks programmeerimist, vaid piisab statistika ja analüüsi teadmistest. Excel ei ole esmane vahend andmetöötluses, vaid pigem abivahend testimiseks ja töötlemiseks esmaseid kontrollandmeid [9]. Alates Microsoft Excel versioonist 2007 kuni versioonini 2016 on võimalik sisestada tabelitesse maksimaalselt 1 048 576 rida ja 16 384 veergu [10].

Excelit on võimalik kasutada ainult Microsoft Windows ja Apple OS operatsioonisüsteemides. Enamus Microsofti operatsioonisüsteemidele on Excel juba paigaldatud, mistõttu ei ole tarvilik kasutajal midagi lisaseadistada, vaid saab koheselt kasutama hakata. Eelnevalt kirjeldatud makrodega on võimalik edastada arvutiviiruseid, mis võib andmestike jagamisel kasutaja ettevaatlikuks muuta.

2.2 OpenRefine

OpenRefine, endise nimega Google Refine, on töölaua rakendus, mis lihtsustab andmete manipuleerimist ning pakub võimalusi salvestada esialgne fail teistesse formaatidesse. OpenRefine on oma olemuselt analoogne MS Exceli tabeliga, kuid filtreerimine ja sorteerimine käib läbi päringukeele *Google Refine Expression Language* (GREL). Samuti on võimalik OpenRefines kasutada ka veebiliidest. [11]

2.2.1 OpenRefine omadused

OpenRefine kasutatakse peamiselt järgmiste tegevuste jaoks:

- Puhastada esialgset andmestikku. Andmete puhastamine on tehtud väga lihtsaks. Kasutajal on võimalik luua alam tabeleid juba puhastatud andmetest ning neid salvestada.
- Andmeid on võimalik transformeerida, normaliseerida ja muuta formaati.
- Veebist tulevaid andmeid saab parsida. OpenRefine'l on olemas platvormist ja keelest sõltumatu dokumentidega suhtlemise liides (ingl *Document Object Model*). Selle tõttu on võimalik veebist tulevaid andmeid lisada juba olemasolevasse andmestikku.

OpenRefine’le seab piirangud failiformaatidega töötlemine. Nimelt on kindel hulk faili formaate, mida programmi importida saab. Neid on kuus ja nendeks on: *.csv, .txt, .xml, .rdf,*

.json ja Google tabelid. Failitüüpe, mida eksportida saab, on aga neli ja nendeks on: *.tsv, .csv, .xls, .html.* [11]

2.3 R

Tegemist on vabavaralise programmeerimiskeelega, mida peamiselt kasutatakse statistika valdkondades. R programmeerimiskeel on enim levinud tööriist andmetöötluses, kus üle 35% andmeteadlase seda kasutab [9]. Erinevalte Excelist töötab R enamikel platvormidel, sealhulgas Linux, Mac OS X ja Windows. R-i tugevuseks on tema hierarhiline struktuur, mis koosneb põhiosast ja lisapakettidest. Samuti tuleb R toime keerukate andmestruktuuridega. Kõik keeles kasutatavad muutujad on defineeritud kui üks objekt. Objekt võib sisaldada nii arve, loogilisi suuruseid, teksti, kui ka näiteks keerukamaid programmilõike. Keelele on implementeeritud vektor- ja maatriksarvutuste toetus, väga hea indekseerimine, mis võimaldab kiiresti andmeid kätte saada ning matemaatiliste meetodite kogu. [13]

2.3.1 R-i omadused

Vabavaralise keele puhul on kõigil oskajatel võimali keele arengusse panustada. Sellest tulenevalt on keele arendamise taga väga tugev kogukond [14]. Keeles on olemas kõik standard statistika ja analüüsi tööriistad, mudelid ja võimalused. Tihti just kogukonna tõttu on uued tehnoloogiad ja ideed implementeeritud enne R-is, kui see teistesse programmidesse või teekidesse jõuab. R on litsentseeritud *General Public Licence* (*GNU*) alla, mille tõttu ei ole keelel ühtegi keeldu ega piirangut.

R'ile on loodud üle 8000 paketi mitmetest eri valdkondadest, sealhulgas majanduses, andmekaeves ja bioinformaatikas [15]. R sobitub hästi peamiste analüüsi võimalustega – andmete importimine erinevatest formaatidest (*.csv*, *.xls*, *.sql*). Samuti võimaldab väljastada graafikuid tuntud formaatides nagu *.pdf*, *.jpg*, *.png* kui ka *.svg*. R ise on kirjutatud C programmeerimiskeeles. Seetõttu on peamiseks murekohaks mäluhalduse õige kasutamine. R võib keeruliste operatsioonide korral kasutusele võtta enamuse arvuti vabad ressursid ning seeläbi üle koormata kogu arvuti töö [16].

3. Rakenduse nõuded

Järgmises peatükis vaadatakse rakenduse üldnõudeid. Rakenduse jaoks vajalikud nõuded on esitatud Tartu Ülikooli Eesti Geenivaramu töötajate poolt.

Üldnõuded

- Rakendus peab avama kõiki tuntumaid failitüüpe (*.csv, .data, .txt, .xls*)
- Peab olema võimalus kuvada tulemused graafikul.
    - Graafik peab olema vähemalt joondiagramm.
    - Kasutaja saab graafikult lugeda eelmise sammu tulemusi.
- Kasutusel peab olema filtreerimissüsteem.
    - Kasutusel peab olema vähemalt *AND* ning *OR* filtreerimis võimalus.
    - Kasutaja peab teostada saama veergude kohta eraldi filtreerimist tüüpiliste matemaatiliste operatsioonide abil (> ; < ; = ; ≠).
- Kasutajal peab olema võimalus ise otsustada faili päiste olemasolu kohta.
    - Kui päiseid ei ole, peab rakendus ise looma vaikeväärtustega päised (*V1, V2, ..*)
- Kasutaja saab valida faili eraldajate sümboleid ja vajadusel ise lisada uue sümboli.
- Kasutaja peab saama kokkuvõtte andmestikust.
- Kasutaja peab saama infot andmestiku struktuuri kohta.
- Kasutajal peab olema võimalus näha enda tehtud tegevusi ja vajadusel salvestada need.
- Kasutajal peab olema võimalus oma tulemused salvestada.
- Kasutaja saab eemaldada faili algusest vajaliku sissejuhatava informatsiooni.
- Rakendus peab toime tulema failis esinevate probleemidega.
    - Kui faili struktuur ei vasta sisu üldistele nõuetele peab rakendus sellest kasutajale märku andma.
    - Kui failis esinevad anomaaliad väärtuste kohta, tuleb kasutajat sellest teavaitada.
    - Kasutajal peab olema võimalus otsustada, mida anomaaliatega teha. Anomaaliateks võivad olla näiteks vale suurusega read või valet tüüpi väärtused.

Rakendusse on implementeeritud kõik üldnõuded. Rakenduse arendusvõimalusi arutatake kuuendas peatükis.

4. Rakenduse arhitektuur

Selles peatükis kirjeldatakse rakenduse ülesehitamiseks kasutatud tehnoloogiad ja tehtud arhitektuurilisi valikuid. Rakendus on vabavaraline ning kõigile kättesaadav. Kogu programmi lähtekoodi allikas on viidatud peatükis „Lisad“.

4.1 Kasutatud tehnoloogiad

Java

Java on platvormist sõltumatu objektorienteeritud programmeerimiskeele, mis keskendub põhiliselt töölaua rakenduste (*desktop applications*) ja serverite arendamisse. Java pakub rikkalikku kasutajaliidest, kõrget jõudlust, mitmekülgsust ja turvalisust, mida tänapäevased rakendused eriti vajavad [16]. Käesoleava rakenduse loomiseks kasutati Java 1.8 versiooni, mis võimaldab kasutajal mugavalt kasutada erinevaid Lambda avaldisi.

```
public Double findMaxValue(List<String> column){

    List<Double> array = new ArrayList<>();
    array.addAll(column.stream().map(Double::parseDouble).collect(Collectors.toList()));
    return array.stream().max(Comparator.comparing(i -> i)).get();

}
```

Joonis 6. Java 1.8 koodinäide

JavaFX

JavaFX on kogum graafika- ja meediapakettidest, mis võimaldab arendajatel luua ja kujundada ja nii töölaua- kui ka veebirakendusi. JavaFX pakub kasutajatele rikkalikke lahendusi uuenenenud kasutajaliidese näol [17]. Kogu prototüübi kujundus on kirjutatud JavaFX’i võimalusi kasutades.

Maven

Maven on projektide automaatika vahend, mida peamiselt kasutatakse Java projektides. Maveni peamine eesmärk on vastutada projekti pakettide, dokumentatsiooni ja projektis esinevate vigade eest. [18] Maven katab projektides kaht aspekti. Esiteks, ta kirjeldab lahti, kuidas tarkvara on ehitatud ning teiseks, määrab ja haldab pakette, mida rakenduses kasutama hakatakse.

Nashorn

Alates Java 1.8 versioonist on sisse toodud JavaScripti põhine liidestus nimega Nashorn [19]. Nashorni eesmärk on võimaldada kergekaalulist, kuid samas suure jõudlusega JavaScripti keelt mastaapses Java maailmas. Nashorn on disainitud selliselt, et oleks võimalik kasutada maksimaalselt uusi JavaScripti tehnoloogiaid läbi Java virtuaalmasina. Nashorn on täielikult uue koodibaasiga, mis fokusseerib uutele tehnoloogiatele. [20] Projektis kasutati Nashorn'i peamiselt kasutaja sisestatud loogika operatsioonide koostisosade ja seoste tuvastamiseks.

```
public static List<Integer> findFilteredRows(Map<String, List<String>> map, String query) throws ScriptException {

    List<Integer> result     = new ArrayList<>();
    ScriptEngine  engine     = new ScriptEngineManager().getEngineByName("nashorn");
    SimpleBindings variables = new SimpleBindings();

    Set<String>  keys    = map.keySet();
    List<String> headers = new ArrayList<>();
    headers.addAll(keys);

    Integer numberOfElements = map.get(headers.get(0)).size();

    int counter = 0;
    for (int i = 0; i < numberOfElements; i++) {
        for (String header : headers) {
            variables.put(header, map.get(header).get(i));
        }
        if(engine.eval(query, variables).equals(true))   {
            result.add(counter);
        }
        counter++;
    }
    return result;
}
```

Joonis 7. Näide Nashorn'i kasutamisest

4.2 Rakenduse kirjeldus

Järgnevas peatükis kirjeldatakse komponentide kaupa rakenduse erinevaid osi. Näidatakse põhilised kasutajaliidese komponendid ning kirjeldatakse nende omadusi ja eripära. Rakendust on võimalik kasutada kõikidel platvormidel.

4.2.1 Rakendus vaated

Selles alapeatükis on kirjeldatud üks näitlik töövoog (ingl *workflow*). Seejärel on toodud põhilised rakenduse vaated ja kirjeldused, kuidas vaated täidavad töövoogu. Iga vaate juurde on kirjeldatud nõuetele vastav funktsionaalsus.

Töövoog:

1. Kasutaja vajutab "Vali fail" nuppu. Avanenud hüpikaknast valib andmestiku, mida rakendus töötlema hakkab.

2. Kasutaja ei muuda välja "Eralda faili algusest read" väärtust. Vaikeväärtus 0 jääb.

3. Kasutaja valib "Eraldaja valik" rippmenüüst andmestiku eraldajaks ",".

4. Kasutaja valib "Kas failil on päised?" valikutest "Ei".

5. Kasutaja valib "Kuidas toimida N/A väärtustega" nupust valiku "Ignoreeri".

6. Kasutaja vajutab "Info". Kasutaja näeb esmast ülevaadet andmestikust.

7. Kasutaja vajutab "Ava filtri toimingud". Avaneb uus modaalaken, kus saab sisestada otsitavat filtreerimis tingimust.

    7.1 Kasutaja sisestab avaldise, mida soovib leida.

    7.2 Kasutaja vajutab nupule "OK". Modaalaken sulgub. Põhitegevus jätkub peaaknas.

8. Kasutaja vajutab nupule "Start". Kasutaja poolt sisestatud tingimused ja avaldised kuvatakse töövoo konsooli. Rakendus hakkab töötlema kasutaja poolt sisestatud toiminguid.

9. Rakendus kuvab tulemused konsooliaknasse ning graafikule.

10. Kasutaja vajutab nuppu "Salvesta". Avaneb hüpikaken, mis lubab valida asukoha, kuhu andmed salvestatakse.

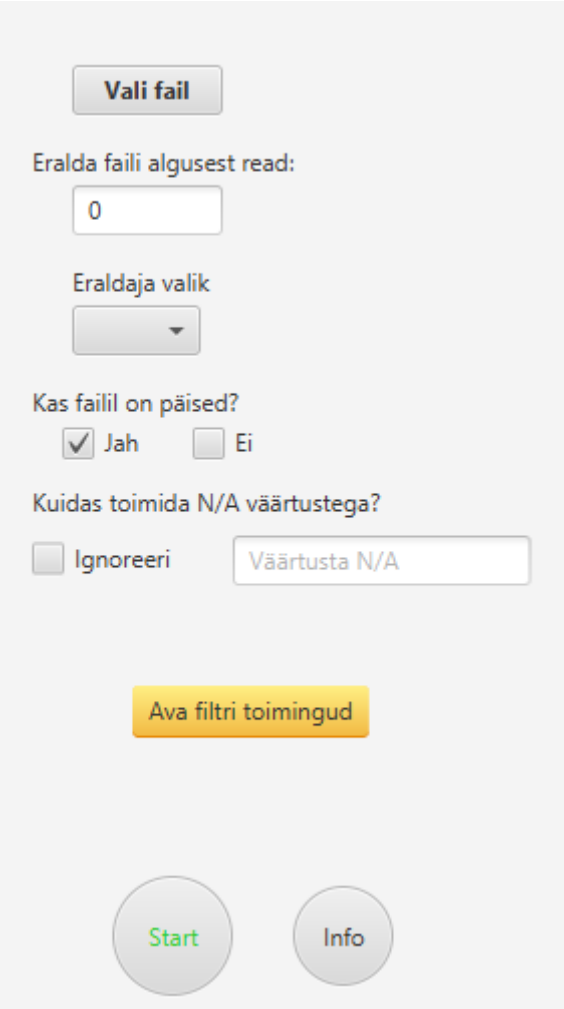
11. Kasutaja sulgeb rakenduse. Töövoo lõpp.

Järgmiseks on toodud vaated, mis vastavaid ülesandeid täidavad.

Joonisel 8 on toodud põhiakna peamine funktsionaalsus. Kasutaja saab valida faili ning vastavalt sellele täita toimingud – eemaldada faili algusest vajalikud read, valida eraldaja, kinnitada päised ja ignoreerida või asendada *N/A* väärtusi.

Peamine töövoog näeb välja selline:

1. Kasutaja valib faili.
2. Kui kasutaja on teadlik, et faili päises on sissejuhatav tekst, saab ta selle eemaldada.
3. Kasutaja valib eraldaja
4. Kasutaja kinnitab päiste olemasolu või nende puudumise.
5. Täidab ära filtri toimingud funktsiooni „Ava filtri toimingud“ alt
6. Vajutab „Start“ ning programm täidab vastavalt kasutaja otsingule toimingud.

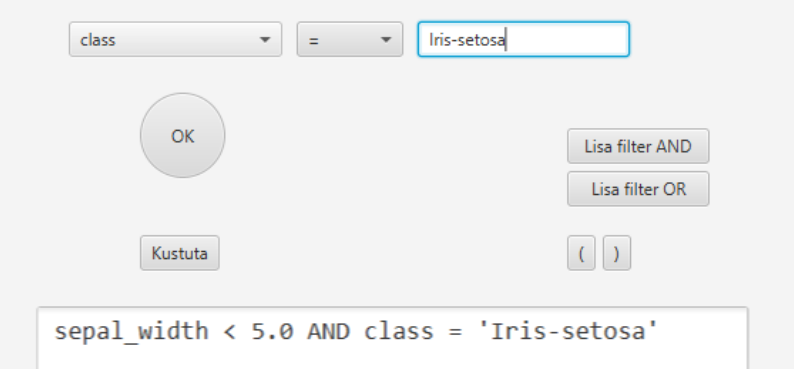


Joonis 8. Faili esmaste toimingu vaade

Joonisel 9 on toodud andmestiku filtreerimise põhiline vaade koos näite sisendiga. Kasutajal on võimalus valida kas ehitab rakenduse poolt antavate funktsioonidega päringu kokku või kirjutab käsitsi all asuvasse aknasse päringu. Kui päring on valmis, vajutatakse nuppu „OK“ ning tegevus jätkub edasi põhiaknas.

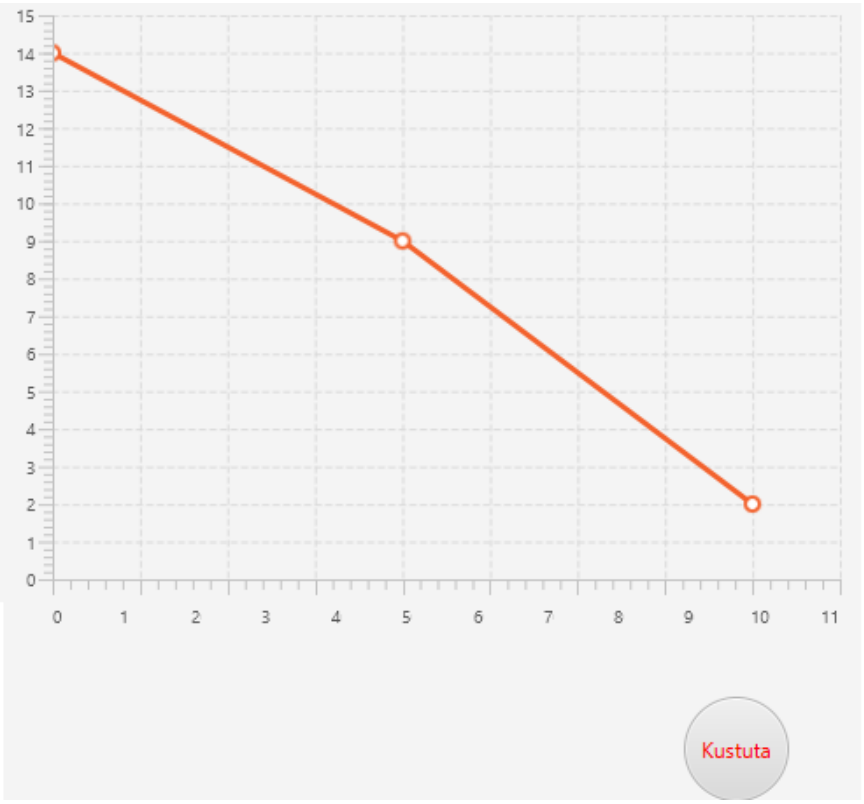
Peamine töövoog, kuidas vastavas vaates toimida:

1. Kasutaja valib rippmenüüst vastava veeru.
2. Kasutaja valib filtreerimiseks ühe tegevuse - >, <, =, ≠ .
3. Sisestab tühja lahtrisse otsitava arvulise või mittearvulise tunnuse.
4. Kasutaja sisestab AND või OR funktsiooni ja kordab tegevusi 1-3 seni kuni soovitud päring on valmis.
5. Kasutaja vajutab „OK“.



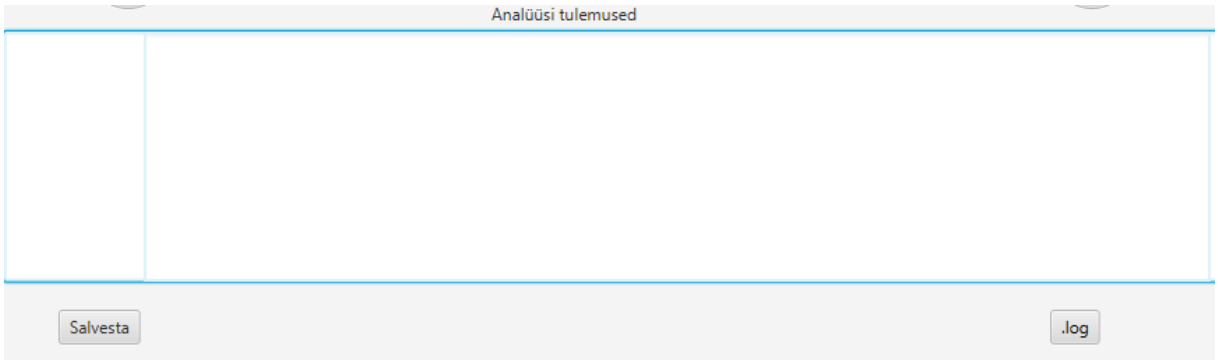
Jooni 9. Filtreerimise funktsioonid

Joonisel 10 on toodud peavaate graafiku osa. Antud graafikule kuvatakse kõik kasutaja poolt sooritatud filtreeringute tulemused. Graafiku punktidelt on võimalik vaadata eelnevalt sooritatud protsessi: sisestatud päringuid ja nende tulemusi. Funktsionaalsus „Kustuta“ alustab kogu failitöötlus protsessi algusest peale. Eemaldatakse nii graafikul esinev informatsioon kui ka valitud fail koos toimingutega.



Joonis 10. Graafiku vaade

Joonisel 11 on näha kasutajale loodud väljundi konsool koos salvestamise ja töölogi funktsioonidega. Kõik andmestikuga tehtud toimingud kuvatakse antud aknas. Kasutajal on võimalus koheselt uus tulemusfail salvestada. Samuti saab kasutaja salvestada logi faili, kuhu kirjutatakse kasutaja tehtud sammude ajalugu



Joonis 11. Tulemuste aken koos Salvesta ja logimise funktsionaalsusega.

5 Rakenduse jõudluse testimine

Peatüki põhirõhk on jõudlustestide käsitlemisel, mis annavad ülevaate sellest, kuidas rakendused üksteisest erinevat. Jõudlusteste sooritati Microsoft Exceliga, käesolevaks tööks valminud rakendusega ning RStudios.

5.1 Jõudlustestide keskkonnad

Testide täitmiseks kasutati kaht erinevat arvutit. Nende masinate parameetrid on toodud kahes erinevas tabelis. Arvutite parameetrid annavad ettekujutuse ka sellest, kui palju ning millisel määral mõjutab andmete töötlemist riistvaralised parameetrid.

Tabelis 1 on välja toodud masina A parameetrid ja tabelis 2 toodud masina B parameetrid. Eelnevalt defineeritud heade jõudlustestide tulemina võib eeldada, et masina A tulemused peaksid olema kõikide eelduste kohaselt olema paremad.

Arvuti nimetus	Samsung Notebook 300V5A-S02
Protsessor	Intel Core i3-2330M
Muutmälu (RAM)	4GB
Operatsioonisüsteem	Linux Mint 17.1 32-bit

Tabel 1. Masina A riistvara info

Arvuti nimetus	Samsung Notebook 300V5A-S02
Protsessor	Intel Core i3-2330M
Muutmälu (RAM)	4GB
Operatsioonisüsteem	Linux Mint 17.1 32-bit

Tabel 2. Masina B riistvara info

5.2 Testide kirjeldused

Testimiseks kasutati merikarpide ja limuste andmestikku (ingl. *abalone*) [21]. Valitud andmestik on valitud mitmekesisuse tõttu. See tähendab, et ridades esinevad nii tühjad väärtused, mittearvulised väärtused ning samuti *N/A* väärtused. Tabelis 3 on välja toodud andmestiku täpne kirjeldus. Originaal andmestikus esineb 4178 rida. Simulatsioone teostati erinevate algandmestiku ridade arvu korral. Kasutatud ridade arv andmestikus oli 1000, 10000, 100000 ning 1000000. Kiirustestide jaoks on väikseimate ridade arvu jaoks võetud algandmestikust juhuslik 1000 realine alamosa ning suuremate (10000, 100000, 1000000) korral võimendatud andmestiku suurust juhusliku tagasipanekuga valikumeetodi abil. Joonisel 12 on päringute olemuse selgitamiseks antud andmestiku kokkuvõtte näide.

Nimetus	**Andmetüüp**	**Mõõtühik**	**Kirjeldus**
Sugu	Nominaal		M, F, I
Pikkus	Pidev	mm	Karbi pikim väärtus
Diameeter	Pidev	mm	Karbi läbimõõt
Kõrgus	Pidev	mm	Karp koos sisemusega
Kogu raskus	Pidev	grammides	Kogu limuski raskus
Sisu rasksus	Pidev	grammides	Limuski enda raskus
Sisikonna raskus	Pidev	grammides	Limuski sisikonna raskus
Karbi raskus	Pidev	grammides	Pärast karbi kuivatamist
Ringide arv	Arvuline		+1.5 vanuse aastates

Tabel 3. Andmestiku kirjeldus

	Length	Diam	Height	Whole	Shucked	Viscera	Shell	Rings
Min	0.075	0.055	0.000	0.002	0.001	0.001	0.002	1
Max	0.815	0.650	1.130	2.826	1.488	0.760	1.005	29
Mean	0.524	0.408	0.140	0.829	0.359	0.181	0.239	9.934
SD	0.120	0.099	0.042	0.490	0.222	0.110	0.139	3.224
Correl	0.557	0.575	0.557	0.540	0.421	0.504	0.628	1.0

Joonis 12. Andmestiku kokkuvõttev kirjeldus.

Testide sooritamiseks kasutati järgmisi meetodeid:

- Java rakenduse korral lisati start nupu meetodi algusesse aja mõõtmist teostava funktsiooni (*System.currentTimeMillis()*) ning meetodi lõpus väljastati lõpp tulemus. Esialgne tulemus saadi millisekundites.
- Exceli korral kasutati makrosid, mille käivitamisel leitakse vastavate avaldiste väärtused. Toimib analoogselt Java koodile, kus protsessi käivitamisel alustatakse aja mõõtmist, töö lõpus väljastastatakse tulemus. Esimese avaldise kohta on toodud makro näide joonisel 13.

```
Sub Macro()

Dim StartTime As Double
Dim SecondsElapsed As Double

'Remember time when macro starts
  StartTime = Timer

    ActiveSheet.Range("$A$1:$E$1000").AutoFilter Field:=1, Criteria1:="=M", _
        Operator:=xlAnd
    ActiveSheet.Range("$A$1:$E$1000").AutoFilter Field:=2, Criteria1:="<0.4"

        'Determine how many seconds code took to run
  SecondsElapsed = Round(Timer - StartTime, 2)

'Notify user in seconds
 MsgBox "This code ran successfully in " & SecondsElapsed & " seconds", vbInformation

End Sub
```

Joonis 13. Makro koodinäide.

- R-i korral kutsuti programmi töö alguses välja meetodi *Sys.time()* ja töö lõpus arvutasin tehtud töö vahe lõpp ajast.

Kokku sooritatakse 4 erinevat päringut. Päringute avaldused limuste andmestikus on antud järgnevalt:

1. V1 AND V2, mis on defineeritud järgmiselt:

   Sugu = 'M' AND *Pikkus* > 0.4

2. (V1 OR V2) AND V3, mis on defieeritud järgmiselt:

   (*Sugu* = 'M' OR *Pikkus* > 0.4) AND *Diameeter* = 0.5

3. (V1 OR V2) AND (V3 OR V4), mis on defineeritud järgmiselt:

   (*Sugu* = 'M' OR *Pikkus* > 0.4) AND (*Diameeter* = 0.0 OR *Kõrgus* > 0.1)

4. V1 AND V2 OR (V3 OR V4) AND V5, mis on defineeritud järgmiselt:

   Sugu = 'M' AND *Pikkus* > 0.4 OR (*Diameeter* = 0.0 OR *Kõrgus* > 0.1) AND *Kogu raskus* > 0.5

Iga päringuga eesmärk on kasvatada veergude läbimise ja kasutamise mahtu ning seeläbi hinnata ajakulu. Igat testi sooritati viis korda. Tulemustena väljastati iga päringu ja ridade kombinatsioonil saadud aegade keskmine. Viimaks toimub erinevate rakenduste kiirustestide keskmiste võrdlus kasutades t-testi. Kuna erinevate ridade arvude korral on aegade hajuvus erinev, siis teostatakse iga ridade arvu korral grupi siseselt aegade standardiseerimine. Selleks lahutatakse iga ridade arvu korral kõigi rakenduste aegade keskmine ning jagatakse läbi kõigi ridade arvu korral mõõdetud aegade standardhälbega

$$(\frac{x-avg(x)}{sd(x)}).$$

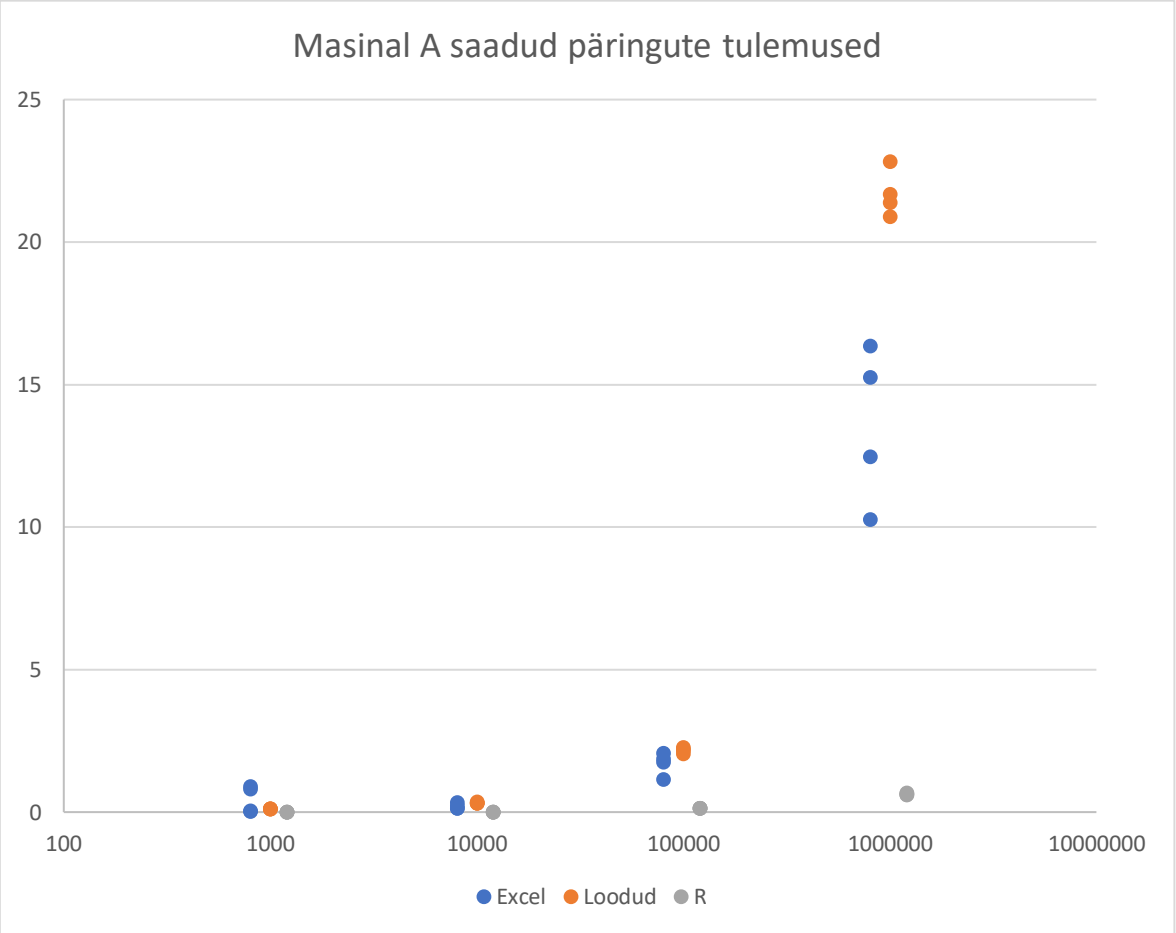
Järgnevalt teostatakse standardiseeritud aegadele kokku 4 t-testi. Esiteks võrreldakse Exceli ja antud rakenduse keskmiste aegade erinevust masinal A ning masinal B. Teiseks võrreldakse samal meetodiga kahel erineval masinal ka R ja loodud rakenduse kiirustestide aegade keskmisi.

5.3 Testide tulemused

5.3.1 Tulemused masina A peal

Joosiselt 14 on näha, et masinas A sooritatud testide tulemused olid kiireimad programmeerimiskeeles R. Päringuaeg kasvas nii suuremate avaldiste kui andmestiku mahtude kasvamise, kuid ületas selgelt Excelit ning antud töö käigus loodud rakendust. On näha, et Exceli töö aeg suureneb, kui päringu kogu pikkus kasvab. Miljoni reaga päringute korral suudab Excel kiireimal juhul filtreerida tulemused 10.257 sekundiga. Pikima päringu ja sama ridade arvu korral võtab päringu sooritamine kuus sekundit kauem aega. Antud töö käigus loodud rakendus sooritab lühemad testid samuti kiiresti. Tuhande rea korral kulub aega maksimaalselt 0.098 sekundit ning 100 000 rea korral maksimaalselt 0.358 sekundit. Kui

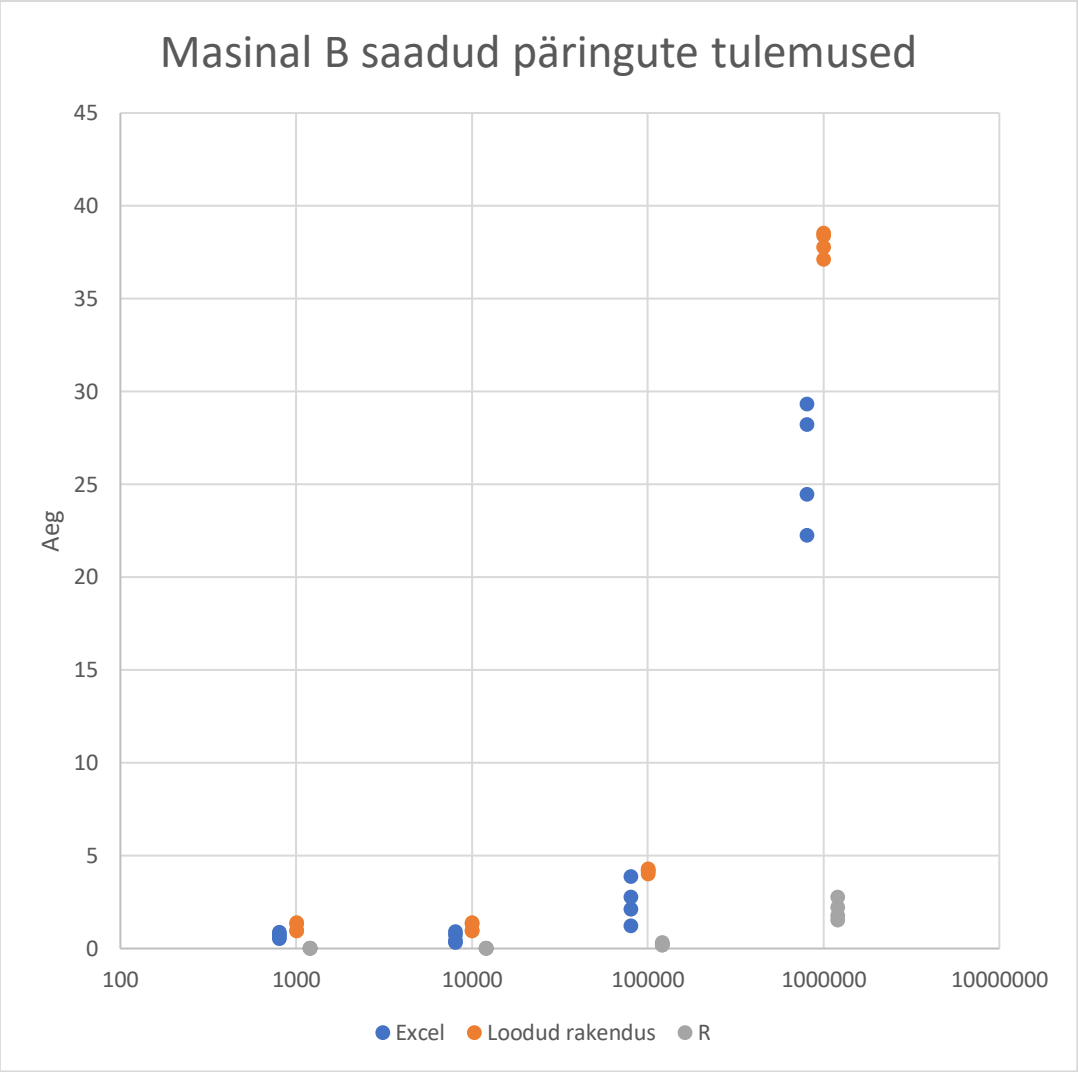
andmestiku ridade arv kasvab, kasvab ka tulemuste leidmise aeg. Loodud rakenduses ei oma päringute pikkused tähtsust. Miljoni realise andmestiku kõige lühem päring võtab aega 20.887 sekundit ning kõige pikem 22.814 sekundit.



Joonis 14. Päringute tulemused masina A peal

5.3.2 Tulemused masina B peal

Joonise 15 kujutab masinal B tehtud kiirustestide tulemusi samades tingimustes kui masinal A. Arvutite riistavaralised omadused mõjutavad testide tulemusi. Loodud rakenduse miljoni realise andmestikuga läheb masinas A aega 22.814 sekundit ning masinas B 38.531 sekundit. Erinevused tulevad ka programeerimiskeelt R kasutades, kus masinas A, samuti miljoni realises andmestikus, võtab pikim avaldis aega 0.666 sekundit, kuidas masinas B juba üle ühe sekundi.



Joonis 15. Päringute tulemused masina B peal

5.4 Järeldused

T-testide tulemusena saadi neli p-väärtust. Masinate A ja B peal võrreldi Excelit, R loodud rakendusega. Omavahelisteks p-väärtusteks saadi:

- Masin A: Excel *vs* Rakendus: 0.325 ja R *vs* Rakendus: 9.8e-5
- Masin B: Excel vs Rakendus: 1.3e-11 ja R *vs* Rakendus: < 2e-16

Nende väärtuste pealt on näha, et tööks valminud rakendus jääb mõlema masina korral R-le selgelt alla. Samas on näha, et masina A peal ei ole Exceli ja loodud rakenduse vahel olulist kvalitatiivset erinevust, mistõttu sobib loodud rakendus ideaalselt kasutajamugavust otsivale kasutajale, kes töötab väiksemate lamefaili andmetega.

6. Kokkuvõte

Bakalaureuse töö käigus loodi andmete visualiseerimise programm, mis lihtsustab kasutajatel andmete esialgset töötlust. Rakenduses on kasutajal võimalik valida töötlemist vajav fail, seadistada valitud faili esmased tingimused ning seejärel kuvatakse rakenduses tulemused, mis visualiseerivad tulemusi graafikul. Tulemusfaili on võimalik salvestada ning kasutada seejärel edasistes protsessides.

Rakenduse juures oli suurim väljakutse vajaliku andmemudeli konstrueerimine. Vaja oli luua mudel, mis suudaks kiiresti andmeid töödelda ning samas ka teataval määral andmeid salvestada. Kasutaja nõuete analüüsimisel alustati probleemi lahendamist kasutajapoolsest sisendist. Vastav rakendus ei ole küll täiuslik ning edasiarendusvõimalusi jagub hulganisti. Mitmekülgsemad lisainformatsiooniga graafikud edastaks infot veelgi tõhusamalt. Samuti saaks funktsionaalsust võimendada andmebaasi ühenduste võimekuse lisamisel. Esineb küll mõningane allajäämine võrreldud tööriistadele, kuid kvalitatiivselt pole erinevus suur.

Viidatud kirjandus

[1] Christensson, P. Flat File, *TechTerms*, 2006, https://techterms.com/definition/flatfile (18.04.2017)

[2] Bloom, M. Data Integration Glossary. *U.S Department of Transportation* 2001, https://www.fhwa.dot.gov/infrastructure/asstmgmt/010394.pdf (20.04.2017)

[3] Al Hashami, Z. XML Files Types and Their Differences and Denominators with Plain TXT File, 2015. https://www.omicsonline.org/open-access/xml-files-types-and-their-differences-and-denominators-with-plain-txt-file-2376-130X-1000130.pdf (30.04.2017)

[4] Oja, P. Hulgateooria, Tartu: Tartu Ülikooli Kirjastus. 2006.

[5] Constantinou, G. Relational Algebra and SQL Query Visualisation, 2010, 6. https://www.doc.ic.ac.uk/~pjm/teaching/student_projects/gc106_report.pdf (21.04.2017)

[6] Ehliar, A. Liu, D. Benchmarking network processors, 2004. http://www.da.isy.liu.se/pubs/ehliar/ehliar-ssocc2004.pdf (21.04.2017)

[7] Jakobson, E. Tõenäosusteooria ja statistika, SA Archimedes. 2013.

[8] Harvey, G. Excel For Dummies, Wiley Publishing, Inc. 2007.

[9] Muenchen, A. R, The popularity of Data Science Software, 2017. http://r4stats.com/articles/popularity (05.05.2017)

[10] https://support.office.com/en-us/article/Excel-specifications-and-limits-1672b34d-7043-467e-8e27-269d656771c3 (05.05.2017)

[11] https://github.com/OpenRefine/OpenRefine/wiki (04.05.2017)

[12] Verborgh, R. De Wilde, M. Using OpenRefine, Packt Publishing. 2013.

[13] R, www.r-project.org/about.html (25.04.2017)

[14] https://www.r-project.org/contributors.html (09.05.2017)

[15] de Vries, A. On the growth of CRAN packages, 2016, 4. https://www.r-bloggers.com/on-the-growth-of-cran-packages/ (23.04.2017)

[16] Java SE, Oracle Technology Network, Oracle http://www.oracle.com/technetwork/java/javase/overview/index.html (25.04.2017)

[17] http://docs.oracle.com/javafx/ (25.04.2017)

[18] Porter, B. van Zyl, J. Lamy, O, Apache Maven, https://maven.apache.org (25.04.2017)

[19] Information about Java 8, https://java.com/en/download/faq/java8.xml (25.04.2017)

[20] OpenJDK: Nashron, http://openjdk.java.net/projects/nashorn (25.04.2017)

[21] Waugh, S. Abalone data, 1995.
https://archive.ics.uci.edu/ml/machine-learning-databases/abalone/abalone.names

Lisad

Lähtekood

Prototüübi lähtekood on saadaval aadressil https://github.com/tjoosep/DataVisualizer. Repositooriumist leiab Javas kirjutatud koodifailid. Lisadena on kaasas töös kasutatud andmefail ning rakendusfail *.jar* kujul.

Mõõtetulemused masinal A

Excel	1000 rida	10 000 rida	100 000 rida	1 000 000 rida	2 000 000 rida
1. päring	0.023 sek	0.026 sek	1.132 sek	10.257 sek	*N/A*
2. päring	0.045 sek	0.135 sek	1.725 sek	12.456 sek	*N/A*
3. päring	0.789 sek	0.232 sek	1.825 sek	15.245 sek	*N/A*
4. päring	0.890 sek	0.323 sek	2.067 sek	16.345 sek	*N/A*

Tabel 4. Exceli tulemused masinal A

Loodud rakendus	1000 rida	10 000 rida	100 000 rida	1 000 000 rida	2 000 000 rida
1. päring	0.115 sek	0.306 sek	2.033 sek	20.887 sek	34.683 sek
2. päring	0.098 sek	0.321 sek	2.172 sek	21.389 sek	37.253 sek
3. päring	0.098 sek	0.320 sek	2.164 sek	21.671 sek	37.977 sek
4. päring	0.098 sek	0.358 sek	2.192 sek	22.814 sek	37.672 sek

Tabel5. Loodud rakenduse tulemused masinal A

R	1000 rida	10 000 rida	100 000 rida	1 000 000 rida	2 000 000 rida
1. päring	0.002 sek	0.002 sek	0.120 sek	0.592 sek	1.002 sek
2. päring	0.005 sek	0.005 sek	0.120 sek	0.614 sek	1.077 sek
3. päring	0.007 sek	0.007 sek	0.133 sek	0.666 sek	1.567 sek
4. päring	0.007 sek	0.007 sek	0.141 sek	0.650 sek	1.532 sek

Tabel 6. R-i tulemused masinal A

Mõõtetulemused masinal B

Excel	1000 rida	10 000 rida	100 000 rida	1 000 000 rida	2 000 000 rida
1. päring	0.523 sek	0.126 sek	1.232 sek	22.257 sek	*N/A*
2. päring	0.645 sek	0.435 sek	2.125 sek	24.456 sek	*N/A*
3. päring	0.789 sek	0.732 sek	2.765 sek	28.245 sek	*N/A*
4. päring	0.890 sek	0.923 sek	2.935 sek	29.345 sek	*N/A*

Tabel 7. Exceli tulemused masinal B

Loodud rakendus	1000 rida	10 000 rida	100 000 rida	1 000 000 rida	2 000 000 rida
1. päring	0.961 sek	0.961 sek	4.033 sek	37.124 sek	44.321 sek
2. päring	0.998 sek	0.998 sek	4.226 sek	37.773 sek	47.253 sek
3. päring	1.398 sek	1.398 sek	4.163 sek	38.389 sek	47.217 sek
4. päring	1.328 sek	1.328 sek	4.292 sek	38.531 sek	47.112 sek

Tabel 8. Loodud rakenduse tulemused masinal B

R	1000 rida	10 000 rida	100 000 rida	1 000 000 rida	2 000 000 rida
1. päring	0.012 sek	0.012 sek	0.180 sek	0.703 sek	1.528 sek
2. päring	0.025 sek	0.025 sek	0.190 sek	0.724 sek	1.773 sek
3. päring	0.027 sek	0.017 sek	0.253 sek	0.992 sek	2.235 sek
4. päring	0.027 sek	0.027 sek	0.341 sek	1.002 sek	2.765 sek

Tabel 9. R-i tulemused masinal B

Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina, **Tanel Joosep**,

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose **Rakendus andmete visualiseerimiseks**,

   mille juhendajad on Toomas Haller ja Tõnis Tasa

   1. reprodutseerimiseks säilitamise ja üldsusele kättesaadavaks tegemise eesmärgil, sealhulgas digitaalarhiivi DSpace-is lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;
   2. üldsusele kättesaadavaks tegemiseks Tartu Ülikooli veebikeskkonna kaudu, sealhulgas digitaalarhiivi DSpace´i kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.
2. olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile.
3. kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest tulenevaid õigusi.

Tartus, 11.05.2017