TALLINNA TEHNIKAÜLIKOOL

Infotehnoloogia teaduskond

Tarkvarateaduse instituut

Ergo Siht, 178019IABM

OSKUS-TEST MUDELIL PÕHINEVATE SÜNTEETILISTE ANDMETE GENEREERIMINE

Magistritöö

Juhendaja: Ants Torim, PhD

Tallinn 2020

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autor: Ergo Siht

11.05.2020

Annotatsioon

Käesoleva töö eesmärgiks on välja töötada mudel, mis võimaldab genereerida sünteetilist andmet Oskus-Test andmemudeli põhjal. Selleks tuginetakse vaadeldavas töös FCA ja sünteetiliste andmete genereerimise põhimõtetele ja tehnoloogiatele. Loodav mudel peab võimaldama kasutajal kontrollida lihtsalt antavat sisendit ja selle alusel loodavat andmestruktuuri. Selleks, et vastav mudel luua, tuleb analüüsida lähteandmeid ja olemasolevaid lahendusi ja formuleerida hüpoteetiline mudel, mille järel see realiseerida ja tulemusi katsetada. Katsetamist ja mudeli kohandamist tuleb jätkata kuni saavutatakse soovitus tulemus.

Põhiprobleemiks on, et otseselt puuduvad eksisteerivad lahendused, mis laseks kasutajal selge sisendiga genereerida sünteetilist andmestiku Oskus-Test andmemudelina. Olemasolevad lahendused eksisteeriva küll klasterdamise meetodil, kuid need ei ole efektiivsed ärikasutaja poolest ja antud valdkonna uudsus viitab sellele, et antud andmestike genereerimist FCA-põhiste andmete jaoks ei ole palju uuritud.

Töö tulemuseks on analüüs, sellel põhinev mudel ja mudeli tarkvaraline realisatsioon. Lõplik analüüs põhineb olemasolevate lahenduste ja tehnoloogiate analüüsi ning näiteks võetud andmestiku vaatlustel põhjal. Tarkvaraline lahendus on loodud tuginedes analüüsi tulemustel.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 61 leheküljel, 8 peatükki, 49 joonist ja 1 tabelit.

Abstract

Generation of synthetic data based on Skill-Test model

The aim of this thesis is to develop a model, which is capable of generating synthetic data based on Skill-Test data model. To achieve this we are going to rely upon FCA and synthetic data generation principles and technologies. Created model needs to allow the user to control easily given inputs and to generate the desired data structure. To create such model, firstly we need to analyse the source data and existing solutions and then formulate hypothetical model, after which we need to develop and test it. Testing and correction must be continued until desired results have been established.

Main problem is that such solutions do not exist, which allow the user to generate synthetic Skill-Test data model based data with clear and easy to understand input. Existing solutions exist based on clustering technology, but they are not effective in the hands of business user and due to the novelty of the FCA area, the area has not been much researched.

As a result of this work, it is expected that produced is analysis, analysis-based model and programmatically developed solution. The end analysis shall be based on existing solutions and technologies analyses and source data observations.

The thesis is in Estonian and contains 61 pages of text, 8 chapters, 49 figures and 1 table.

Lühendite ja mõistete sõnastik

CSV	*Comma-separated values*, komadega eraldatud väärtused
FCA	*Formal concept analysis*, formaalne kontseptianalüüs
IAM	*Identity and access management*, identiteedi- ja juurdepääsu haldus
GDPR	*General Data Protection Regulation*, andmekaitse üldmäärus

Sisukord

Jooniste loetelu

Joonistе loetelu

1 Sissejuhatus

1.1 Taust ja probleem

Üha IT-kesksemaks muutuvas maailmas on suurenenud rõhk tarkvaraarendusel ja seda toetavatel valdkondadel selleks, et hoida töös nii riiki, majandust kui ka pakkuda inimsõbralikemaid teenuseid nii teenindava sektori kui ka riigi poolt. Mistahes tarkvaralahenduse puhul, mille peamine eesmärk on samaaegselt teenindada mitut klienti, on vajalik, et teenustase säiliks ilma liigsete viivitusteta. Samuti on suurem potentsiaal erinevatele riistvara piirangutest põhjustatud tõrgetele, kui jooksutatakse halvasti optimeeritud rakendust. Kõige seetõttu on tähtis, et tarkvara arenduses loodud tarkvara ise oleks optimaalselt töötav - s.t. et kasutatud on ajakohaseid ja efektiivseid meetodeid protsessi käivitamisel, mille tõttu on tähtis et tarkvaras kasutatud algoritmid ja üldised meetodid on sobivad vastavalt valitud ärilisele protsessile ja andmestikule. Pahatihti ei ole lihtne teada, milline lähenemine on äriliselt kõige efektiivsem, mille tõttu on võimalus vigade tegemiseks juba tarkvara arenduse alguses väga suur. Sellest probleemist tulenevalt üritabki see lõputöö tuua valgust valikute paremaks tegemisel töötades välja selleks mudeli, mis võimaldab arendajatel efektiivsemalt testida algoritme söötes sellele ette vastava ärilise valdkonnale omapärast sünteetilist andmestiku, näiteks e-kaubanduses toodete informatsioon ehk andmed, mis on näevad ja käituvad sarnaselt päris andmetele, kuid ei pärine päris äri tegevusest ja on enamasti genereeritud programmi poolt.

Arenduses on tihti peale vajalik katsetada erinevaid algoritme ja testida tarkvara töövõimet äriliselt sarnase, kuid ligilähedase andmestiku peal. Sellisest andmestikus on võimalik tarkvara osas langetada palju targemaid otsuseid efektiivsuse, jõudluse kui ka arhitektuuri osas. Tihtipeale on sellise andmestiku saamise puudusteks konfidentsiaalsuse tase või ebatäpsus ning puuduvad vahendid genereerimaks sobivat andmestiku ning seda seejärel kohandada - s.t. et genereerida saab sünteetilist andmestiku, kuid selle loomine on ajakulukas ehk palju programmeerimist või käsitööd nõudev tegevus ja olemuselt muutumatu. Seetõttu on suureks praktikaks paljudes IT ettevõtetes omavoliliselt kasutada kliendi andmeid või katsetada päris andmete vastu, mis on omakorda lepingu vastane või koguni seadusevastane. Erinevate valdkondade andmeklastrite objektidel on omad iseärasused, suurandmete kogus on erinev ja nende relatiivsus (tähtsuse kaal) on volatiivsed. Sellest tulenevalt on aktuaalne, et ei teata kasutatavate algoritmide sobivust ja raske on testida tarkvara korrektset töövoogu. Seetõttu on antud töö raames peamiseks probleemiks meetodite ja algoritmide sobivus - nende efektiivsus, sobivus kui ka õigsus.

Käesolevas töös keskendume me kontseptiahelatele ja nende aluseks olevale formaalse kontsepti analüüsi teooriale (FCA) selleks, et sünteesida Oskus-Test mudeli laadset andmestiku. FCA on meetod teadmiste esindamiseks, teabe haldamiseks ja andmete analüüsiks - antud meetod aitab leida ja visualiseerida kontsepte ning nende sõltuvusi sisendandmetest. Sellist sorti analüüsimine on vajalik erinevates valdkondades nagu näiteks värbamisel kandidaatide hindamine, hangetes tarkvara sobivus, kaubanduses erinevatel eesmärkidel toodete sobivuse / seostamise määratlemine, IAM-s ligipääsuõiguste väljatöötamine ja ajakohastamine kui ka formaalsete kontseptidega seotud testandmete genereerimiseks arenduses jms. Sellist laadi probleemide lahendamiseks püüab antud lõputöö luua meetodi, millega saab genereerida sünteetilisi andmeid nii, et see on kasutajasõbralik, vabavaraline ja võimaldab andmestiku struktuuri kasutajal kontrollida. Sellest tööst saavad kasu otseselt arendajad kui ka andmeanalüütikat teostavad insenerid, kes töötavad pidevalt erinevate andmestikega ja suurtele kasutajatele, suurele kasutaja arvutile või väga kõrge käideldavuse tasemega mõeldud tarkvara lahendustega. Veel enam tõstab töö tähtsust asjaolu, et suurte andmete süsteemid ja andmeanalüüs on muutumas lähitulevikus laialt kasutatavaks automatiseeritud kujul, mille tõttu muutub andmeanalüüs kergesti kolmandate osapoolte teenusena väikeettevõtjatele kättesaadavamaks või jõudes meie kodudesse tarkade tehniliste lahenduste läbi nagu näiteks sõrmejäljelugejad ustel ja arvutitel, näotuvastus mobiiltelefonidel, häälkäsklused, biomeetria abil isikute tuvastamine jms.

1.2 Ülesande püstitus

Töö peamiseks tulemuseks on luua mudel, mis võimaldab genereerida kasutajale sobivaid Oskus-Test andmemudeli laadset andmestiku. Selleks vaatleme lähemalt sünteetiliste andmete ja FCA valdkonda, analüüsime Oskus-Test andmemudeli omapärasid ja vajadusi, vaatleme seni valdkonnas tehtud tööd, kavandame hüpoteetilise mudeli, realiseerime selle ning teostame seejärel testimist, et valideerida mudelit.

Selleks on seatud järgmised täpsemad eesmärgid, mille abil on plaanitud soovitud tulemus saavutada:

1. Uurida sünteetiliste andmete ja FCA valdkonna seoseid ning varem teostatud tööd FCA põhiste sünteetiliste andmete genereerimiseks.
2. Töötada välja mudel sünteesimaks Oskus-Test laadseid andmeid.
3. Realiseerida hüpoteetiline mudel, mis võimaldab Oskus-Test mudelil põhineva FCA struktuurile sobiva andme sünteesimist.

4. Hinnata mudeli täpsust ning kasutusvõimet FCA struktuuriga andmete genereerimisel.

Oodatavaks tulemuseks on analüüs ja sellele tuginev mudel, mis võimaldab genereerida erinevaid interpretatsioone alusandmetest erinevate ärivaldkondadele sobival kujul.

1.3 Metoodika

Eesmärkide saavutamiseks kasutatakse peamiselt analüüsimeetodeid. Esialgu käiakse üle analüüsi tööd selgitav ja toetav teave ning kirjeldatakse meetodeid ja tehnoloogiaid. Seejärel analüüsitakse olemasolevaid lahendusi ja Oskus-Test andmemudelile iseloomulike andmeid. Analüüsi järel loodakse esialgne hüpoteetiline mudel, mis tugineb analüüsi tulemustel. Hüpoteetilise mudeli kavandamise järel realiseeritakse mudel Python-i moodulina. Programmilise realisatsiooni järel testitakse moodulit ja hinnatakse mudeli tulemust. Testimisel vaadeldakse mooduli tööpõhimõtet ja testitakse FCA-põhiste meetoditega selgitamaks välja kas moodul suudab sünteetilise andme genereerimisel tagada kontrollitava struktuuri.

Loodav tarkvara moodul peab olema eraldiseisev lahendus ja mitte eeldama kasutajalt kõrgeid tehnilisi teadmisi. Töö tegemisel tuginetakse vabavaraliselt leitavatel informatsiooni allikatel, potentsiaalsetel alam-moodulitel ning nende dokumentatsioonil ja Tallinna Tehnikaülikooli äriinfotehnoloogia bakalaureuse kui ka magistri erialalt saadud teadmistele.

1.4 Töö struktuur

Töö on jagatud peatükkideks vastavalt töö eesmärkidele - esimestes peatükkis käsitleme teemat puudutavat tausta ja seotud tehnoloogiaid ehk uurime lähemalt formaalseid kontseptiahelaid (FCA), vaatleme sünteetiliste andmete genereerimist ja FCA-põhiste andmete genereerimise eelnevat tööd ning kirjeldame Oskus-Test andmemudelit, mida üritame töö käigus sünteesida. Seejärel vaadatakse esmast analüüsi olemasolevatele näite andmete põhjal. Analüüsi järgi tuleb mudeli realiseerimise analüüs, milles käsitletakse hüpoteetilise mudeli formuleerimist. Lõpuks vaadatakse eelviimases peatükis mudeli tulemusi ja seejärel võetakse tulemused kokku.

2 Oskus-Test andmemudel ja FCA

Antud peatükk kirjeldab lähemalt, milline on vaadeldava töö käigus analüüsitavate andmete mudel ja millistele reeglitele see allub. Samuti kirjeldatakse seotud FCA tehnoloogiat.

2.1 Oskus-Test andmemudel

Oskus-Test andmemudelil põhinevaid andmeid käsitletakse antud töös nii sisend andmetena kui ka töö tulemusena genereerivate sünteetiliste andmete osas. Oskus-Test andmemudeli andmed on esitatud erinevate objektide võrdlusest tuleneva maatriksi alusel selliselt, et maatriksit defineeriv üks objekti tüüp on defineeritud ridadega ja teine objekti tüüp veergudega. Võrreldes Oskus-Test mudelit klastri kui ka formaalse kontsepti mõistetega, siis klastrite puhul on tegemist objektide hulgaga. Formaalse kontsepti korral on tegemist aga objektide ja atribuutide hulgaga [21]. Oskus-Test mudeli puhul on keskmisteks struktuuri elementideks testitavad (objektid) ja testid (atribuudid), mille puhul on atribuudid määratud järjestusega ehk permutatsiooniga ehk lineaarse järjestusega. Oskus-Test andmemudeli väärtused on defineeritud tõeväärtustena ehk tulenevalt sarnasustest tõeväärtustabeliga, saame kattuvuse alusel lugeda andmetest välja selles peituvaid kontseptsioone, mille alusel saame FCA teooria kohaselt kirjeldada andmestruktuuri keerukust ja objektide seoseid. Näitena saame käsitleda järgmist tabelit, võttes alus hilisemas peatükis analüüsitavate õpitulemuste maatriksi. Võttes aluseks, et andmeid kirjeldavates ridades on reastatud testitavad objektid, näiteks üliõpilased ja nendele vastupidiselt veergudes õppeaineid kirjeldavad objektid, siis nende omavahelise võrdlemise tulemusena saadakse risttabel, mis kirjeldab kahendväärtuse korral üliõpilase tulemust antud õppeainet sooritades. Kahendarvuna on määratud 0-na üliõpilase põrumine aines ja 1-ga selle läbimine. Tulemuse formaliseerimine eeldab, et nii üliõpilase kui ka õppeaine puhul eksisteerib ühiselt defineeritav tingimus, mille alusel on võimalik hinnata üliõpilase oskust või omadust antud õppeaine sooritamisel. Kuna tabel sisaldab ainult kahendväärtusena läbimise tulemust, siis ei kajasta tabel objektidega seotud oskuslike andmeid, mida loeme antud töö mõistes aspektideks. Selliseid tulemuse formuleerimises tähtsaks osutuvaks faktoriks võib olla rohkem kui üks tingimus ehk läbimise tulemust võib mõjutada rohkem kui üks selline aspekt. Võttes näiteks õppeaine X, mille läbimiseks vajab üliõpilane omadusi Y, U, I, siis sõltuvalt üliõpilase selliste omaduste Y, U, I alusel formuleeritakse ka lõplik

tulemus. Reaalses elus saame kõrvutada sellist võrdlust näiteks jooksuga aja vastu, mille puhul on mõjuvateks tingimusteks hinnataval objektil kiirus, füüsiline vorm, varustuse sobivus jms. Tabel kirjeldab näidis väärtusi andmemudelis.

	A1	A2	A3	A4
B1	1	1	0	0
B2	0	0	1	1
B3	1	1	0	0
B4	0	1	0	1

Joonis 1. Oskus-Test andmetabeli näide, kus testitavaid objekte on kirjeldatud tähisega B ja teste tähisega A.

2.2 FCA

Formaalne kontseptsioonianalüüs (FCA) on välja arendatud andmeanalüüsi meetod, mille populaarsus on hiljuti kasvanud erinevates valdkondades, mille peamine rõhk on andmete sise struktuuri kirjeldamisel ja peidetud teadmist ehk formaalsete mõistete kaevandamise. Meetodi peamine rõhk on objektide ja andmehulgas peituvate atribuutide risttabeli genereerimisel ja selle põhjal formaalsete mõistete kaardistamisel ning nende ahelate genereerimisel. [1]

Olemuselt kirjeldavad kontseptid formaalseid mõisted andmeklastreid ehk hulki, mis esindavad loomulike arusaamasid, nagu näiteks “Vees elavad organismis”, “Ülikoolis õppivad tudengid”, “Tagaveolise teljega sõidukid”, “Kõik 2-ga jagatavad täisarvud” jne. [22]

Selliseid mõisteid saame kirjeldada kui ühikuid, mis on inimmõtete formaalsed abstraktsioonid. Sellised abstraktsioonid võimaldavad väljendada andmete tähenduslikkust ja tõlgendada ühese arusaadavusega läbi kontseptide. Sellised konseptid võimaldavad kirjeldada tunnetuslike objekte läbi nendele kohaldatavate tunnuste alusel. Näiteks võib olla üheks selliseks kontseptiks metallist vahend, mis koondab endas kokku erinevaid tööriistu, masinaid jms. FCA-d võib seega vaadelda kui konseptuaalset klasterdamistehnikat, kuna see võimaldab andmeid konkreetsete kirjelduste alusel koondada sarnasesse abstraktsesse mõistesse. [22]

2.2.1 FCA rakendus

Võttes näiteks aluseks andmestiku erinevatest loomariigi esindajatest, suudab FCA kirjeldada erinevaid kontseptuaalseid arusaamasid ja neid andmestikust välja lugeda. Kontseptsioon ehk kontsept on FCA mõistes sarnaseid objekte ja nende atribuute kirjeldav üksus, mille puhul saame olla kindlad, et valitud hulgas olevad objektid on mingi kindla tingimuse alusel sarnased. FCA puhul kehtib seaduspärasus, et mida vähem on võetud kontseptsiooni kirjeldamise aluseks konkreetseid objekti atribuute (edaspidi omadusi), seda spetsiifilisem on kontseptsioon ja väiksemat alamhulka tervest andmest kirjeldava kontseptsiooniga. Sarnaselt vastupidi, mida suurem on kirjeldamiseks valitud omaduste valim, seda üldisemalt ja vähem kirjeldav on kontseptsioon. [23]

Näiteks võime kontseptuaalse analüüsi tulemusena leitavaid alamhulki kõrvutada loomariigis isendite klassifitseerimise süsteemiga. Kõige suurem kontseptsioon oleks sellisel juhul "Liik", kus kõik andmestikus leitavad objektid on mingit sorti liigi esindajad. Seejärel minnes objektide omadusi alusel spetsiifilisemaks ja võttis rohkem aluseks kontseptsiooni kirjeldavaid tingimusi, saame määrata perekonna, sugukonna, seltsi, hõimu, domeeni jms. kuni konkreetse isendini välja. Sellised kontseptid põhinevad ekstreemsemal juhul kattuvuse teoorial sarnaselt näiteks Karnaugh tõeväärtuste tabelile, kus tabeli kattuvuse järel leitakse seda kirjeldav täielik funktsioon. FCA puhul loetakse katvusega välja sarnaseid kontseptsioone.

Muus osas võimaldab meetod kirjeldada ka andmetes eksisteerivat müra, objektide põhjal otsuste langetamisel ebaolulisi omadusi ja selliste objektide vaheliste kontseptside vahelisi seoseid. Juhul kui andmestik sisaldab suurel määral üksikuid erinevaid kontseptsioone, kuhu kuuluvad tervikust valimist üksikud objektid, siis saame selle alusel defineerida terve valimi kohta eksisteerivaid ebanormaalsus ehk müra. Samuti on FCA-l kasutust andmestikust ebaoluliste omaduste välja sõelumisel. Kui kontseptsioonianalüüsi järel on tuvastatav, et mingisugune konkreetne omadus ei osale otseselt kontseptsiooni defineerimiseks - näiteks on terve andmestiku peale kontantselt sama või sarnase väärtusega, siis sellisel juhul saame suure tõenäosusega väita, et antud väärtus on kontseptsiooni defineerimisel ebaoluline. Võttes aluseks eelmise näite loomariigi klassifitseerimisest, oleks sellisel juhul näiteks konkreetse liigi all üldist liiki kirjeldava omadusega - näiteks koerte puhul käppade arv. Sarnaselt eelmisele näitele saaksime määrata objektide omaduste väärtuste vahelisi seoseid kontseptide vahel teades, et kui üks konkreetne omadus kirjeldab alati ainult mõnda kõrgema taseme kontseptsiooni tervest valimist, siis kehtib eeldus, et alati on madalam kontseptsioon kuuluv ka kõrgemasse kontseptsiooni. Selliseks näiteks oleks

loomariigi näitel delfiin, kellel on alati olemas uim, mis teeb temast 100% juhul alati ka veeloom või siis kõik kuuega jagatavad numbrid, mis on alati jagatavad ka kahega ja kolmega. [24]

Kokkuvõtvalt on FCA eripäraks andmete ja teadmiste kontseptuaalsel töötlemisel kolm lahutamatut komponenti: 1) andmeanalüüsis lisaväärtuse avastamine kontseptsiooni alusel ja kontseptidega nende põhjendamine, 2) andmete sõltuvuste avastamine ja nende põhjendamine ning 3) kontseptide ja sõltuvuste visualiseerimine ning lahtipakkimise võimekusega. Nende komponentide kooslus teeb FCA-st võimsa tööriista, mida on võimalik rakendada mitmesuguste probleemide lahendamiseks. Sellisteks näideteks on veel otsingu tulemuste klassifitseerimine üldtuntud teemadeks, geenianalüüs, informatsiooni kaevandamine olemasolevatest andmetest, programmikoodi analüüsis ja selle mõistmises, dokumentide ja failihalduses objektide organiseerimisel, tekstianalüüsis kui ka näiteks riikliku turvalisuse tagamisel inimkäitumise analüüsis. [2]

Viimase näite puhul on kasutatud formaalset kontseptianalüüsi selleks, et USA-s jälgida ja analüüsida riigis tehtud kõnesid eesmärgiga tuvastada ning profileerida potentsiaalselt pahatahtlike individuaale. Kontseptianalüüsi põhimõte seisneb antud näites selles, et iga potentsiaalne kontseptsioon esindab ühte konkreetset isikut või isikute gruppi. Sellisel juhul on võimalik tehtavate kõnede eristuse alusel kaardistada potentsiaalselt pahatahtlike mustreid ja selliste mustrite või isikute vahelisi seoseid - näiteks isikud, kes on omavahel kõnesid pidanud. Sellest andmetest on võimalik ainuüksi analüüsida ja leida vastavat inimgruppi ühendav keskne isik, kes on ka suure tõenäosusega vastava grupi liider. Alternatiivselt, kõik sellised isikud, kes jagavad palju samu ühendusi üksteise vahel, grupeeritakse ühte sarnasesse gruppi. Täpsemaks jälgimiseks muidugi ei piisa ainult telefoni numbritest ja väljavõttest ehk lisaks on vajalik rakendada ka muud kaasnevat informatsiooni nagu näiteks kellaajalised harjumused, kõne pikkused, kõne ja tekstisõnumite sisu, asukohta ja sellistes asukohtades asuvate kohtasid puudutav informatsioon jne. Lisandandme võimaldab sellisel juhul objektide omaduste spetsiifilisuse tõttu palju täpsemalt määrata selliseid inimgruppe, kes vastavad teada tuntud ohtlikele profiilidele. [3]

2.2.2 FCA haardelisus

FCA ei ole loomulikult piiratud ainult ühesuguste andmete rakendamisega, kuid on võimalik rakendada mitmesugustel andmekogudel eeldusega, et selliseid andmeid töödeldakse FCA meetodile vastavaks. See eeldab andmekogult, et selles sisalduvaid andmeid on võimalik diskreetida ja tõeväärtuse protsessi abil määrata selle hinnatav väärtus. [4]

Samuti on FCA tehnika rakendamisel omad piirangud, mis võivad erisuguste andmete analüüsimisel muuta meetodi rakendamise ebaefektiivseks. Sõltuvalt andmestruktuurist, võib ka

keskmise suurusega andmestik sisaldada sadu tuhandeid formaalseid mõisteid, muutes kontseptsioonivõre (võre, mis koosneb kontseptsioonide järjestusest puu kujul) loetamatuks ja juhitamatuks. Erinevalt kontseptide genereeritavast kogusest, on eriti ekstreemsemal juhul kujutavad andmetes leitavad objektid selliseid formaalseid mõisteid, kus mõned üksikud objektid kuuluvad sellesse mõistesse ja sarnasel tasandil on suur varieeruvus. Sellist nähtust nimetatakse andmestikus müraks ja põhjustavad üksikute väär kalduvuste põhjal omakorda erandlike formaalsete mõistete genereerimise. Näiteks oleks selliseks väärtuseks juhuslik arv X, mille väärtused võivad erineda nullist kuni mitmete sadade tuhandete - pangandusest võetud näitena pangakonto saldo või eelnevas NSA näites käsitletud mobiilinumber. Selliste välja X arvude väärtuste töötlemisel on tähtis, et teostatakse esialgset andmetöötlust või välja ignoreeritakse. Viimasena esitab tehnika ka piirangu andmekogudele, sest praegust tehnoloogilist taset arvestades puuduvad lahendused, mis võimaldaks suurte andmete korral kalkuleerida selles sisalduvate kontseptide arvu tulenevalt praegu olemasolevate tarkvarade limiteeringutest. Vahendid nagu näiteks ToscanaJ ja Concept Explorer võimaldavad kalkuleerida ja visualiseerida formaalseid mõisteid, kuid need ei ole disainitud käsitlema suurt mõistete koguseid. [4]

2.2.3 FCA tööpõhimõte ja formaalne kontekst

Formaalse kontseptianalüüsi alusel on võimalik tuletada andmestikus peituvad erineval tasemel peituvat ontoloogilisi sidemeid ehk objektide ja nende omaduste kogumite vahel peituvaid erisusi, sarnasusi ja loogilisi järeldusi. Iga formaalne mõiste ehk kontseptsioon ehk kontsept esindab hierarhias objektide hulka ja sellele kindlaks määratud omadusi. Sellisesse hulka kuuluvad objektid omavad ligilähedaselt sarnaseid või samasuguseid atribuudi väärtuseid valimit defineerivate väljade alusel - valimit võivad määrata kõik objekti väljad, kuid võivad ka olla osaliselt esindatud. Sellises hierarhias leiduva hulga objektide alamhulgad on oma koguselt vähem objekte sisalduvad ja seeläbi spetsiifilisemad, arvestades rohkem väljasid ja nende väärtuste klassifitseeritust. FCA tööpõhimõte toetub seega hulga ja võre teooriatele. [2]

Täpsemaks tööpõhimõtte selgitamiseks kasutame järgmist näidet tabelitest A, B ja diagrammist C. FCA kohaselt eksisteerib andmetest vastavalt objektide väljade alusel nende erisustel põhinev kogus formaalseid mõisteid ehk kontseptsioone. FCA eeldab, et kõik andmeväljad on võimalik olemuselt määrata tõeväärtuse põhjal ehk kas vastav väide andmevälja kohta vastab tõele või on väär. Kui andmed on tõlgendatavad tõeväärtuste põhjal, saab objekte tõeväärtuslikul kujul kujutada maatriks tabelis, kus veergu (või rida) defineerivad objektid ja reas (või veerus) on defineeritud andme tõeväärtusi kirjeldavad väljendused. [2]

Tabeli A põhjal on näites välja toodud kolm objekti, mida kirjeldavad seitse erinevat tõeväärtust. Selliste objektide vahelisi sarnasusi saab hinnata selle alusel, kui palju on tõeväärtuse veergudes C väljenduste kattuvust. Toodud näitest on näha, et kõik objektid on unikaalsed ja seega sisaldab kolme objekti hulk kolme erinevat formaalset mõistet. Mõistet täpsustavad igas veerus leiduvad väljendid. C väljendid ei oma otsest seost objektide omaduste väljadega vaid nende väärtustega. Näiteks kui andmes leidub väli, millel võib olla rohkem kui 2 erinevat väärtust, siis pole väidet võimalik kahendsüsteemis määratleda ja sellisel juhul on tabelis A väljendatud need väidete C1, C2 ja C3-ga. [2]

Formaalsete mõistete defineerimisel lähtutakse printsiibist, et hierarhiat ehitatakse ja määratletakse alustades kõige spetsiifilisemast väljast, mis defineerib suurima objektide hulga andmestikus. Antud näite tabelis B on selleks tabelist A väide C7, mis on kõigil objektidel ühine. Enne Sellest edasi võetakse kuni kõigi väidete ammendumiseni erinevaid väidete kombinatsioone järjest kasvavas väidete kogusega kombinatsioonis. Vaadeldes näidet, defineerib järgmisel tasemel objekte F1 ja F2 väited C4, objekte F1 ja F3 väited C6, C7 ja objekte F3 ja F2 väited C5, C7. Antud näites on kokku kontseptide koguarv lõplikult 6. Kontseptide hulka loeme selliseid andmehulki, mis eristuvad mingi kindla väite alusel. Näiteks kui andmestikus oleks ainult väide 7, siis oleks koosneks terve andmehulk ainult ühest samast kontseptsioonist. [2]

Diagrammil C on võimalik näha kontseptide hierarhilist ülesehitust ehk kontseptivõre. Antud hierarhias kirjeldatud kontseptid on kirjeldatud alt üles lahti selliselt, et alumine aste on kõige spetsiifilisema väärtuste valimi alusel määratletud kuni kõige kõrgema tasemeni välja, kus vaadeldakse objekte kõige pealiskaudsem tasemel. Antud näites on välja toodud näiteks kontseptsioon 5, mis sisaldub samuti kõrgema taseme hulka kirjeldavates kontseptides 1 ja 3. [2]

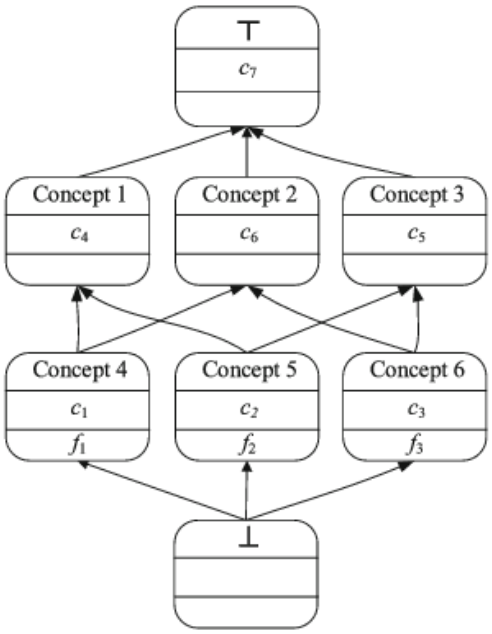
	c_1	c_2	c_3	c_4	c_5	c_6	c_7
f_1	x			x		x	x
f_2		x		x	x		x
f_3			x		x	x	x

(a)

⊤	$\{f_1, f_2, f_3\}$, $\{c_7\}$
Concept 1	$\{f_1, f_2\}$, $\{c_4, c_7\}$
Concept 2	$\{f_1, f_3\}$, $\{c_6, c_7\}$
Concept 3	$\{f_2, f_3\}$, $\{c_5, c_7\}$
Concept 4	$\{f_1\}$, $\{c_1, c_4, c_6, c_7\}$
Concept 5	$\{f_2\}$, $\{c_2, c_4, c_5, c_7\}$
Concept 6	$\{f_3\}$, $\{c_3, c_5, c_6, c_7\}$
⊥	$\{\emptyset\}$, $\{c_1, c_2, c_3, c_4, c_5, c_6, c_7\}$

(b)

(c)



Joonis 2. Näited FCA konseptsioonidest. [5]

Kokkuvõtvalt võime diagrammi C kirjeldada sedasi, et kõrgema taseme kontseptsioon sisaldab alati selle madalamaid tasemeid ja madalama taseme kontseptsioon omab ühiseid jooni kõrgema taseme kontseptidega, kuid eristuvad ühe või rohkema tingimuse alusel. Sellisel kujul formaliseerib FCA tehnika semantilised seosed kontseptide vahel. [2]

Kõige efektiivsemaks FCA rakendamiseks eeldab lähenemine fokusseerimist kontseptide arvule, rakendades andmetele eelnevalt teada tuntud minimeerimise ja filterdamise meetodeid, et eemaldada andmetest üleliigsed andmeväljad ja väikesi kontseptsioone (relatiivselt andmehulka üksikult kirjeldavate objektide väljad) kirjeldavad väljad (näiteks telefoni number). [4]

2.2.4 FCA rakendamine lõputöös ja seos Oskus-Test mudeliga

FCA teooria on kaudselt, kuid eelnevalt mainitud olemuselt, seotud töös kasutatava Oskus-Test andmemudeliga, kuid peamine rõhk on FCA-l andmekaeve alusel loodava sünteetilise andmemudeli valideerimisel, kus vaadeldakse andmete genereeritud sisemist struktuuri ja antakse lõplik hinnang loodavale hüpoteetilise mudeli realisatsioonile ja kasutatavusele. Valitud meetodi sobivust toetab mõlemal juhul kasutusel olev andmestruktuur, mis sisaldab endas konkreetseid väiteid. Nendeks väideteks on Oskus-Test mudelis testitava objekti tõeväärtus testi läbimisel. Oskus-Test mudelis vaadeldakse FCA tõeväärtustabeliga kõrvutades teste kui eraldiseisvaid väiteid, mille läbimisel loetakse seda kui üheks objekti kirjeldavaks välja väärtuseks. FCA põhjal on võimalik analüüsida kui palju on selliseid erinevaid objekte ja nende kontseptsioone ning sisemisi sarnasusi ette antud mudeli sisendist tulenevalt, kui palju on sarnaseid testide läbimise

stsenaariumeid. Oskus-Test andmemudelis on rohkem kontseptsioone, mida rohkem on erineval moel kombinatsioone läbitud testidest. Tuginedes FCA teooriale on võimalik eeldada, et kui sisendiks antakse laiemad parameetrid ja tingimused, siis on võimalik genereerida suuremas koguses erienvaid andmeid ning tingimuslikul mustri spetsifitseerimisel / kitsendamisel peab FCA andmeanalüüs tagastama vähem kontseptsioone ja tõendama, et andmetes on struktuurselt sarnasemad objektid.

3 Sünteetilised andmed ja genereerimine

Antud töö keskmeks on sünteetiliste andmete genereerimine. Sünteetilised andmed kujutavad endast sellist andmehulka, kus kõik andmed on kas täielikult või osaliselt tehislikult genereeritud imiteerimaks reaalelulist andmestiku, kuid pole kasutatavad reaalsetes äriprotsessides väljaspool arendusprotsessi või visualiseerimist. Sellistele andmetele ei rakendu seega ka konfidentsiaalsuse tingimused ja andmete omanikuks on genereerija mitte sisestaja. Selliseid andmeid genereeritakse konkreetsele juhusele vastavalt ja on mõeldud peamiselt kasutamiseks inseneride poolt analüütika ja mudelite disaini eesmärgil. Seega võib öelda, et kui sünteetiline andme imiteerib reaalelulist ärilist andmet, siis peab vastama sünteetiline andme sarnastele tingimustele, milleks on potentsiaalne rakendus äriprotsessides sarnaselt reaalsetele andmetele ja see peab olema samaaegselt täpne, dokumenteeritud ja hallatud, et seda saaks käsitleda asendusena tootmisseadmetele. Vastasel juhul on tegemist juhusliku andmega, millele puudub rakendus ja antud ärilises kontekstis ei saa lugeda andmeid enam ärijuhtumile vastavalt sünteetiliseks. Sünteetiliste andmete genereerimisel on unikaalseks omaduseks, et genereeritud andmed on alati unikaalselt määratletud ehk genereerida on võimalik sarnaseid andmeid, kuid andmed ei ole kahe genereerimise vahel kattuvad ehk samasugused. [25]

Eelnimetatud sünteetilise andmete genereerimine konfidentsiaalsuse eesmärgil on ainult üks reaalsetest põhjustest, miks sünteetilist andmet on vaja luua. Alternatiivselt konfidentsiaalsusele ja lepingulistele piirangutele võib esineda vajadus andmeloomes selleks, et kaitsta autentseid andmeid või andmete taga peituvat isikut [26]. Eriti tänapäeva infotehnoloogilises sfääris, kogutud andmed ja nendes peituv sisu on tihtipeale IT teenuseid pakkuva ettevõtte peamine väärtuse allikas ja tootmise vahendiks. Selles peituva informatsiooni levik konkurendile või muul viisil avalikustamine võib kaotada ettevõtte andmetest tuleneva eelise turgudel. Tihtipeale just viimast riski avaldavad ettevõtted endale, kui paljastavad andmeid kolmandatele osapooltele olenemata ärilise suhtluse tasemest.

Samuti on loodavad andmed kordades odavamad kui nende kogumine. Kogumine eeldab, et andmeid saadakse mõnest algallikast, kuid igal andmeallikal on enamasti määrav väärtus ja nende kogumine on kallis protseduur. Andmete sünteesimine pärsib olukorda, kus teadus- või äritegevus

jääb finantsallikate tõttu piiratuks. Loodavatel andmetel puudub otsene väärtus ja neid on võimalik luua genereerijale sobivas koguses tihtipeale lihtsalt ja kiiresti vastavalt muutuvale vajadusele.

Veel enam võimaldavad sünteetilised andmed luua täpselt sellist struktuuri andmevälja väärtustele, millist genereerija soovib saavutada. Võttes aluseks näiteks mitmete sadade isikute sisestatud andmestiku, on kindel, et andmed vajavad mingil moel korrigeerimist või kogumise meetod tugevaid kontrolle, et kogutud andmed oleks täpsed ja selgesti loetavad. Tihtipeale kulub tavapärase kogumise meetodit rakendades kas aega arendamisele või andmete korrigeerimisele. Kasutades andmete loomeks sünteesimiseks algoritme on võimalik kasutajal määrata täpselt millisel kujul on väljade väärtused määratud ja saavutab kindluse, et andmed kehtivad etteantud reeglitele kohaselt.

Sünteetilisi andmeid saab genereerida mitmel viisil, milleks on 1) tootmisandmete modifitseerimine ja 2) täielikul genereerimisel, milleks esimesena lahkame modifitseerimist. Modifitseerimiseks on mitmeid erinevaid meetodeid, mis kõik eeldavad, et esialgset tootmisandmed muudetakse sedasi, et see ei ole enam äriliselt konfidentsiaalne ja seega lubab kasutada andme modifitseerijal neid vabavaraliselt. Siiski eeldab see, et andmed säilitavad nendes määratud täpsuse ja struktuuri omaselt esialgsetele andmetel ning seetõttu ei ole see nii kergesti teostatav. Samuti eeldab selline protseduur pikemat protsessi, kui andmete täielik nullist genereerimine ja alati jäävad üles potentsiaalsed ohud, mis võivad andmetest tulenevad piirangud taasluua. Modifitseerimise parim näide on kõrvutatav GDPR-ga ja selles käsitletav andmete anonüümsus. Kui võtta hulk isikuandmeid, mis suudavad mingil konkreetsel viisil identifitseerida selles kajastatud objektide põhjal vähemalt mõnda konkreetset indiviidi, siis on tegemist juba äriliselt konfidentsiaalsete andmetega ja rakenduvad töötlemisel kindlad nõuded nagu näiteks isiku teavitamine ja nõusoleku saamine, et temaga seotud andmeid kasutatakse tootmises, profileerimises jms. Eemaldades sellest andmestikust isikule viitavad andmed või neid modifitseerides näiteks vale nimedega, numbritega, emaili aadressidega nii, et objektile külge jäänud andmete alusel ei ole võimalik tuvastada mõistliku vaevaga lähteandmete isik, on võimalik anonümiseerimise järel selliseid andmeid kasutada täpselt sellisel eesmärgil nagu modifitseerija neid kasutada soovib, ilma omamata vajadust teavitada ja saada sellekohast nõusolekut andmesubjektilt või andmeallika valdajalt. Näiteks vaadeldes haiglasse sisse registreeritud patsientide andmeid, siis need sisaldavad näiteks sisemist identifikaatorit, vanust, visiidi aega, haiglat, saabumise aega, osutatud ravi, sugu jms. Sellisel juhul sisaldaks anonümiseerimine ka muude sisemiste andmete modifitseerimist, mis võivad võimaldada haiglas tuvastada isikut mõne teise andmeallika alusel - näiteks sisemine identifikaator või visiidile saabumise aeg. [6]

Antud töös vaatame lähemalt sünteetiliste andmete genereerimise meetodit ja sellega seonduvalt täpsemalt struktuuri genereerimist. Sellisel juhul on eelduseks, et eksisteerib selge mudel, mille alusel on võimalik genereerida sünteetilist andmed vastavalt soovitud tingimustele. Tihtipeale on sellised algoritmid väga spetsiifilised ja võimaldavad üldkasutatavaid andmeid genereerida. Võttes näiteks eelmises lõigus käsitletud isikuandmed, on võimalik genereerida täielikult isikutega seotud erinevaid andmeid ja neid kombineerida. Kuna külastuste andmestruktuur on lihtne, siis andmeid on võimalik lihtsalt genereerida, kuid täpsuse tagamiseks on vajalik imiteerida päris andmetele korrektset sisemist korrelatsiooni ja mustreid. Selliseid mustreid tingimusi võimaldavad täita ainult kas eksperdi teadmiste rakendamine või andmeanalüüsi käigus leitavad seosed sarnaste andmete sisust. Viimase näite põhjal võivad selleks olla näiteks eksperditeadmised haigla protseduuridest, isikute külastuste tiheduse korrelatsioonid, kurdetud murede sagedused ja korrelatsioonid, mis võivad olla seotud väliste teguritega nagu näiteks aastaajad, haiguspuhangud jms. kui ka sisemises andmestruktuurid peituvad teadmised, mida ekspert ei ole võimeline ilma andmeanalüüsi tuvastama. Sellest tulenevalt viime ka käsitlevas töös läbi andmete analüüsi ja kaasame samuti ka mudeli koostamisel eksperditeadmisi ja kaasame tuvastatud andmestruktuuri iseärasusi ja tingimusi hüpoteetilise mudeli disainimisel.

Tänapäeval on sünteetilistel andmetel suur rõhk IT lahenduste arenduses. Insenerid vajavad nii algoritmide kui sisemise arhitektuuri ja ka testimise käigus reaalseid andmeid, et tagada loodava lahenduse efektiivsus ja optimeeritus kui ka veakindlus testimise järel. Samuti dikteerivad andmed tugevalt loodava lahenduse andmestruktuuri andmebaasis ja andmetele rakendatavaid kontrolle ning kitsendusi. Väga laialdast kasutamist on sünteetilistel andmetel andmekaeve ja masinõppe valdkonnas. Masinõppe Mudelid vajavad suures koguses algandmeid, mille alusel on võimalik neid õpetada konkreetset äri funktsiooni täitma - selliseks funktsiooniks on näiteks panganduses pettuse tuvastamise süsteem. Selliste andmetega õpetatakse mudelile kuidas mudel peaks reageerime erinevate tingimsute ja olukordade kokkulangemisel. Antud juhul määrab sünteetiline andme olulise rolli tulenevalt sellest, et reaalsed andmed ei pruugi tegelike ohuolukordi sisaldada ja sellisel juhul ei ole nendest ka kaetud vastavaid tingimusi ja olukordi, mis võimaldaks eelmises näites mudelil tuvastada toimuvat või toimunud pettust. Viimane on näide olukorrast, kus sünteetiline andme omab modifikatsioonidest tulenevat eelist päris elulise andme ees. [9]

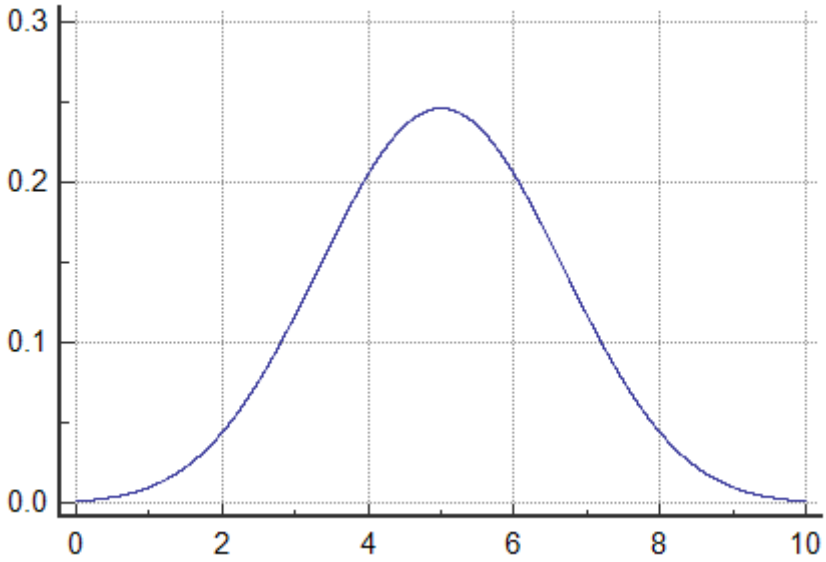
4 Sünteetiliste FCA struktuuriga andmete genereerimine

Tulenevalt FCA valdkonna uudsusest ja otstarbe spetsiifikast, milleks on peamiselt andmeanalüüs ja selle käigus kontseptide kaevandamine, on vähe selliste kontsepte genereerimisega seotud uuringuid. Antud töö raames tuvastati, et FCA põhiste kahendarvulise süsteemiga kontseptivõrede genereerimiseks on kasutusel väheseid meetodeid ja nendest peamiseks on Dirichlet jaotus [1]. Kõigi järgnevate näidete põhjal on ühiseks jooneks samuti protsessi juhuslikkus ehk andmetel ei ole rakendatud erilisi mustreid ega tingimusi, mida konkreetne äriline juhus võiks vajada. Järgmistes alapeatükkides vaatleme lähemalt erinevaid genereerimise meetodeid.

Kõige primitiivsem ja lihtsamaid meetodeid on kahendsüsteemis andmete genereerimiseks rakendada tavapärast mündi viskamise metoodikat. Lähenemise kohaselt tähendab see seda, et iga aspekti väärtust on võimalik initsialiseerida 50% tõenäosusega tõeseks ja järele jäänud 50% tõenäosusega vääraks. Antud mudel on oma iseloomult lihtne ja ei sisalda ühtegi mustrit ega muud implementatsiooni, mis aitaks selge struktuuriga andmestiku genereerida ja seda kontrollida. Metoodika aspektide arvust oleneb lõplik tulemus. Üldiselt võib öelda, et kui võtta küllalt suur arv aspekte ehk omadusi ja objekte ning tulemusi nende vahel genereerida, koondub erinevate objektide koondumine võrdselt erinevatesse kontseptidesse. Antud mudel on otseses sõltuvuses mündiviset kirjeldava funktsiooniga. Konkreetses uurimuses määratleti, et mündivise on kirjeldatav tavapärase Bernoulli jaotusega. Mündi loopimine on oma olemuselt alus sellele, et sellist tüüpi FCA-l põhinevate andmemudelite genereerimisel tuua sisse juhuslikkuse faktor. [1]

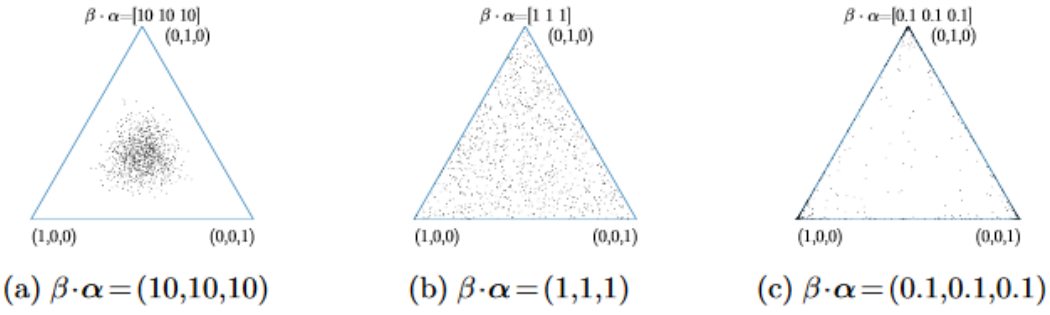
Juhul kui mündi viskele rakendada mõnda tõenäosusliku jaotust, võttes näiteks sama Bernoulli jaotuse, siis on ka genereerimisel leitavad tõenäosused korrelatsioonis etteantud jaotuse tõenäosus liikumist kirjeldava funktsiooniga. Näiteks määrates bernoulli tõenäosusliku jaotuse korral kalduvuse tõese väärtuse poole, seda tõenäolisem on, et tekib vähem kontseptsioone ja atribuutidele määratakse suurem tõenäosusega tõesemaid väärtusi. Alati säilib võimalus, et andmetes tekib müra ja kõige ebarealistlikumad kombinatsioonid võivad realiseeruda, mille tõttu funktsiooni alusel kirjeldav andmete kontseptide tekke arv suuresti seotud katsetega objekte luua. Siiski tekib kalduvus spetsiifilisema suunas rohkem tõeseid väärtusi omavate konseptsioonide suunas. Sellest järeldada on võimalik väita, et vastavalt tõenäosuslikule jaotusele on mudelil määratletav kallutatavus. Kui Bernoulli jaotuse alusel on mõlema mündi poole saamine samaväärne, siis on väga ebatõenäoline, et genereeritakse kontseptsioon, mille kõik väärtused on tõesed või vastupidiselt, kõik väärtused väärad ja rohkem genereeritakse kontseptsioone nende vahepealsete kosneptide jaoks. Alljärgneval joonisel on kujutatud Bernoulli tavapärane jaotus ilma

ühegi mõjutuseta, kus on kujutatud 50/50% võimalus saada väärtuseks tõene või väär vastus mündi heitmisel.



Joonis 3. Näide Bernoulli jaotusest. [7]

Eelmises lõigus vaatasime mündi viskele Bernoulli jaotuse rakendamist ja tõendasime, et erinevate jaotuste rakendamine tõenäosusele võimaldab genereerida erienvat sorti konseptsioonilise jaotusega sisalduvat andmestiku. Täiendavaks modifikatsiooniks mündi loopimisele on võimalik seega rakendada Dirichlet jaotus mudelit. Dirichlet jaotus võimaldab antud andmestikule rakendada tavalise beta jaotuse asemel rohkem kui kahte dimensiooni ehk muutujat, mille abil määratakse tõenäosuslik jaotus andmeväljade genereerimisel. See tähendab, et võttes aluseks alljärgneva joonise, siis jaotuse a korral on kontseptsioon tugevalt koonduvad ühte konkreetsesse klastrisse ja sellisel juhul on loodavad kontseptid tugevalt sarnased. Vaadeldes aga joonise jaotust b, on selline kontseptide jaotus üldiselt võrdne ja jaotuse c korral on klasterdatud tugevalt kolmnurga tippudesse, mille korral saame väita, et Dirichlet jaotuse (0.1, 0.1, 0.1) korral tekib kolma suuremasse gruppi koonduvaid kontseptsioone. [1]



Joonis 4. Dirichlet jaotuse näited erinevatest klasterdustest. [1]

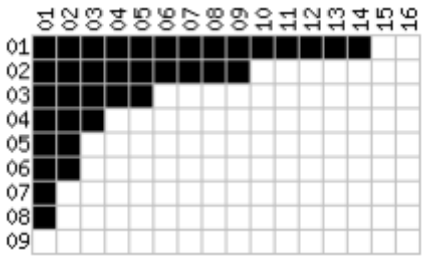
Alternatiivselt on võimalus samuti rakendada lihtsat moodust, milleks on olemasolevate andmete laiendamine andmetes leiduvate kontseptsiooni alusel. Selline tehnika eeldab siiski esialgset andmeanalüüsi, mille alusel on võimalik leida andmetes esinev struktuur ja seda genereerimisel imiteerida. Alternatiivselt ja sarnaselt võrdväärset saab ka andmete mitmekordistamist ehk skaleerimist rakendada olemasolevale andmekogule ja dublantide modifitseerimist, kui andmestikus esineb vajadus unikaalsete andmete olemasolule. Näiteks on see hästi rakendatav selles töös kasutataval algandmetel, kus on 429 üliõpilast ühe kursuse kohta võetud valemiks ja seega valim on küllalt suur tõendamaks selles peituvaid mustreid, et võtta tõsiseltvõetavaks aluseks.

5 Andmete analüüs

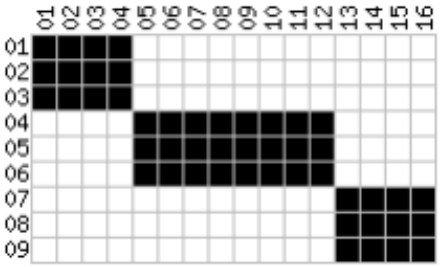
Antud peatükis analüüsitakse Oskus-Test andmemudeli omapärasusi ja koostatakse sünteetiliste andmete genereerimiseks mudelile nõuete nimekiri. Andmete analüüsiks kasutatakse lähteandmeid Tallinna Tehnikaülikooli reaalsetest ainete läbimiste nimekirjast. Eelnevalt on vaadeldavat sarnast andmehulka ja analüüsitud FCA kaevandamise meetodite abil Ants Torimi, Marko Metsa ja Kristo Rauni poolt ehk selles peatükis tugineme töö autori ja antud uurimise käigus leitud avastustele, et formuleerida vajalikud nõuded ja reeglid, mida mudel peab olema võimeline realiseerima.

5.1 FCA kaevandamise tehnika

Eelnevalt mainitud metoodika alusel saame hinnata mudeli poolt genereeritud andmestruktuuri keerukust. Kasutatav metoodika põhineb erinevate kontseptide leidmises ja nende põhiselt ahelate tuvastamisel. Leitavate ahelate koguse põhjal on võimalik hinnata kui erinevad on struktuuris loodud andmeobjektid üksteisest. Kontseptiahelaid analüüsitkase failistruktuuri põhjal kõige madalamast tasemest ehk alates andmeobjektist. Andmeobjekt on kontseptiahela mõistes kõige spetsiifilisem üksus, mille kirjeldamiseks on kasutatud kõiki objekti parameetreid ehk antud töö kontsekstis omadusi. Näiteks võime tuua näite kasside kolooniast ja eksisteerib eristumine ainult karvkatte värvuse alusel, siis eksisteerivad testid, millest üks testib valge värvuse ja teine musta värvuse kohta. Kuna kõik teised väited on samad ja eksisteerib kahte tüüpi kasse, kellest osa on valget ja teised musta värvust, siis eksisteerib antud näite andmestikus ainult kaks kontseptsiooni. Kõik kõrgema taseme kontseptid on omavahel samad, seega need ei eristu erinevatesse ahelatesse. Kontseptiahelaid hindame andmestruktuuris kas negatiivse või positiivsete konsepte kinnitavate väärtuste alusel. Tehnika selgituse lihtsustamiseks vaatame järgmisi konsepte. Vaadeldavatel joonistel on tõesed väärtused värvitud mustade kastidega ja negatiivsed valgetega.



Joonis 5. Ühe kontseptiahela näide [8]



Joonis 6. Mitme kontseptiahela näide [8]

Antud näite puhul on andmestruktuuris kirjeldab ainult üks kontsepti ahel. Vaadeldes näiteks ridasid 1 ja 2 saame öelda, et rida 2 on kõrgem kontseptsioon ja rida 1 on selle kontsepti spetsiifilisem vorm. Seda võib ka kirjeldada ainult üks andme rida, sest arusaama on täiendatud viie parameetri võrra ja ei ole juurde tekkinud erinevaid objekte, mis üksteisest saaksid spetsiifiliselt mõne tõese tingimuse alusel eristuda.

Vaadeldes nüüd ülemist jaotust, siis on selgelt eristatav, et klasterdumine on toimunud kolme suuremasse objektide hulka. Sellise näite korral toimub kontseptiahelateks hargnemine kohe analüüsi alguses, sest kõiki andmeobjekte ei ole võimalik kirjeldada ühegi kindla parameetri alusel, mis oleks kõigil ühiselt rakenduv.

5.2 Lähteandmete kirjeldus ja vaatlus

Antud töös on kasutatud analüüsimiseks lähteandmeid Tallinna Tehnikaülikooli kursuste tulemustest. Antud andmed on elulised ehk on tehtud autorile kättesaadavaks vaadeldava lõputöö juhendaja Ants Torim-i poolt. Kuna antud töö raames vaadeldakse Oskus-Test andmemudeleid, siis on aluseks võetud antud andmestik, mis vastab juba oma olemuselt binaarsuse nõuetele ja samuti on kinnitatava tähtsusega, et andmed on reaalsed ja neid ei ole modifitseeritud. Samuti on ülikoolis kursuste õppimine ja läbimine väga sarnane mudeli põhimõttele, sest ülikoolis teostatakse erinevaid teste ja testide teostajateks on üliõpilased, kellel kõigil on omad oskused või omadused, mis mõjutavad läbimist.

Lisaks teame, et kuna tegemist on ülikooli andmestikuga ja seetõttu on andmetel ülikooli protsessiline taust, siis saame antud andmete analüüsil tugineda ka eksperdi teadmistele. Eksperdi teadmisi kasutame defineerimaks protsessilisi iseärasusi, mida mudel peab arvestama ja samuti täpsustusi, mis jäävad antud töö raames vabatahtlikuks realiseerimiseks ehk nõuded, mis on liiga ülikooli valdkonna spetsiifilised. Viimane nõue on tähtis, sest mudel peab olema laialdaselt

kasutatav, seega on üheks põhiliseks nõudeks loodava mudeli universaalsus kasutada erinevates valdkondades, mis kasutavad algandmeid Oskus-Test andmemudeli põhjal. Järgneva andmestiku põhjal on iseloomulik sarnaselt Oskus-Test andmemudelile, et omavahel ei võrrelda ühte tüüpi andmeobjekte ehk üliõpilasi vaid üliõpilasi võrreldakse õppeainete vastu.

Lähteandmed on lisatud kaasa tööga ja on leitavad Näidis_andmed.xlsx failist.Vaadeldavatel andmetel on järgmised parameetrid:

1. Täielik tudengite arv on 423.
2. Teada ei ole õpilasi ja nende omadusi kirjeldavad parameetrid.
3. Täielik Õppeainete arv on 33.
4. Õppeainete arv on lõplik. Sellest tulenevalt kirjeldavad ained terve ühe Tallinna Tehnikaülikooli kursuse õpilaste hulka, kes on ühel ja samal ajal alustanud.
5. Teada ei ole õppeaineid ja nende omadusi kirjeldavad parameetrid.
6. Õppeainete sooritamise järjekord ei ole lähteandmetest teada ega tuletatav.
7. Lähtudes tausta teadmistest saab eeldada, et esimesena läbitakse üliõpilaste poolt eeldusained, mis on antud kursuse raames kergemad.
8. Sama eriala õppeainetel on omavahelised seosed. Osa aineid on eeldusained või tagavad õpilasele vajaliku tausta teadmisi, mida õpilane saab vajadusel kasutada järgmises aines läbimiseks.
9. Iga õpilase kohta on välja toodud aine läbimine binaarsel tasemel ehk kas õpilane läbis või ei läbinud ainet.
10. Antud andmestiku põhjal ei ole teada, milline on üliõpilase reaalne tase. Seda on võimalik parimal juhul ennustada terve andmestiku pealt võttes aluseks õpilase kõigi individuaalsete õppeainete läbimise taseme.
11. Antud andmestiku põhjal ei ole võimalik eristada kas läbi põrunud ained on proovitud sooritada või mitte. Andmestik ei kirjelda üliõpilase välja kukkumist. Seda on võimalik eeldada ainult selle põhjal kui palju on üliõpilane mahult põrunud ja sellest tulenedes hinnata tausta teadmiste alusel. Antud juhul kehtib ülikooli nominaalmahu läbimise reegel, mille puhul saab aluseks võtta fakti, et iga semestri kohta, mida õpilane sooritab, tuleb potentsiaalselt läbiga kuuest õppeainest 75% semestri kohta.
12. Õpilaste keskmine läbivuse protsent on ligikaudu 47%.
13. Õppeainete läbimise kõrgeim tase oli 66,67% ja madalaim 17,97%.

14. Andmed sisaldavad tudengeid, kes on kõik õppeained läbinud ja samuti neid, kes on kõik õppeained läbi põrunud.
15. Andmed kirjeldavad 6 erineva teaduskonna aineid, mille läbimise protsendid on:
    a. T, ~41,25%
    b. I, 48,31%
    c. Y, 52,72%
    d. H, 44,09%
    e. E, 34,99%
    f. U, 21,51%
16. Kokku on nendes andmetes osakondi 14, mille läbimise protsendid on:
    a. TE, 58,86%
    b. IT, 54,65%
    c. ID, 48,67%
    d. IE, 62,41%
    e. IS, 41,90%
    f. YM, 53,59%
    g. IR, 43,97%
    h. HL, 50,12%
    i. EK, 34,98%
    j. IA, 35,58%
    k. YF, 50,12%
    l. UT, 21,51%
    m. TT, 23,64%
    n. HH, 38,06%
17. Näidisandmetes sisaldub 21 õppetooli, mille läbimise protsendid on:
    a. TET, 58,87%
    b. ITV, 47,99%
    c. IDU, 59,04%
    d. IDK, 45,98%
    e. IED, 62, 41%
    f. ITI, 61,31%
    g. ISS, 37,75%
    h. ISP, 54,37%
    i. YMR, 63,12%

j. IRT, 43,97%
k. HLX, 50,12%
l. YMA, 56,26%
m. EKE, 34,99%
n. IAF, 47,52%
o. YFR, 50,12%
p. IDN, 17,97%
q. UTT, 21,51%
r. TTO, 23,64%
s. IAY, 23,64%
t. HHF, 38,06%
u. YMM, 41,37%

5.3 Formaalsete kontseptiahelate analüüs

2019. aastal viisid TTÜ tudengid läbi sarnase andmeanalüüsi andmestikule, mille käigus hinnati üliõpilaste taset. Antud uuringus vaadeldi andmeid 852 üliõpilaste tulemuste kohta ning tulemusi hinnati läbivuse põhjal sarnaselt nagu on Oskus-Test mudel defineeritud FCA-põhiselt ja kahend arvuliselt. Tulemuste põhjal avastati, et ~80% üliõpilaste tulemustest oli võimalik katta ühe kontseptsiahelaga. Vaatleme järgnevalt kolme esimest kontseptiahelat vaadeldava andmestiku põhjal, mille puhul vaatleme detailsemalt näidet läbipõrumistest ja nendest tekkivatest ahelatest.

Vaadeldavas analüüsis olid kontseptiahelates järjestatud raskuse järgi järjestatud õppeained. Õppeainete järjestus on korrelatsioonis selle aine õpetamise semestriga. Järgmisena on välja toodud väljavõte vastavast artiklist.

> 5. Loogiline programmeerimine (ITI0021), 5. Andmebaasid II (IDU0230), 6. Veebipõhiste rakenduste arhitektuur, disain ja tehnoloogia (IDU0200), 5. Algoritmid ja andmestruktuurid (ITI0050), 4. Andmebaasid I (IDU0220), 4. Tõenäosusteooria ja matemaatiline statistika (YMR0170), 6. Kasutajaliidesed (ITV0130), 5. Tarkvaratehnika (IDK0071), 5. Elektroonika (IED0150) Tud: 195 &
>
> 4. Süsteemiteooria (ISS0010), 2. Arvutid (IAF0041), 3. Programmeerimise põhikursus Javas (ITI0011), 4. Objektorienteeritud programmeerimine keeles Java (IDK0051), 3.

Side (IRT3930), 3. Võrgurakendused I (ITV0110), 3. Sissejuhatus infosüsteemidesse (IDU3350) Tud: 151 & 2. Füüsika I (YFR0011), 2. Mõõtmine (ISS0050), 2. Programmeerimise põhikursus keeles C (IDK1031) Tud: 126 & 4. Arvutivõrgud (ISP0040), 2. Keskkonnakaitse ja säästev areng (EKE0140), 3. Akadeemiline võõrkeel (HLX0020) + 3. Lineaaralgebra (YMA3710) Tud: 112

Uncovered ratio: 0.256523729589501

---

6. Mikro- ja makroökonoomika (TET3070), 3. Lineaaralgebra (YMA3710) Tud: 75

Uncovered ratio: 0.19594079047764382

---

1. Väljendusoskus (ISS0110) Tud: 27 & 1. Filosoofia (HHF3080), 1. Õpingukorraldus (UTT0010) Tud: 9 & 1. Matemaatiline analüüs I (YMM3731), 1. Õigusõpetus (TTO3160) Tud: 6

Uncovered ratio: 0.17808637265374638

Tabel 1. Näidis andmete kontseptianalüüsi näited põrumiste andmetele. [10]

Antud näite mõistes on õppeained meie mudeli mõistes testi objektid. Sellise ahela näitel on märgatav, et tugevalt on läbi kukkunud viimaste kursuste õppeaineid, mis on tavaliselt ka keerulisemad ja samuti on selles õppeaineid, mis paistavad omavat eeldusaineid, nagu näiteks Andmebaasid II (IDU0230). Antud ahelas on samuti kajastatud õppeaine Andmebaasid I (IDU0220), mis tõendab, et selle läbi kukkudes on ka kukutud suurel määral läbi ka Andmebaasid II (IDU0230) või seda pole sooritatud. Samuti olenevalt kursusest, on mõned ained nendest valikulised ja seetõttu ei ole andmestiku puhul teada, kas üliõpilane pidi seda läbima või mitte. Selliseks näiteks on näiteks Tõenäosusteooria ja matemaatiline statistika (YMR0170), kuid selliseid võib ka veel esineda.

Vaadeldes õppeainete õppetoole ja osakondasid ning ka nende õpetamise teaduskondi, siis on selgelt ka väljenduv muster, et põrumisel võib olla potentsiaalseks põhjuseks eelneva kogemuse puudumine, sest õppeainete koodid on selliste õppetoolide ja osakondade mõistes erinevad, kuigi õppeained on enamasti leitavad samast teaduskonnast. Sellest võib järeldada mustrit, et kuna

erinevad õppetoolid, osakonnad ja teaduskonnad käsitlevad reeglipäraselt erinevat sorti teemasid, siis omandavad tudengid vastavast valdkonnast saadavaid teadmisi rohkem või vähem, vastavalt selles esinevate ja vajaminevate omaduste alusel, mida siis konkreetselt selliste teaduskondade, osakondade, õppetoolide kombinatsioonis valdavalt vajatakse. See loob eelduse, et selliste õppeainete läbimisel määrab tugevat tähtsust valdkonnapõhine teadmine ehk Oskus-Test mudeli mõistes tudengile määratud omadus. Näiteks on andmetes välja toodud sellised ained nagu Lineaaralgebra (YMA3710), Arvutid (IAF0041), Side (IRT3930), Füüsika I (YFR0011), Keskkonnakaitse ja säästev areng (EKE0140), Akadeemiline võõrkeel (HLX0020), Mikro- ja makroökonoomika (TET3070) ja Objektorienteeritud programmeerimine keeles Java (IDK0051). Vaadeldes neid aineid on selgelt eristatav, et Mikro- ja makroökonoomika (TET3070) on majandus valdkonnaga seotud ja Lineaaralgebra (YMA3710) rohkem matemaatika ja loogilise mõtlemisega. Samas aga Arvutid (IAF0041) ja Side (IRT3930) on tehnilisemad ained. Samas on ka loodusteaduste valdkonda kuuluvaid aineid mitu, mis pole otseselt tehnilised, kuid võivad ka oma olemuselt vajada teist sorti omadusi nagu näiteks valdkonna vastu esinev huvi.

Vaadeldavast andmetest ei ole võimalik järeldada põrumise põhjust, kuid üldiselt paistab mustrina välja, et varasemate õppeainete põrumisel on väga tõenäoline, et põrutakse ka tulevased andmed. Vaadeldes esimest ahelat, siis on selgelt näha, et ahelas on näha suuremat osakaalu 195 üliõpilase näol, kes on põrunud 4., 5. ja 6. semestri aineid. Samas aga on ahelas koheselt pea 1/3 vähem tudengeid 126 õpilase näol, kelle ained jaotuvad rohkem keskmiste semestrite õppeainetega. Vaadeldes kolmandat ahelat, siis selles on välja toodud valdavalt kõik esimese semestri kursused. Selles andmehulgad kaetakse ainult 6 sellist üliõpilast, kes põrusid esimese semestri õppeaineid. Lõplikult saame järeldada sellest, et ühine muster on ahelatel, et vähem põrutakse ja seega läbitakse rohkem esimeste semestrite õppeaineid, mis on olemuselt lihtsamad ja põrutakse rohkem viimaste semestrite õppeaineid, mis on raskemad ja suurema tõenäosusega tuginevad eelnevatel teadmistel.

Viimase näite põhjal võime ka eeldada, et kuna semestrite koguarv on 6 ja tegemist on sellisel juhul 3 aasta pikkuse protsessiga kõigi selliste valitud ainete läbimiseks, siis võib ainete läbimisel muutuda tähtsaks ka ajalooline andmestik läbitud õppeainete näol sedasi, et kogemuslikel andmetel tekib ajaloolise progresseerumisega väärtuse langus või on need kontrollitud mõne muu tingimuse alusel, mis suudab määrata saavutatud taseme tegeliku väärtuse. Näiteks võib andmestikust eeldada, et kui põrutakse läbi rohkem viimaste semestrite õppeaineid, siis võib olla sellele eelneva kogemuse järel omandatud teadmised kas aegunud, unustatud või need on osaliselt omandatud.

Sellisel juhul esitavad andmed endas maksimaalse enesekindluse 50% juhusega, sest läbimiseks loetakse mis tahes üle 50% kursuse teadmiste esitamist hinnatavate testide näol.

Kui nüüd lähtuda ka ülikoolis kasutusel olevat süsteemi semestri lõikes õppe edukuse hindamisel, siis on potentsiaalselt viimaste semestrite kursuste põrumiseks ka potentsiaalne põhjus peituv selles. TTÜs on kasutusel hetkel süsteem, mille kohaselt peab tasuta õppimiseks läbima vähemalt 22,5 EAP mahus õppeaineid. See tähendab, et iga üliõpilane peab suutma nominaalmahust 75% täita iga semesti kohta ja võlgnevuse tekkimise korral tagantjärele neid õppeaineid sooritama. Sellisest õppekorraldusest tingituna on suur ka tõenäosus, et mõned tudengid kukuvad erialaõppelt välja või jätavad pooleli mõnel muul mõjuval põhjusel erialaõppe enne seda, kui nad jõuavad viimaste semestrite õppeaineteni ja seetõttu loetakse tabeli alusel õppeaine automaatselt läbi kukkunuks.

5.4 Lähteandmetest ja eksperdi teadmistest tulenevad nõuded

Selles peatükis vaatleme lähemalt andmetest tuletatavaid mustreid ja tingimusi, mida loodav mudel peab olema võimeline realiseerima ning mida ka mudeli hindamisel järgime.

Vaadeldavate andmete ja eksperditeadmiste kohaselt saame järeldada, et järgmine loodav mudel peab Oskus-Test andmete genereerimiseks järgima järgmiseid reegleid või vähemalt võimaldama kasutajal neid määrata või ignoreerida mudeli genereerimisel. Nendeks tingimusteks on:

- Määratav peab olema üliõpilaste arv.
- Määratav peab olema testide arv, mida üliõpilased saavad sooritada.
- Määratavad peab olema omaduste arvuline kogus. Selgelt on vajalik õppeainete läbimisel erinevat sorti teadmisi ja kasutajale peab olema see määratav.
    - Mudel peab toetama ühte või enamat omaduse määramist. Juhul kui testitaval objektil on üldine hinnatav tase, siis see on määratletud sedasi, et kasutaja saab määrata ühe domineeriva omaduse hulgast või ainult ühe omaduse.
    - Sellistele omadustele peab olema võimalik määrata kasutajale sobival kujul osakaal testist.
    - Sellised omadused määravad testitava objekti baas oskused ehk üliõpilaste näitel esialgsed isiksused tulenevad omadused.

- Kasutaja peab saama anda enda poolset sisendit võimalusel ka genereerimise vahendite läbi. Mudel ei tohi muuta kasutajale kohustuslikuks, et mudelit saab kasutada ainult kasutaja poolt sisestatud andmete alusel.
- Mudel peab jälgima üldiselt universaalsuse mustrit ehk see ei tohi muutuda spetsiifiliseks
- Mudel peab suutma rakendada võrdlus momenti, mille korral on võimalik võrrelda adekvaatselt testitavat testi objekti vastu sedasi, et eksisteeriks reaalne protsentuaalne võimalus läbimiseks kujul, mida rakendatakse reaaleluliselt. Antud näites ei piisa, et realiseeritakse lihtne võrdlus kahe taseme vahel, vaid peab tekkima võimalus, mille alusel läbitakse test ja rakendatakse juhusliku läbimise reeglit.
    - Juhul kui testi ja testitava objekt on omavahel sarnaste tasemetega määratletud, siis on võimalus, et üliõpilane läbib testi, ligilähedane 50%. Juhul kui aga test on palju keerulisem, kui üliõpilase antud hetkel määratletud omaduse tase, siis peab funktsiooni kohaselt olema määratletud palju suurem põrumise tõenäosus, kui see oleks testi ja testitavate objektide vahest tulenevalt. Näiteks kui test on kõigest 1,5 korda raskem, siis ei ole reaalne, et tudengi läbimise võimalus on 30-40%.
    - Mudel peab olema võimeline genereerima igal katsel unikaalseid kontseptsioone. Juhusliku läbimise reegel on see, mis tagab, et sisendandmete korral ei teki alati samasugune andme.

Valikuliselt on potentsiaalne võimalus, et ka järgmised tingimused võivad mõjutada tulemuse täpsust ja sellisel juhul peaksid olema realiseeritud kasutajale valikuliselt:

- Õppeainete läbimisel peab olema võimalus simuleerida õppeainete vahelistest seostest kogutavaid eelteadmisi, mida on võimalik siis tulevikus üliõpilasel enda kasuks rakendada.
    - Sellise funktsionaalsuse rakendamisel peab kasutaja saama määrata ka omandatavate teadmiste koguarvu kaalu. Sellega on kasutaja võimeline kontrollima kui palju on potentsiaalsest läbimisest võimalik teadmist leida ja millise kaaluga seda edasi kasutatakse.
- Õppeainete vaheliselt eksisteerivad teadmiste seosed ja kasutaja saab seda määrata või seda määratakse kasutaja eest. Võimalusel peab olema selline funktsionaalsus välja lülitatav. Käsitleme sellist funktsionaalsust edaspidi kui kogemusliku teadmiste kogumist.
    - Eksisteeriv võimalus, et ajalooline andme mõjutab tulevasi sooritusе tasemeid. Sellisel juhul peab mudel toetama võimalust määrata kindluse taset omandatud teadmistele, kui neid rakendatakse testide läbimise hindamisel.

- Tulenevalt ülikoolis esitatava õppeainete järjestusest peab olema mudelis võimalik määrata sooritatavate testide järjekorda vastavalt nende raskusastmete alusel.
- Mudel peab suutma imiteerida ülikooli siseseid õppeprotsesside spetsiifilisi korraldusi.
    - Näiteks on sellisteks täieliku põrumise faktori tutvustamine mudelisse, mille korral peab olema kontrollitav nominaalõppe tsükli raames soorituste tase ja üliõpilase jätkamine testide sooritamisel.

6 Mudeli realiseerimine

Mudeli koostamisel lähtume järgmisest moduleerimise printsiibist. Selleks, et mudelit määrata, oleme tuvastanud 4 suuremat alam moodulit, mida mudel peab realiseerima. Nendeks on 1) Sisendandmete määramine, 2) Objektide genereerimine, 3) valikuliselt Test tüüpi objektide sorteerimine ja 4) Sünteetiliste andmete genereerimine. Modulaarsuse eesmärk on võimaldada kasutajal rakendada osalise või täieliku eelandme olemasolul ainult kindlat moodulit. Siiski, antud töö raames vaadeldakse edaspidi mudelit täieliku tsüklina, millest vabatahtlikuks mooduliks on objektide järjestuse uuendamine sorteerimise vahendusel.

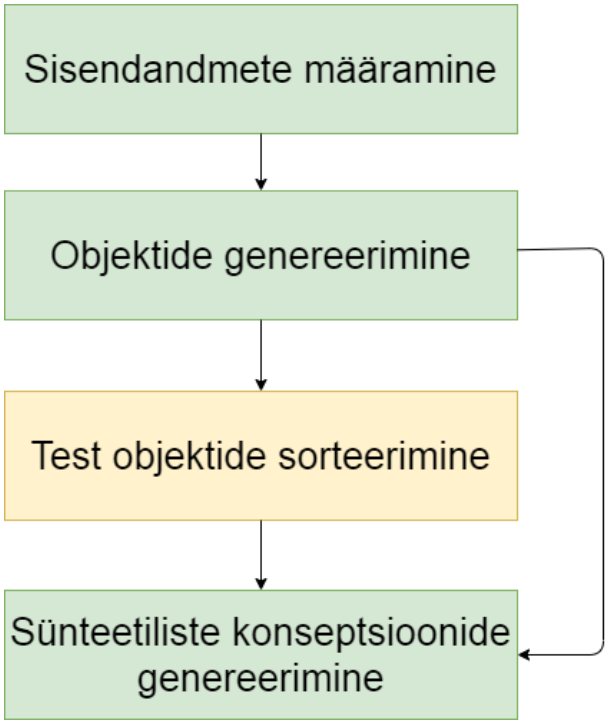
Sisendandmete määramise alam mooduli eesmärk on pakkuda ühtselt liidestuvat lahendust kasutajal erineval viisil sünteesimisele sisendi andmiseks. Sisendandmete määramisel vaatleme eelnevalt defineeritud üldnõudeid, milleks on 1) võimalus sisestada käsitsi kasutaja kogutud sisendandmeid (testide ja osksute objektide väärtused, kaalud jms.) ja 2) sellise sisendi puudumisel tulenevalt reeglite alusel määratleda mõlema objektide väärtuste jaotumise tõenäosus vahemikud. Sisendandmete töötlemisel väljastab moodul objektide genereerimiseks vajalikud tõenäosusjaotused.

Objektide genereerimise alam mooduli eesmärgiks on genereerida mõlemat tüüpi objekte. Mooduli eesmärk on olla üheselt pakendatud ja universaalne sedasi, et vajadusel saavad mõlemat sorti objektid käituda sarnaste reeglite alusel ja tulenevalt sisendi põhjal saab kasutaja määrata erisusi, mis objekti hulkadele kehtivad. Antud moodul on üks põhi moodulitest, mis määrab andmete sisemise struktuuri.

Test objektide sorteerimiseks kasutame eraldi moodulit eesmärgiga, et võimaldada reaalelulisemat lähenemist testide sooritamisele. Eesmärk on moodulil reastada testi objektid kasvavasse järjekorda imiteerides reaalses elus esinevat olukorda, kus potentsiaalsete testide lahendamisel esitatakse isikule esialgu lihtsamaid ja pärast raskemaid test vastavalt isiku arengule ja erinevate kursuste / õppe taseme muutusel. Antud moodul on kasutajale vabatahtlik ja selle tööpõhimõte on testide sisemise struktuuri alusel hinnata ja võrrelda ning seejärel järjestada tulenevalt üldise raskusastme alusel kasvav järjestus.

Viimaseks ja kõige tähtsamaks mooduliks on sünteetiliste andmete genereerimise moodul. Antud moodul ei ole kasutajale valikuliselt kasutatav ehk on alati kohustuslik. Mudeli eesmärk on sisend objektide ja tingimuste alusel realiseerida reaaleluliste andmete areng ja seeläbi genereerida erinevad testide ja testitavate objektide kombinatsioonidena.

Antud mudel realiseerib kõik eelmainitud alam moodulid ühte kasutajaliidesesse ehk eraldi võimalust pea funktsiooni kaudu ei võimaldada mudeli kriitilistele alam moodulitele - valikuliseks jääb kasutajale ainult test objektide sorteerimine.



Joonis 7. Mudeli sisemine kõrgel tasemel vaadeldav struktuur.

Mudeli peamiseks disaini jooneks on selle universaalsus, modullaarsus ja peab võimaldama imiteerida reaalelulisi olukordi, mida tavapäraselt näeme Oskus-Test mudelite korral võttes antud mudeli disainimisel aluseks ülikoolis õppivate tudengite näite. Viimane eeldab, et eksisteerib korrigeeritav ja kontrollitav viis simuleerimaks sisemist testitava objekti arenemist vastavuses sooritatud testidega. Samuti eeldab see, et objektidel on määratud erinevad oskus- või omaduste tasemed, mida testid peavad kontrollima. Reaalsest elust võetava näitena tähendab see näiteks seda, et kui õppeaine kehaline kasvatus hindab füüsilist omadust rohkem, kui alternatiivselt loogilise mõtlemise võimet, siis isegi kui objekti näitel on tegemist omaduste komplektiga kui geeniusega, siis reaalsel hinnatavate väärtuste põhjal on objekti läbimise võime piiratud füüsiliste omadustega ja seeläbi tegelik sooritus ka piiratud. Alternatiivselt võib ka esineda teste, mis ei hinda üldse objekti kõiki omadusi. Lihtsuse mõttes määrab automaatika kõigile omadustele alati mingisuguse

kaalu ja sellise funktsionaalsuse rakendamine on määratav käsitsi kasutaja poolt sünteetilise andme genereerimise moodulisse.

6.1 Sünteetiliste andmete genereerimise alam moodul

Kuna sünteetiliste andmete genereerimise moodul on mudeli tuumik komponent, siis sõltuvad muude moodulite realisatsioonid tugevalt selle realisatsioonist tekkivatest vajadustest. Selleks vaatleme järgmiselt järgmisi tingimusi, mida antud alam moodul peab suutma realiseerida. Andmete sünteesimisel on vajalik, et lisaks juhuslikult valitud objektide omadustele toetab mudel ka järgmisi kasutusjuhte, mis on omased Oskus-Test mudeli realiseerimisel:

1) Mudel peab realiseerima iga testitava objekti kohta baastaseme. Baastase on näiteks õpilase puhul olemuslik oskus, mis on Oskus-Test andmemudeli eelnevalt paika määratud ja võib tuleneda omaduste omapäradest, näiteks isikute puhul testide sooritamise eelne eelis teiste sarnaste isikute ees. Osa isikuid on oma loomuselt näiteks paremad loogilises mõtlemises ja teistel on geneetiliselt antud edukam füüsis, kui hinnatakse füüsilist omadust rohkem.
2) Mudel peab suutma imiteerida reaalelulist arengut objekti kohta;
    a) Testid on eraldiseisvad objektid ja ühe testi läbimine ei garanteeri, et järgmises testis eksisteerib seotud omadust sellisel tasemel nagu vajalik. Näiteks võttes alternatiivideks programmeerimise algkursuse ja ettevõtluse alused, siis neil kahel testil õppeainete näite põhjal ei eksisteeri potentsiaalselt ühtegi seost, seega sellistest õppeainetest ühe aine läbimisel ei garanteerita ega ei saa ka otseselt olla seost ega mõju teise õppeaine läbimisele.
    b) Eksisteerivad arengu sammud. Testi läbides omastab testitav objekt uue reitingu tema tasemele, mille kohaselt väljendatakse, et omandatud on uus teadmine või toimunud muud sorti tunnuse areng.
    c) Omandatud testidel on testide vahelised seosed. Läbides testile eeldus-testi võimaldab see kanda üle omandatud teadmisi seose protsentuaalse kattuvuse mahus.
3) Mudel peab toetama mitme omaduse samaaegset hindamist objekti kohta. Näiteks peab olema võimeline test hindama viiest omadusest peamiselt kolme, millega ta seotud on ja nendes määratud jaotuse alusel.
    a) Mudel peab samuti võimaldama määrata testidele tingimuslike kitsendusi, et milliseid omadusi test kõige rohkem realiseerib. Näiteks matemaatilise taustaga

aines realiseeritakse rohkem loogilist mõtlemist ja kehalises kasvatuses füüsilist taset kirjeldavat omadust.

4) Mudel peab võtma arvesse omaduse kindlusastet. Kindluse tase määrab ajaloolise tausta põhjal potentsiaalselt tegeliku omaduse taseme omandatud teadmistele. Kuna antud mudel töötab FCA-põhiste formaalsete kontseptidega, mis sisaldavad ainult tõeväärtuseid, siis ei ole teada tegelikult millisel tasemel sooritati potentsiaalne test. Kindlusaste võimaldab dünaamiliselt määrata algoritmil potentsiaalset taset ja ajaloolise soorituse alusel hinnata tegeliku omandatud võimekust.

6.1.1 Mudeli objektide omaduste reaaleluline areng ja võrdlus

Eelnevalt välja toodud FCA-põhiste kontseptide genereerimise meetodid mündi viske ja Dirichlet jaotuse alusel. Lähemal uurimisel puudub selliste meetodite kasutamisel oluline funktsionaalne komponent, milleks on otsuse rakendamine tulenevalt etteantud testitavate ja testide objektide põhjal, mille tõttu pole need otseselt kasutatavad. Samuti on nad tugevalt sõltuvad etteantud jaotuse struktuurist ja selle juhuslikkusest. Sellised realisatsioonid ei võimalda realiseerida ega kontrollida efektiivselt mudeli sisemist struktuuri ja genereerimise metoodika mustreid andmete genereerimisel.

Lähemal uurimisel eksisteerivad lahendused, mis kasutavad ELO reitingu süsteemi õpitulemsute hindamisel kui peamise taseme määramise komponendina. Samuti kasutavad sellised süsteemid ELO reitingut kui mitme omaduse valdkonna määratlemise vahendina ja samuti imiteerivad sisu kuvamisel isiku arenevat taset, et kuvada talle asjakohasemat õppematerjale. Sarnaselt on ELO väga levinud vahendina taseme hindamisel erinevates sportlikes kommuunides ja aladel nagu näiteks male, jalgpall, e-sport jms [17]. Kõigi nende ühiseks jooneks on ajas muutuv võistleja tulemus, mida väljendatakse ELO punktiskoori alusel ja mille muutumisel kasutatakse logaritmi, mis muudab võiduvõimaluste tõenäosuse määramise palju täpsemaks ja ligilähedamaks alternatiivse ajaloolise andme hindamisele [17]. Samuti on see laialdaselt levinud kihlveo saitidel, mille peamiseks eesmärgiks on tulemusi kõige täpsemini hinnata tulenevalt ärivaldkonna täpsuse kriitilisest [18]. Eksides tähendab see sellistel ekspertidel finantsilist kaotust. Kuna ELO reitingu süsteem põhineb tuleviku ennustamisel, siis saame sarnast kontseptsiooni rakendada andmete genereerimisel, mille tulemusena peaksid genereeritud andmed olema ligilähedased tulevikule ja sõltuv eelnevalt genereeritud andmetest igas objekti sünteesimise tsüklis.

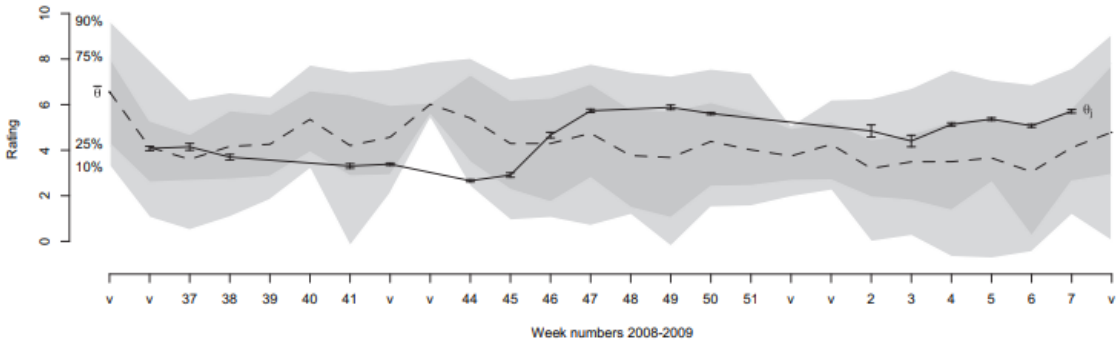
Näiteks on kasutusel ELO reiting ja selle edasiarendus Glicko kasutusel tugevalt süsteemides, mis hindavad kasutajate oskustasemeid. ELO peamine rõhk sellistes süsteemides on tuua sisse adaptiivse õppimise faktor. Adaptiivse õppimise faktori alusel suudab süsteem hinnata kasutaja praegust taset ja selle alusel soovitada õppematerjale ja kursuseid, mida kasutaja peab läbima. Sarnaselt Oskus-Test andmemudeli genereerimisele, ei kaota õpilane ka sellistes süsteemides juba omandatud teadmisi. Süsteemi eesmärk on kasutada ELO reitingut sedasi, et kasutajale pakutakse õppematerjale, mis on juba seotud eelnevate läbitud testide või selle tasemega ja seda teostada nõnda, et motiveerida kasutajat jätkama õpingutega. Alternatiivselt oleks kasutajale ette pakutud kõik erinevad kursused olenemata selle nõutavast tasemest ja tulenevalt mahust või testide kohesest keerukusest on võimalik, et kasutaja jätab palju suurema tõenäosusega õpingud pooleli. Sarnaselt kasutab ka Coursera süsteem meie esitatud nõudele kursuste vahelisi seoseid. [11]

Sarnaselt on kasutatud ELO veel ka teistes süsteemides nagu näiteks RIPPLE. Uuringu kohaselt leiti, et tavalise ELO asemel mitmik-ELO kasutamine ehk antud töö kontekstis ELO initsialiseerimine mitmele erinevale omadusele, andis mitu positiivset tulemust tavalise ELO ees. Esimeseks eeliseks oli valdkonna tasemel omaduste hindamine, mida on võimalik kasutada kasutaja suunamisel kursuste läbimisel. Teiseks, mitmik-ELO rakendamine mõjus positiivselt õpilaste tulemusele suurendades nende motivatsiooni taset kursuste läbimisel. Samuti suurendas see platvormi poolt pakutud kursuste soovituse usaldusväärsust kasutajate silmis. Lõpuks andis mitmik-ELO ka paremat sisendit õpilaste juhendajatele, sest tavalise ELO reitingu asemel oli võimalik ka mõista täpsemalt valdkonniti omistatud taset ja millised vahed õpilaste ning valdkondade vahel erinesid. Antud süsteemis on praktikas kasutatud keskmise ELO arvutamist ehk leitakse keskmine kompetentsus, et hinnata isikut sarnastel alusel. [12]

Mitmik-ELO rakendamise puhul on täheldatud 2006 aasta uuringus, et mitme ELO reitingu rakendamisel on määratulevad erinevad kaalud erinevatele omadustele, mis määratlevad tegeliku võidu tulemuse. Antud uuringu tulemusel leiti, et kõige tugevam tulemuse määramise omadus määras ainult kuni 25% tulemuse tegelikust võidu võimalusest ja selleks oli kogemus. Samuti määras erinevate muude määradega rolli male turniiridel osalemsie aktiivsus, intelligentsuse tase ja iseloomu omadused, kuid koos kogemusega olid need ainult 55% ulatuses otsustavad tegeliku tulemuse määramisel. [13]

Kolmanda näitena on veel ELO kasutusel Maths Gardeni keskkonnas. Antud süsteemi implementeerimisel Maths Gardenis uuriti õpilaste tulemuse kasvu ja leiti, et süsteemi rakendamine andis positiivse tulemuse. Antud näite puhul suutsid õpilased, kes olid oma

tulemustelt viimaste seas tõusta tippu. Õpilaste näite puhul märgati, et uue õppe alusel suutsid õpilased nelja nädalaga jõuda top 25% õpilaste hulka kasutades Maths Gardeni uut adaptiivset õppesüsteemi. Järgneval pildil on võimalik näha ajateljena kujutatult kahe süsteemi õpilaste keskmise taseme muutust. [14]



Joonis 8. Õpilaste tulemuste võrdlus ajas ELO põhise adaptiivse õppemudeli kasutusele võtmisel. [14]

Võrreldes samuti alternatiivsete masinõppe meetoditega on ELO reitingu rakendamine süsteemis ka jälgitav, mille tõttu on võimalik õpilase tulemusi vaadelda ka ajaloos. Samuti on see suurel skaalal vajaliku arvutuste põhjal palju lihtsam ja seega ka paremini sobivam töös kasutatava mudeli genereerimise rakendamisel, sest selle peamine rõhk on andmete genereerimisel suuremas koguses kui jooksvalt tulemuste uuendamisel. Samuti on ELO meetodi juurutamine palju lihtsam ja olemuselt hoomatavam kui keerulised masinõppe mudelid. [14]

Vaadeldes lähemalt ELO süsteemi positiivseid külgi saab välja tuua antud mudelisse sobivuseks järgmiste punktide alusel:

1. Osapoolte tugevuse tasemed
2. Initsiatiiv võita
3. Tulemuse sõltuvus võrdluste järjekorrast

Alternatiividena on võrreldud erinevaid spordi tulemuste ennustamise meetodeid: Võidu-Kaotuse meetod, Massey meetod, Colley meetod, Markovi meetod ja ELO meetodit. Antud meetodite ühiseks jooneks on see, et neid kasutatakse sarnaselt Oskus-Test andmemudelile objektide paaris võrdlemiseks. Meetodite võrdluse tulemusel leiti, et igal meetodil esinesid piirangud, kuid tulemuste põhjal saame hinnata, et ELO on sobivaim valik nendest valikutest, et rakendada seda tarkvara mooduli peamise loogika elemendina ja samuti tõendab, et eelnevalt välja toodud

haridussüsteemides on ELO kasutamine olnud optimaalseim. ELO kohaselt on kasutajal alati määratud võrdluses erinevad tugevused, mille alusel saab selgelt võrrelda objekte ja määrata võitjat. Samuti omad ELO süsteem võitmise initsiatiivi soosimist. Võrdluse kohaselt oli ELO puuduseks ainult tulemuse sõltuvus võrdluste järjekorrast. Loodava andmemudeli puhul teame eksperdi teadmistest, et tegelikult on järjestus sooritatavatel testidel, ülikooli näitel õppeainetel, üpris konstantne ja keerukuse kasvamise järjekorras, mille tõttu ELO reitingu puudus ei määra otsest rolli valiku teostamisel. Alternatiivide puhul olid puudusteks vähemalt üks teistest tingimustest. Kriitiline on antud loodava süsteemi puhul selgelt vajadus positiivset tulemust tagada testitaval objektil ja ka võrdluses sõltuvus vastase tugevusest, mille tõttu teised meetodid ei ole enam pädevad. [15]

Method	Property I	Property II	Property III
Win-Loss	×	✓	✓
Massey	×	✓	✓
Colley	×	✓	✓
Markov	✓	×	✓
Elo	✓	✓	×

Joonis 9. Võrdlus ELO ja alternatiivsete süsteemidest [15]

Eelneva tulemusel võib enesekindlalt väita, et ELO reitingu süsteemil on potentsiaali nii objektide tasemete määratlemisel kui ka nendevaheliste võrdluste teostamisel, mille tõttu realiseerib mudel ELO reitingu süsteemi peamise otsustamise komponendina. ELO süsteem on võimeline selgelt määrama reaalelulisemat otsustuse teostamise kvaliteeti ja samuti on sarnastes valdkondades efektiivselt kasutusel, mis tagab enesekindluse, et realiseeritud mudel on samuti võimeline sarnaseid tingimusi täitma. Seetõttu pühendub ülejäänud mudeli analüüs ELO reitingu põhjale ja realiseerib mudeli funktsionaalsuse selle ümber.

6.1.2 ELO võrdluse põhjal otsuse lahendamine

ELO reitingu süsteemi põhimõte seisneb selles, et reitingute muutuseid kontrollitakse kahe omavahel võrreldava objekti alusel. Võttes kahe objekti ELO reitingu, saab nende võrdlusest tõenäosuse, mille muutumise kiirus on seotud objektide ELO reitingu vahe suurusega. Mida suurem on vahe, seda ebatõenäolisem on võita või kaotada. ELO suudab sellist olukorda reaaleluliselt simuleerida kasutades selleks logistilist funktsiooni. ELO võidu tõenäosus on kalkuleeritud järgmise valemi alusel, milles Ra on võitu sooviva objekti tugevuse ELO reiting ja Rb teise võrreldava objekti ELO reiting. Mõlemale objektile leides antud valemiga ja liites tõenäosused annavad alati tulemuseks 1. Ea = 1/ (1 + 10**((Rb-Ra/400))). [16]

Antud töös kasutame sarnast lähenemist nagu on kasutatud eelnevalt mainitud õppe süsteemides adaptiivse õppe korraldamisel ehk testitava objekti võrdlusel testi objektiga leiame relevantsete omaduste ELO-de alusel tähtsuse kombinatsiooni ja kasutame seda objektide võrdlemisel. Alternatiivselt vaatame ka teisi kombinatsioone ELO-de põhjal otsustamiseks. Nendeks on 1) kõigi omaduste ELO võrdluste läbimise nõue, 2) juhuslikult valitud omaduse otsus läbimisel ja 3) läbitud testide põhjal protsentuaalselt läbimise otsustamine (näiteks viiest kolm läbib, siis 60%).

6.1.3 Dünaamiline omaduste rakendamine hindamisel

Standard ELO korral tähendab sündmust mõlema objektide muutust, mille korral kaotajal ELO vähendatakse ja võitjal ELO reitingut seeläbi suurendatakse. Suletud süsteemis on ELO reiting konstantne ja seega säiliv. Alternatiivselt on rakendatav ELO ülekandele kaotajalt võitjale eraldi konstant K-d, mille muutuse kiirust saab määrata dünaamilisemalt. Standard ELO puhul on see määratud üheselt ja jääb muutumatuks terve süsteemi elutsükli jooksul. Antud töös kasutame standard meetodit K konstandi määramiseks, millega saab kasutaja määrata käsitsi endale sobiva reitingu muutuse kiirust. [16]

Selleks, et pakkuda loodavas mudelis dünaamilisust ELO näol, peame realiseerima järgmised funktsionaalsused:

- kogemuslike andmete kogumine ja kogemuse ELO kalkuleerimine
- Testide vaheliste seoste määratlemine
- Kindluse taseme määratlemine
- Põrumise reegli rakendamine

Kogemuslike andmete kogumiseks kasutame tavapärast ELO reitingu uuendamist. Selleks arvutame välja eraldi ELO kandena läbitud testi korral selle järel lisatava ELO välja valemiga E = K(Sa-Ea), mille kohaselt ELO on võrdväärne K väärtuse ehk muutumise sammu korrutisega eeldatava ja tegeliku tõenäosuse vahega. [16]

Kuna süsteem hindab erinevaid omadusi erinevalt, siis uuendatakse ka omadusi vastavalt testitavale domeenile. See tähendab, et vastavalt testis kasutatud omaduste jaotusele määratakse ka kogemuslike ELO väärtuste tõus vastavalt. Näiteks kui test sisaldab 75% loogilist mõtlemist, siis uuendatavast ELO tasemest sisaldab kogu ELO-st loogiline mõtlemine 75% ja ülejäänud omaduste vahel jagatakse ülejäänud 25%. Selline lähenemine imiteerib mudelit, kus uuendatakse ainult seotud omaduste väärtust, mis on olnud kasutusel testi sooritamisel täpselt nagu õppeainete

sooritamisel, kus näiteks matemaatikas täiendatakse matemaatilisi teadmisi, kuid mitte füüsilist vormi.

Selleks, et rakenduks kogemuslikud omaduste ELO väärtused, tuleb rakendada testide vahelisi seoseid. Seose kohaselt on määratud millise protsentuaalse suuruse alusel on kaks erinevat testi omavahel seotud. Näiteks kui matemaatikas kasutatakse samu teadmisi nagu majanduse aines, siis võib olla seos näiteks 25%. Sellisel juhul, kui matemaatika on läbitud, siis kandub kogemuslik omadus üle läbi testide vahelise seose. Alternatiivse näitena ei ole kanduv üle matemaatika teadmised kehalisse kasvatusse, sest näiteks nende vahel ei eksisteeri ühtegi seost. Olemuselt üritab lahendus lahendada teadmiste jälgimist nagu on näiteks realiseeritud Bayese teadmiste jälgimise mudelis, kuid lihtsamas vormis.

Kuna mudel genereerib kahendsüsteemis määratletud väärtusi selliselt, et määratud on kas testi läbimine või põrumine, siis peab mudel suutma määrata kogutud kogemuse kindluse taset. Kindluse tase määrab millisel määral saab mudel usaldada eelnevalt kogutud ELO kogemust ja kasutab selleks ajaloolist andmet. Näiteks kui õpilane on läbinud ajalooliselt pooled testid, siis on tõenäoline, et tema sooritus ei ole kõigis testides 100%. Selleks üritab mudel imiteerida Bayese interferentsi sarnast loogikat, kuid kahe alternatiivse objekti tõenäosuse puudumise tõttu arvestab mudel ajaloos läbitud testide põhjal läbitud / proovitud testide suhet. Kuna läbimise puhul ei ole teada 50% testi tulemust, siis interference on määravaks poolele omandatud ELO-le ajaloos.

Viimasena peab mudel realiseerima õppekorralduslike reegleid. See on vajalik, et imiteerida kursuselt välja kukkumist õppeainete ebaeduka sooritamise tõttu. Mudeli keerukuse kontrollimiseks ei määra me testidele erinevaid testi kaale, kuid määrame intervalli, mille järel kontrollitakse maksimaalselt lubatud õppeainete põrumise kogust intervallide koguarvu kohta. Sedasi suudame me imiteerida 75% EAP-de omandamist nominaalõppe korral.

6.2 Sisendandmete määratlemise ja töötluse alam moodul

Sisendandmete määratlemise mooduli eesmärk on lugeda sisse kasutaja poolt vastavalt sisendile ette antud failidest äri juhtu kirjeldavat objektidele initsialiseeritavate algväärtuste sisendit. Selle mooduli tööpõhimõte jaotub kaheks eraldiseisvaks osaks, millest 1) võimaldab kasutajal anda ette eelnevalt genereeritud jaotuse mõne reaal elulise näite põhjal ja 2) selliste andmete puudumisel imiteerida läbi sisseehitatud funktsioonide alusel potentsiaalset jaotust. Mooduli lõpliku

tulemusena väljastatakse iga objekti tüübi ja tema omadusele omane jaotus, mille alusel saab andmete generaator luua vastavale jaotusele sobivad kuid juhuslikult määratletud objektid.

6.2.1 Sisendandmete sisselugemine olemaoleva taustandmed alusel

Antud mooduli selle alamtüki tööpõhimõte on olemuselt lihtne - moodul peab suutma lugeda sisse kasutaja poolt määratud failidest potentsiaalse jaotuse ja määrata selle alusel millised väärtused on tõenäolisemad objekti väärtuste genereerimisel. Sellise sisendi struktuur peab olema mõlema objekti tüübi kohta sarnane ning määratlev paindlikult. Failitüübi lihtsustamise ja üldkasutatavuse eesmärgil kasutame antud näites CSV failide sisendit ja sellest väärtuste lugemist.

Mudel arvestab oma põhimõttelt, et objektide puhul on määratud prioriteet omaduste määramisel testitavale objektile ehk kui testitaval objektil on omadusi määratud ainult kolm ja testidel neli, siis neljas ja testitaval objektil kirjeldatud puudulik omadus muutub veakontrolli järel antud andmemudelis ignoreeritakse. Alternatiivselt, kui testitavatel objektidel on rohkem omadusi kui testidel, siis sarnaselt eelmisele kombinatsioonile määratakse testidele küll sellised omadused, kuid nende kaal on alati null ehk ei oma tähtsust hindamisel.

Kasutaja peab saama määrata mõlema objekti korral konkreetsete objektide sisendiks samaväärse arusaamaga jaotused. Selleks peab suutma mudel sisend andmetest lugeda sisse jaotused õiges järjekorras. Tagamaks lihtsust ja vältimaks ka samaaegselt keerukaid konrolle ning kasutaja lisasisendi nõudmist valideerimisel, peavad sisendandmed andma sisse omadustega seotud andmed samas järjekorras. Kuna selliseid omadusi võib iga objekti kohta olla mitu, siis on mõistlik, et mudel realiseerib rea-põhise faili sisse lugemist.

Lihtsuse mõistes ja kasutajale arusaadavalt peab sisendandme imiteerima tõenäosusjaotust. Selleks peab andme andma ette tõenäosusjaotuse punktid koordinaat teljestikul. Sellised punktid koosnevad ainult positiivsetest koordinaatidest. Koordinaadid on määratud kahe tunnuse alusel: 1) ELO väärtus kui X-telge kirjeldab väärtus ja 2) ELO potentsiaalne kaal tervest andmestikust kui Y-telge kirjeldav väärtus ELO väärtuse kohta. Kasutaja peab saama selliseid väärtusi määrata koguseliselt selliselt nagu on vastav kogus algandmetes määratud. Minimaalseks nõudeks on, et vähemalt 1 punkt on määratud, kuid sellisel juhul oleks andmete varieeruvus määratud konstantselt. Sisse antav mudel imiteerib reaalses elus rakendatavaid histogramme, miles on määratud väärtus skaalal objektide arv seda konkreetset väärtust omavast alamhulgast. Mudel peab lugema automaatselt omaduse kohta ELO vahemikuks vastavalt sisendandmetes määratud X-telje vahemiku. Selles vahemikus määratud minimaalne väärtus on minimaalne ELO vahemik ja maksimaalne määratava ELO maksimaalne

6.2.2 Sisendandmete genereerimine ekspert teadmiste alusel

Kasutaja saab ekspert teadmistele tuginedes määrata sünteetilistele andmetele tingimuslikud sisendid. Vaadeldes varasemalt eelnevate FCA-põhiste andmete genereerimise Dirichlet põhist ja mündi viske meetodeid ning eelnevas lõigus käsitletud histogrami põhist kaalude / väärtuste seoste esitusviisi, siis üheks selliseks tingimuseks on määratav sarnaselt Dirichleti ja mündi heitmise meetodile sarnaselt sünteetilise jaotuse ette määramine, mille põhjal saab kasutaja edasiselt määrata jaotuse tingimuslike mutatsioone. Mudel realiseerib selliste jaotuste määramise kasutajale arusaadava sisendina. Jaotuste valim määratakse jaotuste tüüpide alusel sedasi, et oleks küllaldaselt erinevad jaotused funktsioonide põhjal määratletud ja kasutaja saaks erinevat tüüpi andmeid nende läbi kirjeldada. Lisaks peavad jaotused suutma määrata erinevat tüüpi mustreid.

Jaotuste funktsioonide alusel määratlemiseks realiseerib mudel seega järgmised funktsioonid:

1) Bernoulli
2) Eksponentsiaal
3) Poisson
4) Gamma
5) Normaal
6) ühtlane (uniform)
7) Binoom

Antud funktsioonid on valitud sedasi, et kasutaja saaks määrata lihtsalt laialdaselt kasutatud tõenäosuse funktsioonide alusel sisendandmeid. Näiteks on laialdaselt kasutusel Binoomjaotus, mille alusel võib kirjeldada mündi viset või IQ jaotust populatsioonis [20]. Eelmainitud jaotused on realiseeritud Pythoni Scikit statistiliste funktsioonide abil [19]. Funktsioonide lisaks on kasutajatele jäetud erinevad lihtsamad mustrid ja juhuslike arvude genereerimise võimalus etteantud vahemikule.

Lisaks oodatakse kasutajalt, et ta määrab baas ELO väärtused ja nendega seotud mürad mõlema objekti kohta. Müra on määratud samuti ELO reitinguna ehk kui baas ELO on objektile 1500 ja müra 200, siis minimaalne ja maksimaalne ELO väärtus on 1300 ja 1700, mida generaator saab objektile funktsiooni piires rakendada. Ka vajab mudel sisendina kasutajalt soovitud objektide koguseid.

6.2.3 Sisendandmete töötlemine

Juhul kui generaatorile antakse ette failid CSV kujul, siis loeb mudel eraldi failidest välja testide ELO jaotused ja mõlemat tüüpi objektide kaalutud jaotused iga omaduse kohta. Testide ELO

sisemise struktuuri genereerimiseks (millises mahus millist omadust test rakendab) piisab nimekirjast iga aspekti tõenäosuse kohta samal real. Juhul kui funktsioon loeb välja omaduste objektide pealt rohkem tõenäosusi, siis puudulikud tõenäosused määratakse kas 0 väärtusega või vaba tõenäosuse olemasolul võrdseks. Selliseid tõenäosusi kasutatakse selleks, et programmiliselt genereerida testidele omaduste protsentuaalne osakaal läbimiseks. Sisse loetud väärtused on maksimaalne tõenäosuse kaal, mida väärtus saab omada. Näiteks 3 omaduse korral oleks 0.7, 0.2 ja 0.1. Sellise jaotuse põhjal omab esimene omadus kõige suuremat kaalu potentsiaali.

Sarnaselt saab sisestada CSV failina objektide kaale. Iga omaduse kohta sisaldab CSV fail kaht rida. Kaks rida on vajalik selleks, et määratleda kasutajale sobivas mahus sisendid. Read käituvad oma olemuselt koordinaat teljestikuna, kus esimene rida määrab teisel real samal positsioonil leitavale väärtusele kaalu tervest jaotusest. Kuna omadusi loetakse ridade kaupa, siis saab kasutaja saab määrata selle tulemusena lõpmatu arvu omadusi ja samuti pole limiteeritud jaotuse punktide määratlus.

Objektide puhul on väärtused alati määratletud vahemikuna. Selleks, et vähendada kasutaja pingutusi ja suurendada etteantud jaotuste kasutatavust, rakendatakse skaleerimise funktsiooni ja laiendatakse ette antud mahtu 1000-le andmepunktile. Kahe kõrvalise väärtuse vahel kalkuleeritakse sujuv üleminek võttes arvesse neid ümbritsevaid punkte.

6.3 Objektide genereerimise alam moodul

Objektide genereerimiseks kasutatakse vastavat objekti tüüpi kirjeldavate omaduste jaotusi. Iga objekti tüübi kohta on määratud programmiliselt omadused ja iga omaduse kohta ELO jaotus. Etteantud koguste alusel genereeritakse nimekiri objektidest, mida edaspidi antakse sisendiks andmete sünteesimise moodulile.

7 Mudeli testimine ja tulemused

Antud peatükis vaatleme lähemalt realiseeritud mooduli tulemusi ja hindame selle ning eelnevalt määratud universaalsuse tingimuste põhjal mudeli sobilikkust probleemi lahendamisel. Mudeli tööpõhimõtte hindamiseks kasutame Ants Torimi poolt loodud kontseptiahelate analüüsi tööriista, mille alusel on ka kaardistatud sarnaste õppeedukuste andmed peatükis 5. Vaatame esmalt loodud ELO põhise sünteesimise mudelit lähemalt, testime seda, vaatleme alternatiivset mündi viskamise mudelit ja võtame kokku võrreldavad tulemused ja mudeli loome edukuse.

ELO põhjal andmete sünteesimise pseudokood üliõpilaste, testide ja omaduste näitel:

```
def generate_sünteetiline andmestik(Üliõpilased, Testid,
Testide omaduste kaalud, Dünaamiline ELO, Põrumise
tingimus = 0, Testide seosed, Põrumise tsükli testide
arv, Tsükli lubatud põrumiste koguarv, Testide erinevad
kaalud aktiveeritud, Kogemusliku ELO maksimaalne leidmise
kogus, ELO otsustamise loogika):
     sünteetiline andmestik = []
     for üliõpilane in range(0, Üliõpilaste pikkus):
          üliõpilase ELO tulemused = []
          Kogutud kogemuslikud ELOd = []
          Üliõpilane on välja visatud = 0
          for test in range(0, Testide pikkus):
               if (test % Põrumise tsükli testide arv ==
0) and Põrumise tingimus == 1:
                    lubatud põrumiste koguarv =
(test/Põrumise tsükli testide arv)*Tsükli lubatud
põrumiste koguarv
                    üliõpilase ELO tulemuste summa = 0
                    for tulemus in range(0, üliõpilase
ELO tulemuste pikkus):
                         üliõpilase ELO tulemuste summa =
üliõpilase ELO tulemuste summa + üliõpilase ELO
tulemused[tulemus]
                    if (üliõpilase ELO tulemuste summa +
lubatud põrumiste koguarv) < üliõpilase ELO tulemuste
pikkus:
                          Üliõpilane on välja visatud = 1
               if Üliõpilane on välja visatud == 0:
                    üliõpilase objekt =
Üliõpilased[üliõpilane]
                    if Testide seoste pikkus > 0 and
```

```
Dünaamiline ELO == 1:
                        üliõpilase objekt =
calc_experienced_aspects(test, üliõpilase objekt, Testide
seosed, üliõpilase ELO tulemused, Kogutud kogemuslikud
ELOd, Testid[test])
                    tulemus = compare_elo(üliõpilase
objekt, Testid[test], Testide omaduste kaalud[test],
Testide erinevad kaalud aktiveeritud, ELO otsustamise
loogika = ELO otsustamise loogika)
                    lisa üliõpilase ELO tulemustesse
võrdluse tulemus
                    lisa kogutud kogemuslike ELOde sekka
testist saadud uus kogemus (calc_added_ELO(tulemus[1],
Testide omaduste kaalud[test], tulemus[0], Kogemusliku
ELO maksimaalne leidmise kogus))
               else:
                    lisa üliõpilase ELO tulemustesse
negatiivne tulemus
          lisa sünteetiline andmestiku nimekirja ELO
tulemused (üliõpilase ELO tulemused)
     return numpy.array(sünteetiline andmestik)
```

Joonis 10. ELO põhjal andmete sünteesimise pseudokood.

Genereerimise ELO otsustamise alammooduli pseudokood (alammoodul otsuse langetamiseks):

```
def compare_elo(üliõpilase omadused, testi omadused,
testi omaduste osakaalud, testi omaduste osakaalud sisse
lülitatud, ELO otsustamise loogika):
     tulemus = []
     üliõpilase elo = calc_object_elo(üliõpilase
omadused, testi omaduste osakaalud, testi omaduste
osakaalud sisse lülitatud)
     testi elo = calc_object_elo(testi omadused, testi
omaduste osakaalud, testi omaduste osakaalud sisse
lülitatud)
     if ELO otsustamise loogika == 'ELO_combination':
          ELO võimalus võita =
calc_elo_probability(üliõpilase elo, testi elo)
          lisa tulemustesse testi
tulemus(calc_test_pass(ELO võimalus võita))
     elif ELO otsustamise loogika == 'Random_ELO_choice':
          üliõpilase suvalise omaduse ELO =
choice(üliõpilase omadused)
          testi suvalise omaduse ELO = choice(testi
omaduse d)
          ELO võimalus võita =
calc_elo_probability(üliõpilase suvalise omaduse ELO,
testi suvalise omaduse ELO)
          lisa tulemustesse testi tulemus
```

```
(calc_test_pass(ELO võimalus võita))
     elif ELO otsustamise loogika ==
'All_pass_requirement':
          testide tulemused = []
          for omadus in range(0, üliõpilase omaduste
pikkus):
               ELo võimalus =
calc_elo_probability(üliõpilase omadused[omadus], testi
omadused[omadus])
               lisa testide tulemustesse
tulemus(calc_test_pass(ELo võimalus))
          if all(testide tulemused) == True:
               lisa tulemustesse positiivne tulemus
          else:
               lisa tulemustesse negatiivne tulemus
     elif ELO otsustamise loogika ==
'ELO_Probability_combination':
          üliõpilase koguvõimalus = 0.0
          testi koguvõimalus = 0.0
          for omadus in range(0, üliõpilase omaduste
pikkus):
               üliõpilase koguvõimalus = üliõpilase
koguvõimalus + calc_elo_probability(üliõpilase
omadused[omadus], testi omadused[omadus])
               testi koguvõimalus = testi koguvõimalus +
calc_elo_probability(testi omadused[omadus], üliõpilase
omadused[omadus])
          ELO võimalus võita = üliõpilase
koguvõimalus/(üliõpilase koguvõimalus + testi
koguvõimalus)
          lisa tulemustesse testi tulemus
(calc_test_pass(ELO võimalus võita))
     else:
          print('Wrong ELO mode input. Correct and try
again')
          sys.exit()
     return tulemus
```

Joonis 11. ELO otsustamise loogika pseudokood.

7.1 Mudeli testimine formaalse kontseptiahelate leidmise meetodiga ja tulemuste analüüs

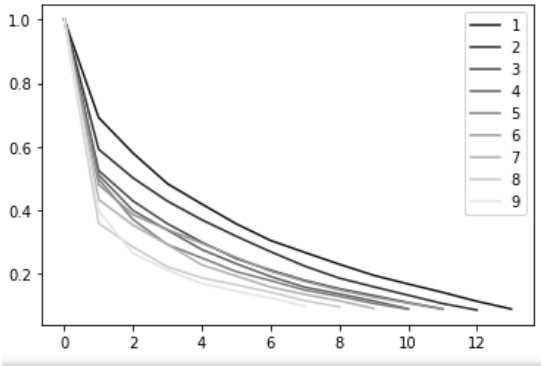
Järgmisena vaatleme sisendandmete muutuseid ja nende mõju mudeli abil sünteesitud andmetele. Selleks kasutame FCA-põhist kontseptiahelate leidmise tööriista. Kontseptiahelate alusel saame hinnata kui keerukas on sisemiste andmete väärtustest tulenev struktuur ja milline on sellise struktuuri andmestiku ülemineku kiirus keerulisemale tasandile. Järgnevatel joonistel kirjeldab Y-

skaala kattuvuse protsentuaalset vahemiku (Skaala on määratud 1.0-st kuni 0.0-ni, mis näitab kui palju on iga sammu kohta jäänud veel konseptse, mida pole üheski ahelas veel leitud). X-skaala näitab mitu kontseptiahelat on ajas leitud. Joonisel välja toodud parempoolne skaala väärtustega ühest kuni üheksani näitab andmestruktuurile sisendiks antud keerukust omaduste arvu alusel. Näiteks kui väärtus on neli, siis on iga objekti kohta määratud neli omadust. Testide tegemiseks kasutame arvuti poolt arvutatud objektide sisendiga funktsiooni. Teste jooksutame sama sisendandmete kohta kõigi nelja ELO võrdluse meetodi kohta.

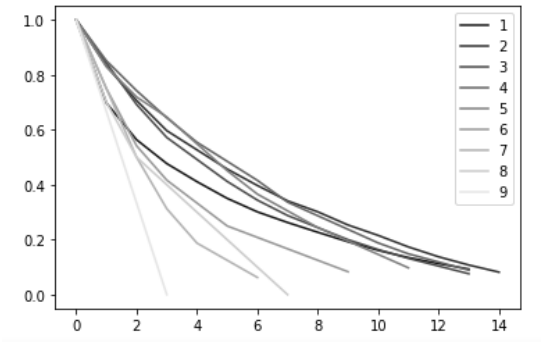
Testidel kasutatakse baas väärtustena väärtusi järgmiste sisendina:

- generate_samples_amount = 100,
- generate_tests_amount = 20,
- aspects_amount = i (määratud programmi poolt väärtuste 1-9 vahel),
- sort_tests_asc = 0,
- dynamic_elo = 0,
- fail_condition_samples = 0,
- interference = 0,
- sample_base_elo = 1500,
- sample_noise_elo = 200,
- test_base_elo = 1500,
- test_noise_elo = 200,
- sample_distribution_type = 'random',
- test_distribution_type = 'random',
- scramble_aspects = 1,
- scramble_objects = 0,
- sample_tilt_pos_elo = 1350,
- test_tilt_pos_elo = 1350,
- reverse_aspects = 0,
- test_aspect_ratios_random = 1,
- test_aspect_ratios_active = 1,
- experience_maximum_gain_ELO = 1500

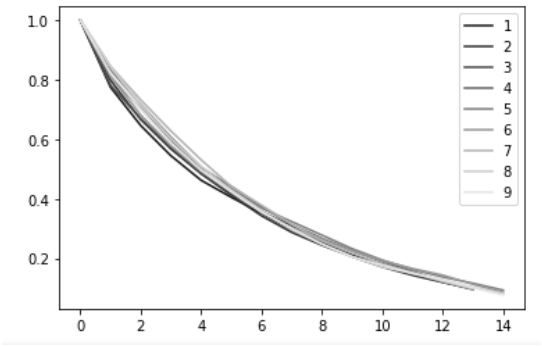
Võrdluspunkti määramiseks arvutame standard väärtused:



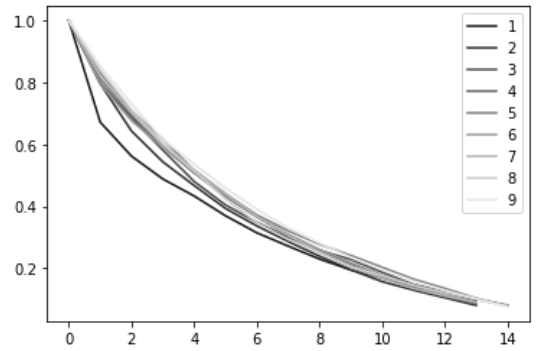
Joonis 12. ELO Katse 1: ELO_combination



Joonis 13. ELO Katse 1: All_pass_requirement

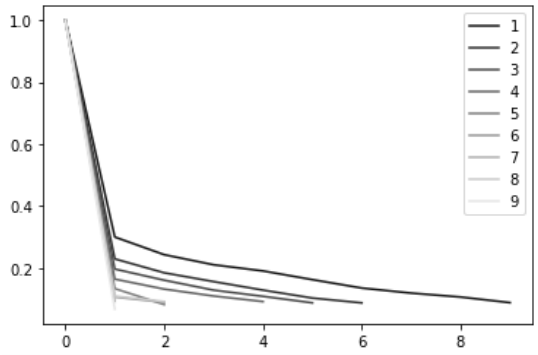


Joonis 14. ELO Katse 1: Random_ELO_choice

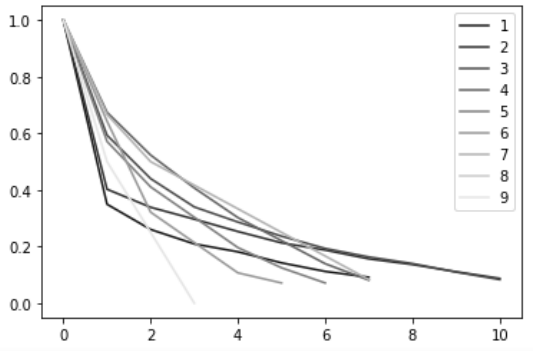


Joonis 15. ELO Katse 1: ELO_Probability_combination

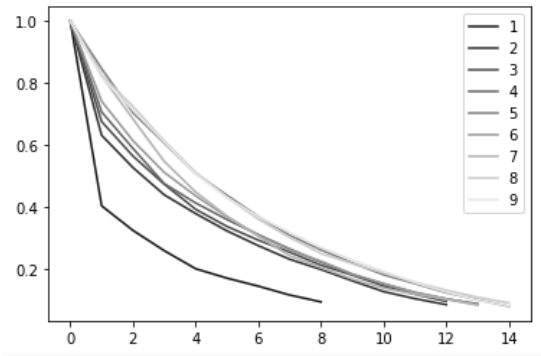
Järgnevalt võrdleme müra taseme muutuse mõju. Selleks suurendame Müra taset standard väärtuselt 200-lt 600-le.



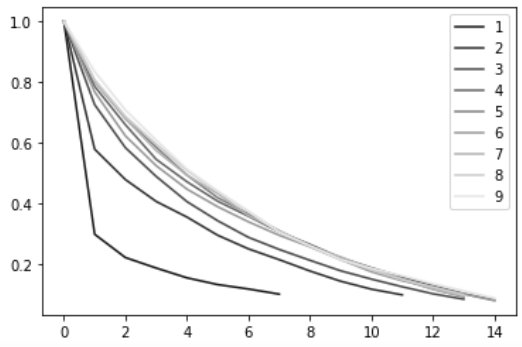
Joonis 16. ELO Katse 2: ELO_combination



Joonis 17. ELO Katse 2: All_pass_requirement



Joonis 18. ELO Katse 2: Random_ELO_choice

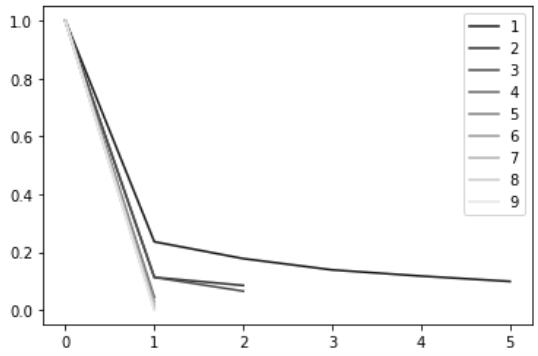


Joonis 19. ELO Katse 2: ELO_Probability_combination

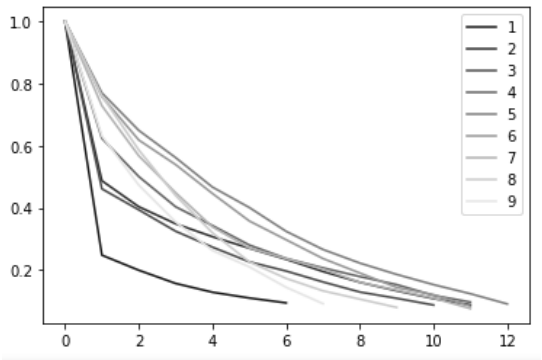
Müra suurenemisest on märgata, et üldiselt on suurenenud ka kontseptiahelate hargnemise paigutus omaduste arvu kohta. Seda on näha näiteks "Random_ELO_choice" ja "ELO_Probability_combination" algoritmide korral, kus madalamate omaduste arvu korral on hargnevus erinevateks ahelateks palju sügavamal kui varasema näitega võrrelda. Kontseptiahelate koguarv andmestikus ei ole muutunud. Alternatiivselt vaadates "All_pass_requirement" ja "ELO_combination", siis mõlema puhul on kontseptiahelate koguarv vähenenud. Samuti on näha sarnast mustrit, kus esimesed ahela lülid katavad suuremat osa. Antud tulemuste põhjal võib väita, et müra lisamine antud sisendandmete näite põhjal pigem pärsib andmestruktuuri keerukust ja liigutab hargnemist kontseptiahelate puhul andmetes sügavamale ehk erinevus hakkab esile paistma rohkem mitme välja võrdlusel kui esialgsete üksikute väljade kontseptidega võrrelda. Antud juhul saab järeldada, et kuna andmestruktuuris tekivad suuremad erisused ELO tasemete osas, siis väheneb juhuslikkuse faktor ehk tugevamad objektid võidavad palju tihedamini. Sellest tulenevalt tekib ridades või vertikaalides maatriksis konstantsemad järjestused samadest tõeväärtustest ehk nagu eelnevalt ka seletasime, siis kuna tõeväärtused on samad, on ka kontseptid palju sarnasemad. Struktuuri keskpunkt on defineeritav genereeritud objektide ELO erinevuste

põhjal ehk mida suurem on kallutatud keskpunktist objektide jaotuse võrdsusest, seda selgemalt on struktuur määratletud.

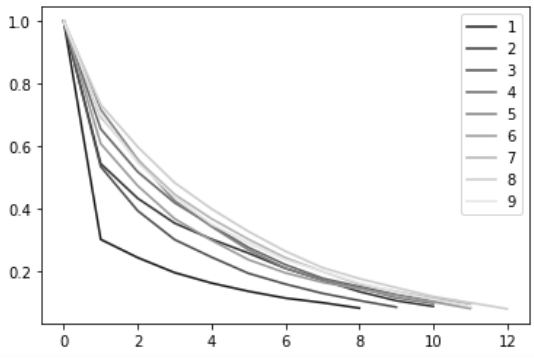
Järgmises testis jätame müra taseme 600-le ja suurendame testitava objekti ELO baasväärtust 1500-lt 1800-le.



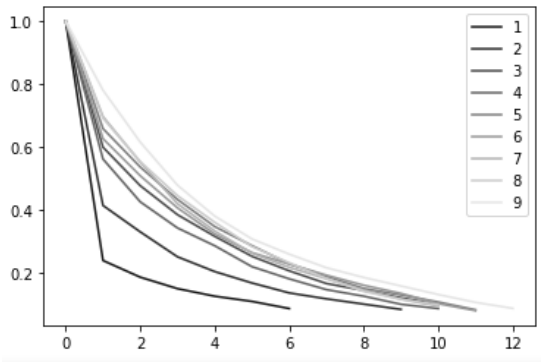
Joonis 20. ELO Katse 3: ELO_combination



Joonis 21. ELO Katse 3: All_pass_requirement



Joonis 22. ELO Katse 3: Random_ELO_choice

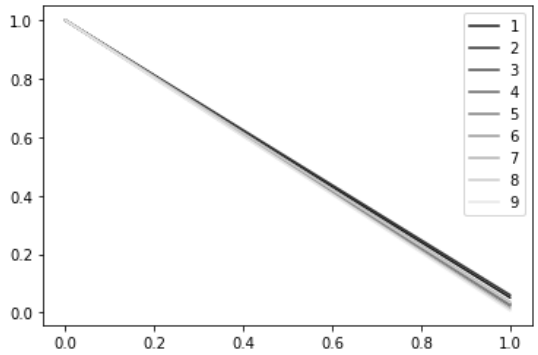


Joonis 23. ELO Katse 3:
ELO_Probability_combination

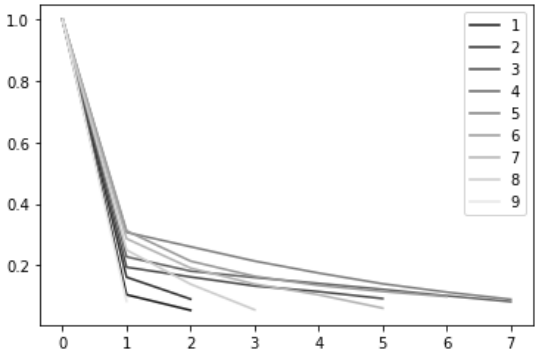
Võrreldes eelmisega on muudatused palju väiksemad, kuid vaadeldava tulemustiku pealt võib järeldada, et sarnaselt eelmisele on kontseptiahelate koguarv vähenevas trendis ja esimesed ahela osad katavad suuremat osa andmestikust võrreldes varasema tulemusega. Antud reegel on kehtiv kõigi otsustamise loogikate korral välja arvatud “All_pass_requirement” loogikaga. Vaadeldes teisi saame öelda, et kõigi puhul on kontseptiahelate arv keskmiselt kahe kuni kolme võrra vähenenud ja kõigi omaduste arvu korral on esinenud ahelate vähenemist. Antud juhul on selge, et kui suurendasime 300 võrra ühe osapoole ELO väärtust, siis see muutis keskmiselt antud objekti tüüpi objekte tugevamaks ja kuna objektid on tugevamad, siis on ka nende võrdlusest palju tõenäolisem, et vastav objekt võidab suurema tõenäosusega lähtudes ELO logistilisest funktsioonist ja sellest tuleneva muutuse kallakuga. Vaadates eraldi “All_pass_requirement” loogikat, saame lugeda välja erandi, mille kohaselt suureneb sisemine keeruksu andmetes, sest suureneb ka üldine juhuslikkuse efekt. Kuna testitava objekt peab testi läbimiseks läbima kõigi

oma omaduste põhjal positiivselt testi omaduste vastu, siis on antud loogika korral keskmine punkt (mündi heite korral 50% peal) kallutatud tugevalt testitava objekti suunas. See tähendab, et potentsiaalselt on 50% keskpunktist samaväärne võimalus võita ja läbi põruda sõltuv sisestatud omaduste arvust ja mida rohkem on selliseid arvuliselt, seda rohkem nihkub võrdsete võimaluste keskpunkt testi sooritaja objekti suunas ehk seda kõrgemat ELO reitingut vajab antud objekt, et konkureerida madalamate hinnangutega ELO testide vastu.

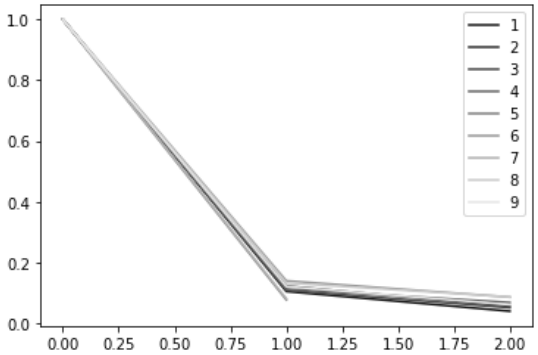
Järgmistes testides määrame müra taseme on 300-le ja muudame objektide ELO on 1500-le. Lisaks lülitame sisse dünaamilise ELO kalkuleerimise, mille käigus saab testitav objekt kasutada eelmistest testidest kogutud ELO väärtusi järgmistes seotud testides, kui läbitud test on vaadeldava testida protsentuaalselt seotud.



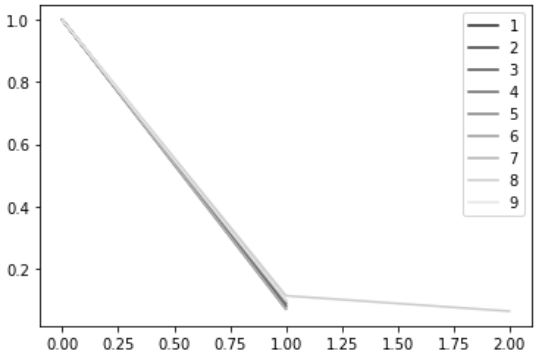
Joonis 24. ELO Katse 4: ELO_combination



Joonis 25. ELO Katse 4: All_pass_requirement



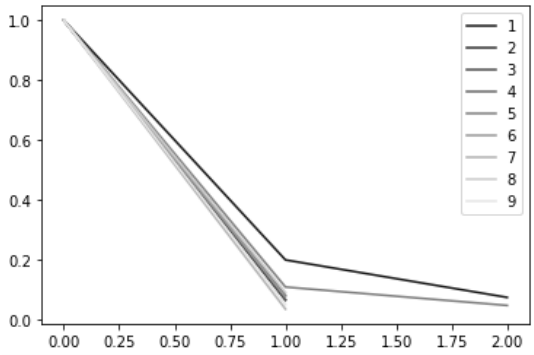
Joonis 26. ELO Katse 4: Random_ELO_choice



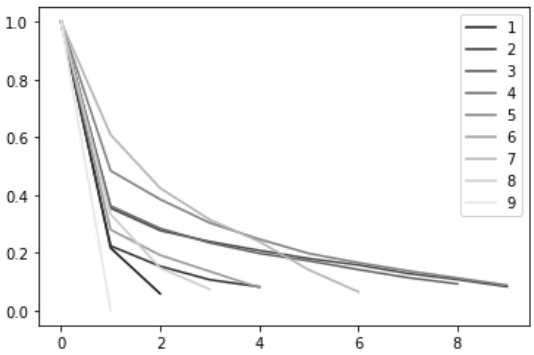
Joonis 27. ELO Katse 4: ELO_Probability_combination

Dünaamilise ELO sisselülitamisel on märgata suuri muutusi, mille tulemusel on tugevalt vähenenud kontseptiahelate koguarvud üle kõigi otsustamise loogikate. Erinevatest näidetest tulenevalt saame ka siin lähtuda juhuslikkuse faktorist, sest andmed tekivad oma olemuselt väga sarnased. Jällegist on märgata, et otsustamise loogika "All_pass_requirement" eristub tugevalt teistest, andes teiste andmete ühe-kahe kontseptiahela asemel seitse ahelat. Antud juhul saame andmetest lugeda, et kuna loogika suurendab testide positiivsel läbimisel testitava objekti

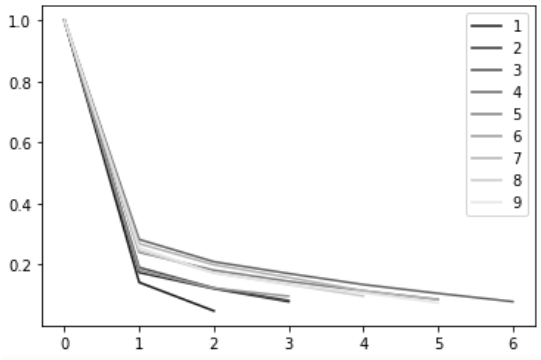
ELO kogemusliku kogutud ELO arvelt, siis muutuvad ajas objektid palju tugevamaks ja seeläbi on tulemused ka palju spetsiifilisemalt tõeseks määratletavad. Antud näide on kõrvutatav ülikoolis õppimise näitega, kus enamasti need, kes on edukad, on ka lõpuni edukad ja välja kukutakse suhteliselt varakult või jäädakse testide vastu nõrgaks. Kokkuvõtvalt on potentsiaalne, et juhuslikkus langeb sellisel juhul drastiliselt pärast paari testi läbimist. Kuna ajas läbivad kõik objektid mõningal määral mõnda testi ja koguvad sellest eelteadmisi teiste testide ees, siis ühel hetkel muutuvad kõik testitavad objektid ilma välja kukkumise faktorita tugevamaks kui testid. "All_pass_requirement" puhul säilib osaline juhuslikkus, sest selle sarnaste võimaluste kese on juba oma olemuselt nihutatud. Tugevalt määrab antud näites rolli omaduste arv, sest süveneva omaduste arvuga nihkub ka võrdsete võimaluste kese, mille tõttu on ka andmed palju sarnasema ahelaga määratletud kõrgemate omaduste arvu korral. Näiteks kui võimalus on 50% omaduse põhjal, siis kahe omaduse korral on kahe võimaluse kombinatsiooni korral tõese väärtuse saavutamiseks loogika mudeli korral ainult 25% ja seda edasi. 9 omaduse korral tähendab see seda, et andmestikus on ligi 0.002 võimalus, et saab tekkida tõene väärtus, seega peab ka võrdsete võimaluste kese olema väga oskuse objekti poole nihutatud, et saaks tekkida rohkem kontseptiahelaid ja sügavam struktuur. Selline muutus kehtib võimaluse astendamsie reegli põhjal, kus astendajaks on omaduste arv. Kuna dünaamilise ELO lisamisel on tulemused muutunud üpris spetsiifiliseks, siis vaatleme dünaamilise ELO muutuse mõju suurendades müra taseme 300-lt 600-le.



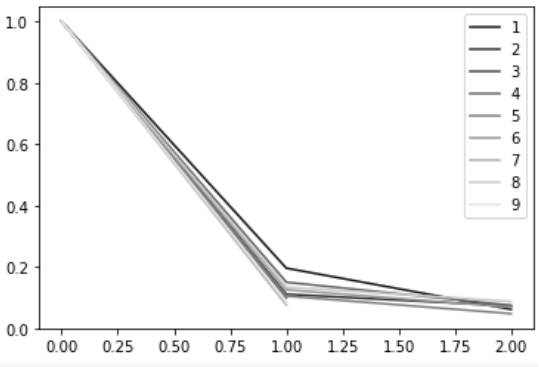
Joonis 28. ELO Katse 5: ELO_combination



Joonis 29. ELO Katse 5: All_pass_requirement



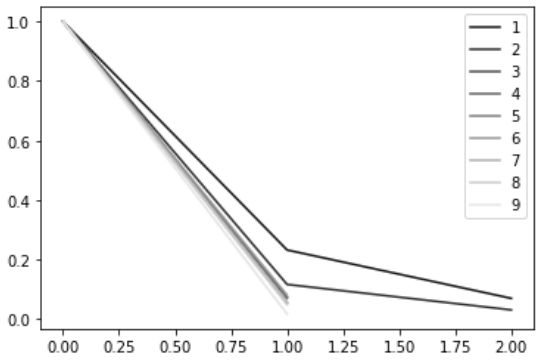
Joonis 30. ELO Katse 5: Random_ELO_choice



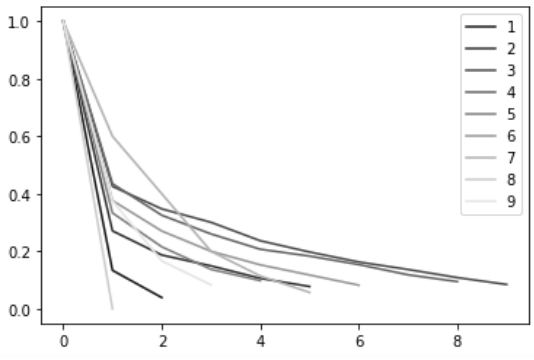
Joonis 31. ELO Katse 5: ELO_Probability_combination

Eristuv on, et kontseptiahelate arv on kasvanud ja eriti “Random_ELO_choice” loogika korral. Ka on väga palju liikunud “ELO_Probability_combination” korral, kus on palju iteratsioone saavutanud senise ühe asemel kaks kontseptiahelat. “All_pass_requirement” korral on märgata ,et muutused tulenevad sisse andmestruktuuris varasemate kontseptide korral ja samuti on suurenenud kahe võrra kontseptiahelate koguarv. Eelnevalt leidsime, et müra suurenedes langeb andmete juhuslikkus. Antud näites näeme, et see võib töötada ka vastupidiselt, kui andmed on väga ühte laadi määratletud. Selgelt tulevad erisused sisse palju varasemates andmetes, ehk müra avaldab küll antud loogika korral mõju, kuid seda varasemates testides ja hilisemates saavutab testitav objekt küllalt kogemusi, et püsida testist tugevam. Selge on, et dünaamilise ELO efekt muudab andmed üksluiseks juhul kui kogutud kogemusliku ELO kogutakse kiiremini ehk kui määrata madalama ELO maksimaalset taset / kordajat, siis on vaadeldavad muutused vähem drastilisemad ja seega ka tekib rohkem kontseptiahelaid andmestikes. Sarnaselt saame järeldada ka, et kui testid järjestada raskuse alusel, siis läbitakse varasemalt rohkem teste ja kogutakse rohkem kogemusliku ELO, mille tulemusena väheneb kontseptiahelate koguarv, sest testitavad objektid saavad tugevamaks palju varem kui suvalise järjestuse korral.

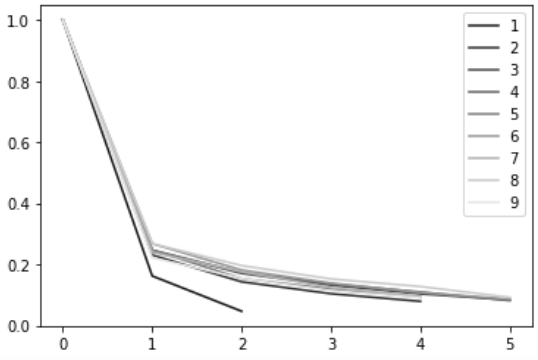
Järgmiseks vaatame dünaamilise elo mõju muutust, kui lisada kindluse taseme mudeli kalkulatsioonidesse sisse. Nagu eelnevalt mainitud, siis kindluse tase hindab kogemusliku ELO tegeliku taset ja määrab protsentuaalselt saadud ELO osas selle dünaamiliselt tuginedes ajaloolisel infol. Kindluse tase on määratud 50% ulatuses kontrollima kogemusliku ELO tegeliku taset.



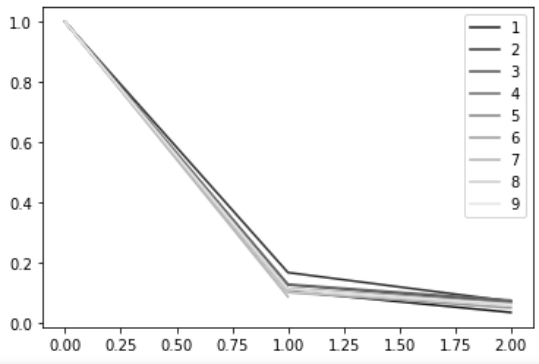
Joonis 32. ELO Katse 6: ELO_combination



Joonis 33. ELO Katse 6: All_pass_requirement



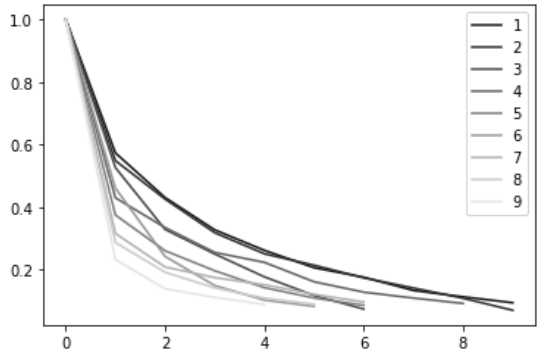
Joonis 34. ELO Katse 6: Random_ELO_choice



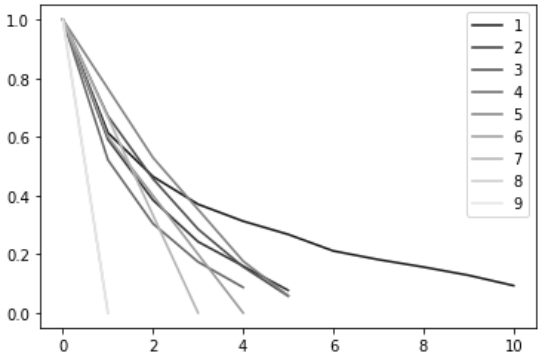
Joonis 35. ELO Katse 6:
ELO_Probability_combination

Selgelt on eristatav, et kõigi loogikate korral on kuidagi mõjutatud sisemine struktuur sarnasemate objektide suunas. See väljendub kas kontseptiahelate koguarvu vähenemises või kaetuse osas, sest kõigi näidete korral katavad muudatustes esimesed ahelad suuremat osakaalu tervest andmestikust kui varasemalt ilma kindluse tasemeta.

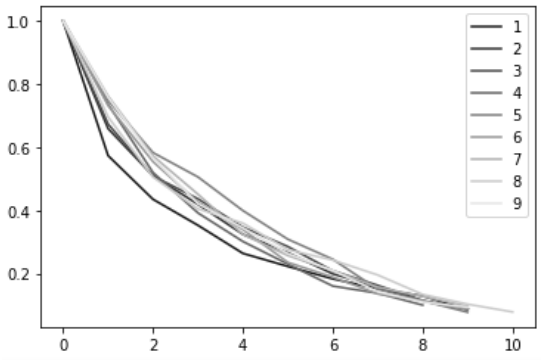
Edasi vaatame mudelis kajastatud põrumise tingimuse mõju. Selleks lülitame sisse põrumise tingimuse, mille kohaselt testitakse iga 6 testi järel testitava objekti jätkamise pädevust. Jätkata saab objekt ainult siis, kui tal on kuni 2 põrutud testi tsükli kohta. Objektide ELO-d on jäädavalt 1500 ja müra hoiame 200 peal. Eemaldatud on dünaamilise ELO.



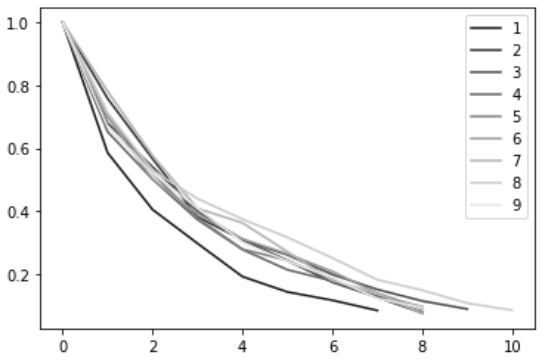
Joonis 36. ELO Katse 7: ELO_combination



Joonis 37. ELO Katse 7: All_pass_requirement



Joonis 38. ELO Katse 7: Random_ELO_choice



Joonis 39. ELO Katse 7: ELO_Probability_combination

Antud viimase näite korral esineb kaheti vaadeldavaid loogikaid ja nende tulemusi. Kuna välja kukkumise tingimus eemaldab iga 6 testi korral läbi põrunud objektid edasi positiivset tulemust saamast, siis on 20 testi kohta selliseid punkte 3 ja seetõttu peavad ka kontseptiahelate arvud olema suurenenud või need ahelad katma korraga väiksemat andme üldist kogust. Antud piirangu rakendamisest on näha, et see on kõigi otsustamise loogikate korral näha v.a "All_pass_requirement". Siin saab nähtust selgelt selgitada antud loogika erisusega, kuna võrreldes eelnevate testidega on toimunud rohkem keskmise suunas koondumist. Esialgse sarnaste andmete testist ilma piirava tingimusega olid tulemused jaotatud palju rohkem laiali, mille tõttu on andmetes juhuslikkus tugevalt suurem ja seda antud näite korral on seega mõjutatud, et juhuslikkus väheneks ja andmetes tekiks suuremat koondumist kolme kuni viie ahela juurde.

7.2 Alternatiivse mudeli testimine mündi heite meetodil

Selleks, et võrrelda ja valideerida mudeli töövõimet, siis realiseerime ja mitte-ELO põhise mudeli eelmainitud mündi viskamise meetodil ja vaatame konkureerivaid tulemusi. Mündi viskamise funktsiooni algseteks sisenditeks on testimisel:

- samples_amount = 100,
- tests_amount = 20,
- aspects_amount = i,
- coin_flip_1_chance = 0.5

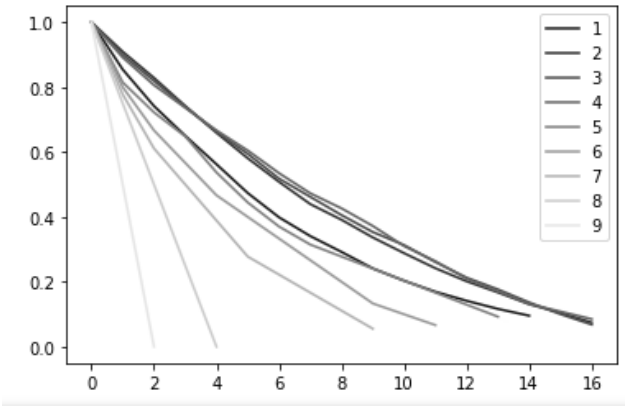
Mündi viskamise mudeli realiseeringud kirjeldab järgmine pseudokood. Antud mudelis on testitavad objektid määratletud kui üliõpilased ja testid testidena.

```
def generate_data_by_coin_flip(üliõpilased, testid,
omadused, positiivse viske võimalus, otsustamise
loogika):
     tulemused = []
     for üliõpilane in range(0, üliõpilased):
          üliõpilase tulemused = []
          for test in range(0, testid):
               viske tulemused = []
               for omadus in range(0, omadused):
                    lisa viske tulemustesse otsus
(positiivse viske võimalus >= loositud võidu tulemuse
piir)
               if otsustamise loogika ==
'all_aspect_pass':
                    if all(viske tulemused):
                         lisa üliõpilaste tulemustesse
positiivne tulemus
                    else:
                         lisa üliõpilaste tulemustesse
negatiivne tulemus
               elif otsustamise loogika ==
'aspect_pass_to_percent':
                    edukate visete arv = 0
                    for p in range(0, len(viske
tulemused)):
                         if viske tulemused[omadus] == 1:
                              edukate visete arv =
edukate visete arv + 1
                    positiivne võidu võimalus = edukate
visete arv/viske tulemuste pikkus
                         lisa viske tulemustesse otsus
(positiivne võidu võimalus >= loositud võidu tulemuse
piir)
               elif otsustamise loogika ==
'random_aspect_pass':
                    lisa viske tulemustesse otsus
(suvaline viske tulemus viske tulemustest)
               else:
                    print('Wrong draw mode input. Correct
and try again')
                    sys.exit()
```

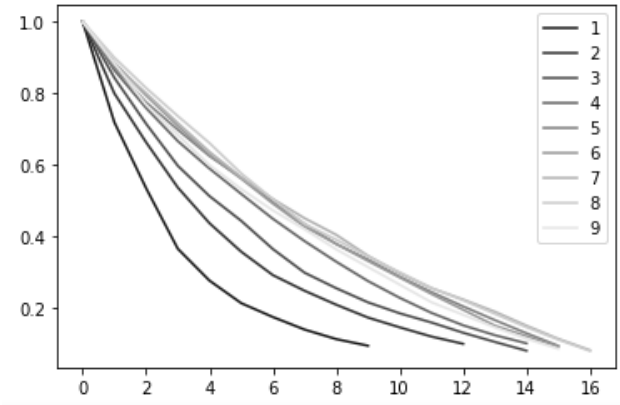
```
        lisa üliõpilase tulemused tulemustesse
    return numpy.array(tulemused)
```

Joonis 40. Mündi viskamise pseudokood

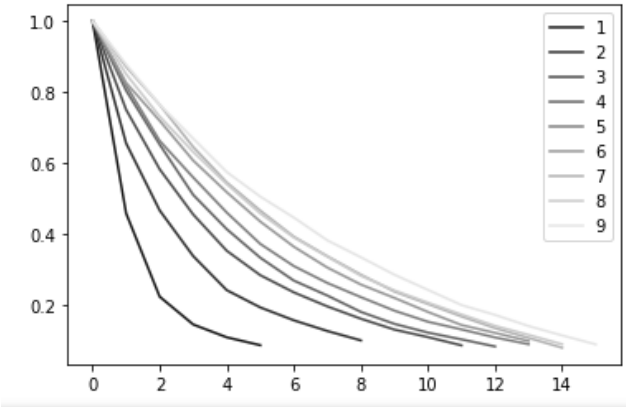
Järgnevalt vaatame otsustusmudelit (draw_mode), mille alusel peavad kõik omadused läbima testi. Sellega koos vaatame kuidas muutuvad kontseptiahelad muutes mündi heite positiivse viske tulemuse võimalusi.



Joonis 41. Mündi vise kõigi omadustega, coin_flip_1_chance = 0.5

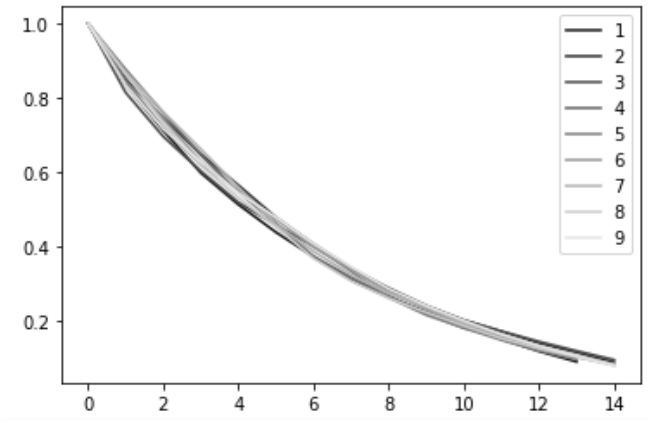


Joonis 42. Mündi vise kõigi omadustega, coin_flip_1_chance = 0.8

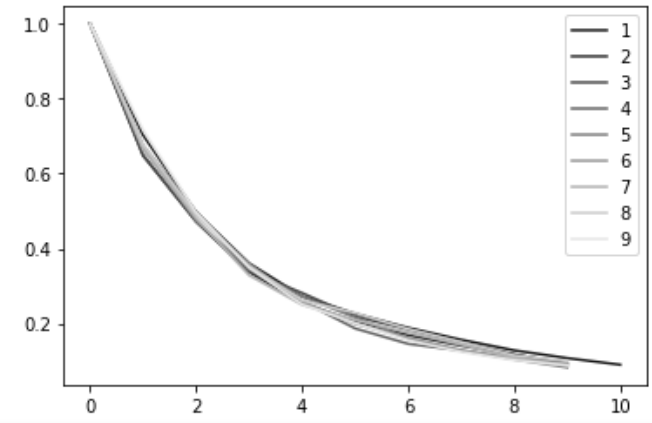


Joonis 43. Mündi vise kõigi omadustega, coin_flip_1_chance = 0.9

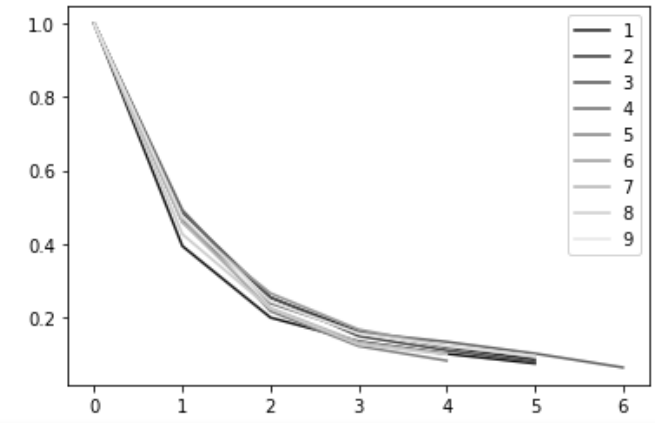
Järgnevalt vaatame otsustusmudelit (draw_mode), mille alusel läbinud omadused moodustavad läbimise protsentuaalse osakaalu. Kui näiteks viiest testis läbitakse kolm, siis on tõenäosuseks 60%.



Joonis 44. Mündi vise protsentuaalse osakaaluga, coin_flip_1_chance = 0.5

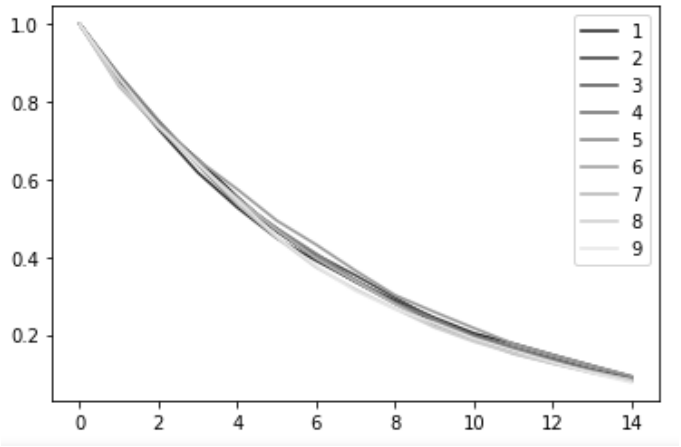


Joonis 45. Mündi vise protsentuaalse osakaaluga, coin_flip_1_chance = 0.8

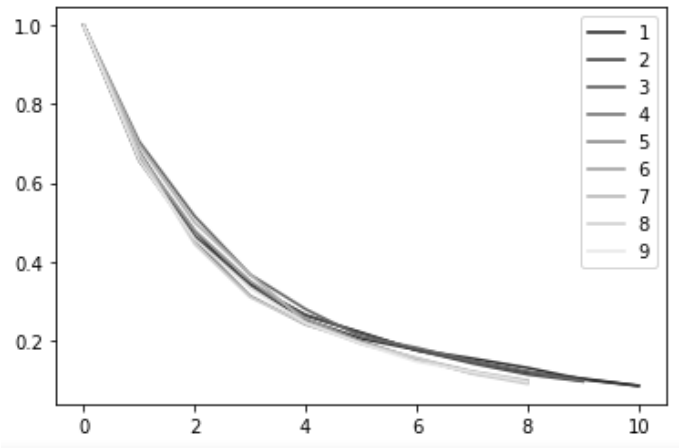


Joonis 46. Mündi vise protsentuaalse osakaaluga, coin_flip_1_chance = 0.9

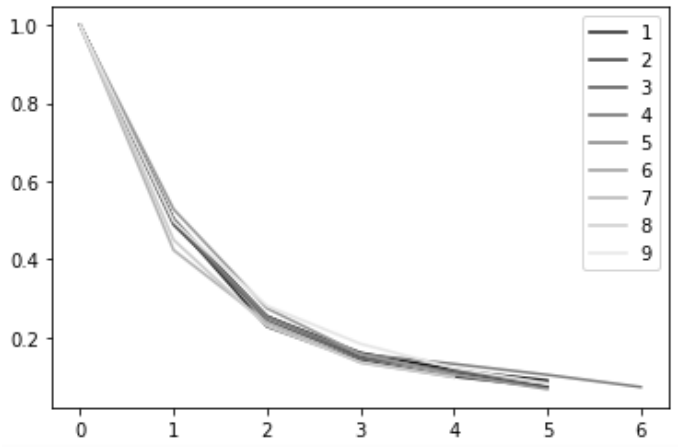
Järgnevalt vaatame otsustusmudelit (draw_mode), mille alusel valitakse juhusliku omaduse tulemus testi hindamisel.



Joonis 47. Mündi vise juhusliku omaduse otsusega, coin_flip_1_chance = 0.5



Joonis 48. Mündi vise juhusliku omaduse otsusega, coin_flip_1_chance = 0.8



Joonis 49. Mündi vise juhusliku omaduse otsusega, coin_flip_1_chance = 0.9

7.3 Tulemuste hindamine ja võrdlemine

Kuna peamiseks genereerimise allikaks on juhuslikkus ja selle kontrollimine, siis vaatleme eelnevate testi tulemuste põhjal kuidas on sisse ehitatud juhuslikkus kontrollitav. Selleks vaatame realiseeritud mudeli testi tulemusi ja võrdleme neid mündi viske mudeli vastu ning hindame lõplikult mudeli kasutatavust, lihtsust ja võimekust. Nagu eelnevalt on mainitud, siis peamiseks hindamise mehhanismiks on kontseptiahelate analüüsimine ja sellest tulenevalt määratleme, et kui tõhus ja täpne on loodud mudel reaaleluliseks kasutatavuseks.

Vaadates peale eelnevalt määratud tulemusi, siis on mudel suutnud ELO põhjal, nii dünaamilise kui tavalise korral suurendada andmete juhuslikkuse kontrollimist ja võimaldab kontrollida ärikasutajal tugevamalt millist struktuuri andmestik omab. Võrreldes tavalise mündi viskamise loogikaga, siis ei oma ükski alternatiivne loogika mündi viskamisel otsest tulemust, sest omaduste arvu kasvades on siiski sisemine struktuur üldiselt jääv. Erinevus tuleb sisse ainult mündi viskamise kõigi läbimise loogikas. Seal on selgelt näha samuti, et kõigi omaduste läbimise tingimuse tõttu on osaliselt pööratud loogika 50% positiivse heite korral, sest suurema omaduste arvu korral muutub positiivse tulemuse saamine palju väiksemaks iga kord poole võrra ehk see on väljendatav valemiga: positiivse viske võimalus astmes objekti omaduste koguarv. Seetõttu on aspekti kasvades ka andmestruktuur samalaadsem. Nihutades positiivse viske tulemust rohkem positiivsemas suunas on ka näha kuidas sarnaselt nihkuvad ka kontseptiahelate arvud.

Võrreldes eelneva põhjal saame hinnata, et loodud mudel on palju efektiivsem ja laiemalt kasutatav kui traditsiooniline mündi heitmise metoodika. Samuti oleme tuvastanud kuidas on antud mudeli puhul erinevad sisendid seotud andmestruktuuri genereerimisega. Lõpuks oleme ka tõestanud antud lõputöö raames, et ELO reitingu mudelit ja sellel põhinevat jooksvat võrdlust objektide vahel on võimalik kasutada, et genereerida Oskus-Test mudelil põhinevat andmestiku.

Loodud mudeli lihtsus on üldiselt keskmine. Mudeli puhul on palju sisendeid võimalik määrata, kuid kõik mittekohustuslikud ja lisa sisendid muudavad ainult täpsust ja loogikat. Peamisteks sisenditeks saame lugeda baas ELOsid, müra ELOsid, dünaamikat, selle kindluse taset, dünaamilise ELO muutuse kordajat ja otsustamise loogika, mis teevad kokku kaheksa sisendit. Nendest on kohustuslik määrata ainult baas ELOd ja ELO mürad. Samuti on selge kuidas mõjutavad erinevad sisendid mudeli genereerimist. Selle kõige põhjal saame olla kindlad, et kasutaja saab mudeli kasutamisest aru ja sisendite andmine ei ole keeruline, kuid samas keskmise keerukusega, kui kasutaja soovib loogika tööpõhimõtet täpsustada.

Võimekuse osas on näha, et vastavalt sisenditele ja nende testimise tulemusena on võimalik kontrollida kontseptiahelate koguarvu ja nende koondumist. Generaator ei võimalda küll luua kahte

erinevat struktuuri omavat andmestiku genereerimist, kuid selleks saab kasutaja jooksutada sünteesimise protsessi mitu korda ja andmehulki liita. Viimase tingimuse põhjal saab öelda, et mudeli loomine on edukas ja mudel täidab selle esmase eesmärki üpris edukalt.

7.4 Mudeli kasutamise kirjeldus

Mudelis on peamisteks kontrollitavateks väärtusteks:

- generate_samples_amount (Testitava objektide kogus. Objektide arvu suurenedes, suureneb potentsiaalselt andmestruktuuri keerukus.)
- generate_tests_amount (Testide objektide kogus. Objektide arvu suurenedes, suureneb potentsiaalselt andmestruktuuri keerukus.)
- aspects_amount (Omaduste kogus objekti kohta. Omaduste arvu suurenedes, suureneb potentsiaalselt andmestruktuuri keerukus.)
- dynamic_elo (Lülitab sisse kogemusliku ELO kogumise testitavale objektile, mille käigus testi positiivsel läbimisel objekti ELO tõuseb järgmsit testi teostades, kui testidel on omavahelised teadmsite seosed määratud. Kogemusliku ELO tõustes muutuvad testitava objekti tugevused suuremaks ja vastavalt sellele kas võrdsustub esialgu objektide tase ja tagatakse võrdsemad võimalused või suureneb objektide vaheline tugevuste kaugus ja seeläbi väheneb juhuslikkus.)
- interference (Muudab dünaamilise ELO kasutamise mahtu dünaamiliselt, võttes aluseks ajaloolise andmestiku. Edukamad objektid saavutavad sama taseme mis ennem, kuid vähem edukamad objektid kasvavad ELO tasemelt aeglasemalt. Sunnib peale objektide tüüpide eristumist tulenevalt läbitud testide koguarvust ja suhtest positiivsete tulemustega. Kokkuvõtvalt vähendab dünaamilisest ELOst tulenevat ELO kasvu vähem edukamatele objektidele.)
- sample_base_elo (Genereeritud objektide lähte ELO, mis on vaikeväärtusena. Määrab objektide keskmise tugevuse)
- sample_noise_elo (Genereeritud objektide hälve positiivses ja negatiivses suunas. Määrab ELO maksimaalse muutuse vahemiku vaikseväärtusest. Liigutab andmete struktuuri genereeritud objektide tugevuse osakaalu suunas.)
- test_base_elo (Genereeritud objektide lähte ELO, mis on vaikeväärtusena. Määrab objektide keskmise tugevuse)

- test_noise_elo müra (genereeritud objektide hälve positiivses ja negatiivses suunas. Määrab ELO maksimaalse muutuse vahemiku vaikseväärtusest. Liigutab andmete struktuuri genereeritud objektide tugevuse osakaalu suunas.)
- experience_maximum_gain_ELO (Määrab kui kiiresti ja kui palju kogemusliku ELO kogutakse, kui dünaamiline ELO on sisse lülitatud.)
- elo_mode (Otsustamise loogika valik. Vastavalt kasutusjuhule saab rakendada järgnevatest loogikatest sobivaimat.)
    - ELO_combination (Arvutamisel kasutatakse loodud objektide omaduste keskmisi ELOsid tulenevalt võrreldava testi omaduste kaalu vastu. Kombineeritakse kokku üldine ELO tase. Andmestruktuuri juhuslikkus on suurenev objektide müra vähenemisega.)
    - Random_ELO_choice (Valitakse üks juhuslik omadus ja võrreldakse selle ELO. Objekti võime võita on juhuslikum koos müra tasemega.)
    - All_pass_requirement (Nõuab, et testi läbivad kõik objekti omadused testi omaduste vastu. Andmestruktuuri keerukus on kontrollitav tugevalt omaduste arvu järgi, sest võrdsete võimaluste keskpunkt on nihkuv testitava objekti suunas valemiga: positiivse viske võimalus astmes objekti omaduste koguarv)
    - ELO_Probability_combination (Läbimise protsent on määratav testimisel objekti omaduste võrdlus tulemuste põhjal. Lõplik tulemus on osaliselt kontrollitud omaduste koguarvu ja objekti omaduste tugevuste põhjal)

Alternatiivselt saab kasutaja määrata:

- test_aspect_ratios (Sisend fail, defineerib failis suure mahuliste aspektide kaalud objektide genereerimisel)
- sort_tests_asc (Tõese väärtuse korral sorteeritakse testid raskuse alusel alustades kergematest. Kasutatav dünaamilise ELO korral simuleerimaks reaalset õpet.)
- fail_condition_samples (Simuleerib semestri kohta läbi kukkumist ja selle tingimuse kontrolli. Kontrollitakse iga 6 testi kohta kas läbitud on 75% eelnevatest testidest, et testitav objekt saaks jätkata testide sooritamist. Loob andmetesse erisuste tekitamiseks iga kuuenda objekti korral erisuse potentsiaalselt kontrollides kontseptiahelate arvu andmestiku keskme suunas.)
- samples_file (Sisend fail, laseb kasutajal defineerida eelneva andmeanalüüsi põhjal sisendi või soovitud objektide struktuuriga. Kasutatav ainult faili toega funktsioonis.)

- tests_file (Sisend fail, laseb kasutajal defineerida eelneva andmeanalüüsi põhjal sisendi või soovitud objektide struktuuriga. Kasutatav ainult faili toega funktsioonis.)
- sample_distribution_type (Kasutaja saab määrata objektide ELO jaotust funktsioonide sisseehitatud alusel.)
- test_distribution_type (Kasutaja saab määrata objektide ELO jaotust funktsioonide sisseehitatud alusel.)
- scramble_aspects (Genereeritud omaduste struktuuri on võimalik segada juhuslikkuse alusel.)
- scramble_objects (Genereeritud objektide struktuuri on võimalik segada juhuslikkuse alusel.)
- sample_tilt_pos_elo (Kasutaja saab määrata erilistele ELO jaotuse funktsioonidele nagu näiteks gamma ja poisson, keskse koha ELO jaotusel, kust alustatakse tõenäosuste genereerimist.)
- test_tilt_pos_elo (Kasutaja saab määrata erilistele ELO jaotuse funktsioonidele nagu näiteks gamma ja poisson, keskse koha ELO jaotusel, kust alustatakse tõenäosuste genereerimist.)
- reverse_aspects (Muudab genereeritud objekti omaduste järjestust vastupidiseks.)
- test_aspect_ratios_random (Muudab testide omaduste kaalude tähtsuse muutuvaks objektide genereerimisel.)
- test_aspect_ratios_active (Aktiveerib testide omaduste tähtsuste kaalud.)

8 Kokkuvõte ja tulevane töö

Vaadeldav uurimistöö käsitles Oskus-Test mudelil põhinevate andmestike genereerimist, toetudes selleks FCA ja sünteetiliste andmete genereerimise põhimõtetele. Eesmärgiks oli uurida ja uurimuse põhjal luua mudel, millega on võimalik vastavalt kasutaja sisendile kontrollida ja genereerida Oskus-Test mudeli andmeid.

Uurimistöö raames on lähemalt uuritud formaalsete kontseptiahelate tausta ja sünteetiliste andmete olemust ning nende genereerimist. Samuti on vaadeldud alternatiivseid meetodeid kontseptide genereerimiseks. Seejärel on analüüsitud Oskus-Test andmemudelil põhinevat andmestiku ja vaadeldud sellest tulenevaid nõudeid, mida on mudelis seejärel realiseeritud. Realiseeritud mudeli tööpõhimõtet testiti ja tuvastati millist mõju esialgu loodud hüpoteetilise mudelise sisendid tegelikult andmestruktuuri genereerimisel omavad. Seejärel hinnati loodud mudeli töövõimet ja võrreldi alternatiivse mündi viskamise mudeli tulemuslikkusega. Lõpuks anti definitsioonid mudeli sisendite ja avastuse põhjal kasutajale lihtsalt hoomatavaks juhendiks.

Uurimistöö peamiseks tulemuseks on loodud ELO põhine Oskus-Test andmete sünteesimise mudel, mis võimaldab genereerida erinevate ELO suuruste, müra suuruste ja nende jaotuse, omaduste arvu, jooksvate reeglite ja dünaamiliste tingimuste põhjal andmestiku ehk määrata milline on sünteetilise andmestiku sisemine struktuur. Täies mahus saab kasutaja määrata andmete sisest struktuuri 11 peamise ja 14 täiendava sisendandme tüüpide alusel. Lisatulemuseks on tõestus, et ELO hindamissüsteemi on võimalik kasutada sünteetiliste andmete genereerimisel, kui selliste andmete puhul kehtib objektide omavahelise võrdluse moment, nende initsiatiiv võita ja mõningal juhul jooksva seisu muutus. Seetõttu on olulised ka antud töös kajastatud analüüs ja selle tulemused. Töö tulemusena on loodud unikaalne ELO põhine mudel, missuguseid varasemalt ei ole realiseeritud sünteetiliste andmete genereerimiseks.

Mudeli tulemused on kergesti kontrollitavad ka lihtsamate struktuuride põhjal ja sisenditeks on vaja väikese arvuline kogus sisendandmeid, et genereerida juba selgelt eristuvaid struktuure. Mudel võimaldab genereerida nii kõrge kui ka madala sarnasusega andmeobjekte. Analüüsitud kontseptiahelate meetodi kohaselt võimaldab mudel kontrollida üpris täpselt mudelis peituvate sünteetiliste andmete juhuslikkust kui ka andmete struktuuri hargnevusi. Enamik juhtudel on

mudeli tulemused ootuspärased, mille kohaselt muutub omaduste alusel andmestruktuur keerulisemaks. Osadel juhtudel ja sisendite kombinatsioonist tulenevalt funktsioneerib mudel vastupidiselt sellisel juhul, kui juhuslikkus on omaduste põhjal muutuv vastupidiseks ehk omaduste arv hakkab tulenevalt valitud loogikast töötama vastu juhustlikkuse loomele. Näiteks kehtib see tugevalt kõigi omaduste läbimise nõude otsustamise loogika korral, kus omaduste arv on keskmist tõenäosust läbimise hindamisel astendav.

Järgmise sammuna saab loodud ELO põhjal realiseeritud mudeli abil genereerida valdkonna spetsiifilist andmestiku ja katsetada loodud mudelit valdkonna andmete vastu, kasutades selleks valdkonna reaalsetest andmetest tuletatud jaotustega, mida antud mudel juba võimaldab parameetritena ette anda. Samuti on võimalik kasutada mudelit andmete genereerimiseks, et võrrelda erinevate algoritmide nagu näiteks klasterdamise, kontseptiahelate, dendrogrammide jms FCA-põhiste meetodite kirjeldusvõimet erisuguste andmestruktuuride ja keerukuste alusel. Eelnevalt ei ole tehtud mudelit, mis võimaldaks erinevaid selliseid algoritme omavahel võrrelda. Kuna selline mudel on unikaalne, siis oleks sellise võrdluse tulemused huvipakkuva väärtusega. Antud võrdlus oleks unikaalne, sest ka loodud mudel on senini unikaalne ja selline algoritmide võrdlus võiks olla teemaks järgmisele magistri tööle.

9 Kasutatud kirjandus

[1] Felde, M. & Hanika, T. Formal Context Generation using Dirichlet Distributions. September 2018. [WWW] https://arxiv.org/pdf/1809.11160.pdf (03.05.2020)

[2] Belohlavek, R. INTRODUCTION TO FORMAL CONCEPT ANALYSIS. 2008. [WWW] http://belohlavek.inf.upol.cz/vyuka/IntroFCA.pdf (03.05.2020)

[3] Farley, J. The N.S.A.'s Math Problem. 2006. [WWW] https://www.nytimes.com/2006/05/16/opinion/16farley.html (05.05.2020)

[4] Andrews, S. & Orphanides, C. Analysis of Large Data Sets using Formal Concept Lattices. [WWW] http://ceur-ws.org/Vol-672/paper10.pdf (03.05.2020)

[5] Chaikalis, T. Example-of-Formal-Concept-Analysis-Formal-Context-a-Concepts-of-the-Formal-Context. [WWW] https://www.researchgate.net/profile/Theodore_Chaikalis/publication/257665299/figure/fig1/AS:339195223658499@1457881851857/Example-of-Formal-Concept-Analysis-Formal-Context-a-Concepts-of-the-Formal-Context.png (03.05.2020)

[6] Anonymisation with Synthetic Data Tutorial. [WWW] https://github.com/theodi/synthetic-data-tutorial (04.05.2020)

[7] medcalc.org. [WWW] https://www.medcalc.org/manual/_help/functions/pdfbinomial.png (05.05.2020)

[8] Liiv, I. Mustrite avastamine kasutades järjestamist ning maatriksi ümberkorrastamist: unifitseeritud vaade, edasiarendused ning rakendus ladude juhtimises. August 2008. [WWW] https://digikogu.taltech.ee/et/Item/e1dd1b54-c2d4-428c-84ca-03c41803939d (08.05.2020)

[9] Barse, E., Kvarnstrom, H. & Jonsson, E. Synthesizing test data for fraud detection systems. Detsember 2003. [WWW] https://ieeexplore.ieee.org/document/1254343/ (05.05.2020)

[10] Torim, A., Mets, M. & Raun, K. Covering Concept Lattices with Concept Chains. Juuni 2019. [WWW] https://link.springer.com/chapter/10.1007/978-3-030-23182-8_14 (08.05.2020)

[11] Reddick, R. Using a Glicko-based Algorithm to Measure In-Course Learning. 2019. [WWW] https://files.eric.ed.gov/fulltext/ED599237.pdf (05.05.2020)

[12] Abdi, S., Khosravi, H. & Sadiq, S. A Multivariate Elo-based Learner Model for Adaptive Educational Systems. 2019. [WWW] https://drive.google.com/file/d/1Ha-rG38xbjl5agBWsArdSVlGbU_nmP-n/view (06.05.2020)

[13] Grabner, R., Stern, E. & Neubauer, A. Individual differences in chess expertise: A psychometric investigation. Juuli 2006. [WWW] https://ethz.ch/content/dam/ethz/special-

interest/gess/ifv/professur-lehr-und-lernforschung/publikationen-stern/ab-2005/Artikel/Grabner_Stern_Neubauer_Acta_2006.pdf (06.05.2020)
[14] Klinkenberg, S., Straatemeier, M. & van der Maas, H. Computer adaptive practice of Maths ability using a new item response model for on the fly ability and difficulty estimation. November 2010. [WWW] https://hvandermaas.socsci.uva.nl/Homepage_Han_van_der_Maas/Publications_files/papers/klinkenberg.pdf (07.05.2020)
[15] Vaziri, B., Dabadghao, S., Yih, Y., & Morin, T. L. Properties of sports ranking methods. Journal of the Operational Research Society, 69(5), 776-787. 2018. [WWW] https://pure.tue.nl/ws/portalfiles/portal/89687107/properties_of_sports_ranking_methods.pdf (06.05.2020)
[16] Dorsey, J. ELO REGRESSION EXTENDING THE ELO RATING SYSTEM. Mai 2019. [WWW] https://etd.ohiolink.edu/!etd.send_file?accession=akron1555587309160256&disposition=inline (07.05.2020)
[17] Aldous, D. Elo Ratings and the Sports Model: a Neglected Topic in Applied Probability? Märts 2017. [WWW] https://www.stat.berkeley.edu/~aldous/Papers/me-Elo-SS.pdf (10.05.2020)
[18] Calestini, L. Elo Against the Spread. [WWW] https://medium.com/sports-analytics/elo-against-the-spread-b70a6cc2210a (08.05.2020)
[19] scikit-learn. [WWW] https://scikit-learn.org/stable/ (08.05.2020)
[20] BCCampus. Estimating the Binomial with the Normal Distribution. [WWW] https://opentextbc.ca/introbusinessstatopenstax/chapter/estimating-the-binomial-with-the-normal-distribution/ (10.05.2020)
[21] Kuznetsov, S. & Napoli, A. Formal Concept Analysis: Themes and Variations for Knowledge Processing. Juuli 2015. [WWW] http://ijcai15.org/downloads/tutorials/T23-FCA.pdf (10.05.2020)
[22] Amgoud, L. & Prade, H. A formal concept view of abstract argumentation. [WWW] https://www.irit.fr/~Leila.Amgoud/AmgPraECSQARU.pdf (07.05.2020)
[23] Ignatov, D. Introduction to Formal Concept Analysis and Its Applications in Information Retrieval and Related Fields. Detsember 2015. [WWW] https://www.researchgate.net/publication/287483929_Introduction_to_Formal_Concept_Analysis_and_Its_Applications_in_Information_Retrieval_and_Related_Fields (11.05.2020)
[24] Ganter, B. Formal Concept Analysis: Methods and Applications in Computer Science. 2002. [WWW] http://www.math.tu-dresden.de/~ganter/cl03/stumme/chapter1_2.pdf (10.05.2020)

[25] Ritchie, F. & Smith, J. Confidentiality and linked data. Detsember 2018. [WWW] https://arxiv.org/ftp/arxiv/papers/1907/1907.06465.pdf (11.05.2020)