Matematiikan ja tietotekniikan tehtävien automaattinen tasoarviointi adaptiivisella testauksella

TURUN YLIOPISTO
Informaatioteknologian laitos
Tietotekniikka
Diplomityö
Mikko Kondratjeff
Kesäkuu 2016

TURUN YLIOPISTO
Informaatioteknologian laitos

KONDRATJEFF, M.: Matematiikan ja tietotekniikan tehtävien automaattinen tasoarviointi adaptiivisella testauksella

Diplomityö, 59 s.
Tietotekniikka
Kesäkuu 2016

Tämän diplomityön tavoitteena on ollut kehittää adaptiivista testausta hyödyntävä menetelmä, jonka tuottaman datan avulla voidaan havaita ja ennakoida oppimisvaikeuksia perusopetuksen peruskoulun sekä lukion matematiikan ja tietotekniikan opetuksessa. Tietojen analysointi toteutetaan ohjelmallisesti tilastollisia menetelmiä hyödyntäen anonymisoidusta aineistosta. Adaptiivisuutta hyödynnetään tällöin tehtävien asettelun apuvälineenä sekä tehtäviä suoritettavaksi tarjoavana menetelmänä muiden käytössä olevien tehtäväasettelujen ohella. Opiskelijan osaamista selvitetään joko aikaisempien tehtävien perusteella tai opiskelijan ominaisuuksien perusteella arvioidun tyypillisen profiilin mukaisesti. Tehtävien kategorisointi tasoarviointitehtävien ja yleisten tehtävien välillä tehdään osittain automaattisesti, mutta tehtävien lajittelu käytettävissä olevien tehtävien joukkoon vaatii käsityötä.

Selvitettävän aineiston tutkimisen avuksi tarvitaan kategorisoivaa aineistonvalintaa, joka voidaan toteuttaa osin ohjelmallisesti mutta johon tarvitaan myös käyttäjän arviointia. Lähtökohtana käytetään primitiivisiä työvälineitä, joiden jatkokehittämisen tarve tulevaisuudessa on välttämätöntä aineistojen muuttuessa ja tutkittavan joukon laajentuessa.

Avainsanat: adaptiivinen testaus, automaattinen arviointi, tietokoneavusteinen oppiminen, oppimisanalytiikka, oppimisvaikeudet

UNIVERSITY OF TURKU
Department of Information Technology

KONDRATJEFF, M.: Automatic assessment of assignments in Mathematics and Computing Science by adaptive testing

Masters Thesis (tech.), 59 p.
Software Engineering
June 2016

Purpose of this thesis has been to create a method to find figures in assessment of mathematical and computational assignments primary and secondary school level, with goal to screen possible problems in learning mentioned subjects. Adaptive testing has been the method which is used to offer assignments depending to the earlier results or typical student properties. Assignments are categorized between evaluation and generic subjects by using partly automatic method.

Categorizing subject choosing is needed to help to find the suitable data for assignment offering, which needs some user evaluation along with automatic program. In this task relatively primitive tools can be used to begin with, but which need further clarification to process the continuously growing and developing data set.

Keywords: adaptive testing, automatic assessment, computer aided learning, learning analytics, learning difficulties

KIITOKSET

Kiitokset – kaikille joille se kuuluu, eli diplomityöni ohjaajille Mikko-Jussi Laaksolle ja Rolf Lindénille, opetusteknologian tiimille tuesta ja opastuksesta työn aikana, opinto-ohjaajille, opettajille sekä kollegoille, ja perheellenikin kannustavasta taustatuesta.

Turussa 1.6.2016

Mikko Kondratjeff

SISÄLLYSLUETTELO

1 JOHDANTO

Tutkielman tarkoituksena on ollut selvittää, voidaanko tilastollisten menetelmien ohjelmallisilla toteutuksilla löytää ViLLE -oppimisympäristöön tallennetussa datassa tyypillisiä oppimisvaikeuksia indikoivia, solmukohdiksi kutsuttuja piirteitä. Tietoaineistoa on kerättävissä perusopetuksen alakoulun matematiikan opintopolkua seuraavilla tasokokeilla että yläkoulun ja lukion tietotekniikan ja matematiikan tasokokeilla. Oppimisvaikeuksien analysointi ja havaitseminen kvalifioidusta datasta toimii lähtökohtana hyödynnettäessä oppimisparametritietoa tehtäväasettelussa. Tehtävien asetteluun on runsaasti tutkittuja toimintatapoja ja viitekehyksiä. Tämän tutkielman lähtökohtana on ollut selvittää opiskelijan tehtävien suoritustapoja, joiden avulla järjestelmä voi sopeuttaa tehtävätarjonnan opiskelijan kykyjä ja osaamista vastaaviksi. Tällaista tehtävien suorittamiseen vaikuttavien piirteiden avulla tehtävätarjontaa muokkaavaa asettelua kutsutaan adaptiiviseksi, toisin sanoen käytettävissä olevien mitattujen käyttäytymistä kuvaavien parametrien mukaan sopeutuvaksi järjestelmäksi.

Eräs oppimissuorituksiin liittyvien tietojen keräämisessä sovellettavista toteutustavoista on adaptiivinen testausmenetelmä, joka tallentaa opiskelijan suorituksesta parametritietoa, mahdollistaen tehtäväasettelun yksilöllisen suoritustasoon sopeutetun muokkaamisen. Toisin sanoen voidaan tarjota suoritettaviksi niitä tasokokeen tai tutoriaalin tehtäviä joilla on merkitystä oppimisen tason määrittelyssä. Menetelmän tueksi validoidaan oppimisvaikeuksien konkreettinen esiintyminen saman opiskelijan aineistoista anonymisoidusti siten, ettei tietoja voi suoraan yhdistää tiettyyn henkilöön. Toisaalta tietojen yhdistäminen henkilöön suoritusaineistojen tarkastelun ja arvioinnin osalta on tärkeää. Tällöin käytettävissä on tietoa, joka auttaa selvittämään yksittäisen opiskelijan suorituksissa esiintyviä, virheellisiin vastauksiin johtavia tulkintoja. Opiskelijan pakottaminen oikein ratkaistuihin suorituksiin ei kuitenkaan palvele oppimisen tavoitteita, vaan on käytännön suoritustapahtumaa ohjaava tekijä. Koska

opiskelijaa voidaan yksilöllisesti muotoiluilla tehtäväkokoelmilla kannustaa pohtimaan suorittamiaan tehtäviä ja ratkaisemaan niissä esiintyvät ongelmat, hänen motivoimisensa omatoimisiin ratkaisuihin hyödyttää sekä kurssin järjestäjää että opiskelijaa itseään.

Adaptiivista testausmenetelmää hyödyntävässä rakenteessa voidaan käyttää paljon erilaisia aineistoja ja arviointitapoja, joista pääasiallisimpia mahdollisuuksia tarjoaa matemaattisen suorituskyvyn mittaaminen. Suorituskyvyn mittaamisen menetelmiä on käytössä useissa tehtävissä ja tehtäväkohtaista opiskelijan suorituksiin viittaavaa aineistoa on satoja tuhansia yksiköitä. Tätä tietoa voidaan soveltaa sekä yksilökohtaisessa arvioinnissa sekä myös koko kurssin yleisen vaikeustason arviointiin. Matematiikan ja tietotekniikan opetuksessa käytetään opintojen jäsentämiseen opintopolkuja, joiden rakenne on pääosin samanlainen eri oppilaitosten välillä ja joiden avulla voidaan mitata oppimisen tavoitteita ja opetuksen sekä oppilaiden menestyksen tasoa ([1]). Opintopolkurakenteilla pyritään siirtymään tietokoneistamaan opetuksen sisältö. Matematiikan oppimisen vaikeuksia indikoivien erityisiä hankaluuksia aiheuttavien ongelmakohtien, niin kutsuttujen matematiikan solmukohtien ([2]), tarkastelu tuo opintopolkujen soveltamiseen tehtävätason näkökulman. Tietoaineistoa hankitaan jatkuvasti tasokokeilla, laskuharjoituksilla että kotitehtävillä.

Laskutaitojen korrelaatio oppimisvaikeuksiin avaa tarkastelunäkymän tarkempaan ja monimuotoisempaan analyysiin. Yhdistämällä esimerkiksi adaptiivisen testauksen avulla kerättyä tietoa muista aspekteista koottuun tietoon, saadaan kokonaiskuvan soveltamiseen vähintäänkin monipuolisempi joukko tarkastelutapoja. Oppimisvaikeudet havaitaan tutkimusten perusteella etenkin perusopetuksessa usein vasta kolmannen ja viidennen luokan aikana, ja tässä vaiheessa oppimisvaikeuksien kokemukset ovat saattaneet jo heikentää itsetuntoa sekä oppimiseen liittyviä asenteita, mistä johtuen oppimisvaikeuksien aikaisempaan havaitsemiseen kohdistuvista menetelmistä on ehdottomasti hyötyä. Toisaalta myös menestyvät oppilaat kaipaavat turhautumisen kokemusten välttämiseksi tukea tilanteisiin, joissa vaaditut tehtävät ja mahdolliset lisätehtävät on suoritettu tai ne eivät muuten tarjoa tarpeeksi haastetta. Menetelmissä

onkin tarpeen huomioida kumpikin oppimisprofiililtaan todennäköisesti toisistaan merkittävästä poikkeavat oppilasryhmät, ja työssä voidaan hyödyntää sopeutuvia oppimisympäristöön integroituja osaamista ja oppimiskäyttäytymistä mittaavia ohjelmistokomponentteja. Työn sisältöä ja toteutuksilta edellytettävien taustatekijöiden sisältöä ohjaavat kutakin koulutusastetta koskevat OPS2016-määräysten asettamat perusteet ([3]).

2 TIETOKONEAVUSTEINEN OPPIMINEN

Tietokoneavusteisesta matematiikan ja tietotekniikan opetuksesta on tehty lukuisia tutkimuksia, joissa on selvitetty tietoteknisten välineiden vaikutusta opetettavan aineiston sisällölliseen omaksumiseen sekä lisäksi kerätyn tiedon pohjalta on pystytty määrittelemään mallia sille, mikä on oleellista oppimisen mittaamiseksi tieteelliseen tutkimukseen riittävällä tarkkuudella. Historiallisesti tarvetta tietokoneavusteisen oppimisen kehittämiselle on toteutettu pitkään kokeellisten rajattujen tutkimusten kautta ([4], [5]). Tarve automatisoidulle arvioinnille on esiintynyt myös koko 2000-luvun aikana laadituissa opetussuunnitelmissa. Järjestelmiä ja kehysmalleja oppimisympäristöille on lukuisia, ja kehitystyötä tehdään sekä yliopistojen että tutkimuslaitosten lisäksi myös kaupallisten toimijoiden toteutuksissa. Tutkimuksissa on selvitetty sähköisen opetuksen vaikutusta matematiikan ja tietotekniikan oppimiseen laskutehtävin sekä ohjelmointisuorituksin tehtäväympäristöissä ([6]), ja keskitetysti oppimiseen liittyvää aineistoa kerääviin järjestelmiin liitännäisiksi toteutetuilla opetuspeliohjelmilla, tehtävälomakkeina sekä kokeina. Tässä tutkielmassa keskitytään Turun yliopistossa kehitettävään tietokoneavusteiden opetuksen ja arvioinnin automatisointiin ja oppimiseen liittyvän tietoaineiston hallintaan käytettävään Turun yliopistossa kehitettävään oppimisympäristöön (ViLLE, a collaborative education tool).

Tämän tyyppinen kollaboratiivinen oppimisympäristö on työväline, joka integroi yhtenäiseksi kokonaisuudeksi laajan joukon tehtävämuotoja, tehtävien esittämistapoja sekä oppimisen tulosten analyysimenetelmiä. Järjestelmään on myös luotu lukuisa joukko erilaisia tehtäviä, jotka noudattavat yleisesti usean koulutusasteen oppiaineksen asettamia edellytyksiä. Tehtäviä myös käytetään laajasti, ja niitä voidaan yhdistellä hyvin monimutkaisiksi kokonaisuuksiksi. Tehtävien avulla tuotetusta datasta voidaan selvittää oppimisen tuloksia ja oppimiseen liittyviä tapoja, ja järjestelmä sallii myös käsin suoritettavan tarkastus- ja arviointityön tarpeen vähentämistä automaattisen arvioinnin toteutuksilla.

Tutkimuksissa joissa tarkastelun painopiste on ollut oppijan näkökulmassa, on tehtävien sisällöissä pyritty käyttämään oppimiseen kannustavia ja havainnollistavia menetelmiä. ViLLE:n ollessa järjestelmä jossa voidaan käyttää edellä kuvatun kaltaisia oppimiseen kannustavia oppimisympäristöjä, on se myös tärkeä työkalu opettajille jotka voivat oppimisympäristön liitännäisten avulla kerätyn tiedon avulla suunnitella tehtäväkokonaisuuksia opintopolkujen mukaisiksi tasoryhmittäin tai joissakin tapauksissa myös yksilöllisesti tapauksissa joissa tuki- tai muu lisäopetus on tarpeellista. Yksilöllinen tarkastelu kohdistuneekin tämän tyyppisissä tapauksissa erilaisten oppimisen ongelmien ennakointiin ja tukiopetuksen suunnitteluun ([7]).

Nykyaikaiset oppimisympäristöt jakautuvat järjestelmiin joissa suoritetaan etäluento-opetusta ja joihin suoritetut tehtävät ovat usein keskustelusisältöisiä ryhmätöitä, ja järjestelmiin joita käytetään tukemaan kontaktiopetusta ja mahdollistamaan tarkka opetuksen ja tehtävien suorituksen kirjaaminen sovellukseen. Tässä tutkielmassa tarkastellaan jälkimmäisiä kontaktiopetuksen yhteydessä käytettäviä, oppimiseen kiinteästi liittyviä parametreja tutkimusaineistoksi tallentavia opetussovelluksia.

2.1 Matematiikan oppiaineksen tarkastelutapoja

Matematiikan keskeisen oppiaineksen kokeiden avulla saadaan eksklusiivisesti selvitetyksi, mitä oppilas osaa ja mitä hän ei osaa matematiikan käsitteistöstä. Tässä tarkastelussa voidaan käyttää hyväksi aikaisempien oppiaineksen kokeiden sisältöjä, ja niistä koottuja tuloksia. Tulosten pohjalta on mallitettu menetelmiä, joiden tarkastelu kohdistuu erityisesti haluttuihin oppimisen vaikeuksia aiheuttaviin tekijöihin. Oppimisvaikeuksien esiintymistä myöhemmässä matematiikan oppimisessa voidaan näiden tulosten pohjalta arvioida, kun on tiedossa joidenkin erityisten käsitteiden kuten lukusarjojen tai lukujen suuruuden sekä lukujen välisten operaatioiden vaikeudet joidenkin erityisiä ongelmia aiheuttavien tilojen esiintyvyydessä. Tällaisen tiedon hyödyntäminen kuitenkin vaatii erityisiä tietoja ja taitoja, joita tyypillisesti voidaan edellyttää erityisopettajilta. Kuitenkin jokaisen matematiikan opettajan on syytä kyetä

havaitsemaan, missä vaiheessa oppimisessa esiintyy sellaisia ongelmia jotka vaativat erityisopettajan konsultaatiota. Oppiaineksen osaamisen mittaamiseksi kerätyn aineiston pohjalta, käytetään tehtävien suunnittelussa hyväksi niin kutsuttuja matematiikan solmukohtia. Näiden käsitteiden oppiminen, ymmärtäminen ja muistaminen tuottaa oppilaille yleisesti ongelmia ([2], [8]). Niiden yhteys oppimisvaikeuksiin ei kuitenkaan ole aina ilmeistä, vaan oppimisen vaikeuksia saattaa esiintyä liitoksellisena johonkin muuhun oppimisen ongelmaan kuten dysleksiaan eli lukihäiriöön, ja silloin matematiikan oppimisen vaikeus on osana laajempaa oppimisen vaikeuksien kokonaisuutta. Tapoja tarkastella näitä oppimiseen liittyviä ongelmia on suorituttaa oppilailla tutkimuksellisia tehtäviä.

Vaikka tehtäviä voidaan laatia käsin tai käyttää valmiiden mallien mukaisia oppiaineksen kokeita, on nykyaikana opetuksessa mahdollista hyödyntää sähköisiä oppimisympäristöjä. Näissä tietokoneavusteiden oppimisen menetelmissä on useita niiden käyttöä puoltavia hyötyjä, kuten automaattinen tehtävätulosten arviointi, mahdollisuus tuottaa tehtäväsisältöjä yhteistyössä muiden opettajien kanssa sekä mahdollisuus kerätä automaattisesti tutkimuksessa hyödynnettävää kvalitatiivista ja kvantitatiivista tietoa. Tietokoneavusteinen oppiminen sallii myös suuremman tehtäväjoukon laadinnan ja suuren suoritusten määrän yhtenäistetyn hallinnoinnin, sekä voi tarjota mahdollisuuden yksilölliseen ja ryhmäkohtaiseen tehtäväsisältöjen muokkaamiseen. Kerätystä tiedosta voidaan myös mallintaa kullekin oppimistavalle ja oppimiskäyttäytymisprofiilille ominaisia piirteitä.

2.2 Tietokoneavusteisen oppimisen menetelmiä

Tietokoneavusteisessa oppimisessa (CAL, Computer-assisted Learning) käytettävistä menetelmistä yleisimpiä ovat järjestettyjen aineistojen käyttö, satunnaisesti tehtäviä tarjoavan järjestelmän toteutus sekä käyttäjän toimiin sopeutuvan, adaptiivisen menetelmän soveltaminen. Järjestetyillä tehtäväkokonaisuuksilla voidaan saavuttaa erilaisille käyttäjäprofiileille laadittuja kokoelmia. Näitä kokoelmia erilaisille käyttäjäprofiileille esittämällä voidaan luoda tilanne, missä kerättävästä tiedosta voidaan

tilastollisilla menetelmillä havaita tyypillisiä suorituspiirteitä sekä myös oppimisvaikeuksia indikoivia tai oppimisvaikeuksiin johtavia tyypillisiä poikkeavuuksia. Tällaisen järjestelmän ongelmaksi saattaa muodostua laajan tehtäväkokoelman kyseessä ollessa laadittujen kokoelmien hallinnointi, ja siksi laajojen kurssien tai kurssikokonaisuuksien tapauksessa tehtäväaineiston lisääntyminen johtaa kasvavaan resurssitarpeeseen. Tehtäväkokoelman muodostamisen osittainen tai kokonaan tapahtuva automatisointi olisi siis etu, joka järjestelmällä voidaan välittää kurssin opettajien työtä tukevaksi. Ympäristö voi välittää myös motivoivia tekijöitä tukemaan ja kannustamaan opiskelijan suoritusta, kuten erilaisia vihjeitä ratkaisun jäsentämisestä, ratkaisun teoreettista taustaa tukevia harjoituksia, tai eri tyyppisiä havainnollistavia peliohjelmia. Näiden kiinnostavuutta lisäävien tekijöiden käsitteleminen tarjoaa myös väylän tietojen tallentamiseen näiden suoritusten tapahtumasisällöstä mikä osaltaan voi auttaa myös järjestelmän sopeutuvuuden kehittämisessä ([9]).

Tietokoneavusteisen oppimisen järjestelmiä (LMS, Learning Management System) on useita, eri tasoilla toimivia toteutuksia. Aktiivisessa käytössä olevia ovat esimerkiksi kurssien yleisten sisältöjen jakeluun käytettävät työkalut, jotka tarjoavat myös työvälineitä tehtävien palautukseen sekä lisäksi mahdollistavat erilaisten rajapintojen kautta tapahtuvan arviointimenetelmien laadinnan. Tällaisissa järjestelmissä on kuitenkin ongelmana usein se että niiden avulla toteutettavia yhteistyökäytäntöjä rajaa järjestelmien painottuneisuus hallinnolliseen osaan opetustehtävää. Tehtäväsisältöjen toteutuksiin keskittyviä oppimisympäristöjä ja työvälineitä onkin luotu erityisesti tiettyihin oppiaineisiin kuten ohjelmoinnin ja matematiikan opetuksen tarpeisiin. Usea järjestelmistä toteuttaa myös automaattisen arvioinnin palautettujen tehtävien tarkastamiseen, mikä sallii kursseja järjestäville opettajille laajemman ja monipuolisemman aineiston sisällyttämisen kurssin aihepiiriin. Järjestelmissä on myös huomioitu opetuksen kehittämisen tarpeet, ja ne tarjoavatkin mahdollisuuden kerätä ja raportoida oppimiseen liittyviä tasoa ja osaamista mittaavaa dataa ([10]).

Usea nykyisistäkin järjestelmistä on kuitenkin rajattu usein vain kunkin oppilaitoksen henkilöstön ja opiskelijoiden käytettäväksi. Järjestelmien kollaboratiivisen sisällöntuotannon etu tuleekin esille niissä tapauksissa, joissa samaa aineistoa voidaan hyödyntää useamman eri oppilaitoksen opettajien ja opiskelijoiden kesken. Toisaalta moni järjestelmä sallii kirjautumisen esimerkiksi valtakunnallisesti käytössä olevalla, korkeakoulujen tietokonetunnuksen haltijoiden käytettävissä olevalla kertakirjautumistunnuksella ([11]). Tällaisen tunnuksen käyttäminen ei kuitenkaan välttämättä johda konkreettiseen käyttöoikeuteen kaikkiin kirjautumisen kohdejärjestelmässä laadittuihin aineistoihin, vaan usein pääsy on hyvin rajattu esimerkiksi yliopistojen yhteistyössä järjestämien kurssien aineistoihin. Nämä aineistot useasti tuotetaan kussakin yhteistyöhön osallistuvassa yliopistossa erikseen, minkä lisäksi myös kurssien harjoitukset ja muut palautettavat aineistot hallinnoidaan tyypillisesti kunkin yliopiston vastuuhenkilöiden toimesta. Sisällöllinen hallittavuus aiheuttaakin useasti moninkertaista työtä, ja sen vuoksi järjestelmien kollaboratiivisuuden astetta onkin pyritty kasvattamaan siihen, että yhden opettajan laatimat tehtävät saatetaan myös muiden kyseistä oppimisympäristöä soveltavien oppilaitosten opettajien käytettäviksi ([12]). Erityisesti automatisoidun arvioinnin tarjoaminen samojen tehtäväsisältöjen osalta käytettäviksi kaikkien yhteistyöhön osallistuvien tahojen kesken mahdollistaa arvioinnin ja tiedon keräämisen hallinnan helpottumisen

2.3 Tietokoneavusteisen arvioinnin menetelmiä

Tietokoneavusteisen arvioinnin (CAA, Computer-assisted Assessment) kehittäminen on tärkeä osa oppimisympäristöjen kehittämistyötä. Automaattinen arviointi on erityisen merkittävässä roolissa silloin, kun tehtävätoteutuksien määrä on tuhansia tai kymmeniä tuhansia yksittäisiä tehtäviä, ja joista oppiaineksen edellyttämät kurssien sisällölliset kokonaisuudet kootaan. Automaattisessa arvioinnissa voidaan käyttää monia erilaisia menettelyjä, kuten tapahtumien ajankohtien sekä keston tallentamista ja yhdistelemistä tehtävien tuloksiin, sekä tapahtumien lukumäärää tehtäväkierrosta kohti, ja tällaisen metatiedon jälkikäteisanalyysin avulla laadittua opiskelija- tai ryhmäkohtaista

suoritusprofiilia. Näitä tietoja voidaan myös soveltaa tehtävän suorituksen aikaiseen tehtäväkokonaisuuden yksilöintiin jopa jokaiselle kurssin opiskelijalle erikseen. Automaattinen arviointi voi myös mitata yksittäisten tehtävän sisältämien tyypillisen mallinnettujen ongelmia aiheuttavien kohtien analysointia, ja näiden esiintyvyyden sopeuttamista käyttäjän vasteeseen ja aikaisempaan suoritusdataan perustuen. Tällöin automaattisen arvioinnin toteutuksella muodostetun arvostelullisen aineiston lisäksi tallennettua tietoaineistoa voidaan hyödyntää tukevana tekijänä oppimisen tavoitteiden parempaan kokonaissaavutukseen tähtäävässä kehitystyössä. Matemaattisten tehtävien tietokoneavusteiseen arviointiin on käytössä algebrallisia lausekkeita ja niiden tekijöiden oikeellisuutta testaavia malleja ([13], kappale 7). Ohjelmointitehtävien arvioinnissa voidaan hyödyntää ohjelmointikielten kääntäjien ja ajoympäristöjen tarjoamia syntaksin ja semantiikan tarkistukseen käytettäviä sovelluksia. Lisäksi tunnettaessa tarkastelun kohteena olevan ohjelman toteuttama algoritmi, voidaan sen lausekkeellinen malli tarkistaa myös algebrallisesti. Automaattisen arvioinnin erityinen etu, sen tarjoama välitön palaute tehtävää suorittavalle tukee kurssin sisällöllisiä ja suorituskyvyllisiä tavoitteita. Aikaisemmin mainitun, suorituksista kerätyn metatiedon automaattisessa arvioinnissa on hyötyä tehtävien suorituksiin liittyvästä, opiskelijakohtaisesta suorituskykyyn liittyvästä aineistosta. Tehtäväkokoelman vaikeustason määrittely tällaisen tiedon avulla muodostetun profiilin mukaan toimii adaptiivisena, toisin sanoen käyttäjälle ominaisiin parametreihin sopeutuvana vaikuttajana. Adaptiiviseen testaukseen perustuvaa vaikeustasoa kuvaavan lähtökohdan määrittelytapaa voidaankin soveltaa osana automaattisen arvioinnin toteuttavaa järjestelmää.

2.4 Tietokoneavusteisen adaptiivisen testauksen yleiskuvaus

Eräs menetelmistä arvioida oppimisen tasoa ja lähtötilannetta on tietokoneavusteista adaptiivista testausta hyödyntävä tehtäväkokoelman laadinnan kokonaan tai osittain automatisoitu menetelmä. Adaptiivisen testauksen mahdollinen hyöty onkin, että tehtäväkokoelmat voidaan laatia kurssin suorittajille yhteisiksi, ja kurssien osallistujien suorittaessa kursseja oppijalle ominainen onnistuneiden ja virheellisesti suoritettujen

tehtävien parametriprofiili muodostuu järjestelmässä kunkin tehtävän suorituksen jälkeen. Tällöin opettajan työ oppimisen analysoinnissa helpottuu kun käytettävissä on oppimistapoja ja oppimistuloksia esittävää data-aineistoa. Tässä tutkielmassa painopiste on adaptiivista testausmenetelmää hyödyntävässä mittausmenetelmässä, jota käytetään myös tehtävien suoritusvaiheessa suoritettaviksi tarjottavia tehtäviä painottavana tekijänä. Adaptiivisuuden vaikutus tässä kontekstissa kohdistuukin sekä käyttäjän toimiin sopeutuvaan tehtävien asetteluun, että tehtävien arvioinnin menetelmissä hyödynnettävään dataan, minkä lisäksi sen vaikutus tarkasteltavan kurssin arvosteluperusteiden lähtökohtiin korostuu ([14]).

Adaptiivinen metodi on riippuvainen luokittelumenetelmistä. Kun käsitellään tietoa josta osa ei vielä ole ajallisesti tapahtunut, ollaan riippuvaisia kahdesta seikasta:

- tallennetuista tulosaineistoista, riippumatta onko data hankittu adaptiivisesti, rakenteellisesti tai satunnaisesti järjestetyin tehtävin
- mallinnetuista suoritusnormeista, joihin aineistoa korreloidaan.

Tarkasteltavan kurssin ollessa kesken, tarkasteluajankohtaan saakka hankittua aineistoa ei kuitenkaan voida pitää epätäydellisenä sillä tehtävien tarjontatavan kannalta vain jokin edellinen, yhdistelmä joistakin tai kaikki edellisistä tehtäväiteraatioista ovat lähtökohtaisesti oleellisia. Luokittelijan kannalta tarkasteltuna muutoskuviot voivat olla äkillisiä, inkrementaalisia, asteittaisia tai toistuvia, mutta myös ulkokohtaisia, harha-askelmaisia, ja siten vaikeasti mallinnettavia ([15], kappaleet 2 & 3).

Lähtökohtaisesti tässä menetelmässä testataan testijoukossa esiintyvien tehtävätapahtumien esiintyvyyttä. Tätä havaintoa voidaan käyttää apuna myös erillisessä oppimisanalytiikassa, sillä edellytyksellä että tehtävätapahtumien esiintyvyydellä on korrelaatio oppimisnormissa. Tehtäväyksiköt ovat samassa oppimistapahtumassa ketjutettuja, eli tehtävän arvioimiseksi edellytetään jokaisen yksikön suorittamista joko hyväksytysti tai virheellisesti. Adaptiivisuutta voidaan hyödyntää virhesuorituksen uusintaan. On kuitenkin muistettava, että hyväksyntä tällaisessa tapauksessa tulisi

arviointia tarkastellen iteraatioiden määrästä riippuvaiseksi: Suoraviivaisessa, niin kutsutussa naiivissa arvioinnissa on oleellista huomioida opiskelijan taso, ja tasolle normalisoitu tehtäväyksikön tulos tulisi vastata yksittäistä, tai mahdollisimman pientä tehtäväyksikön iteraatioiden lukumäärää.

2.4.1 Adaptiivinen valintajärjestelmä

Adaptiiviseen testaukseen perustuva valintajärjestelmä koostuu päätasolla proseduurin toteuttavista toimintalohkoista, joista kukin (datalukija, luokittelija, tehtäväsuoritin ja -muokkain sekä tuloskirjaaja) suorittaa ominaista funktiotaan:

a) Datalukijan toiminnot
Datalukija tarkastelee yksittäisen opiskelijan kurssijakson aikana suorittamien tehtävien ja tehtäväosioiden tuloksia ja käsin tehtyjä arviointeja. Datalukija kokoaa opintojakson aikaisen datan ja opiskelijan edellisten samaan oppiaineeseen sisältyvien jaksojen tasoarvioinnit yhtenäiseksi tietojoukoksi, syötteeksi luokittelijaproseduurille.

b) Luokittelijan valikointikriteerit
Luokittelija kohdistaa opiskelijan käytettävissä olevan datan perusteella normijoukkoon, joka määrää seuraavan toimintalohkon perusjoukon. Luokittelija perustaa kriteerinsä aikaisempien tasotehtävien ja opiskelijan aikaisempien kurssien aikana saamiinsa arviointeihin.

c) Tehtäväsuorittimen valintakriteerit
Tehtäväsuoritin noudattaa ensisijaisesti luokittelijan toteuttamaa valikointikäytäntöä. Toissijaisesti sovelletaan myös jakson aikana kumuloituvaa data-aineistoa ja automaattista arviointia. Arviointimenetelmä sisältyy tähän lohkoon, tai vaihtoehtoisesti lohkon metodi kutsuu erillistä arviointifunktiota. Kurssin alkuvaiheessa arviointiaineistoa ei kuitenkaan ole saatavilla ensimmäisen tehtäväkierroksen aikana, ja valintaproseduurin täytyy pohjautua edellisten kurssijaksojen taso- tai

loppuarviointeihin. Lisäksi suorittimessa määrätään, kuuluuko tarkasteltava opiskelija adaptiiviseen vaiko verrokkiryhmään. Opiskelijan kuuluessa adaptiiviseen ryhmään määrätään osiossa käytettävä tehtävä aikaisempien arviointien perusteella ja esitetään ne opiskelijalle. Verrokkiryhmään kuuluville tehtävät esitetään yleisesti käytössä olevaa satunnaismetodia käyttäen, ja tehtävät voivat tällöin olla mitä tahansa osion tehtävien joukosta.

d) Tuloskirjaajan tallennusmenetelmä

Tuloskirjaaja on kiinteä osa oppimisympäristön toteuttavaa järjestelmää, ja siten se on ulkoinen osa adaptiivista valintajärjestelmää. Sen kutsuminen tehtäväsuorittimen keräämillä parametreilla (data-aineistolla) tapahtuu kuitenkin myös osana adaptiivista prosessia. Tuloskirjaaja toteuttaa suoraviivaista datan keruun ja tallennuksen peruskäytäntöä, jota edellytetään jokaisessa järjestelmään kytkeytyvässä prosessissa. Tutkimuskelpoista ja anonymisoitua raakamuotoisesta aineistosta muotoillaan siinä vaiheessa, kun järjestelmän tietokannasta noudetaan tutkijan tarkoitukseensa tilaamaa dataa.

Tunnettaessa valintajärjestelmän toiminta, voidaan tutkimukselle tuottaa aineistoa sekä adaptiivisella tehtäväjaolla että erilaistumattomalla verrokkiryhmällä suoritettujen tehtäväsisältöjen kautta. Koska adaptiivisessa opetuksessa järjestelmä muokkaa esitettävää materiaalia suhteessa opiskelijan suorituskykyyn ja oppimistapaan, tarvitaan luokittelijaan menetelmiä jotka seuraavat oppimisesta kerättyä tietoa tarkoitusta varten toteutetuilla menetelmillä (katso CAT -proseduurin kuvaus alla). Tehtäväsuorittimen ja -muokkaimen avulla saadaan rajattua hyvin tarkasti esimerkiksi yhteen opiskelijan kannalta tärkeään tehtäväkategoriaan, oppimisen solmukohtaan tai osioon jossa esiintyy mallin ulkopuolisia erityisvaikeuksia. Solmukohtien esiintyminen erilaisissa tehtävätyypeissä tunnetaan, sillä niihin kohdistuvaa tutkimusta on tehty runsaasti että ne ovat oleellisia sisältöjä matematiikan oppiaineksessa ja joiden hallitsemista edellytetään seuraavilla tehtävävaikeustasoilla.

Tarkasteltaessa pisteytyksen merkitystä tehtävien vaikeustason, havaitaan että kokeen vaikeustason manipulointi ennen tehtävän suoritusta voi vaikuttaa testauksessa havaittuihin tuloksiin. Vaikeustason manipulaatiolla voidaan tällöin asettaa lähtökohtainen tilanne sellaiseksi, missä haluttu vaikeustaso asettuu pisteytyksen perusteella. Suosittelumekanismi ei tällöin tarjoa välttämättä konkreettisesti vaikeampia tehtäviä niitä suorittavalle käyttäjälle, mutta tällä tavoin voidaan tarjota esimerkiksi määrättyyn tehtäväkategoriaan kuuluvia tehtäviä suoritettavaksi muita useammin ([16]).

2.4.2 Tietokoneavusteinen adaptiivinen testausproseduuri

Tietokoneavusteisen adaptiivisen testausproseduurin (CAT, Computerized Adaptive Testing) yleinen kuvaus ([17], s. 362) määrittää toiminta-arkkitehtuurin IRT -kontekstissa (Item Response Theory; Yksikkövasteteoria) seuraavasti:

a) yksikkövastemalli (Item Response Model)
CAT voidaan toteuttaa 1-3 -parametrisoidulla logistiikalla tai normalisoidulla kumulatiivisella jakaumalla (ogive). Mallin valintaan vaikuttaa tarkasteltavien tehtäväyksikköjen luonne (vapaa- tai monivalintamuoto) ja yksikkövastedatan sopivuus kyseiseen malliin.

b) yksikköjoukko (Item Pool)
Missä tahansa kontekstissa käytetylle CAT -operaatioille oleellista on yksikköjoukko, joka voidaan projisoida arviointeihin perustuvaan IRT -parametrijoukko, ja jonka yksikköparametrit on normalisoitava tarkastelukontekstin yleiseen metriikkaan. Mitään yleistettyä ohjetta tämän toteuttamiseksi ei kuitenkaan ole olemassa. Tällaisessa tilanteessa on edellytettävä CAT -proseduurissa käsiteltävien yksiköiden ominaisten vaikeusasteiden jakautuvan yksikköjoukon ja parametrijoukon koko tarkasteluskaalan ulottuvuudelle.

c) lähtötaso (Entry Level)

Adaptiivinen testaus mahdollistaa tasoryhmille ominaisten vaikeustasojen tarkastelun ja arvioinnin. Tasoryhmän määrääminen kullekin opiskelijalle tapahtuu tasokoekategoriaan lukeutuvien tehtävien suorittamisella ja niiden pisteyttämisellä (skaalana kerralla onnistumiset – iteraatioiden määrä, ja näiden käänteinen vaikutus pisteytykseen). Tarkkaa lähtötason analyysiä ei välttämättä voida ensimmäisen yksikköjoukon kohdalla asettaa, mutta seuraavalla tasolla tämän tulisi olla jo mahdollista mahdollisimman pienellä virhemarginaalilla.

d) yksikkövalintasääntö (Item Selection Rule)

Yksikkövalinnassa on vaihtoehtoisina käytettävissä maksimi-informaatioproseduuri sekä frekvenssiparadigman mukainen (Bayesilainen) proseduuri. Maksimi-informaatiovalinnassa on käytettävissä maksimimäärä tehtäväkohtaista dataa edellisen tasokoetehtävän jälkeisen tason määrittämisen arviosta. Bayesilaisessa valinnassa pyritään minimoimaan tehtävän jälkimmäisen arvioinnin odotusarvon varianssi. Molemmat menetelmät vaativat joka tapauksessa koko ennalta käsittelemättömän tehtäväyksikköaineiston läpikäyntiä jokaisen yksikön kohdalla.

e) pisteytysmetodi (Scoring Method)

Opiskelijakohtaiset piirteet pyritään arvioimaan joko enimmäistodennäköisyyteen tai Bayesilaiseen modaaliseen arviointiin ([18]) perustuen. Jälkimmäinen menetelmä tarjoaa yksilölliset ominaispiirrearviot yksittäisille tehtäväyksiköille, tai vastauskuvioille jotka sisältävät pelkästään oikeita tai virheellisiä vastauksia. Maksimitodennäköisyydellä ei voida tarkastella edellä esitettyjä tekijöitä sisältäviä joukkoja. Tarkemmin, pisteytyksellä painotus kohdistuu aikaisempaan piirrearvioon, kun taas maksimitodennäköisyysarviot ovat asymptoottisesti painottumattomia. Käytännössä voidaan alkupään tehtäväyksiköissä käyttää modaaliseen arviointiin perustuvaa pisteytystä siihen saakka kunnes parametreista on mahdollista havaita toiminnan malli eli kunnes tarkasteltavan testikäyttäytymisen osalta voidaan siirtyä maksimitodennäköisyyspisteytykseen ([19]).

f) päätelmäkriteerit (Termination Criterion)

CAT-metodiikassa testausta jatketaan vain niin kauan kuin on tarpeen kunkin opiskelijan kohdalla. Mitattaessa opiskelijan kykyjä, metodiikkaa tulee käyttää kuitenkin niin kauan että ennalta määrätty tarkkuus arvioinnissa on saavutettu, jolloin kunkin tehtävän suorittajan kohdalla tehtyjen päätelmien kriteerit täyttyvät tasavertaisesti suhteutettuna muihin jakson osallistujiin.

2.5 Adaptiivisen testauksen käyttömahdollisuudet

Kiinteäpituisiin harjoitus- ja koetilanteisiin sovellettu tehtäväasettelu on ollut ja on edelleen käytössä kaikilla koulutusasteilla. Tämän tyyppisissä klassiseen testausasetteluun pohjautuvissa tehtävissä ei kuitenkaan pystytä huomioimaan oppimisvaikeuksien ilmenemiseen johtavia yksilöllisiä oppimisen piirteitä, eikä voida myöskään havaita ovatko johonkin määrättyyn kategoriaan sisältyvät tehtävät vaikeita yksittäisille tehtävien tekijöille tai erityisesti jollekin arvosanajakaumassa esiintyvälle normitetulle ryhmälle. Tehtävien suorittamiseen käytettävää aikaa on vaikeaa mitata, eikä se kokeisiin tai tehtäviin osallistuvien suhteellisen suuren joukon vuoksi ole käytännöllistä. Useimmissa tapauksissa joissa koetilanteelle on varattu rajallinen ajankohta, ei mitata myöskään kokeen suorittamiseen kulunutta aikaa vaan usein voidaan havaita vain kokeen suorittajalta jääneen tekemättä osa tehtävistä tai kunnes kaikki tehtävät on suoritettu. Arvioinnissakin voidaan silloin selvittää ainoastaan, onko tehtävän suoritus onnistunut tai epäonnistunut, ja arvioinnin pisteytyskin pohjautuu pitkälti tentaattorin tietoihin ja osittain subjektiiviseen arvioon. Kokeita tai laskuharjoituksia suunniteltaessa voidaan tietenkin tarjota samaan tehtäväkategoriaan kuuluvia tehtäviä peräkkäisissä koe- tai harjoitustilaisuuksissa, mutta sekin vie tarpeettomasti aikaa muilta samalla jaksolla esiintyvien aiheiden testaamiselta. Lisäksi paperimuotoisille kokeille, vastauksien arvioinneille sekä arvioinnin palautteelle on varattava kiinteä aika kontaktiopetuksesta, tai sovittava opettajan ja opiskelijan välisestä erillisestä palauteajankohdasta ([20]).

Tietokoneistetun tehtäväjärjestelyn myötä saavutetaan tilanne, missä harjoituksia

voidaan pitää paitsi kontaktiopetuksen aikana mutta opiskelijat voivat suorittaa tehtäviä oppimisympäristössä oman aikataulunsa asettamin edellytyksin. Tietokoneistetussa järjestelyssä voidaan myös saada selville muitakin seikkoja tehtävän onnistumisen tai epäonnistumisen lisäksi tietyn tilaisuuden aikana, kuten tehtävän suorittamiseen käytetty aika, tehtävien tulokset, sekä suoritusajankohta. Lisäksi voidaan myös tarjota mahdollisuus yrittää ratkaista tehtävä uudelleen epäonnistumisen tapauksessa, ja sallia tarvittaessa useampikin ratkaisutilaisuus.

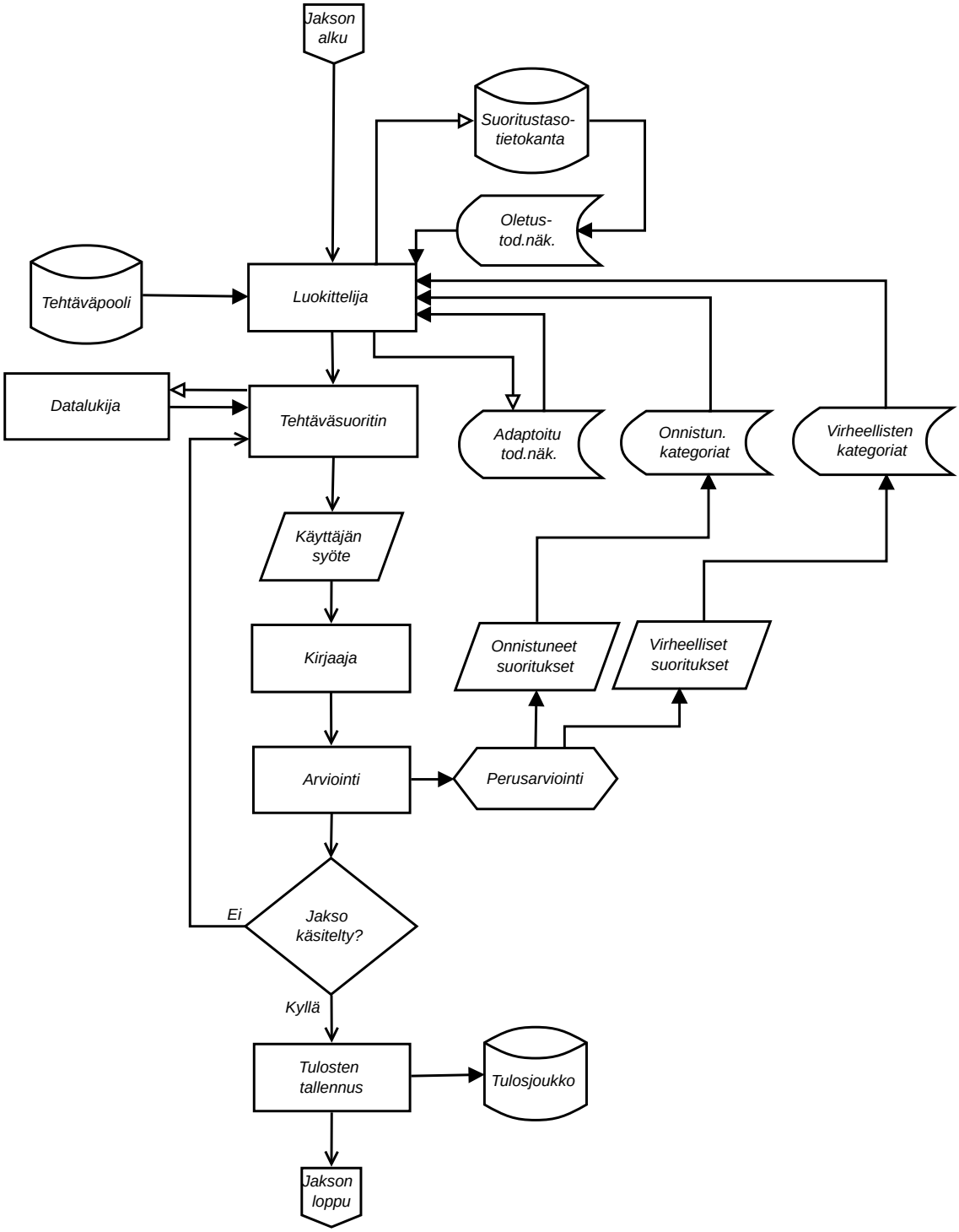
Suorituksista voidaan tällöin edellä mainitun mukaisesti kirjata kuhunkin iteraatioon kulunut aika. Kun koe- tai harjoitustilanteita on ollut jakson suorittamiseen vaadittu määrä, saadaan kullekin opiskelijalle ominainen oppimisprofiili muodostettua näistä peräkkäisistä tapahtumista. Aikaisempien jakson instanssien aikana kerätyn oppimisprofiileja sisältävän aineiston perusteella muodostettuun malliin pohjautuen voidaan muokata oppimistilanteessa esitettäviä tehtäviä adaptiivisesti oppimisprofiilille ominaiseen suoritustapaan, mutta on myös syytä huomioida kurssia kehitettäessä tapahtuneet sisällölliset muutokset ja niiden vaikutus koottuun aineistoon ([16]).

2.6 Adaptiivisen testauksen soveltaminen opetusaineistoihin

Adaptiivista testausta on hyödynnetty analysoimalla yliopisto-opetuksessa käytetyllä yhteistyöpohjaisella oppimisalustalla koottua aineistoa jälkikäteen, tarkoitukseen kehitetyillä ohjelmistoilla. Menetelmällä on mallinnettu normitetut arvosteluluokitukset, joiden korrelaatio kunkin opiskelijan tyypilliseen oppimisprofiiliin auttaa mahdollisen lisäopetuksen tarpeen arvioinnissa. Varsinaisen harjoituksen aikana adaptiivisuutta ei ole ollut merkittävissä määrin hyödynnetty, vaan harjoitus- ja koetilanteet ovat satunnaistetulla tehtävävalikoinnilla toteutettuja. Aineiston perusteella on kuitenkin pystytty havaitsemaan mallia soveltaen, missä vaiheessa oppimisprofiili alkaa konvergoida johonkin normiin. On siis selvitettävä, voidaanko oppimisen tuloksiin vaikuttaa muuttamalla opetuksen muotoa tarjoamalla opiskelijan profiilille normisovitettuja tehtäviä tukemaan tavoitetta hyväksyttävästä suorituksesta

2.7 Adaptiivisten ja satunnaistettujen tehtäväkulkujen eroavaisuuksia

Faktoihin perustuva kategorisointi ja kategorioihin perustettu adaptiivinen soveltaminen vaatii tietoa proseduurin aikaisista tapahtumista. Näiden tapahtumien malli voi olla suoraviivainen (peräkkäinen analyysinäkökulmasta), jolloin jakso sisältää täysin samanlaisen skeeman mukaisen joukon tehtäviä jokaisen tehtävän suorittajan kohdalla ja tällöin oppimisen näkökulmasta voidaan mitata opiskelijoiden suoritusten yhdenmukaisuutta. Tämä ei välttämättä tarjoa etua oppimiseen liittyvien vaikeuksien kohtaamisessa ja niiden korjaamisessa tarjoamalla opiskelijoille erityisopetusta, sillä kerätty tieto ei suoraan osoita mitään erityistä ongelmakohtaa jolloin tukiopetuksen suunnittelu on myös haasteellista ja voi vaatia tarpeettoman paljon aikaa. Tätä mallia on lähes välttämätöntä käyttää tilanteissa, joissa tehtävät tai kokeet suoritetaan perinteisesti paperilomakkeita käyttäen.



Kaavio 1: Adaptiivisen testausproseduurin yleisrakenne

Testausmallissa voidaan toisaalta pisteyttää tehtävät todennäköisyysarvoilla, pitäen lähtötasona että jokaisella tehtävällä on onnistumisen sekä epäonnistumisen

todennäköisyys 0,5. Lähtökohtana yhtäläisellä todennäköisyyspisteytyksellä merkittyä tehtäväjoukkoa voidaan silloin kohdistaa kuhunkin opiskelijaan, ja mikäli tehtävää suorittava opiskelija onnistuu tai epäonnistuu useassa peräkkäisessä tehtäväsuorituksessa, voidaan valita hänelle seuraavaksi suoritettavaksi kokonaisjoukosta joko vaikeampia tai helpompia tehtäviä ([20]). Tällaisessa tilanteessa kannustetaan opiskelijaa suorittamaan tehtäviä lukumäärällisesti enemmän, mutta tässäkin järjestelyssä ongelmaksi muodostuu ettei oppimisvaikeuksia välttämättä pystytä haarukoimaan tälläkään menetelmällä riittävän tarkasti. Siksi onkin oleellista identifioida epäonnistumisten tapauksessa tehtäväjoukosta ne tehtäväkategoriat, joissa opiskelijalla on vaikeuksia ja tarjota niitä suoritettavaksi muiden kategorioiden tehtäviä useammin. Tällaisella järjestelyllä saadaan suoritettua opiskelijalle tärkeä kertaava harjoitus ja voidaan kannustaa opiskelijaa yrittämään tehtävän suoritusta uudelleen. Tehtävärakennetta voidaan silloin noudattaa seuraavassa kaaviossa kuvatussa prosessissa ([21]).

3 VILLE-OPPIMISYMPÄRISTÖ

ViLLE on yhteistyökäytäntöihin perustuva oppimisympäristö, joka tarjoaa mahdollisuuden opiskelijoiden ja opettajien väliseen vuorovaikutukseen sekä välittömään oppimispalautteeseen joko tehtävä-, osio- tai jaksokohtaisesti. Välittömän palautteen ohella voidaan suorittaa myös jaksojen tai muiden kokonaisuuksien arviointeja. Enimmäkseen ViLLE:ä käytetään matematiikan ja tietotekniikan opetuksessa jossa voidaan käyttää tehtävien pohjana havainnollistavia ja visualisoivia ohjelmointiesimerkkejä, matemaattisia funktioita ja näihin liittyviä yhdistelmärakenteita. Järjestelmä tukee myös yleistä tehtäväformaattia, jossa opettajat voivat laatia vapaamuotoisia tehtäviä omien opetus- tai tutkimustarpeidensa mukaisiksi ja sisällyttää niihin näin sekä yleisiä että matematiikan ja tietotekniikan opetukselle tyypillisiä tehtäviä. Tehtäviin liittyvien tulosten ja tapahtuma-aikojen kirjaamisen lisäksi voidaan tarkkailla myös opiskelijoiden läsnäoloja ja harjoitustapahtumia, ja sillä voidaan tallentaa myös järjestelmän ulkoisten tehtävien tuloksia. Järjestelmää voidaan siis käyttää täysin tai osittain automatisoituun arviointiin, sekä sen avulla voidaan tuottaa luentoja tukevia materiaaleja että tutoriaalien aineistoja. Järjestelmään voidaan tuottaa yllä mainittuja tapahtumatietoja hyödyntäen myös sähköisiä koetilanteita, joissa tehtävän palautuskertojen määrää sekä kokeen suoritusaikaa rajataan. Tällaisessa kokeessa suoritetut tehtävät voidaan tarkastaa sekä käsin että automatisoidusti, ja jälkimmäistä tapaa voidaan täydentää myös osittaisella käsin tapahtuvalla tarkastuksella, ja tietoa niiden suoritustapahtumista voidaan tallentaa samalla tavoin kuin minkä tahansa muunkin järjestelmässä suoritettavan tehtävän aikana.

3.1 ViLLE:n yleiskuvaus ja aiempia tutkimuksia

Tietokoneavusteisen oppimisen perusteita voidaan hyödyntää tarjoamalla oppijalle tehtäväkokonaisuuksia, joissa tehtävän suorittamisen jälkeen voidaan antaa välitön palaute samassa sovellusistunnossa. Tällaisissa käyttötilanteissa hyödytään käyttäjän omaksumasta sovelluksen käyttötavasta ja asennoitumisesta tehtävän suorittamisen

yhteydessä välittömään palautekäytäntöön. Opiskelijalla on tällöin matalampi kynnys kokeilla suorittaa tehtävä uudelleen, esimerkiksi käyden järjestelmällisesti läpi kaikki vastausvaihtoehdot tai harkintansa mukaan selvittää tehtävä tausta-aineiston tuella oppimistapansa muodostamin edellytyksin. Käyttötapa saattaa silloin myös edesauttaa ennakoivaan tehtäväkäyttäytymiseen siirtymistä jolloin opiskelija miettii asiaa teorian lähtökohdista ja siirtyy suorittamaan tehtävänsä vasta sen jälkeen. Oppimisalustana ViLLE tukee molempia toimintatapoja ja tehtävien suorituksista kerättyä dataa voidaan hyödyntää opintopolkujen kehittämiseen liittyvässä tutkimuksessa ([22], [23]).

Myös koetilanteita voidaan toteuttaa järjestelmään. Näihin koetilanteisiin pätevät samankaltaiset tehtävän suoritussäännöt kuin muidenkin tehtävien suoritukseen, sillä erotuksella että tehtävien palautusta ja suoritusaikaa on mahdollista määrätä. Lisäksi automaattinen arviointi hyödyttää sekä kurssin ajankohtaista suoritusta että tarjoaa hyödyllisiä lähtökohtia kurssin seuraavien toteutusten suunnittelemiselle. Sähköisesti järjestettäviä tenttejä on toteutettu sekä matematiikan että ohjelmoinnin peruskurssien suorituksista useamman vuoden ajalta, ja näistä suoritustapahtumista on mallinnettu kurssin kehittämiseen soveltuvia tietokokonaisuuksia, erityisesti koskien ohjelmoinnin peruskurssien opetusaineistoa ([24]). Kyseisiä aineistoja käytetään aktiivisesti sekä tietokoneavusteisen opetuksen kehittämisessä että siihen liittyvässä tutkimuksessa.

Opintopolut voivat poiketa toisistaan hyvinkin paljon, kun tarkastellaan perusopetuksen koko oppimäärää alakoulutasolta oppivelvollisuuden loppuun saakka, kuten myös lukiossa eri laajuisissa opintojaksokokonaisuuksissa. Oppimisalustaan voidaan rakentaa sovellus useita erillisiä saman sisällön aspekteja varten, jos halutaan tarkastella jotain määrättyjä oppimiseen liittyviä parametreja. Tästä johtuen on mahdollista toteuttaa myös sovellus, joka sallii opettajan toimesta tapahtuvan tehtäväkokonaisuuksien laadinnan – datan kerääminen tapahtuu silti tietosuojan ([25], 14 §) toteuttamiseksi vain oppimisalustan taustajärjestelmään, johon on rajattu pääsy ja josta tietoa luovutetaan ainoastaan anonymisoidusti tutkimuskäyttöön.

3.2 Datan formaatti ViLLE:ssä

Järjestelmä kirjaa oppimisympäristössä suoritetuista tehtävistä suoritustiedot opiskelija-, tehtävä- ja jaksokohtaisiin taltioihin, jotka on laadittu taulurakenteina MySQL -relaatiotietokantaan. Relaatiomallin mukaan tietokannasta on opiskelijan tunnisteen lisäksi oltava kyseltävissä tietoaineistoja useasta tiedollisesta aspektista, ja tiedon formaatin osalta on kiinnitettävä huomiota kannassa esiintyviin tietotyyppeihin. Tietotyyppien ominaisuudet ovat tärkeitä erityisesti tilanteissa, joissa syntyy tarve muodostaa uusi relaatiotaulua tietokantaan. Adaptiivisen tehtäväsäiliön kannalta oleellista on opiskelijan perustietojen lisäksi tieto tehtävistä, niiden hypoteettisista tuloksista, onnistumisen todennäköisyyden arvioinnista sekä näiden suhteesta tarkasteltavaan opintopolkuun. Arviointitietona käytettävä numeerinen tieto on aina oltava yhdistettävissä sekä tehtäväsäiliöön että opiskelijaan jota onnistumisen todennäköisyys koskee, etenkin tilanteissa joissa opiskelijan suorittamat tehtävät ovat osa ennalta määriteltyä opintopolkua.

Uuden data-aineistoluokan laatiminen järjestelmään on syytä määritellä järjestelmässä käytössä olevalla modulaarisella suunnittelutavalla. Tässä työssä kuvattavaa adaptiivista testausta käyttävän metodiikan soveltamiseen tarvittava riippuvuus muihin järjestelmän komponentteihin muodostuu väistämättä valikoitaessa tarvittavaa lähtöaineistodataa sekä luotaessa näkymä tietoihin adaptiivisen metodiikan kautta. Tällöin uuden data-aineiston aspektia voidaan ajatella jo toteutetun, olemassa olevan järjestelmän ja näkymien laajennuksena toimivana liitännäisenä.

3.3 Datan hankinta ViLLE:n välityksellä

Tietojen keräämiseen matematiikan tai tietotekniikan opetuksessa voidaan käyttää näitä oppiaineita varten toteutettuja tehtäväsovelluksia, joiden tyyppejä on kuvattu järjestelmän opettajadokumentaatiossa ([26], kappaleet 4.3 & 4.4). Lähtökohtaisesti sovelluksessa ajetut tehtävät keräävät perustietoja suoritustapahtumista, kuten opintojakson ja opiskelijan tunnisteen, tapahtuman aikaleiman, binäärisen tiedon siitä

onko tehtävä suoritettu onnistuneesti, monesko suorituksen yrityskerta on ollut ja kuinka paljon aikaa tehtävän suorittamiseen on kulunut. Tehtävät voivat olla muodoltaan myös sellaisia, että niissä vaaditaan kaikkien tehtävän osioiden suorittaminen lopputulokseen pääsemiseksi. Ehtoja tehtävän hyväksyttyyn suorittamiseen voidaan muodostaa myös edellä mainittujen yhdistelminä.

Tutoriaaleja ([26], kappale 6) eli tehtäviä jotka sisältävät esimerkiksi ohjelmametodin havainnollistavan läpikulun, voidaan hyödyntää siten että opiskelijan täytyy vastata ennalta määrätyissä kohdissa tutoriaalin kulkua ohjelman suorituksen sen hetkiseen tilanteeseen liittyvään kysymykseen. Näitä tehtäväkulun yksittäisiä kappaleita voidaan toistaa, mikäli tehtävään on vastattu virheellisesti, tai antaa kokonaisarviointi tehtävän suorittamisen jälkeen ja tehtävän suorittaja voi kerrata tausta-aineiston ja suorittaa tehtäväjakson uudelleen haluamanaan ajankohtana.

3.4 Adaptiivisen testauksen mahdollisuudet ViLLE:ssä

ViLLE:en voidaan rakentaa eri tyyppisiä tehtäväkokonaisuuksia matematiikan ja tietotekniikan opetusta varten. Suoritukseen liittyvien perustiedollisten parametrien taltioon kirjatuista tiedoista voidaan tietokantakyselyillä sekä tilastollisesti selvittää, mitkä tehtävät on jouduttu toistamaan useimmin. Silloin voidaan myös havaita, esiintyykö tämä tieto joillakin yksittäisillä tai määrättyyn ryhmään kuuluvilla opiskelijoilla. Perusasetelmassa on siis käytössä joko kronologinen tai satunnaistettu tehtävänkulku. Tehtävän osioiden epäonnistuessa niitä voidaan toistaa välittömästi kyseisen kohdan jälkeen, tai esittää virheellisesti suoritetut tehtävät osion päätyttyä kerrattaviksi.

Tilastollista profilointia erilaisten oppimiseen liittyvien vaikeuksien osalta on tutkittu korkeakouluopetuksen alkuvaiheen opintojaksojen aikana suoritettavaksi laadittujen tehtävien tuloksiin perustuen. Malleissa on käsitelty kullekin opiskelijalle ominaista tehtävätapahtumista kirjattua numeerista informaatiota. Kunkin opiskelijan yksilöllinen data voidaan havainnollistaa toisiinsa verrattavissa oleviksi määrämuotoisiksi

vektoriaineistoiksi, joita voidaan edelleen suhteuttaa normitettuihin arvosanajoukkoihin. Normijoukkojen muodostamiseksi on toteutettu usean vuosikurssin oppimistapahtumien tietoaineistoihin pohjautuvaa mallintavaa tutkimusta ([27], tutoriaaliopetus).

Virheellisesti suoritettujen tehtävien osalta voidaan käyttää satunnaistetun tarjonnan ohella adaptiivista esitystapaa. Tällöin virheellisesti suoritettujen tehtävien kanssa samaan kategoriaan kuuluvia tehtäviä esitetään opiskelijoille, joilla on havaittu useita hyväksymättömiä suorituskertoja kyseisen tyyppisissä opintopolun ajankohtaista tasoa mittaavissa tehtävissä. Kategoriat täytyy määrittää tehtäväjaksoa laadittaessa etenkin tasokoetyyppisille tehtäville, mutta muissakin tehtäväjaksoissa adaptiivisuutta voidaan hyödyntää. Lisäksi adaptiivisuuden jatkuvuus eri tehtävien ja jaksojen välillä vaatii yksilölliseen käyttäjän tunnisteeseen perustuvaa tietojen luokittelua taustajärjestelmässä. Tällöin oppimista tukevien tehtävien tarjoaminen on mahdollista aloittaa opintojakson aikaisessa vaiheessa, tarkentaen tilanteissa joissa oppimisvaikeuksien varhainen havaitseminen on kriittisessä asemassa kurssilla menestymisen kannalta ([28]).

3.5 Tietoaineistojen kategorisointi ja validointi

Oppimisympäristössä opintojaksoa koskeva jaksolle ominainen binäärinen, numeerinen tai tekstimuotoinen tietoaineisto kootaan kurssille ominaisten vaatimusten perusteella, aikaisemmin kuvatun perusmuotoisen tiedon oheen. Yleisesti ottaen mitä tahansa aineiston osaa voidaan pitää arvioinnin kannalta validina. Automaattisen arvioinnin kannalta oleellisimpana voidaan pitää binääristä ja numeerista aineistoa. Tekstimuotoisen aineiston osalta voidaan joutua validoimaan ja kategorisoimaan tiedon sisältö merkitsemällä se toisella tai molemmilla edellä mainituista merkintätavoista, sillä lähtökohtaisesti ohjelmallinen käsittely on mahdollista vain kun tiedolla on suoraviivainen koneellisesti tulkittavissa oleva arvo.

Tehtäväkohtaisen sisällöllisen tiedon lisäksi on kutakin suoritusta koskevaan arviomerkintään koodattava tieto myös siitä, onko tehtävä ohitettu, onko se ehditty suorittaa ja myös ajankohta jolloin tehtävä on suoritettu, mikäli tehtäviä voi suorittaa

jossain muussa kuin ennalta määrätyssä peräkkäisessä järjestyksessä. Ajankohta voidaan tallentaa myös siinä tapauksessa, missä tehtävät on suoritettava tietyssä järjestyksessä. Tehtävän suoritusajankohtien lisäksi voidaan hyödyntää tehtävän suorittamiseen kokonaisuudessaan kulunutta aikaa.

Lähetettäessä tehtävä arvioitavaksi, suoritetaan sille tehtävän luokittelu joko välittömään automaattiseen tai käsin suoritettavaan arviointiin. Välittömään arviointiin luokitellut tehtävät, yleisesti matemaattisten ja ohjelmointikielitehtävien osalta, voidaan arvottaa suorituskohtaisten väärintulkintojen (ViLLE -oppimisympäristön terminologiassa math_misconception) ja oikein suoritettujen osioiden yhdistelmänä ja suoritusta voidaan tämän jälkeen verrata mallisuorituksen mukaiseen lopputulokseen.

Matematiikan numeerisissa tehtävissä tämä on yleensä hyvin suoraviivainen proseduuri, ja tulos voi olla vain oikein tai väärin. Algebrallisissa tehtävärakenteissa väärinymmärryksen aiheuttama ongelmatilanne voi johtaa algoritmin tai rakenteen virheelliseen suoritukseen, mutta tällaisessa tilanteessa kirjataan tieto virheen esiintymiskohdasta. Ohjelmointitehtävissä hyväksyttävä ohjelman suoritus voi toteutua usealla eri tyyppisellä metodilla. Toisaalta voidaan tarkistaa myös kierroksen eri lohkojen välitulokset ja päätellään, hallitseeko opiskelija jonkin määrätyn algoritmisen ongelman ratkaisemisen. Nämä suoritukset voidaan merkitä epäselviksi käsin tarkastamista varten. Matematiikan keskeisen oppiaineksen kokeiden soveltamisen lisäksi voidaan käyttää ViLLE -oppimisympäristöön opettajien toimesta laadittuja kokeiden sisältöjä.

Perusopetuksen matematiikan osaamistason määrittämiseksi käytetään tasokoetehtäviä, joiden suoritusajankohta sekä suoritukseen käytetty aika tallennetaan. Tasokoetehtävien suoritusajankohtaa määräävät koetilanteen aikarajat ja aineiston käyttömahdollisuuksien rajaaminen. Tehtäväalueita voidaan rajata spesifisiin matematiikan solmukohtiin, joista saadaan havaituiksi ne tyypillisimmät virhetilanteet jotka tasoarviointiin vaikuttavat. Saman kurssin osallistujien tuottamasta datasta muodostuu kullekin opiskelijalle

ominainen tietojoukko ja koko kurssin osallistujista kerätystä aineistosta saatavaa tulostietojoukkoa voidaan mallintaa ja suhteuttaa aikaisempien kurssin iteraatioiden tuloksiin. Kerätyn tiedon normalisointi halutulle tarkasteluvälille toteutetaan ViLLE -oppimisympäristöön rakennetuille tehtäville automaattisesti, ja aineiston normalisointi voidaan toteuttaa jakamalla opiskelijan suorittaman tehtävän pisteytetty tulos kyseisen tehtävän maksimipistemäärällä. Etuna ViLLE:n käytössä on, että myös ne tehtävät jotka opiskelija jättää tekemättä voidaan pisteyttää ja hyödyntää arvioinnissa, ja tarvittaessa on mahdollista selvittää myös opiskelijan tekemättä jättämien tehtävien pohdintaan kulunut aika ([22]).

4 TUTKIMUSKYSYMYKSET

Ensisijaisena tarkoituksena on selvittää olemassa olevasta oppimisaineistosta ohjelmallisesti sopivia tilastollisia menetelmiä ([29], kappale 13, Raschin malli) soveltaen, tarjoaako oppimisympäristön järjestelmä adaptiivisella tehtäväasettelulla merkittävää etua satunnaistettuun tehtävätarjontaan perustuvaan tehtäväasetteluun verrattuna. Lisäksi selvitetään, onko ennakoivalla adaptiivisella testauksella tarjotuilla tehtävillä mahdollista parantaa niiden opiskelijoiden suorituksia, jotka historiallisen datan perusteella ennallistetusti ohjautuisivat ominaisryhmään joka saa hylätyn tai heikon arvosanan suorituksestaan. Mallien avulla toteutettavalla ennakoitavuudella pyritään havaitsemaan lähtötason lisäksi yksittäisten, määrättyihin kategorioihin sisältyvien tehtäväosioiden merkitys opiskelijan opintopolun ohjaamisessa suoritusten suotuisan kehittymisen tukemiseksi.

4.1 Tehtävien laatiminen

Kurssin opetusjakson tehtäväaineisto laaditaan ViLLE -oppimisympäristön opettajan graafisessa käyttöliittymässä, jossa voidaan laatia tehtäväjakson tai -osion kulku sekä siihen sisältyvät yksittäiset tehtävät. Tehtävien laadinnassa ei käytännössä ole rajoittavia teknisiä reunaehtoja, joten kokonaisuuksia laadittaessa on huomioitava muut viitekehykset kuten opetussuunnitelmat ja sisältövaatimukset. Adaptiivinen valinta voidaan toteuttaa kaikissa opetuksellisissa viitekehyksissä eli sitä ei ole sidottu mihinkään erityiseen tehtäväsisältöön tai ohjeistusmalliin, vaan sen tarkoituksena on tarkentaa järjestelmän tuottamaa oppimiseen liittyvää data-aineistoa ja tuottaa näin etua tutkimukselle ja opetuksen kehittämiselle. Adaptiivinen tehtävien esittäminen muodostaa oppimisprofiilille yksilöllisen, tasoryhmälle ominaiseen normaaliprofiiliin suhteellisen puurakenteen, jonka haarautumat ovat muuttujallisia, vastoin traditionaalisissa tehtäväkokonaisuuksissa esiintyvää vakiollista haarautumistapaa.

Tehtävät esitetään opiskelijanäkymässä ViLLE:ä käyttäville, kun käyttäjä on rekisteröity

jakson osallistujaksi. Tehtävät voidaan sallia suoritettaviksi määrätyllä aikavälillä, jolloin tehtävien suorittaminen on mahdollista esimerkiksi vain oppitunti- tai tutoriaali-istunnon aikana, tai vastaavasti opetuskerran jälkeen kotitehtävänä. Adaptiivisuus näkyy tehtäväasettelussa tehtävää suorittavalle yksilöityinä tehtäväkokonaisuuksina tai harjoituksen painottumisena niihin seikkoihin, joissa opiskelija on onnistunut erityisen lahjakkaasti tai vaihtoehtoisesti tarvitsee lisää harjoitusta. Kumpaakin lähestymistapaa voidaan käyttää luokiteltaessa opiskelijakohtaisia todennäköisyyspisteytyksiä tehtävien adaptiivista valintaa varten.

Oppimiseen kiinteästi liittyvät suoritusajankohta sekä suoritukseen kuluva aika tallennetaan osana suoritusta kaikissa tehtävien toteutusmalleissa. Tästä toiminnosta muodostuneesta datasta voidaan selvittää, eroaako oppimisvaikeuksien esiintyminen traditionaalisesta asettelusta ja mikäli eroaa, tuloksen eroavaisuuden suuruus tai laatu pystytään määrittämään. Tämä mahdollistaa oppimisvaikeuksien esiintymisen arvioinnin oppimisessa tallennettujen parametrien eli tehtävätulosten perusteella. Tilanteissa, joissa normiin suhteutuvan opiskelijaprofiilin yhteydessä havaitaan jonkin solmukohdan olevan opiskelijalle hankala – toisin sanoen vaativan useampia toistokertoja, ja palautetuissa ratkaisuissa esiintyy algebrallisia virheitä – voi sen suhde tiettyyn suoritusajankohtaan olla merkitsevä tekijä. Lisäksi voidaan tilastollisesti havaita, esiintyykö samalla opiskelijalla useissa tehtävissä samaan kategoriaan kuuluvissa tehtävissä ongelmia, mikäli ongelmien määrä yleisesti kumuloituu.

Tulosjoukon ja suoritustasotietokannan erillisyys järjestelmässä tukee adaptiivista prosessia. Kummastakin sijainnista voidaan tarkastella adaptiivisen todennäköisyyden vaikutusta tehtäväjoukon muodostamiseen, sillä adaptiivinen prosessi kuten traditionaalinen tehtäväjärjestys on luonteeltaan syklinen. Tästä johtuen tehtävien luokittelussa ja suoritusjärjestyksen asettelussa sekä tasotiedot että aikaisemmat suoritukset vaikuttavat tehtävien onnistuneen tai virheellisen suorituksen todennäköisyyksien ennakointiarvioon ([30]).

4.2 Järjestelmän arkkitehtuurityyli ja suunnittelumallit

Faktoihin perustuva kategorisointi ja kategorioihin perustuva ViLLE -oppimisympäristön järjestelmätoteutus noudattaa arkkitehtonisesti yleistä MVC (Model-View-Controller; malli-näkymä-käsittelijä) -suunnittelumallia ([31]) ja osittain myös palvelukeskeistä arkkitehtuuria ([32], luku 3). MVC -arkkitehtuuria käytetään erityisesti graafisten käyttöliittymien suunnittelun tukena, ja sen avulla järjestelmän kunkin komponentin käsitteellistä merkitystä voidaan tarkastella yksittäisenä komponenttina tai komponenttien yhdistelmänä. Lisäksi komponenttien välisten rajapintojen rooleja ja merkityksiä voidaan suunnittelutyössä tarkastella erikseen. Näissä ohjelmistoarkkitehtuurin malleissa näkymän ja ohjaimen komponentit toteuttavat järjestelmän käyttäjille näkyvät käyttöliittymä- ja syöteosiot, ja sovelluksen malli keskittyy tietojen keräämiseen ja tallentamiseen sekä tausta-aineiston näkymien hallintaan. Tietokannat joihin tutkimusaineisto tallennetaan toteuttavat relaatiomallia, jossa tiedot on tallennettu eri tyyppisillä avaimilla. Tietokantoihin tallennettua aineistoa voidaan kuitenkin tarkastella eri aspekteista muita kriteerejä käyttävien tietokantanäkymien avulla. Toteutuksessa hyödynnetään Vaadin -suunnitteluviitekehystä ([33]), jolla käyttöliittymä sekä siihen liittyvät sovellukset muotoilevat suoritinnäkymän tehtävien suorittajille sekä tehtäväjaksoja laativille opettajille.

Tällaisessa ympäristössä jaksot voidaan koostaa aktiivisessa mallissa sekä järjesteltyjen että jaeltujen tehtäväpoolien kautta. Nämä poolit perustuvat ennalta määritettyihin tehtäväjoukkoihin, joiden suoritusjärjestys on määrätty tai satunnainen. Adaptiivinen pooli toteuttaa tehtävien suoritusjärjestyksen osallistujakohtaisesti, ja sillä on riippuvuus opiskelijan aikaisempiin suorituksiin perustuvaan suorituksen laadullisten todennäköisyyksien pisteytysmenetelmään. Todennäköisyys tehtävän onnistuneelle tai virheelliselle suoritukselle on järjestellyissä ja jaetuissa tehtäväpooleissa kiinteästi määritelty. Adaptiivinen pooli asettaa nämä todennäköisyyspisteytykset aikaisempien tasomäärittelyjen avulla arvoihin, joita voidaan käytyä opiskelijalle esitettävien tehtävien esiintymisen painottamiseen.

Tässä työssä onkin ollut tavoitteena laatia adaptiivista valintamekanismia hyödyntävä tehtäväasettelussa käytettävä järjestelmä. Järjestelmää hyödyntämällä saadaan dataa, josta voidaan selvittää tiedon tarkoituksenmukaisuus oppimisvaikeuksien havainnoinnissa ja ennakoinnissa.

5 JÄRJESTELMÄN KUVAUS JA SUUNNITTELU

Adaptiivisen testauksen soveltamiseksi edellytetään olemassa olevan data-aineiston lisäksi myös aineistoa, jota käytetään aktiivisesti meneillään olevassa opintojaksossa. Toisaalta menetelmän soveltamiseksi tarvittavan aineiston ajankohtaisuus ei ole merkityksellinen kuin tapauksessa, jossa halutaan ennakoitavuus osaksi tunnistamiseen käytettyä menetelmää. Tässä toteutuksessa menetelmää sovelletaan lähtökohtaisesti historialliseen dataan niillä kursseilla, joilla on käytössä sekä satunnaistettu, järjestetty että adaptiivinen tehtävätarjonta.

Koska adaptiivinen tehtäväsäiliö käyttää suodatusmekanismeja, joiden avulla opiskelijan aikaisemmat suoritukset vaikuttavat seuraavien tehtävien sisältöön, voidaan sen luonnehtia olevan rajatun joukon suosittelujärjestelmä. Suosittelujärjestelmiä ([34], suosittelujärjestelmän sovellus sähköisessä oppimisympäristössä) käytetään tyypillisesti palveluissa joissa käyttäjäryhmien välillä voi esiintyä eriytyneitä tarpeita ja vaatimuksia. Ominaiset palvelut joissa tällaista suosittelutapaa käytetään, ovat erilaiset verkkokaupat joissa asiakkaan hankkimien tai selaamien tuotteiden perusteella arvioidaan, mitä asiakas todennäköisesti seuraavaksi selaa sekä valikoi, ja näitä suosituksia voidaan esittää asiakkaalle aina hänen vieraillessaan kaupan sivustolla. Adaptiivisessa järjestelmässä lisätehtävien merkitys voi olla hyvin oleellinen yhdenmukaisille suoritusprofiileille tai jopa yksittäiselle opiskelijalle, ja tällöin suosittelumekanismi arvioi yhden tai useamman parametrin perusteella mikä tehtävätyyppi olisi viiteryhmälle tai yksittäiselle opiskelijalle hyödyllinen.

5.1 Yleistä kehitysympäristöstä

Järjestelmä on toteutettu Vaadin -kehitysympäristöllä, jota ViLLE:en rakennetut tehtävät ja tehtävistä koostuvat kurssikokonaisuudet hyödyntävät tehtävien esittämiseen ja ajamiseen oppimisympäristössä. Kehitysympäristössä on käytettävissä koko tehtäväjoukon ohjaamiseen käytettävä tehtäväsäiliön hallintaobjektien kokoelman

sisältämä tiedon käsittelyn kannalta oleellinen metodiikka, johon kehityksen kohteena oleva adaptiivista testausta käyttävä tehtävävalintamenetelmä sijoitetaan käytettäväksi osana kurssin tehtäväosiota. Kunkin kurssin osalta voidaan päättää, käytetäänkö kaikissa kurssin tehtävissä yksittäisiä tehtäviä, järjestellyn tai tasomääritellyn tehtäväsäiliön sisältämää joukkoa, adaptiivisen tehtäväsäiliön sisältämää joukkoa tai näiden yhdistelmää. Adaptiivisen tehtävän metodiikka perustuu tehtävätapahtuman aikaiseen arvostelussa käytettävään tulosmerkitsemiseen ja tasopisteytykseen aikaisempien opiskelijan tai viiteryhmänsä tasokokeisiin. Myös tehtäväjoukon määrittelyn riippuvuus tehtävien pisteytykseen on tarpeen huomioida toteutuksessa.

Kehitysympäristö tukee graafisia kehitystyökaluja, joiden avulla komponenttien väliset rajapinnat sekä rajapinnat ohjelmiston ulkopuolisiin järjestelmän arkkitehtuuriosiin (kuten tietokantoihin, HTTP-palvelimiin sekä sisällönhallintaan) ovat havainnollisesti hallinnoitavissa. Myös uusien komponenttien laatiminen tai tuominen järjestelmään on mahdollista tuomalla objektit osaksi projektin graafista resurssinäkymää, ja liittymät niiden ja aikaisempien toteutusten välillä on mahdollista suurelle projektille verrattain helposti. Kehitysympäristölle oleellisena osana toimii lähdekoodin hallintaan tarkoitettu versionhallintajärjestelmä, jonka avulla voidaan kehittää uusia tai olemassa olevia komponentteja ilman että niiden ominaisuudet tai toteutuksessa havaittavat virheen aiheuttavat mahdollisimman vähän häiriöitä jo tuotantokäytössä oleviin ominaisuuksiin.

Tässä projektissa käytetään Eclipse IDE -ohjelmointiympäristöä (IDE, Integrated Development Environment), jonka sisäisiin ja liitännäiskomponenttien ominaisuuksiin perustuvat useimmat projektin toteutukset. Tämän tyyppisellä ohjelmistojen suunnitteluun käytettävällä ohjelmointityövälineet kokoavalla ympäristöllä voidaan keskittyä ohjelmiston korkean tason suunnitteluun, jolloin ohjelmiston kokonaiskuvan hahmottaminen ja uusien ominaisuuksien jäsentäminen helpottuu. Järjestelmän lähdekoodi on laadittu suurimmalta osin Java -kielellä ([35]). Järjestelmän vaatiman Java -suoritusympäristön käännöstyökalut on sovitettu makroin ja apuohjelmin Eclipseen. Eclipsen asetuksiin on laadittu myös tyyliohjeisto, jonka avulla suuren

kehittäjäryhmän tuottama ohjelmakoodi saadaan ohjattua yhtenäiseen ja selkeälukuiseen muotoon tallennettavaksi versionhallintaan. Vaadin -sovelluskehyksellä toteutetaan suurin osa käyttäjän kokemista käyttöliittymän ominaisuuksista, ja se on myös Java -kielellä toteutettu. Versionhallintajärjestelmänä käytetään Git -ohjelmistoa, joka yleisyytensä johdosta on tarjolla useille erilaisille käyttöympäristöille ja on siten myös integroitavissa useaan eri tyyppiseen IDE -ympäristöön, myös tässä projektissa käytettävään Eclipseen. Sille on myös useita avoimen lähdekoodin kuin myös kaupallisiakin asiakas- ja palvelinohjelmistoja. Git mahdollistaa laajojen, hajautettujen ja monihaaraisten sovelluskokonaisuuksien muutostenhallinnan. Sen avulla on mahdollista selvittää uusien versioiden julkaisussa mahdollisesti esiintyviä ristiriitaisuuksia komponenttien sisäisissä toteutuksissa että niiden välisissä rajapinnoissa.

5.2 Yleistä tehtäväsäiliöistä

Opetuksessa voi tulla esiin useita tilanteita, joissa yksittäiselle opiskelijalle tarjotaan ratkaistavaksi erilaisia tai yksilöityjä tehtäviä kuin mitä kurssilla yleisesti käytetään. Tehtäväsäiliö sisältää ennalta määritellyn kokoelman tehtäviä, joita esitetään yksittäiselle opiskelijalle. Tehtäväsäiliöitä on käytettävissä kahta erilaista tyyppiä, jaettu ja järjestetty. Jaetussa tehtäväsäiliössä on joukko tehtäviä, joita esitetään satunnaisessa järjestyksessä, kunnes vaadittu määrä suorituksia on tehty. Järjestetty tehtäväsäiliö puolestaan sisältää joukon tehtäviä, jotka suoritetaan säiliötä laadittaessa määrätyssä järjestyksessä, ja joista kunkin suorittamiseksi edellytetään edellisen tehtävän onnistunutta suorittamista. Tehtäväsäiliöiden sisältöä ei ole rajoitettu varsinaisesti millään tavoin, vaan se voi sisältää opiskelijakohtaisesti mitä tahansa ViLLE:ssä käytettävissä olevista tehtävistä, eli ne voivat olla opiskelijan tarpeista riippuen joko oikein/väärin-, monivalinta- tai laskutehtäviä.

Edellä mainittujen säiliöiden lisäksi toteutuksen kohteena on adaptiivista testausmenetelmää soveltava tehtäväsäiliö, joka muistuttaa toiminnaltaan osittain jaettua tehtäväsäiliötä jonka tehtäviä esitetään satunnaisesti määrätystä matematiikan opetuksen

solmukohtaa vastaavasta joukosta. Järjestettyä tehtäväsäiliötä adaptiivinen säiliö vastaa siten, että siihen sisällytetään määrättyä matematiikan solmukohtaa vastaavia tehtäviä ja joita esitetään laaditussa järjestyksessä. Adaptiivinen tehtäväsäiliö hyödyntää myös aikaisemmin opiskelijan suorituksista tallennettua matemaattisen suorituskyvyn ja virheellisten suoritusten tilastollista tietoa. Käytännössä adaptiivisen tehtäväsäiliön toiminta vastaa muuten muita tehtäväsäiliötyyppejä, mutta sen avulla voidaan toimintaa automatisoida osittain opiskelijan tehtäväsäiliökohtaisten onnistumis- ja epäonnistumistapahtumien mukaan lasketulla painoarvotuksella. Painoarvotus voidaan tosin toteuttaa vakiotermisesti, jolloin onnistumistodennäköisyyden ollessa ääriarvoissaan 0 tai 1 tiedetään kaikkien tehtävien onnistuvan tai epäonnistuvan opiskelijalle tyypillisellä tavalla.

Muissa tehtäväsäiliötyypeissä voidaan esittää opiskelijalle suoritettavaksi jokin sen sisältämistä tehtävistä. Adaptiivisessa tehtäväsäiliössä on jokaiselle opiskelijalle määritelty oletuslähtötaso – jonka muodostamiseen voidaan käyttää myös matematiikan lähtötasoarviointitaulua – ja oletuspainoarvo, joka määrittää muutosaskeleen suuruuden tehtävän palautuksen yhteydessä. Adaptiivinen proseduuri on tällöin melko suoraviivainen, koska se esittää opiskelijalle niitä tehtäviä jotka ovat onnistumistodennäköisyydeltään pienempiä kuin tehtävän suorituksen aikainen oletustaso on.

5.3 Adaptiivinen tehtäväsäiliö

Adaptiivinen tehtäväsäiliö on perustoiminnallisuuksiltaan jaettua ja järjestettyä tehtäväsäiliötä muistuttava. Erona adaptiivisessa testauksessa edellä mainittuihin tehtäväsäiliötyyppeihin on, että adaptiivinen tehtäväsäiliö hyödyntää matematiikan suorituskyky- ja virhesuoritusaineistoja. Ottaen huomioon, että kumulatiivista suoritustapahtuma-aineistoa on tätä kirjoitettaessa miljoonia yksittäisiä tapahtumia ja siksi on selvää, että aineiston lukeminen tietokannasta ja laskentaprosessointi vaatii verrattain runsaasti laskenta-, tiedostojärjestelmä- että tietokantaresursseja etenkin jos opiskelijan lähtöarvot määritellään aikaisempien kurssien opiskelijoiden tulosprofiilien

ja matematiikan solmukohtien perusteella. Adaptiivisen tehtäväsäiliön tarkoituksena on tarjota opiskelijalle hänen oppimistapojaan sekä suoritustasoaan vastaavia tehtäviä aikaisemmin kuvatun prosessin mukaisesti. Opiskelijan suorittaessa tehtäväsäiliön muuttuvat tehtävät normaalissa järjestyksessä, ja virheellisesti suoritettujen kanssa samaan kategoriaan kuuluvat tehtävät kierroksen lopuksi, kirjataan tieto kaikista suoritustapahtumista suoritustietokantaan.

On syytä huomioida, että suoritinaikaa adaptiiviseen tehtäväsäiliöön liittyvien algoritmin sisäisten laskentatapahtumien suorittamiseen ei ole käytettävissä mielivaltaista määrää, vaan prosessille käytettävissä oleva aika saa olla suurimmillaan vain muutamia sekunteja. Pitemmät käsittelyajat aikaansaavat käyttäjäkokemuksen heikkenemisen ympäristössä jossa tehtävät suoritetaan muutoinkin rajallisen ajan puitteissa, jolloin käyttäjän ei ole syytä kokea järjestelmän toiminnan tarpeetonta hidastumista. Tehtäväsäilön metodeilla käsitellään järjestelemätöntä joukkoa tehtäviä, johon poimitaan tehtäviä kategorioista. Kun johonkin kategoriaan kuuluvat tehtävät on ratkaistu, pyrkii tehtäväsäiliö painottamaan ratkaisemattomien tai virheellisesti ratkaistujen tehtävien joukkoa lisätehtävinä kierrosta suorittavalle opiskelijalle.

5.4 Adaptiivisen tehtäväsäiliön oliomalli

Adaptiivinen tehtäväsäiliö (Kaavio 2) pohjautuu järjestetyn tehtäväsäiliön (Kaavio 3) toteutukseen, ja se käsittää neljä pääluokkaa:

- Tilaobjekti tallentaa ja pitää yllä tietoa kulloinkin meneillään olevan tehtäväsäiliön sisältämien tehtävien suorituksen aikaisesta tilasta objektin yksityisiin muuttujiin. Objekti vastaa MVC -arkkitehtuurissa näkymän käyttämää tilatietojoukkoa, toisin sanoen näkymä-malli interaktiota. Objekti sisältää myös tietokannan käsittelymetodiikan, joilla tietokantaan luotua relaatiorakennetta käsitellään sekä luku- että kirjoitusoperaatioin.
- Dataobjekti ylläpitää ajantasaista tehtäväsuorituksen tilanteeseen liittyvää pisteytys- ja tulosaineistoa. Se vastaa MVC -arkkitehtuurissa mallia, sisältäen datan käsittelymetodiikan. Tieto on tallennettuna instantioidun objektin

ajonaikaiseen muuttujatilaan.

- Käyttöliittymäobjekti hallinnoi käyttäjän toimia tämän palautettua tehtävän, ohjaa tilatietoa minkä perusteella sallitaan tai estetään tehtäväkierroksen jatkaminen tulosten perusteella, sekä käsittelee kierroksen tehtävien seuraavan suorituskerran onnistumisen todennäköisyysarvoa. Tämä objekti vastaa MVC -arkkitehtuurissa käyttäjärajapintaa ja näkymää joka esittää mallin mukaisia tehtäväaineistoja ja ohjaa niiden palautusta sekä palautuksen tilainformaatiota aliluokan avulla. Suunnittelumalliltaan tämä komponentti on käyttäjärajapintanäkymä.
- Muotoiluobjekti käsittelee tehtäväsäiliön aineiston tallentamiseen sopivaa muotoon ja käsittelee tehtäväinstanssin muuttujien arvosisältöä. Tämä objekti vastaa mallia käyttävää käsittelijäobjektia MVC -arkkitehtuurissa, joka määrittelee kierroksen sisältämän säiliön mallia. Objekti sisältää aliluokan, joka ohjaa kierroksen tehtäväsuorituksen ajankohtaista tilanteen seurantaa.

Keskeisenä objektina adaptiivisen tehtäväsäiliön mallissa on dataobjekti, jonka toteuttavat metodit toteuttavat objektien ja tietokannan välisen kommunikaation. Tämän lisäksi objektit kommunikoivat keskenään metodein, jotka määrittävät niiden suhteen osoitustavan (1-to-1, 1-to-many) tai tilan (0, 1).

Käyttöliittymäobjekti on samankaltainen järjestetyn tehtäväsäiliön rakenteeseen verrattuna, muotoiluobjektin ollessa myös käsitteellisesti samankaltainen järjestetyn tehtäväsäiliön muotoiluobjektiin nähden. Tilaobjekti käsittelee käyttäjän tasopisteytyksen ja tasokohtaisen aikarajoituksen sijasta kierroksessa käytettävän tehtäväsäiliön sisältämien tehtävien onnistumistodennäköisyyksien ja tehtäväsäiliökohtaisen aikarajoitukseen liittyviä tilatietoja. Keskeisimmän dataobjektin käsittelymetodiikka poikkeaa järjestetyn tehtäväsäiliön metodiikasta lähtötason määrittelymetodien ollessa korvattuja tehtävien onnistumistodennäköisyyden pisteytysmetodeilla.

Kutsuhierarkian ja järjestelmän toiminnan kannalta on oleellista että edellä mainitut objektit konstruoidaan niitä ajettaessa oikealla tavalla. Kehitysympäristö huolehtii objektien välisten riippuvuuksien hallinnasta, ja ohjelmiston kutsurakenteessa onkin lähinnä huolehdittava että objektit on esitelty järjestelmään. Kutsuva objekti tuo kuvaajaobjektin järjestelmään sekä numeroi adaptiivisen objektin tunnisteen. Tehtävävalitsin on yleinen luokka, jonka tehtävänä on sijoitella tehtäväsäiliöön kootut tehtävät tehtäväsäiliön ajonaikaiseen tilaan. Luokan kutsurakenne huolehtii automaattisesti muotoiluobjektin suorituksen ajojärjestelmässä, ja opettajanäkymässä hyödynnetään tätä muotoiluobjektin yleistä ominaisuutta. Kehitysympäristössä voidaan myös määrätä tehtäväsäiliötyyppien näkyvyydestä vain kehittäjille, tai kehittäjien lisäksi opettajille. Luokan sisältämillä metodeilla toteutetaan myös tehtävien säiliöön valikointiin käytettävän opettajakäyttöliittymän hallinnointi.

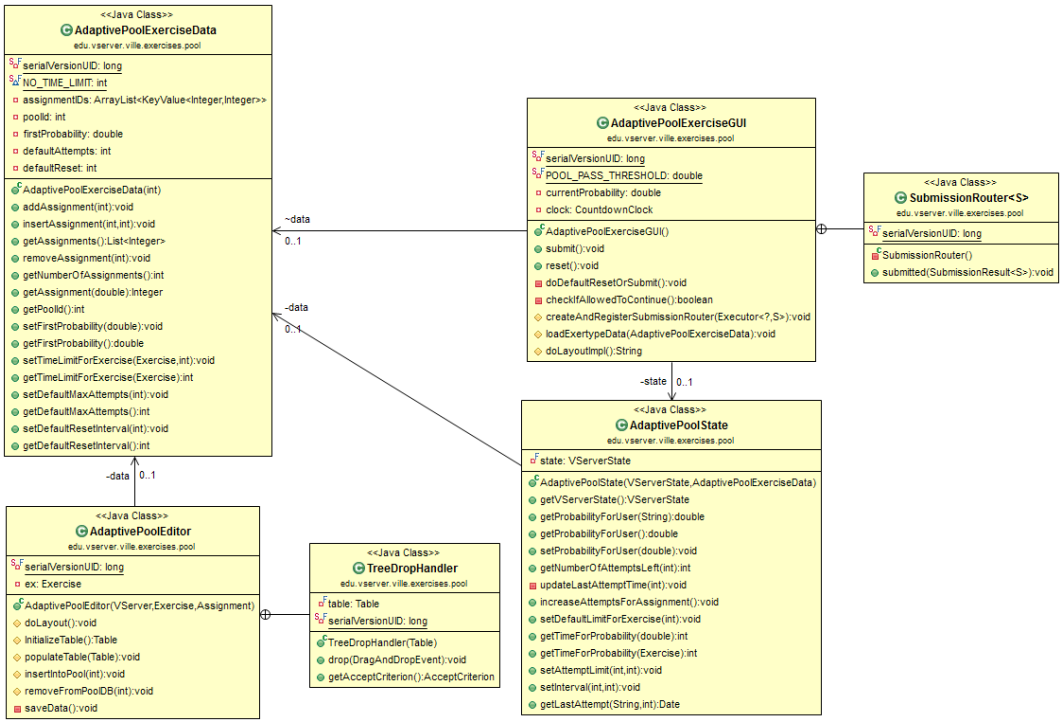
Kun tehtävät suoritetaan kierroksen aikana, niin tilatietoa tarvitaan harjoituskerrassa jäljellä olevien tehtävien listaamiseen ja joita voidaan tarjota ratkaistaviksi palautuksen jälkeen. Tehtävät jotka eivät ole oikein suoritettuja esitetään kierroksen päätyttyä uudelleen suoritettaviksi. Tätä voidaan pitää primitiivisimpänä adaptiivisen testauksen toteutusta tukevana menetelmänä, ja sen avulla saadaan opiskelija harjoittelemaan epäonnistuneiden tehtävien suorittamista mekaanisesti uudelleen. Tämä menetelmä ei sellaisenaan kuitenkaan vielä sovellu adaptiiviseksi menetelmäksi, sillä tehtäväjoukon rajaaminen ja tehtävien esittäminen tulisi olla mahdollista myös muiden kuin samalla harjoituskerralla epäonnistuneiden tehtävien joukon sisältö. Saman tyyppisiä tehtäviä voidaan ottaa suoritettavaksi väärin ymmärrettyjen ja opintojen tasoarvioinnissa tyypillisiä piirteitä osoittavien tehtävien joukosta, niin kutsuttujen solmukohtien avulla.

Matematiikan solmukohtien vaikutus lähtöarvojen koostamiseen kannasta käytetään luokan raportointia varten toteutettua menetelmäosiota. Sillä pystytään kokoamaan kurssille osallistuneiden opiskelijoiden suorituksista ne tiedot, joiden perusteella voidaan arvioida solmukohtien osaamisen taso. Kyseinen metodi ajetaan ajastetusti kerran viikossa palvelimella, ja tietojoukko muotoillaan luettavaan muotoon sekä

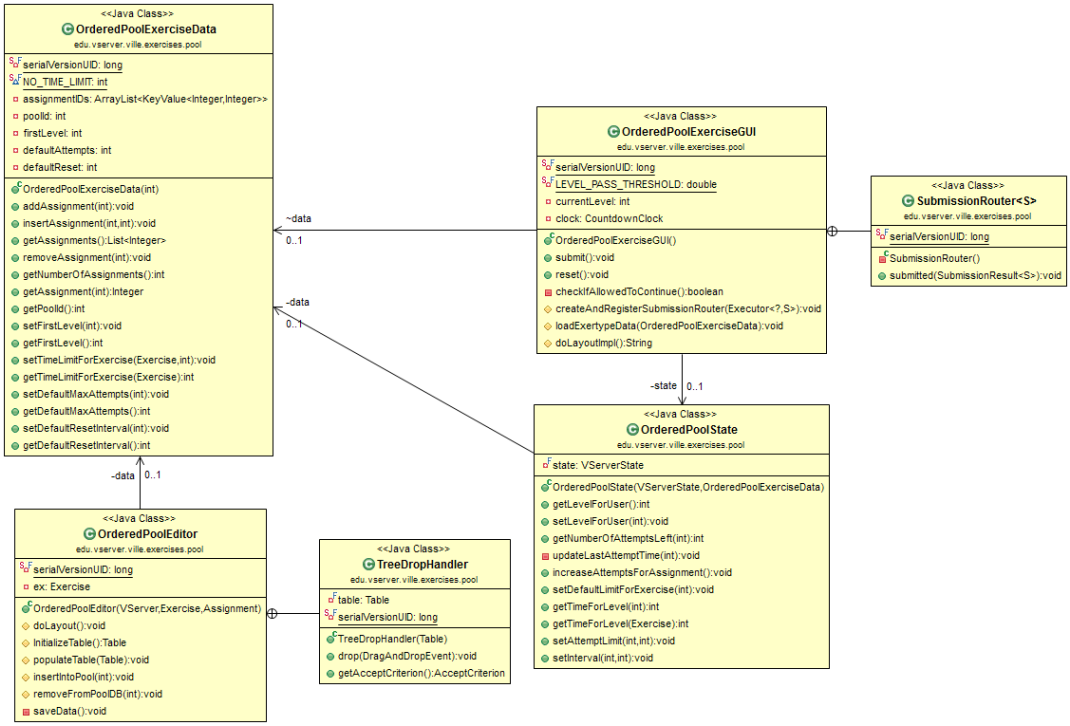
lähetetään sähköpostin välityksellä kurssin sisällöstä vastaavalle opettajalle.

Tätä metodia muokkaamalla on mahdollista saada tämä solmukohtien osaamista osoittava data myös käyttäjäkohtaisesti sekä haluttuun ajankohtaan suoritettavana. Tätä varten on olemassa edellä mainitun viikkoraporttimuotoilijan pohjalta laadittu metodi samaan luokkaan, ja joka palauttaa solmukohtien suorituksista kootun lukuvektorin sitä kutsuvalle metodille. Tämän metodin käyttötarkoitus on viikkoraporttimuokkaimesta erillinen ja sitä voidaan kutsua aina tarpeen mukaan.

Opiskelijakäyttäjille muodostetaan tietokantaan kullekin käyttäjälle ominainen tietoaineisto. Tämä tieto on kurssikohtaisena yksiulotteinen lukuvektori, joka sisältää tiedon kunkin tehtävän suorituksen onnistumistodennäköisyyksistä. Tietokantaan tallennetaan toisaalle tieto tehtävien suorituksen tilasta ja tehtävän yleisestä tapahtumista kuten suorituksen toistoista ja ajankohdista. Tieto on relaatiossa käyttäjän tunnisteeseen, ja onnistumistodennäköisyyksiä sisältävä vektori onkin yksi numeerinen komponentti lisättäväksi käyttäjän tietoihin. Tilaobjektissa on käytettävissä raportointimenetelmä, jonka avulla voidaan kurssilta saada kokonaisarvio sekä poimia kokonaisjoukosta kullekin opiskelijalle ominainen suorituskykydatan sisältävä vektori. Kuitenkin adaptiivinen tehtäväsäiliö vaatii oman tyyppisensä kyselyn, mikä luokan tällä metodilla ei suoraan onnistu. Sen vuoksi sen pohjalta voidaan laatia erillinen kysely, joka hakee yksittäisen opiskelijan suorituskykydatan ja palauttaa sen sitä kutsuvalle luokalle. Menetelmä tekee tietokantakyselyn niistä tiedoista jotka kohdistuvat käyttäjän yksilölliseen tunnisteeseen, ja palauttaa tämän datan.



Kaavio 2: Adaptiivisen tehtäväsäiliön objektimalli



Kaavio 3: Järjestetyn tehtäväsäiliön objektimalli

Relaatiomalli adaptiivisen tehtäväsäiliön käyttämälle tietokannalle koostuu poolille ominaisesta taulusta, joka sisältää tiedon käyttäjästä jota adaptiivisen valinnan vaatima todennäköisyysarvio koskee. Data jäsentyy tauluun käyttäjän ja tehtäväsäiliön tunnisteesta muodostetulla komposiittiavaimella, jonka lisäksi taulusta voidaan hakea dataa myös taululle ulkoisten indeksien kautta. Opiskelijaroolista tarkasteltaessa relaatiossa on vain yksi aspekti, joka kohdistaa käyttäjän perustiedot tehtäviä sisältävälle säiliölle. Opettajaroolista tarkasteltaessa aspekteja on edellisen lisäksi myös tehtävän omistajan tunniste, joka kohdistaa yksittäisen tehtävän sen laatijaan tai omistajaan. Todennäköisyyksien pisteytys voidaan laajentaa koskemaan myös yksittäistä tehtävää, mutta tämä vaatii erillisen taulun määrittelemistä.

Painotusarvo, jonka perusteella määräytyy todennäköisyyden muutos tilanteessa jossa opiskelija palauttaa tehtävän arvioitavaksi ja se arvioidaan onnistuneesti tai epäonnistuneesti suoritetuksi, lisätään painotusarvo todennäköisyyspisteytykseen positiivisena lukuna onnistuneen suorituksen ja negatiivisena epäonnistuneen suorituksen kohdalla. Painotusarvon muuttaminen voi olla tarpeen tilanteissa joissa epäonnistuneiden palautusten määrä ylittää vakiollisen, kokonaislukutyyppisen oletusarvon, tai tehtävissä joissa on useampi kuin yksi virhesuorituksen mahdollisuus. Toisaalta nämä eri tavoin luokitellut virhemahdollisuudet voidaan kukin sijoittaa omaan kategoriseen tehtäväsäiliöönsä, ja tällöin tehtäväkokonaisuuksia laadittaessa täytyy tämä rajoite ottaa huomioon tehtävien sijoittelussa. Painotusarvon käyttöä ei kuitenkaan edellytetä, mikäli aineiston lähtökohdaksi on käytettävissä matemaattista suorituskykydataa.

Suorituskykyaineiston sisältämillä onnistuneiden ja epäonnistuneiden suoritusten joukon avulla on mahdollista selvittää minkä tyyppiset tehtävät aiheuttavat vaikeuksia tai minkä onnistumisen todennäköisyys on oletusarvoista suurempi. Painotusarvon kokeellinen käyttäminen todennäköisyyden arvioinnissa on kuitenkin huomioitu erillisessä menetelmään liittyvässä pohdinnassa (kappale 7.3, suosittelumekanismin parametrisointi). Sen sijaan tiedon yhdisteleminen tietokannan käyttäjätietoja,

tehtäväviittauksia ja todennäköisyyksiä sisältävästä taulusta matemaattisten suorituskykytietojen kanssa edellyttää tietokannan suorituskykyaineiston hyödyntämistä määrätyn relaatiomallin mukaisesti.

5.5 Adaptiivisen tehtäväsäiliön toteutus

Adaptiivisen tehtäväsäiliön toteuttaminen yllä kuvatun objektimallin ja tietokannan relaatiorakenteen avulla melko suoraviivaista. Toteutukseen liittyy erityisenä kysymyksenä, voidaanko uniikkia opiskelijakohtaista dataa käsitellä rakenteellisesti jokaisella tehtäväkierroksen suorituskerralla. Kumulatiivisen luonteensa vuoksi data-aineiston määrä vaikuttaa suoritusaikakustannukseen. Tämän tyyppisessä järjestelmässä jossa aktiivisia opiskelijakäyttäjiä on vuodessa noin kymmenen tuhatta, kierroskohtaisen todennäköisyyden laskennan vaatima aika voi aikakustannuksen johtaa järjestelmän ruuhkautumiseen niinä ajankohtina jolloin palautuksia syntyy runsaasti. Ymmärrettävästi viiveet järjestelmässä voivat johtaa käyttäjäkokemuksen heikkenemiseen mikäli laskentaan kuluu pahimmassa tapauksessa useita sekunteja istuntoa kohti. Todennäköisyyden ja painotusarvon laskentaan käytettävä aikakompleksisuus tuleekin huomioida järjestelmän käytönsuunnittelussa. Tarvittaessa laskenta voidaan toteuttaa eräajonakin, mutta prototyyppi laskee arvot tehtävän interaktiivisen istunnon aikana.

Säiliöön sisällytettävät tehtävät valikoidaan kurssia ja sen kierroksia laadittaessa opettajanäkymän avulla ([26], kappale 2.6). Tehtäväsäiliön tyypiksi tulee valita ”Adaptive” tai ”Adaptiivinen”, käyttöliittymään valitusta kielestä riippuen. Kierroksen pisteytys määritellään tehtäväjakson pituuden tai laajuuden perusteella. Tämän jälkeen tehtäväsisältö jäsentyy tietokantaan osaksi kyseistä säiliötä, ja sieltä tarjotaan tehtäviä järjestyksessä, kunnes mahdollisuus palauttaa tulee ajankohtaiseksi opiskelijan kannalta. Kun opiskelija palauttaa tehtävät, niin epäonnistumisen ja onnistumisten mukaisesti pisteytetään tehtävien onnistumistodennäköisyys. Tämän jälkeen tarjotaan opiskelijalle suoritettaviksi niitä tehtäviä – tukiopetuksen näkökulmasta – joissa opiskelijan todennäköisyys onnistua on pienempi kuin kyseiselle tehtävälle asetettu todennäköisyys.

Adaptiivista tehtäväsäiliötä voidaan käyttää myös lisätehtäväasettelun työvälineenä.

Adaptiivisen tehtäväsäiliön ohjelmallinen toteutus tukeutuu muihin järjestelmässä käytössä oleviin tehtäväsäiliöihin, joista on relaatio useita erilaisia tehtäviä sisältäviin tietokantatauluihin. Kaikki taulut ovat saman tietokantamallin (schema) sisällä, jolloin relaatioita käsiteltäessä ei tarvita erillistä tietokantayhteyttä useisiin erillisiin malleihin. Niitä käsiteltäessä on kuitenkin muistettava että opiskelijoiden ja tehtävien tunnisteet (id) ovat uniikkeja, ja uniikkeja ovat myös niiden erilaiset yhdistelmät (composite key) indekseissä.

Esitetyssä luokkakokoelmassa keskeisimpiä metodeja ovat todennäköisyyden ja painotusarvojen muuttujasisällöt, joiden tilaa ylläpidetään muistinvaraisena opiskelijan suorittaessa adaptiivisen tehtäväsäiliön tehtäviä, ja oletusarvoista poikkeavat arvot tallennetaan tietokantaan. Tämän jälkeen opiskelijan suorittaessa tehtäviä saman tehtäväsäiliön niiden tehtävien joukosta joiden onnistumistodennäköisyys on pienempi kuin opiskelijan henkilökohtainen arvo, nämä joukon tehtävät katsotaan suoritetuiksi kun niihin annetaan oikea vastaus. Mainittujen muuttujien lisäksi voidaan tehtäväsäiliölle määrittää lisäksi aikaraja tai maksimisuoritusmäärä, joiden puitteissa on suoritettava.

5.6 Pisteytysmenetelmän kuvaus

Adaptiivisessa tehtäväsäiliössä on tarpeen selvittää aikaisempien kurssi- tai tehtäväsuoritusten pohjalta muodostettu oppimisen todennäköisyyksien vektori. Tämä voidaan koota tiedoista, jotka on tallennettu tietokantaan ja joita käsitellään matematiikan suorituskykyä mittaavan luokkakokoelman metodeilla. Tärkeimpinä pisteytyksen arvoa määrittävänä materiaalina voidaan pitää tehtävien suoritustapaa kuvaavaa tietoa sekä tietoa niihin palautetuista ratkaisuista. Matematiikan suorituskykytietoa kerättäessä huomioidaan sekä onnistuneet että epäonnistuneet tehtäväsuoritukset ja siihen vaikuttaa myös onnistumiseen päätyneen suorituksen ja sitä ennakoivien suorituksen toistojen lukumäärä. Tämän tiedon ollessa ainutlaatuinen

jokaisen opiskelijan kohdalla mutta muuten määrämuotoista, sen avulla voidaan mallintaa todennäköisyys sitä koskevan tehtävätyypin onnistumiselle. Tämän datan hyöty tulee esille siinä, ettei onnistumistodennäköisyyden laskemiselle tarvita erillistä painotusarvoa jonka suuruus vaikuttaa siihen paljonko suorituksen todennäköisyys muuttuu laskettaessa se tehtäväpalautusten yhteydessä, vaan voidaan jo käytettävissä olevan datan perusteella arvioida ennakollinen todennäköisyys.

Matematiikan suorituskykyaineisto on kohdistettu indeksoinnissa tehtävään (assig_id) ja opiskelijan henkilökohtaiseen tunnisteeseen (user_id). Näiden yhdistelmästä voidaan hakea onnistumiset (correct) ja epäonnistumiset (incorrect) käyttämällä hakukriteerinä edellä mainittuja opiskelijan ja tehtävän tunnisteita ja tuloksena saadaan vektori joka sisältää onnistumis- ja epäonnistumisaineistot. Tätä vektoria voidaan tulkita muodostettaessa onnistumistodennäköisyyttä seuraaville matemaattista suorituskykyä hyödyntäville opiskelijan suorittamille tehtäville. Lisäksi on käytettävissä myös väärinymmärrysten tietokantataulu (math_misconception) josta saadaan selville spesifinen virheen tyyppi epäonnistuneiden suoritusten tapauksessa. Tätä tietoa voidaan käyttää tasapainottavana ja tarkentavana tekijänä onnistumisten ja epäonnistumisten koostevektorin aineiston avulla tapahtuvan onnistumistodennäköisyyden laskennassa.

Pisteytysmenetelmä vaatii tässä muodossaan aineistoa tietokantaan. Toisaalta voidaan myös tarkastella kurssille osallistujia kokonaisuutena, ja arvioida kurssin keskimääräinen opiskelijoiden suorituskyvyn taso kokonaisdatan perusteella. On selvää, että tällainen mittaaminen ei suoraan kykene osoittamaan erityisiä hankaluuksia yksittäisen opiskelijan suorituksessa, mutta tästä saatu vertailutaso voidaan asettaa lähtökohdaksi mikäli kurssilla suoritettavien tehtävien osalta ei aineistoa vielä ole kyseiselle opiskelijalle olemassa.

5.7 Pisteytysohjattu tehtävävalitsin

Tehtävien esittäminen toteutetaan käyttöliittymäobjektissa, joka puolestaan järjestää tehtävät opettajanäkymän tehtäväsäiliön sisältömuokkaimessa tuotetussa järjestyksessä.

Säiliössä olevien tehtävien järjestys saattaa muuttua tehtäviä suoritettaessa, mikäli opiskelija on onnistunut tehtävissä ja opiskelijan onnistumistodennäköisyys kyseisessä tehtävässä on suurempi kuin mikä koko poolin tehtävien onnistumiselle on määritetty opiskelijan tunnistetta vastaavasti. Tällöinkin niiden järjestys on similaarinen alkuperäisen järjestyksen suorituspaikkoja vastaavissa kohdissa harjoitusta. Tehtävävalitsimessa voidaan määrittää ensimmäisen ajokerran jälkeen suoritusta rajaavia tekijöitä, kuten että käyttäjälle esitetään ne tehtävät jotka, ovat epäonnistuneet kyseisellä tai jollain aikaisemmalla tehtävien suorituskerralla. Suosittelumekanismi on tällä perusteella toimivana melko primitiivinen, mutta etenkin solmukohtien aineistoa suoritettaessa voi olla hyötyä myös ennallistamiseen käytettävästä aineistosta. Tällöin opiskelijalle ominainen profiili lasketaan opiskelijan aikaisemmin suorittamien kurssien suoritustiedoista, ja tässä yhteydessä voidaan hyödyntää myös tietoa siitä mitkä solmukohdat todennäköisesti aiheuttavat ongelmia mikäli jokin tietty yksittäinen solmukohta tai jokin erityinen solmukohtien joukko toteutuu opiskelijan kohdalla.

6 JÄRJESTELMÄN TUOTTAMAN DATAN ANALYYSI

Historiallisesta useamman vuoden aikana kootusta datasta voidaan ennallistettavien mallien avulla havaita, mihin normiryhmään opiskelija todennäköisesti sijoittuu jakson päättymisen jälkeen. Vertaamalla aikaisempien vuosien kurssiaineistoista tehtyjä havaintoja ja malliennusteita, voidaan myös testata adaptiivista menetelmää toteuttaneen kurssin aineistoja satunnaistettuun menetelmään nähden. On syytä muistaa käsiteltäessä eri opetusasteiden aineistoa, että kullekin niistä kohdistuu erityisiä edellytyksiä tehtäväaineistojen tarjontaan, kuten on todettu ([26], kappale 2.3.1, kuva 8):

> *Korkeakouluissa on todettu hyväksi sitouttaa opiskelijat kurssin suorittamiseen teettämällä heti kurssin alussa joitain tehtäviä. Tehtävät voivat olla hyvinkin yksinkertaisia, pääasia on, että opiskelijat saavat heti kurssisuorituksia.*

Adaptiivisia tehtäväkokonaisuuksia laadittaessa on myös huomioitava opiskelijoiden lähtötason asettamisen kanssa ekvivalentti onnistumistodennäköisyys. Lähtötaso voidaan tällöin käsittää edellytykseksi tehtävän suorittamiselle, ja tason määrittämiseksi voidaan käyttää perinteisiä tasokokeita. Adaptiivisessa tehtävässä sen sijaan käsitellään tasotehtäväkierroksen tai kokonaisen tasokokeen sisäisiä havaintoja, joiden perusteella opiskelijan tasotehtäväkohtaista pisteytystä muutetaan tulosten perusteella. Lähtötasona pidettävä $n/2$ todennäköisyys onnistumiselle (skaalalla $0 \ldots n$, missä $n = 1$) määrää, tarjotaanko opiskelijalle ratkaistavaksi vaikeampia tehtäviä samassa tehtäväjoukossa (tehtäväsäiliössä) joista samalla tiedetään ennalta niiden vaikeustaso, kuten myös tehtäviä joissa opiskelijalla on havaittu vaikeuksia opiskelijan suorittaessa samaan kategoriaan sisältyviä tehtäviä virheellisesti.

Lähtötason ollessa tasajakoinen onnistumisen ja epäonnistumisen välillä, tarjotaan tehtäviä säiliön oletusarvoisen järjestyksen mukaan. Opiskelijan suorittaessa tehtävän palauttamalla sen arvioitavaksi, kirjautuu myös tieto yrityskerroista kyseisessä tehtävässä. Onnistumisen todennäköisyyttä pienennetään virhesuoritusten tai kasvatetaan kun opiskelija saa ensimmäisellä suorituskerralla onnistumaan useita

kategorian tehtäviä. Arvo tai suhteellinen osuus jolla todennäköisyysarvioita kasvatetaan tai pienennetään, on määriteltävä vielä erikseen joko apurutiinissa tai suoritusistuntoa suoritettaessa, ja tallennettava opiskelijan henkilökohtaiseen suoritustietueeseen tehtäväkategorian lisäksi. Kun opiskelija suorittaa seuraavan kerran tehtäviä joihin on liitettynä todennäköisyyspisteytykselle jokin muu arvo kuin oletusarvoinen tasajako, tämä huomioidaan tarjoamalla opiskelijalle enemmän todennäköisyyteen kohdistuvaan kategoriaan liittyviä tehtäviä.

6.1 Jaetulla ja järjestetyllä tehtäväsäiliöllä toteutetut kierrokset

Jaetulla ja järjestetyllä tehtäväsäiliöllä toteutettujen kierrosten analyysissä huomioitavan sisällön muoto on homogeeninen jokaisen osallistuvan opiskelijaprofiilin kohdalla. Tehtäväkierrosten sisältö on siis vakio jokaisen kohdalla eikä lisätehtävien tarvetta analysoida automatisoidusti. Kurssin järjestänyt opettaja voi suorittaa tämän arvioinnin tehtävistä saatujen tulosten perusteella, ja kumpikin tehtäväsäiliö tarjoaa käytännössä samat mahdollisuudet arviointiin kuin mikä tahansa perinteisellä tavalla toteutettu tehtäväkierros tai koe. Tällöin koneellisesta arvioinnista saatava hyöty ei välity suoraan opettajan ja opiskelijan väliseen vuorovaikutukseen muutoin kuin palautusten analysointivaiheessa. Tällöin voidaan havaita esimerkiksi opiskelijan suorittamien toistojen määrä ja ajankohta jolloin hän määrättyjä tehtäviä suorittaa, ja opiskelijan profiilille ominaisten oppimisparametrien avulla voidaan vuorovaikutussuhteessa käytävillä keskusteluilla vaikuttaa oppimisen tuloksiin.

6.2 Adaptiivisella tehtäväsäiliöllä toteutetut kierrokset

Suorituksen onnistumisen todennäköisyyden merkitys seuraaville suoritettaville tehtäville mitataan perusteiltaan melko yksinkertaisella parametrien erotukseen perustuvalla menetelmällä. Oppimistilanteessa opiskelijan ja opettajan välinen vuorovaikutussuhde laajenee kattamaan myös ne tehtävät, jotka järjestelmä arvioi oleellisiksi opiskelijan suorituskyvyn parantamiseksi. Järjestelmä toimii tällöin välittömän palautteen ja lisätehtäväsuoritusten määräämisen osalta opettajan ja

opiskelijan vuorovaikutussuhteen laajennuksena, ja opiskelijan valmiudet ratkaista tehtävissä esitettyjä ongelmakohtia voivat parantua, kun opiskelijan suorittamassa tehtäväjoukossa painotetaan niiden tehtäväkategorioiden esiintymistä, joissa opiskelijalla on havaittu ongelmia esiintyneen. Toisaalta menetelmää voidaan hyödyntää myös kannustukseksi, tarjoamalla suoritettavaksi niitä tehtäväkategorioita joissa opiskelija on menestynyt. Perinteistä arviointimenetelmää voidaan käyttää myös tämän järjestelytavan kanssa, ja tieto opiskelijan suoritustavasta saadaan yhdistettyä hänen suorittamiensa tehtävien sekä suoriteltujen tehtävien tulosten kanssa vuorovaikutussuhdetta tukemaan. Tarkastelun kohteena on silloin tehtäväsisältöjen yksilöllistämisen tarjoaman edun vaikutus oppimiseen. Tätä tietoa voidaan käyttää seuraavia kurssi- ja kierrossisältöjä laadittaessa osana oppimisprosessista mitattavien parametrien joukkoa, ja ennallistettaessa opiskelijoiden oppimisprofiileille ominaisia mallisuoritteita.

Toteutuksessa ei suoranaisesti rajata mitään oppimisympäristössä käytettävissä olevia tehtäviä tai tehtäväjoukkoja metodiikan ulkopuolelle, mutta lähtökohtaisesti se on kehitetty käytettäväksi matematiikan ja tietotekniikan opintopolkuihin liittyen tehtäväsisältöjen kanssa ja mikäli sisällöille on käytössä automaattinen arviointimekanismi, se on mahdollista sovittaa muihinkin sisältöihin. On silti huomattava että myös matematiikan tehtävissä esiintyvien vaikeuksien tasot voivat olla tehtäväjoukoissa toisistaan voimakkaasti poikkeavia ja tällöin on huomioitava myös kyseisiä tehtäviä sisältävän poolin ominainen todennäköisyyspisteytyksien muutos onnistumisten ja epäonnistumisten tilanteissa. Todennäköisyyspisteytys ei silloin ole vakiollinen tai yhteinen kaikkien tehtäväsäiliöiden välillä eikä välttämättä myöskään saman tehtäväsäiliön sisältämien tehtävien välillä, mikäli tehtäviä siitä on suoritettu ja jonka johdosta todennäköisyys- ja painotusmuuttujien arvoja on muutettu suoritusten perusteella.

6.3 Numeerinen testaus

Tehtäväsäiliöön tulee luoda tarpeeksi paljon tehtäviä, jotta sen sisältämien tehtävien

suorittamisesta saadaan tarpeeksi numeerista suoritusaineistodataa. Mitään erityistä alarajaa tehtäväjoukon koolle ei metodin testaamiseksi voi määrätä, mutta tiedon vertailukelpoisuus tulee esille vasta kun tehtävissä päästään testaamaan tiloja joissa äärimmäiset muuttujien arvot saavutetaan, ja tilanteet joissa saavutetaan tehtäväsäiliölle asetetut aikarajat, sekä myös tilanteet joissa suoritetaan tehtäviä virheellisesti ilman että päästään onnistuneeseen tulokseen. Tämän selvityksen kannalta onkin oleellista hahmottaa proseduurin toiminta edellä mainittujen viitteellisten ehtojen mukaisesti. Tässä kappaleessa selvitetään käytettyjen pisteytysmenetelmien avulla saatuja tuloksia.

6.3.1 Kertoimellinen kokeellinen pisteytysmenetelmä

Tässä kokeellisessa menetelmässä tarkoituksena on, että jokaiselle suorituskerralla painotusarvon pienenee jokaisella tehtävän suorituskerralla riippumatta tehtävän onnistumisesta tai epäonnistumisesta. Tällöin maksimisuoritusten määrä vaikuttaa siihen kuinka monta askelta muutokseen voi sisältyä.

6.3.2 Vakiollinen kokeellinen pisteytysmenetelmä

Tässä kokeellisessa menetelmässä lähtökohtana on, että painotusarvo säilyy tehtäväsäiliökohtaisessa vakioarvossaan. Useimmille tehtäväsäiliöille tyypillinen arvo on kuitenkin järjestelmällisesti saman vakion suuruinen. Tämän pisteytysmenetelmän ongelmana voi olla paljon tehtäviä sisältävissä tehtäväsäiliöissä onnistumisen todennäköisyyden päätyminen ääriarvoihin 0 tai 1. Se ei ole toivottavaa, sillä ääriarvoinen todennäköisyys ei konkreettisessa järjestelyssä salli muutoksia ääriarvonsa ylitse ja tällöin menestyksen kehitystä ei niissä arvoissa voida osoittaa kuin huononevana tai paranevana.

6.3.3 Matemaattiseen suorituskykyaineistoon pohjautuva pisteytysmenetelmä

Tilanteessa jossa tietokantaan on tallennettuna käyttäjäkohtaista suorituskykyaineistoa, voidaan dataa kyselemällä muodostaa opiskelijakohtaisesti kuva suorituskyvyn suhteesta tulevien tehtäväsuoritusten ennakoivaan pisteytykseen.

Suorituskykyaineistossa on käytettävissä tietoa onnistuneista sekä epäonnistuneista suorituksista, ja datan formaatti on samankaltainen kaikkien niiden opiskelijoiden kesken, jotka ovat osallistuneet ViLLE:ssä suoritettavissa oleville kursseille ja joista on koottu suoritustietoja. Suorituskykyaineistosta on saatavilla myös tieto väärinymmärryksistä (math_misconception) tehtävien suorituksessa.

6.4 Oppimisvaikeuksien havainnointi

Tehtäväsäiliöiden sisällöllinen materiaali ei itsessään vielä pysty osoittamaan oppimisvaikeuksia. Sen sijaan, tehtävätulosten analyysissä voidaan osoittaa jokin yksittäinen ongelmakohta, tai kokonainen ongelma-alue sen perusteella, minkä verran opiskelijakäyttäjä on menestynyt hänelle määrätyissä tehtävissä. Tehtävien onnistumistodennäköisyydet liittyvät aina yksittäisen tehtäväsäiliön sisältöön, ja ne onkin syytä koota sellaisiksi että niiden oppimisen kannalta oleellinen sisältö voidaan täsmällisesti kategorisoida. Tämän tiedon avulla, kyseisiä tehtäviä sisältävän tehtäväsäiliön opiskelijaroolikohtainen onnistumistodennäköisyys voi toimia apumuuttujana oppimisen tuloksia analysoitaessa.

Tätä tilatietoa voidaan hyödyntää myös automatisoidussa oppimisanalytiikassa. Epäonnistumisten määrä vaikuttaa suorituksen läpimenon todennäköisyysarvoa aikaisemmissa kokeellisissa pisteytysmenetelmissä kuvatuilla tavoilla laskevasti, ja kasvattavasti kun tehtävien onnistumisten tapauksessa. Kun pisteytysmenetelmän skaala on nollasta yhteen, voidaan periaatteellisesti todeta myös että käyttäjä onnistuu kaikissa hänelle tarjotuissa tehtävissä tai vastaavasti ei saa mitään niistä suoritetuksi oikein. Tämän kaltaisten ääriarvojen esiintymistä datassa tulisi kuitenkin välttää, sillä se ei suoranaisesti pysty osoittamaan menestymisen kehitystä. Painotuksen kehittymisalgoritmia tulisi soveltaa siksi siten ettei ääriarvoja pääse syntymään, jotta jokaiselle suorituksen käyttökerralle on osoitettavissa kehityksen suunta.

6.5 Adaptiivisen tehtävävalinnan hyötynäkökohtia

Suoritettaessa tehtäviä ympäristössä jossa tehtävät valikoidaan joko ennakolta siten, että käyttäjän suoritustason kasvaessa joko lineaarisena onnistuneiden suoritusajankohtien määrän kasvuna toteutuva suoritustason kasvun indikaatio määrää tulevien suorituskertojen vaikeustason tai tehtäväsäiliöön laaditun mekanismin avulla aina opiskelijan suorittaessa tehtäviä onnistuneen suorituksen jälkeen vaikeustaso kasvaa, on osoitettavissa merkittävää etua suhteessa siihen että tehtäviä tarjottaisiin täysin sattumanvaraisessa järjestyksessä. Satunnaisessa valikoinnissa ei välttämättä saada tarjottua opiskelijalle kokonaisuutta tukevaa joukkoa tehtäviä, ja jotkin solmukohdat voivat jäädä suorituskerralla kokonaan käymättä läpi. Tämä on kuitenkin melko epätavallista, sillä satunnaisvalinnassa voidaan käyttää joukkoon sovitettua satunnaisuuden generointia joka huomioi suoritetut tehtävät.

Sama toiminnallisuus kuitenkin on järjestetyissä ja vaikeustasollisissa tehtäväsäiliöissä, ja tehtäväsäiliöiden valinta on lähinnä kehittäjien käytössä. Adaptiivista testausta käyttävä metodiikka sijoittuu vaikeustasollisen ja järjestetyn tehtäväsäiliön kanssa samaan ryhmään, sillä erotuksella että adaptiivinen tehtäväsäiliö käyttää hyväksi tietoa myös muiden matematiikan solmukohtien kuin opiskelijan sillä hetkellä suoritettavana olevan tehtävän viitteelliseen tehtäväkokoelmaan. Adaptiivinen tehtäväsäiliö soveltuu kuitenkin heikosti niiden opiskelijoiden tehtävävalikointiin, joilla ei ole suorituskykydataa tallennettuna ViLLE:n tietokantaan aikaisempien järjestelmässä suoritettujen kurssien osalta. Tällöin toiminnallisuus vastaa lähinnä vaikeustasollisen tehtäväsäiliön toimintaa. Toisaalta adaptiivinen tehtävävalikointi voi toimia hyvin siihen saakka kunnes opiskelijalla on runsaasti suorituksia ViLLE:ssä, mikä tilanne syntyy useimmissa tapauksissa silloin kun opiskelija on käynyt useita kursseja ja edennyt oppimisessaan muutenkin.

Hyötynäkökohdan painopiste on adaptiivisella menetelmällä kuitenkin opiskelun tai kurssin alkuvaiheessa jolloin opiskelijan tiedot aiheesta ovat vielä vajaat ja harjoittelua

tarvitaan runsaasti. Oppimisvaikeuksien osalta oleellisin ajankohta tämän menetelmän soveltamiseen olisi siis alkuvaiheessa ja oppimisessa esiintyvien vaikeuksien varhainen tunnistaminen hyödyttäisi opiskelijaa eniten. Yksiselitteisesti, adaptiivinen menetelmä vaatii lähtökohdaksi jotain aineistoa, sillä sen avulla ei pystytä tyhjästä tietojoukosta saamaan muuta tietoa kuin sen että opiskelijalla on yhtäläinen mahdollisuus onnistua ja epäonnistua jokaisessa niissä tehtävässä jotka säiliö sisältää mutta suorituskertojen kasvaessa epäonnistuneet tehtävät saadaan liitetyiksi niitä koskeviin solmukohtiin. Tämä onkin oletuslähtökohtana. Adaptiivinen menetelmä kuitenkin saa tiedokseen ensimmäisen tehtäväkierroksen jälkeen sen, missä tehtävissä opiskelija on onnistunut ja missä epäonnistunut sillä säiliö kirjaa jokaisen sen sisältämän tehtävän onnistumistodennäköisyyden opiskelijan kyseistä tehtävää koskevaan tietokantatauluun.

Adaptiivinen tehtäväsäiliö voi siis sisältää teoriassa rajaamattoman määrän tehtäviä, mutta käytännössä ViLLE -oppimisympäristö sekä useimmille kursseille käytettävissä oleva aika määrittelee toiminnalle rajatekijät. Adaptiivinen tehtäväsäiliö kirjaa tehtäväkohtaisesti onnistumisen todennäköisyyden aina tehtävän palautuksen yhteydessä. Mikäli yksi tehtäväkierros sisältää useampia yksittäisiä laskutehtäviä, niin yksi kierros määräytyy adaptiivisessa säiliössä yhdeksi tehtäväksi. Tämän vuoksi kannattaa laatia tehtävät siten, että kierrokset niiden sisällä ovat minimimääräiset. Muussa tapauksessa useampia kierroksia sisältävän tehtävän suorituksen yhteydessä saatetaan joutua muuttamaan tehtävämekanismia, ja tältä työltä vältytään edellä kuvatulla tavalla toimittaessa. Se kuitenkin vaatii että aina tehtävän suorituksen jälkeen sen tulos lähetetään arvioitavaksi, ja siirrytään tehtäväsäiliön seuraavaan tehtävään tai tehtäväjoukkoon. Näin saadaan kuitenkin yhtenäinen datavektori suorituskykyaineistoon ja saadaan siksi sidotuksi tehtävä haluttuun solmukohtaan.

7 POHDINTOJA

Tutkielmaa tehdessä on tullut vastaan myös sellaisia sisällön kannalta oleellisia seikkoja, joiden käsittely on täytynyt jättää osittain tai kokonaan ulkopuoliseksi. Näiden seikkojen huomiointi on kuitenkin tärkeää, etenkin jos ne tukevat tutkielman puoltamia menetelmiä. Moni kysymys saattaa tulla esiin vasta sen jälkeen kun menetelmä on otettu tuotannolliseen käyttöön kokeellisen testauksen jälkeen. Lisäksi osa alla mainituista pohdinnoista liittyvät suurimmalta osin ensimmäisen vaiheen empiirisen menetelmän muodostamiseen ja omaksumiseen.

7.1 Tekstimuotoisten aineistosisältöjen kategorisointi

Tehtäväjaksoja tai niiden yksittäisiä osioita tarkastellessa havaitaan usein kirjatun tiedon sisältävän binäärisen ja numeerisen aineiston ohella käytettävissä olevan myös tekstimuotoista sisältöä. Jotta tätä tietoa voitaisiin hyödyntää arviointiprosessissa, nykytilanteessa päädytään tekemään tekstin analyysi käsin ja validoimaan sisältö opettajan subjektiivisesta näkökulmasta. Tietoa voitaisiin myös käsitellä osittain automaattisesti, esimerkiksi merkitsemällä tieto kuuluvaksi johonkin tiettyyn aineistoluokkaan tai arvioimalla se sisällöllisesti arvosanan kaltaisella pisteytyksellä. On selvää että tämän kaltainen analyysi vie aikaa ja mikäli se voitaisiin toteuttaa automatisoidusti ja täten säästää aikaa itse tehtäväsisällön laadintaan, kannattaisi se tehdä sillä aineistoja ja tehtäväluokkia on lukuisia.

Tarkemmin esittäen, yhteistyövälineessä toteutuneiden keskustelusisältöjen kvalitatiivisessa analyysissä voidaan sanoa erityispiirteenä osallistujien ongelmien kokemisen painottuvan joko korjauskelpoiseen tai mahdottomana pidettyyn tyyppiin. Joissakin tapauksissa yksittäisten osallistujien kokemus voidaan erottaa saman vuorovaikutustapahtuman aikana kirjoitetusta keskustelusisällöstä. Keskustelijoiden havaitessa ongelmia ja kokiessaan ne ratkaisukelpoisiksi, pelko epäonnistumisesta ja sen aiheuttama oppimisen paine voi vähentyä toisin kuin niillä keskustelijoilla jotka

kokevat ongelmat mahdottomina. Keskustelusisällöstä havaittavan emootion, käyttäytymispiirteen tai suhtautumistyylin esiintymisen havainnointi vaatii syventymistä koneoppimiseen ja luonnollisen kielen ohjelmalliseen käsittelyyn ([36]). Esitellyllä perusteella automaattiseen arviointiin tarvittava tekstimuotoisen sisällön analyysi vaatisi erillistä tutkimusta, jolloin sen käsittely on syytä jättää tämän tutkielman ulkopuolelle.

7.2 Aikakustannuksen arviointi ja minimointi

On selvää, että järjestelmässä jossa palautustapahtumien määrä opiskelijaa kohti on pieni mutta aktiivisia opiskelijaprofiileja on kymmenen tuhannen luokkaa, todennäköisyyspisteytysten arviointiin tarvitaan runsaasti laskenta-aikaa. Järjestelmän kuormitus aktiivisimpina käyttöajankohtina saattaa rajoittaa käytettävissä olevaa aikaa, ja käyttäjäkokemus on otettava menetelmän suunnittelussa huomioon. Sähköisten oppimisympäristöjen tarkoituksena on sekä nopeuttaa tehtävien suoritusta että vähentää mallia noudattavien tehtävien käsin tehtävän arvioinnin määrää, ja käyttäjäkokemuksen on laadullisesti vastattava järjestelmän tarjoamaa etua.

Järjestelmän kuormitus on siksi analysoitava sekä oppimistilanteessa että myös jälkikäteen. Tehtävän aloitusajankohdan ja tehtävän suoritettavaksi tarjoamisen välinen aika saattaa tällöin muodostua mitattavaksi muuttujaksi, mutta järjestelmä voidaan toteuttaa myös eräajomenetelmällä. Tällöin opiskelijan lähtökohtaiset todennäköisyydet onnistua tai epäonnistua tehtävässä asetetaan oletusarvoihin kurssin alkaessa,ja todennäköisyyspisteytykset lasketaan eräajona ajankohtana, jolloin käyttäjiä on minimaalinen määrä kuten öisin tai viikonloppuisin. Toisaalta laskennan aikakustannus ei välttämättä ole kovinkaan merkittävä, mikäli tapahtumaan liittyy vain opiskelijakohtaisten tietueiden haku tietokannasta sekä näiden perusteella tapahtuva todennäköisyyspisteytysten laskenta.

Laskenta ei prototyyppisovelluksessa ole kovinkaan monimutkaista, vaan se koostuu pääosin peruslaskutoimituksista ja tietokantakyselyistä. Jos järjestelmää kehitetään monipuolisemmaksi siten että se huomioi yhdestä tehtävästä aikaisempien arviointien

perusteella määritellyt tyypilliset virhesuoritusten tavat ja yhdessä tehtävässä voi olla useampi kuin yksi painotusarvo virhesuoritusta kohti, niin kokonaisuutena virhesuorituksen pisteytys koostuu silloin näiden tehtävän osien virhepisteytysten summasta tai niiden monimutkaisemmasta yhdistelmästä. Tällaisessa tilanteessa opiskelijan suoritus voi kehittyä myös yksittäisen virhetyypin alaisuudesta johonkin toiseen, ja virheyksikköjen määrä voi kasvaa tai pienentyä.

7.3 Suosittelumekanismin parametrisointi

Mekanismi jolla tehtävien suorituksia arvioidaan, vaikuttaa siihen minkä tyyppisiä tehtäviä opiskelijalle suositellaan suoritettaviksi saman kierroksen lisä- tai tukiopetustehtävinä. Parametrisoinnissa on lähtökohtaisesti otettu huomioon suoritusten onnistumisen todennäköisyys, joka arvoskaalalla 0…1 puolivälissä määrittää tarkastelun lähtötason ja yli 0,5:n osoittaessa onnistumisen todennäköisyyden ja alle puolivälin osoittaessa epäonnistumisen todennäköisemmäksi. Tilanteessa, jossa välitöntä palautetta tukeva tehtävä palautetaan arvioitavaksi, todennäköisyyspisteytys asetellaan sen suuruisella muutoksella, mikä on kullekin tehtäväkategoriaa edustavalle tehtäväsäiliölle ominainen. Tällöin tehtäväsäiliötä kohti tarvitaan kaksi muuttujaa, joista toinen edustaa onnistumistodennäköisyyttä ja toinen muutoksen suuruutta.

Muitakin muuttujia voidaan ottaa käyttöön tilanteissa, joissa kategorialle on useita tyypillisiä virhesuoritustilanteita, mutta näiden arviointi koneellisesti täytyisi sisällyttää adaptiiviseen proseduuriin. Tehtävän parametrien muuttaminen vaikuttaa kuitenkin siihen, että sen toteuttaman tehtäväfunktion tuloksen ennakointi myös monimutkaistuu. Tehtävät joissa eri ratkaisuilla on erilainen painoarvo, voidaan tarjota ratkaistaviksi monivalintatehtävinä. Tällöin kuitenkin helposti rajautuvat tarkastelun ulkopuolelle ne tehtävät, joissa ratkaisu halutaan numeerisessa, kaava- tai ohjelmaproseduurimuodossa. Tästä johtuen on menetelmän laajuuden huomioon ottaen keskitytty niihin tehtäväjoukkoihin, joissa merkitsevinä parametreina ovat todennäköisyyspisteytys ja sen muutoksen suuruus palautusarvioinnissa ([37], [34]).

Tässä tutkielmassa on huomioitu todennäköisyyden pisteytysmekanismissa kaksi erityyppistä algoritmia, joiden avulla tehtävien esitykseen vaikuttavaa pisteytystä muutetaan. Sekä vakiollinen että laskeva painotusarvo voivat sopia käytettäviksi, mutta vakiollinen painotusarvo vaikuttaa todennäköisyyden ääriarvoja lähestyttäessä suhteellisesti enemmän, mikä voi johtaa arvioinnin radikaalisuuteen niiden opiskelijoiden kohdalla joiden suorituksissa esiintyy pääasiassa joko virheellisiä tai oikeita tuloksia.

Laskeva painotusarvo sen sijaan määritetään aina tehtävän palautuksen jälkeen, huolimatta siitä ovatko tehtävän edelliset suoritukset olleet viimeisimmästä suorituksesta poikkeavia vaiko saman laatuisia. Parhaiten sopiva algoritmi vaikuttaisi olevan matemaattisen suorituskykytiedon sisällön avulla koostettu tehtävätyyppikohtainen onnistumistodennäköisyys, mutta jonka kanssa on huomioitava ettei tätä suorituskykyyn tai väärintulkintoihin liittyvää aineistoa ei ole kaikilla kursseilla käytettävissä. Opetusohjelmien vaatimukset saattavat myös asettaa omia rajoituksia ja vaatimuksia algoritmien sisällölle, jolloin niiden vaikutusta voidaan ohjata tarvittaessa menestyksekkäiden opiskelijoiden lisätehtävätarkasteluun tai tukiopetusta tarvitsevien opiskelijoiden harjoitusten kautta saavutettavaan onnistumiseen.

7.3.1 Kertoimellinen kokeellinen pisteytysmenetelmä

Metodiikka ottaa kantaa muuttujien sisällön muutoksiin painotusarvojen avulla. Kun virheellisiä suorituksia on tietty määrä, muutetaan onnistumisen todennäköisyyttä $0{,}5 - 0{,}1 \rightarrow 0{,}4 - 0{,}05 \rightarrow 0{,}35 - 0{,}025 \ldots$ joka kuvataan algoritmilla:

$\{ x = x - y \mid y = y \div 2^{z-1} \mid z = z + 1, z > 0 \}$, missä

- x suorituksen sen hetkinen todennäköisyys
- y suorituksen sen hetkinen painotusarvo
- z kierroksen sen hetkinen järjestysnumero kokonaislukuna.

Algoritmi suoritetaan myös onnistuneiden tehtävävastauksien jälkeen. Tällöin

algoritmin ensimmäisen vaiheen vähennettävän luvun etumerkki muuttuu, jolloin algoritmi toimii inkrementaalisesti kyseisen tehtävän onnistumistodennäköisyyttä arvotettaessa:
$\{ x = x + y \mid y = y \div 2^{z-1} \mid z = z + 1, z > 0 \}$, missä muuttujat samat kuin edellisessä.

Tyypillisesti samaan tehtävään saadaan sekä virheellisiä että vähintään yksi onnistunut suoritus ellei maksimisuorituskertojen määrää saavuteta ja kierroksen kyseisen tehtävän osuus päättyy, jolloin siirrytään seuraavaan tehtävään tai päätetään harjoituskierros.

Suorituksista ei koota erillistä tapahtumavektoria, vaan tehtäväsäiliön sisältämille tehtäville määritellään kullekin opiskelijakohtainen todennäköisyysarvo. Kullekin niistä määrittyy myös painotusarvo, joka vaikuttaa seuraavien suorituskertojen aikana. Kun kaikki tehtävät on tehty, havainnollistuvat tehtäväsäiliön tilamuuttujat vektorimuotoiseksi tietuejoukoksi jota voidaan verrata keskinäisesti muiden samaa tehtäväsäiliötä suorittaneiden arviointeihin. Tietokannasta saadaan noudetuksi suorituksiin liittyvät adaptiivisessa proseduurissa käytettävät arvot lineaarisessa muodossa, jälleen tutkimuksessa hyödynnettäviksi. Painotuskerrointa voidaan tarvittaessa muuttaa myös tehtävätyyppikohtaisena.

7.3.2 Vakiollinen kokeellinen pisteytysmenetelmä

Metodiikka voidaan toteuttaa myös vakiollisella painotusarvolla tehtäväkohtaisesti laskevan painotuksen tilalla. Tällöin menetelmä vähentää tai lisää kiinteän painotuksen oletusarvon tai tehtäväsäiliökohtaisen painotusarvon onnistumistodennäköisyydestä tehtävän palauttamisen jälkeen. Menetelmässä on kuitenkin pidettävä huoli siitä, ettei onnistumistodennäköisyys saavuta ääriarvoja, koska niissä tapauksissa voidaan päätyä säiliön tehtävissä tilaan jossa joko kaikki tai ei yksikään tehtävä määräytyvät uudelleensuoritettaviksi.

Tällaisessa tapauksessa on syytä kiinnittää huomiota tehtäväkohtaiseen

maksimisuoritusten lukumäärään, jolloin ei päädytä vaihteluvälin äärimmäisiin arvoihin. Tämän kaltainen menetelmä voikin sopia lähinnä tilanteisiin, joissa opiskelija ei ole vielä suorittanut säiliön tehtäviä. Tämä algoritmi kuvautuu kaavalla:

$\{ x = x - y \mid y = w \mid z = z + 1, z > 0 \}$, missä

- x suorituksen sen hetkinen todennäköisyys
- y suorituksen sen hetkinen painotusarvo
- z kierroksen sen hetkinen järjestysnumero kokonaislukuna
- w on tehtäväsäiliökohtainen painotuksen oletusarvo.

Yllä mainittu tehtäväsuoritusten maksimimäärä ei suoraan vaikuta tähän algoritmiin, ja tällainen pisteytysmenetelmä sopiikin tilanteisiin, joissa palautusten maksimimäärä on rajoitettu. Tällaisia tilanteita ovat esimerkiksi koetilanteet (tasokokeet, lopputentit) tai muut palautukset joissa tehtävän suorittaja ei voi palata muuttamaan vastauksiaan jälkikäteen.

8 PÄÄTELMÄT

Tutkielmaa tehdessä on ollut selvää, että oppimiseen liittyvien vaikeuksien eli niin kutsuttujen solmukohtien havaitsemista edesauttaa ennakoiva opiskelijalle ominaisen suoritusjoukon – matemaattisesti määritellyn, järjestellyn vektorin – konvergenssi normiryhmään, ja on yleisesti tiedossa että ennaltaehkäisevä suorituksen epäonnistumisen todennäköisyyttä vähentävä eri muodoissa tarjottu tukiopetus edesauttaa kurssin suorittamista. Kognitiivisessa kontekstissa oppimista tukee myös onnistumisen kokemus, joka voidaan saavuttaa joko toistamalla satunnaisesti niitä tehtäviä joissa opiskelijan ongelmat korostuvat, tai adaptoimalla tarjontaa esittämällä tasotehtävistä sen aliryhmän tehtäviä joissa ongelmia erityisesti esiintyy. Laajentamalla tasotehtävien luokituksia sisältämään tarkemmin rajattuja ongelmia, voidaan tarjota hyödyllistä tietoa tutkimukseen että mahdollistetaan opettajan ja opiskelijan välisen vuorovaikutussuhteen hienosyisempi sisällön rajaaminen ja suunnitella niihin perustuen oppimisen tavoitteita.

8.1 Havaintoja ongelmakohdista

Ongelmakohtina opiskelijan työskentelyssä oppimisympäristössä on useassa tapauksessa saman tyyppiset virhetilanteet kuten perinteisissäkin tuntitehtävissä tai kokeissa. Tällaiset virheet havaitaan usein juuri palautuksen jälkeen, mikä pätee adaptiivisessakin tehtäväesitysmuodossa. Jälkimmäistä menetelmää puoltaa mahdollisuus tarjota oppitunnin tai kotitehtävien suorituksen yhteydessä välitön tukiopetuksellisten tai lisätehtävien suoritettaviksi määrääminen. Tehtävien yhteydessä on myös syytä huomioida maksimisuorituskerrat, sillä jos kierroksen palauttaminen jää kesken siitä syystä että jokin tehtävä on väärin ei ole mielekästä tarjota samaa tehtäviä rajattomia määriä, voi se vaikuttaa tehtävien suorittamisen motivaatioon.

8.2 Ongelmien ennakoinnin menetelmiä

Ongelmien ennakointiin on mahdollista käyttää ainakin tietokantakyselyihin

pohjautuvaa kyselytulosten vertaamista mallivastausjoukkoon. Vertailumenetelmällä voidaan osoittaa annettujen vastausten ja mallivastausten pisteytyksen erotusten suuruudet, jolloin voidaan havaita ongelmalliset tehtäväalueet tiedettäessä tarkasteltavien solmukohtien tyypilliset vastaukset. Toisaalta voidaan myös laskea keskiarvot kurssin osallistujien vastausvektoreiden yhteisistä kentistä, jolloin voidaan havaita kurssin keskimääräinen onnistuminen mallivastauksiin nähden. Tätä menetelmää ei kuitenkaan ole juuri selvitetty tai muutoinkaan testattu tässä tutkielmassa. Siitä saattaa olla mahdollista havaita kurssin vaikeustaso ja saatua tietoa käyttää seuraavien kurssijaksojen toteutusta suunniteltaessa.

9 VIITTAUKSET

[1] M.-K. Kortesalmi, *Junnauskoe 0-20 ja peruslaskutoimituksien automatisoituminen*. Jyväskylä: Jyväskylän yliopisto, 2008.

[2] H. Kauppinen, *”14 - 9 = 0” : matematiikan solmuja kolmasluokkalaisilla*. Kajaani : [H. Kauppinen], 2007.

[3] *Perusopetuksen opetussuunnitelman perusteet 2014*. Helsinki: Opetushallitus, 2014.

[4] Edward Dieterle, ”Multi-User Virtual Environments for Teaching and Learning”, teoksessa *Encyclopedia of Multimedia Technology and Networking, Second Edition*, Margherita Pagani, Toim. Hershey, PA, USA: IGI Global, 2009, ss. 1033–1041.

[5] N. Di Blas, A. Bucciero, L. Mainetti, ja P. Paolini, ”Multi-User Virtual Environments for Learning: Experience and Technology Design”, *IEEE Trans. Learn. Technol.*, vsk. 5, nro 4, ss. 349–365, loka 2012.

[6] H. Indzhov, P. Zhelyazkova, ja G. Totkov, ”Supporting adaptive e-Learning: an approach based on open-source learning management systems”, 2013, ss. 321–328.

[7] J. Cao, J. M. Crews, M. Lin, J. K. Burgoon, ja J. F. Nunamaker Jr., ”An Empirical Investigation of Virtual Interaction in Supporting Learning”, *SIGMIS Database*, vsk. 39, nro 3, ss. 51–68, heinä 2008.

[8] H. Ikäheimo, ”Edu.fi - Matematiikan solmukohtia”. [Verkossa]. Saatavissa: http://www.edu.fi/erityinen_tuki/materiaaleja_oppimisen_tueksi/matematiikka/solmukohdat. [Viitattu: 31-touko-2016].

[9] K. De Witte, C. Haelermans, ja N. Rogge, ”The effectiveness of a computer-assisted math learning program: Computer-assisted math learning program”, *J. Comput. Assist. Learn.*, vsk. 31, nro 4, ss. 314–329, elo 2015.

[10] C. Schulte, ACM Digital Library, Association for Computing Machinery, ja Special Interest Group on Computer Science Education, *Proceedings of the 10th Koli Calling International Conference on Computing Education Research*. New York, NY: ACM, 2010.

[11] ”CSC - Haka Identity Federation - All Services”. [Verkossa]. Saatavissa: https://www.csc.fi/-/haka-kayttajatunnistusjarjestel-1. [Viitattu: 30-touko-2016].

[12] M. Masud, ”Collaborative e-learning systems using semantic data interoperability”, *Comput. Hum. Behav.*, vsk. 61, ss. 127–135, elo 2016.

[13] C. J. Sangwin, *Computer aided assessment of mathematics*. Oxford: Oxford University Press, 2013.

[14] D. J. Weiss, ”Computerized Adaptive Testing for Effective and Efficient Measurement in Counseling and Education”, *Meas. Eval. Couns. Dev.*, vsk. 37, nro 2, ss. 70–84, 2004.

[15] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, ja A. Bouchachia, ”A survey on concept drift adaptation”, *ACM Comput. Surv.*, vsk. 46, nro 4, ss. 1–37, maalis 2014.

[16] V. Ponsoda, J. Olea, M. S. Rodriguez, ja J. Revuelta, ”The Effects of Test Difficulty Manipulation in Computerized Adaptive Testing and Self-Adapted Testing”, *Appl. Meas. Educ.*, vsk. 12, nro 2, ss. 167–184, huhti 1999.

[17] D. J. Weiss ja G. G. Kingsbury, ”APPLICATION OF COMPUTERIZED ADAPTIVE TESTING TO EDUCATIONAL PROBLEMS”, *J. Educ. Meas.*, vsk. 21, nro 4, ss. 361–375, joulu 1984.

[18] C. A. W. Glas, W. J. van der Linden, ja L. S. A. Council, *Computerized adaptive testing with item cloning*. Law School Admission Council, 2006.

[19] A. Kavcic, ”Fuzzy User Modeling for Adaptation in Educational Hypermedia”, *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.*, vsk. 34, nro 4, ss. 439–449, marras 2004.

[20] R. J. De Ayala, *The theory and practice of item response theory*. New York: Guilford Press, 2009.

[21] J. R. Finley ja A. S. Benjamin, ”Adaptive and qualitative changes in encoding strategy with experience: Evidence from the test-expectancy paradigm.”, *J. Exp. Psychol. Learn. Mem. Cogn.*, vsk. 38, nro 3, ss. 632–652, 2012.

[22] E. Lokkila, E. Kurvinen, E. Kaila, ja M. J. Laakso, ”AUTOMATIC RECOGNITION OF STUDENT MISCONCEPTIONS IN PRIMARY SCHOOL MATHEMATICS”, teoksessa *EDULEARN15 Proceedings*, Barcelona, Spain, 2015, ss. 2267–2273.
[23] E. Kaila, T. Rajala, M.-J. Laakso, R. Lindén, E. Kurvinen, V. Karavirta, ja T. Salakoski, ”Comparing student performance between traditional and technologically enhanced programming course”, teoksessa *Proceedings of the Seventeenth Australasian Computing Education Conference (ACE2015), Sidney, Australia*, 2015, ss. 147–154.
[24] T. Rajala, E. Kaila, R. Lindén, E. Kurvinen, E. Lokkila, M.-J. Laakso, ja T. Salakoski, ”Automatically assessed electronic exams in programming courses”, 2016, ss. 1–8.
[25] ”Henkilötietolaki 523/1999 - Ajantasainen lainsäädäntö - FINLEX ®”. [Verkossa]. Saatavissa: https://www.finlex.fi/fi/laki/ajantasa/1999/19990523. [Viitattu: 28-syys-2015].
[26] ”ViLLE - Dokumentaatio”. [Verkossa]. Saatavissa: http://villeteam.fi/index.php/fi/tuki/dokumentaatio/opettajan-kirja. [Viitattu: 16-syys-2015].
[27] S. Willman, R. Lindén, E. Kaila, T. Rajala, M.-J. Laakso, ja T. Salakoski, ”On study habits on an introductory course on programming”, *Comput. Sci. Educ.*, ss. 1–16, elo 2015.
[28] N. Kupp ja Y. Makris, ”Applying the Model-View-Controller Paradigm to Adaptive Test”, *IEEE Des. Test Comput.*, vsk. 29, nro 1, ss. 28–35, helmi 2012.
[29] R. M. Furr ja V. R. Bacharach, *Psychometrics: an introduction*. Los Angeles: Sage Publications, 2008.
[30] L. Garcia-Marques, L. D. Nunes, P. Marques, P. Carneiro, ja Y. Weinstein, ”Adapting to test structure: Letting testing teach what to learn”, *Memory*, vsk. 23, nro 3, ss. 365–380, huhti 2015.
[31] E. Gamma, Toim., *Design patterns: elements of reusable object-oriented software*. Reading, Mass: Addison-Wesley, 1995.
[32] R. Cortez ja A. Vazhenin, ”Virtual model-view-controller design pattern: Extended MVC for service-oriented architecture: Virtual Model-View-Controller Design Pattern”, *IEEJ Trans. Electr. Electron. Eng.*, vsk. 10, nro 4, ss. 411–422, heinä 2015.
[33] ”Overview · Vaadin”. [Verkossa]. Saatavissa: https://vaadin.com/docs/-/part/framework/introduction/intro-overview.html. [Viitattu: 28-touko-2016].
[34] J. Bobadilla, A. Hernando, ja A. Arroyo, ”e-learning experience using recommender systems”, 2011, s. 477.
[35] ”The Java® Language Specification - jls8.pdf”. [Verkossa]. Saatavissa: https://docs.oracle.com/javase/specs/jls/se8/jls8.pdf. [Viitattu: 23-helmi-2016].
[36] ACM SIGCHI International Conference on Supporting Group Work, SIGCHI (Group : U.S.), Microsoft Research, National Science Foundation (U.S.), ja Association for Computing Machinery, Toim., *GROUP’12 proceedings of the ACM 2012 International Conference on Support Group Work: October 27-31, 2012 : Sanibel Island, Florida, USA*. New York, NY: Association for Computing Machinery, 2012.
[37] T. Rocklin, ”Self-adapted testing: Improving performance by modifying tests instead of examinees”, *Anxiety Stress Coping*, vsk. 10, nro 1, ss. 83–104, tammi 1997.