TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

JONNE PETTERI PIHLANEN

SUOSITTELIJAJÄRJESTELMÄN RAKENTAMINEN APACHE SPARKILLA

Tiivistelmä

Tarkastaja: Timo Aaltonen

Suosittelijajärjestelmä on ohjelmisto, joka ehdottaa käyttäjälle sisältöä perustuen tämän aiempaan toimintaan. Ne ovat jatkuvasti läsnä jokapäiväisessä elämässämme ja auttavat päätöksenteossa asioissa kuten verkko-ostoksissa, suoratoistopalveluissa, sosiaalisessa mediassa tai yksinkertaisesti uutisten lukemisessa. Yksinkertaisin ja luonnollisin suosittelu tapahtuu ihmiseltä ihmiselle, mutta ihmiset voivat kuitenkin tehokkaasti suositella vain niitä asioita, jotka ovat itse henkilökohtaisesti kokeneet. Tällöin suosittelijajärjestelmistä tulee hyödyllisiä, sillä ne voivat mahdollisesti tarjota suosituksia tuhansista tai jopa miljoonista erilaisista tuotteista.

Suosittelijajärjestelmät eroavat toisistaan kohdistetun toimialan, käytetyn tiedon ja erityisesti suosittelualgoritmin perusteella, eli keinon, jolla suositukset tuotetaan. Tässä työssä keskitytään vain yhteen suositteluteknikoiden luokkkaan: *yhteisösuodatukseen*. Tätä menetelmää käytetään *Apache Spark* -sovelluskehyksessä, joka on myös yksi työn tärkeimmistä osa-alueista.

Yhteisösuodatusta käyttävät suosittelijajärjestelmät perustuvat käyttäjien yhteistyöhön. Niiden tavoitteena on tunnistaa malleja käyttäjän mielenkiinnoista voidakseen tehdä suunnattuja suosituksia. Yhteisösuodatus voidaan jakaa kahteen kategoriaan: *tuotepohjaiseen- ja käyttäjäpohjaiseen yhteisösuodatukseen*. Tuotepohjaisessa suosittelussa on tarkoituksena etsiä samankaltaisia tuotteita, sillä käyttäjän ajatellaan olevan mahdollisesti kiinnostunut samankaltaisista tuotteista myös tulevaisuudessa. Käyttäjäpohjaisessa suosittelussa käyttäjän ajatellaan olevan kiinnostunut tuotteista, joita samankaltaiset käyttäjät ovat ostaneet. Tarkoituksena on siis tunnistaa samankaltaiset käyttäjät, jotta voidaan suositella näiden ostamia tuotteita.

Työn päämääränä on tutustua Apache Spark-sovelluskehykseen, *Scala*-ohjelmointikieleen sekä Amazonin tarjoamaan *Amazon Web Services (AWS)* -pilvipalveluun ja toteuttaa näiden teknologioiden avulla suosittelujärjestelmä. Järjestelmän tarkoituksena on siis suositella elokuvia ja ohjelmiston keskiössä oleva koneoppimismalli opetetaan MovieLens sivuston tarjoamien elokuvien ja käyttäjien arvosteluiden avulla.

AWS on Amazonin tarjoama kokoelma pilvilaskentaan (cloud computing) tarkoitettuja tai siinä avustavia palveluita. Koska yksi työn kriteereistä oli Spark-sovelluksen ajaminen pilvipalvelussa, Amazonin palveluista tässä auttoivat *EMR (Elastic Map Reduce)* sekä *S3 (Simple Storage Service)*. EMR on hallittu klusterialusta, joka yksinkertaistaa big data -sovelluskehysten, kuten Apache Sparkin, käyttämistä AWS:n palveluissa. S3 on tietovarasto, joka on suunniteltu helpottamaan pilvilaskentaa ja

se tarjoaa yksinkertaisen rajapinnan tietovaraston hallintaan.

Työssä käytetty EMR-liukuhihna pystytettiin eräästä opetusvideosta löytyneiden ohjeiden mukaisesti, sillä Amazonin oma ohjeistus koettiin liian monimutkaiseksi ja epäselväksi. Pystytettävän palvelun nimi on *On-Demand Pipeline*, mutta tosiasiassa varatut resurssit pysyvät käynnissä kellon ympäri. Tätä ei varsinaisesti kommunikoitu missään vaiheessa, vaan asiaan havahtui vasta laskun saavuttua. Kuukauden mittaisesta *EC2* (m3.xlarge) -instanssien ajamisesta olisi tullut maksettavaa reilut 1000 dollaria. Tämä olisi ollut mahdollista välttää ottamalla käyttöön asetuksen, joka olisi sulkenut liukuhihnan ohjelman suorituksen jälkeen.

Apache Spark on sovelluskehys, joka mahdollistaa hajautettujen ohjelmien rakentamisen. Hajautetussa ohjelmassa suoritus voidaan jakaa useiden käsittelysolmujen kesken. Jotkin suositteluongelmat on mahdollista mallintaa hajautettuna ohjelmana, jossa kaksi matriisia, käyttäjät ja tuotteet, prosessoidaan iteratiivisella algoritmilla, joka mahdollistaa ohjelman suorittamisen rinnakkain. Jokainen Spark-sovellus koostuu driver-ohjelmasta sekä yhdestä tai useammasta executor-ohjelmasta. Driver on ohjelma, joka ajaa käyttäjän pääohjelmaa ja hajauttaa laskennan klusteriin. Executor on yksi kone klusterissa.

Spark voidaan esitellä kuvailemalla sen edeltäjää, *MapReduce*:a, ja sen tarjoamia etuja. MapReduce tarjoaa yksinkertaisen mallin ohjelmien kirjoittamiseen ja pystyy suorittamaan kirjoitettua ohjelmaa rinnakkain sadoilla tietokoneilla. MapReduce skaalautuu lähes lineaarisesti datan koon kasvaessa. Suoritusaikaa hallitaan lisäämällä lisää tietokoneita suorittamaan tehtävää. Spark säilyttää MapReduce:n lineaarisen skaalautuvuuden ja vikasietokyvyn mutta laajentaa sitä muutamalla merkittävällä tavalla. MapReducessa map- ja reduce-tehtävien väliset tulokset täytyy kirjoittaa levylle kun taas Spark kykenee välittämään tulokset suoraan liukuhihnan (pipeline) seuraavalle vaiheelle. Lisäksi Spark esittelee muistissa tapahtuvan prosessoinnin tarjoamalla abstraktion nimeltä *Resilient Distributed Dataset (RDD)*. RDD tarjoaa kehittäjälle mahdollisuuden materialisoida minkä tahansa askeleen liukuhihnassa ja tallentaa sen muistiin. Tämä tarkoittaa sitä, että tulevien askelien ei tarvitse laskea aiempia tuloksia uudelleen ja tällöin on mahdollista jatkaa juuri käyttäjän haluamasta askeleesta. Aiemmin tämänkaltaista ominaisuutta ei ole ollut saatavilla hajautetun laskennan järjestelmissä.

Apache Spark on rakennettu Scala-ohjelmointikielellä. Scala on monikäyttöinen, moniparadigmainen ohjelmointikieli, joka tarjoaa tuen funktionaaliselle ohjelmoinnille

sekä vahvan tyypityksen. Funktionaalista ohjelmointia varten Scalasta löytyy tuki funktionaalisen ohjelmoinnin käsitteille, kuten muuttumattomat (immutable) tietorakenteet ja funktiot ensimmäisen luokan kansalaisina. Olio-ohjelmointia varten Scalasta löytyy tuki käsitteille kuten luokat ja oliot. Scala tukee myös esimerkiksi kapselointia, perintää, moniperintää ja muita tärkeitä olio-ohjelmoinnin konsepteja. Scala on staattisesti tyypitetty kieli ja sillä kirjoitetut ohjelmat käännetään Scala-kääntäjää käyttäen. Scala on JVM-perustainen (Java Virtual Machine, Java-virtuaalikone) kieli, joten Scala kääntäjä kääntää sovelluksen Java-tavukoodiksi, jota voidaan suorittaa missä tahansa Java-virtuaalikoneessa. Tavukooditasolla Scala-ohjelmaa ei voida erottaa Java-sovelluksesta. Scalan ollessa JVM-perustainen, Scala on täysin yhteensopiva Javan kanssa ja näin ollen Java-kirjastoja voidaan käyttää suoraan Scala-koodissa. Tästä syystä Scala-sovellukset hyötyvät suuresta Java-koodin määrästä. Vaikka Scala tukee sekä olio- että funktionaalista ohjelmointia, funktionaalista ohjelmointia suositaan.

Saadut tulokset eivät ole palveluiden, kuten Netflix, tasolla, mutta ei sitä varmaan kannattanut odottaakaan. Mielenkiintoista oli se, kuinka "huonoja"saadut suositukset olivat. Yllättäviä ja uusia kylläkin, mutta mikään elokuvista ei kuulosta mielekkäältä. Tässä tosin voikin piillä juuri hyvän suosittelun raja, sillä luultavasti ihmisen muodostama mielipide pelkän nimen perusteella saattaa johtaa elokuvan hylkäämiseen. Ihminen ei välttämättä ole täysin objektiivinen valitsemaan sitä, onko jokin suositeltu elokuva katsomisen arvoinen. Elokuvan julkaisuvuosi, ohjaajan tunnettuus, näyttelijät ja jopa kansikuva herättävät mielipiteitä, jotka saattavat johtaa elokuvan hylkäämiseen.