hyväksymispäivä arvosana

arvostelija

Big Datan analysointi

Janne Laukkanen

Helsinki 29.3.2014

HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Laitos — Institution — Department
Matemaattis-luonnontieteellinen tiedekunta	Tietojenkäsittelytieteen laitos

Tekijä — Författare — Author
Janne Laukkanen

Työn nimi — Arbetets titel — Title

Big Datan analysointi

Oppiaine — Läroämne — Subject
Tietojenkäsittelytiede

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
	29.3.2014	15 sivua + 0 liitesivua

Tiivistelmä — Referat — Abstract

Big Datan analysointi

ACM Computing Classification System (CCS):
H.3.1 [Information Storage and Retrieval]: Content Analysis and Indexing

Avainsanat — Nyckelord — Keywords
tieto, Big, Data, Iso

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — övriga uppgifter — Additional information

Sisältö

1 Johdanto

Tämä seminaarityö käsittelee Big Datan analysointia. Big Datan varsinainen analysointi voi olla pienessä mittakaavassa hyvin samanlaista kuin minkä tahansa tiedon analysointi on perinteisesti ollut, mutta poikkeaa erityisesti tiedon saamiseen liittyvien aikarajojen, tiedon määrän ja tiedon monimuotoisuuden osalta perinteisestä tiedon analysoinnista. Nämä haasteet ovat ajaneet uusien teknologioiden kehitystä ja varsinkin hajautettu tiedon analysointi on tullut paljon aiempaa tehokkaammaksi ja helpommaksi. Big Dataa voidaan analysoida pääpiirteittäin kahdella eri tavalla: tekemällä tiedosta visualisointeja, jolloin ihmisen on helpompi hahmottaa kokonaisuuksia, sekä tekemällä koneellisia laskelmia esimerkiksi muuttujien riippuvuuksista toisiinsa. Molempia tapoja tarkastellaan tässä työssä.

Motivaatio tiedon analysoinnille on tärkeää ymmärtää, jotta voidaan kehittää toimivia ja oikeaa hyötyä antavia analyysityökaluja. Big Data on tullut voimakkaasti esille yritysmaailmassa viimeisten viiden vuoden aikana. Siihen kohdistuu kasvava mielenkiinto, sillä Big Datan analysointiin käytettävät tekniikat mahdollistavat arvokkaan tiedon kaivamisen esille ennen vaikeasti hyödynnettävästä tiedosta. Tiedon saaminen on mahdollista jopa reaaliajassa, mutta myös jälkeenpäin tehdyllä analyysillä on paljon liiketoiminnallista arvoa, jos se on tehty tarpeeksi nopeasti. Perinteisesti tiedon analysointiin on tarvittu paljon resursseja sekä ihmisinä että laskentatehona, mutta uudet teknologiat ovat mahdollistaneet valtavien tietomäärien tehokkaan läpi käymisen ja hyödyntämisen murto-osalla aiemmista resursseista. Sitä mukaan kun suuret konsulttitoimistot, kuten McKinseyCo, ovat julkaisseet raportteja Big Datasta saatavista hyödyistä [MCK11], ovat yritykset vähitellen alkaneet ymmärtää, että Big Data ei ole vain uusin yritystrendi, vaan sen avulla on mahdollista ymmärtää paljon laajempia konaisuuksia ja niin ollen tehostaa kaikkea toimintaa.

Big Dataan liittyy myös paljon tieteellisesti kiinnostavia haasteita, joihin suuryritykset ovat pyrkineet vastaamaan saavuttaakseen kilpailuetua markkinoilla. Luultavasti suurin mielenkiinto suuryrityksen näkökulmasta Big Dataan on tullut sosiaalisten medioiden kautta. Tämä johtuu siitä, että ihmisten käyttäytymistä ja aikomuksia on mahdollista ennustaa, kun tiedetään miten he ovat käyttäytyneet aiemmin [PDCE11]. Monesti Big Data mielletäänkin juuri valtavaksi tietomääräksi, jota saadaan sosiaalisten medioiden kautta. Vaikka tämä näkemys onkin väärä, on käyttäjiltä saatava tieto vaikuttanut kehityksen nopeuteen. Koska käyttäjiltä saatava tieto on yleensä rakenteetonta tietoa, on se omalta osaltaan myös edesauttanut sellaisten algoritmien kehitystä, jotka sovelutuvat rakenteettoman tiedon käsittelyyn.

Vaikeasti hyödynnettävä tieto voi olla esimerkiksi tietoa, jota on niin paljon, että sen analysointi ei ole ollut aiemmin mahdollista. Tiedon määrä ei kuitenkaan ole ainoa tiedon käsittelyä hankaloittava suure, vaan usein tiedon heterogeenisuus eli vaihtelevuus tuo myös ongelmia tiedon analysoinnin suhteen. Vaihtelevuus voi esiintyä tiedon erilaisina tallennusmenetelminä, kuten tiedostoina tai tietokantana. Monesti myös kaikki tieto voi olla tallennettuna relaatiotietokantaan, mutta koska jokaisella taululla tietokannassa on oma mallinsa, ei kaikkea tietoa ole mahdollista hakea ilman erilaisia käskyrakenteita jokaiselle halutulle tiedolle. Tiedostoihin tallennettava tieto taas voi yrityksissä tulla useammalta eri taholta, kuten kirjanpitotieto yhdeltä ja myynti- ja markkinointitieto toiselta. Näiden lukujen vertailu on usein tarpeellista, mutta tiedostoihin on molempien tahojen antama tieto tallennettu eri tavalla. Näin ollen tietoa on prosessoitava eli käsiteltävä etukäteen, ennen kuin on mahdollista analysoida koko tietomäärää.

Big Data määritellään tässä tietona, joka on sellaisessa muodossa, että se vaatii jonkinlaista käsittelyä ennen kuin sen hyödyntäminen ja erityisesti analysointi on mahdollista. Tietoa ei siis tarvitse olla paljon, jotta se täyttää määritelmän kriteerit.

Koska Big Dataa täytyy käsitellä ennen kuin sitä on mahdollista analysoida ja käsittely vaikuttaa tulevan analysoinnin mahdollisuuksiin, käydään tässä nopeasti läpi tärkeimmät esikäsittelyyn tarvittavat teknologiat. Ensimmäinen on tiedon yhtenäiseen tallentamiseen tarkoitetut tietokannat (NoSQL), joihin voidaan tallentaa suuria määriä rakenteetonta tietoa. Tälläiset tietokannat mahdollistavat eri formaatissa olevien tietojen tallentamisen, mutta tieto pitää joka tapauksessa muokata yhtenäiseen formaattiin, jotta sitä voidaan NoSQL -tietokannoista hakea. NoSQL-tietokannat ovat erikoistuneita tietokantoja, joten ennen tiedon tallentamista täytyy tarkkaan miettiä sitä, millaisia hakuja kantaan tullaan tekemään. Tämä on erityisen tärkeää tehokkaan analysoinnin kannalta, sillä ennen tiedon analysointia joudutaan tyypillisesti tekemään monia suunnitteluratkaisuja tiedon tallentamistavan suhteen.

Toinen Big Datan käsittelyyn tarkoitettu teknologia on MapReduce -algoritmi, jonka on kehittänyt alunperin Google [DG04]. MapReduce on ohjelmointimalli, joka mahdollistaa tiedon hajautetun käsittelyn pienissä osissa rinnakkain ja tulosten koostamisen vastaukseksi. MapReduce -mallin tärkein ilmentymä on Hadoop -sovelluskehys. Hadoop on järjestelmä, joka hoitaa tiedon hajautetun tallentamisen ja MapReduce -mallin mukaisen laskennan hajauttamisen niin, että tietoa analysoiva koodi on ajetan aina sillä koneella, jolla tieto fyysisesti sijaitsee. Koska MapReduce -malli ja Hadoop ovat Big Datan kannalta oleellisia teknologioita, käsitellään niitä tarkem-

min omassa luvussaan. Huomattavaa on, että MapReduce -ohjelmointimalli on rajoittunut eikä niin ollen sovellu hyvin kaikenlaiseen tiedon analysointiin.

Tämä seminaarityö jakautuu seuraaviin osiin: luvussa 2 käsitellään tarkemmin Big Datan käsittelyyn kehitettyjä teknologioita, jotka mahdollistavat tiedon analysoinnin. Luvussa 3 keskitytään tiedon analysoinnissa käytettäviin tekniikoihin ja kieliin. Luku 4 on yhteenveto.

2 Big Datan käsittely

Kuten aiemmin mainittiin, on Big Datan käsittelyyn ennen sen analysointia kehitetty useita teknologioita. Tässä esitellään niistä tärkeimmät sellaisella tasolla, että on mahdollista ymmärtää miten tiedon analysointityökalut Big Data -ympäristössä toimivat.

2.1 NoSQL

Tässä esitellään lyhyesti NoSQL -tietokantojen tarkoitus ja merkitys erityisesti Big Datan analysoinnin näkökulmasta. Kuten aiemmin mainittiin, joudutaan ennen analysointia tekemään useita päätöksiä tiedon tallennusmuodon suhteen, sillä se vaikuttaa olennaisesti analysointimahdollisuuksiin. Yksi tärkeimmistä päätöksistä tiedon tallentamisen kannalta on tietokantatyypin valinta. Eri tietokannoilla on hyvin erilaiset vaatimukset tiedon rakenteen, määrän ja formaatin suhteen, joten on olennaista valita tiedolle sopiva tietokanta.

NoSQL -akronyymi voidaan helposti ymmärtää väärin niin, että sen tarkoitus on poistaa tarve SQL-tietokannoille. On tärkeä huomata, että NoSQL tulee sanoista "Not Only SQL", eli ei pelkästään SQL. NoSQL tietokannat ovat tulleet perinteisten SQL-tietokantojen rinnalle täydentämään olemassa olevia tallennusmahdollisuuksia, eivät poistamaan niitä markkinoilta. Täydentäminen on olennaista, sillä samoin kuin MapReduce tiedon käsittelyn osalta, eivät NoSQL -tietokannatkaan sovellu hyvin kaikenlaisen tiedon tallentamiseen. NoSQL -tietokantojen tulo näin ollen myös aiheuttanut sekaannusta, kun niitä on arvosteltu ymmärtämättä niitä kunnolla. Erityisesti erilaiset NoSQL -tietokannat soveltuvat eri tarkoituksiin. Esimerkiksi paljon relaatioita sisältävän tiedon mallintaminen onnistuu niin SQL-tietokantaan, avain-arvopari NoSQL-kantaan kuin -verkkotietokantaankin. Yksittäisen tiedon hakeminen jokaisesta näistä tietokannoista onnistuu vaivatta, jos tiedetään esimerkiksi

haettavan tiedon tunniste (id).

Edellisen esimerkin tapauksessa ongelmia kuitenkin ilmenee, kun halutaan hakea tietoa, joka on monen yhteyden päässä siitä tiedon ilmentymästä, jonka tunniste on tiedossa. Perinteiset SQL-tietokannat ovat tunnetusti hitaita aina, kun joudutaan tekemään liitoksia (JOIN) muiden taulujen tietoihin. Varsinkin jos näitä liitoksia voidaan joutua tekemään suuria määriä, eivät SQL-tietokannat ole mahdollinen vaihtoehto tiedon nopeaan käsittelyyn.

Toisaalta tieto voidaan myös tallentaa NoSQL-kantaan, joka on optimoitu avain-arvoparien tallentamiseen ja hakemiseen. Näihin tietokantoihin voidaan tyypillisesti tallentaa todella suuria määriä tietoa ilman havaittavia viiveitä. Tiedon hakeminen on todella nopeaa, jos tiedetään avain, jolla tieto on tallennettu. Avain-arvo -pareille optimoituja tietokantoja ei kuitenkaan yleensä ole tehty relaatioiden näkökulmasta, vaan oletetaan, että haetujen tietojen avaimet ovat tiedossa. Jonkinlainen malli yhteyksien tallentamiselle tietokantaan voi olla olemassa, mutta jos yhteyksien suodattaminen ja tiedoista muodostuvan verkon yhteyspolkujen kulkeminen on yleistä, eivät avain-arvopari tietokannat ole tehokas ratkaisu.

Kolmas vaihtoehto, eli verkkotietokannan käyttäminen osoittautuu parhaaksi vaihtoehdoksi paljon yhteyksiä omaavan tiedon tallentamiseen. Verkkotietokannat, kuten Neo4J [Neo] ja OrientDB [Ori], ovat NoSQL -tietokantoja, jotka on erityisesti suunniteltu yhteyksien nopeaan kulkemiseen verkossa. Tallennettava tieto koostuu solmuista, jotka kuvaavat entiteettejä. Solmulla on yksilöivä tunniste, jonka avulla se on nopea löytää. Solmuille on yleensä mahdollista tallentaa avain-arvo tietoa samalla tavoin kuin edellisen tietokantaesimerkin tapauksessa. Olennaisin osa verkkotietokantoja on yhteyksien tallentaminen solmujen välille ja niiden nopea etsiminen sekä kulkeminen. Periaatteessa verkkotietokannat sopivat minkä tahansa tiedon tallentamiseen, kun tieto vain mallinnetaan tietyllä tavalla. Varsinkin kun käytetään Linkitetyn Datan [BL07] periaatteita ja mallinnetaan tieto RDF-muodossa, on verkkotietokanta luonnollisin tapa tallentamiseen. Koska koko maailman voidaan ajatella koostuvan asioista ja niiden välisistä yhteyksistä, on tiedon mallinnus tässä muodossa todella joustavaa.

Paras puoli verkkotietokannoissa kuitenkin on yhteyksien äärimmäisen yksinkertainen kulkeminen. Tiedosta voi olla saatavilla vain esimerkiksi kaukaisen sukulaisen tunniste sekä tieto yhteydestä haluttuun tietoon. Tällöin halutun tiedon hakeminen on mahdollista suodattamalla aloitussolmun yhteyksiä ja kulkemalla solmusta toiseen, kunnes haluttu solmu löytyy. Solmusta toiseen menevän yhteyden kulkem-

inen on käytännössä vakioaikaista, eikä satojen tai tuhansien kaarien kulkeminen nopeasti ole lainkaan mahdotonta. Tiedon analysoinnin kannalta verkkotietokannat ovat tärkeä tallennusmuoto varsinkin sosiaalisissa medioissa olevan tiedon suodatuksessa, sillä sosiaaliset mediat muodostavat suuria verkostoja, joiden mallintaminen on luontevaa verkkotietokannalla. Verkkotietokannoissa olevan tiedon käsittelyyn on kehitetty omia erikoistuneita kieliään, esimerkiksi Gremlin [Gre].

2.2 MapReduce

MapReduce-malli kehitettiin Googlella alunperin helpottamaan erilaisten kyselyjen tekemistä todella suureen määrään rakenteetonta tietoa. Tietomäärät olivat niin suuria, että laskenta piti hajauttaa usealle koneelle ja tehdä rinnakkain. Ongelmana kuitenkin oli, että koska ei ollut olemassa rajapintaa, joka piilottaisi laskennan ja tiedon hajauttamisen ja hoitaisi virheistä toipumiset, jokaisen erilaisen laskennan tekeminen vaati paljon vaivaa. Tämän ongelman motivoimana Google kehitti järjestelmän, joka piilottaa monimutkaisen hajautetun laskennan yksinkertaisen rajapinnan taakse nopeuttamaan laskennan ohjelmointia [DG04].

MapReduce koostuu kahdesta loogisesta osasta, Map- ja Reduce-osasta. Map-osan tarkoitus on käsitellä jotain osaa koko tietomäärästä ja antaa vastauksena avain-arvopari. Tieto, jonka Map-osa lukee voi olla täysin rakenteetonta, esimerkiksi rivi jostain tiedostosta. Ohjelma lukee rivin, analysoi sen ja palauttaa rivillä olevaan tietoon perustuvan avaimen ja vastaavan arvon. Kun Map-osa on palauttanut avain-arvoparin, hoitaa alla oleva sovelluskehys avain-arvoparien jakamisen Reduce-osille automaattisesti. Näin ollen Map-osien tulosteet menevät Reduce-osien syötteiksi.

Reduce-osat saavat syötteenä Map-osien tulostamat avain-arvoparit. Olennaista on, että saman avaimen omaavat tulosteet menevät aina samalle Reduce-osalle. Reduce-osa yhdistää siis mahdollisesti sadoilta Map-osilta saadut avain-arvoparit ja tulostaa koostetun tiedon. Yleinen esimerkki Reduce-osan hödyllisyydestä on sanalaskuri, jossa jokainen erilainen sana tiedostossa on oma avaimensa, ja samat sanat menevät näin samoille Reduce-osille. Tämän jälkeen Reduce-osa laskee sille tulevien samojen sanojen määrän yhteen ja tulostaa yhteenlasketun sanamäärän. Tällä tavoin on mahdollista laskea sanamäärä todella suuresta määrästä tietoa, aina Teratavuista Petatavuihin saakka.

Tärkeä ominaisuus MapReducessa on myös laskennan ketjuttaminen. Map -ja Reduce -osia voidaan yhdistellä toisiinsa pitkiksi jonoiksi, jolloin on mahdollista suorit-

taa tiedolle hyvinkin monimutkaista analyysiä tai muokkausta.

2.3 Hadoop

Hadoop on Apachen kehittämä avoin versio Google MapReduce-mallista. Hadoopia käytettäessä Map- ja Reduce-osat voidaan ohjelmoida Javalla omina funktioinaan, ja Hadoop tarjoaa hyvät kirjastot helpottamaan toteutusta. Myös muita kieliä voidaan käyttää, varsinkin skripti-kieliä varten on olemassa erityinen rajapinta, joka sallii minkä tahansa kielen käyttämisen. Hadoop käyttää myös omaa tiedostojärjestelmää, HDFS:ää, joka hoitaa tiedon hajauttamisen niin monelle fyysiselle koneelle kuin on tarpeen. Näin ollen analysoitava tiedosto voi olla paljon suurempi kuin yksittäisen koneen tallennuskapasiteetti.

Hadoopia on alettu käyttämään rakenteettoman tiedon hallitsemiseen ja analysointiin, koska sen avulla voidaan todella suurista määristä rakenteetonta tietoa suodattaa olennainen ja hyödyntää sitä. Hadoop ei vaadi paljon järjestelmään kuuluvilta koneilta, vaan hyvinkin heterogeeninen joukko toimii. Tieto täytyy ensin ladata HDFS-tiedostojärjestelmään, minkä jälkeen se on Hadoopin käytettävissä.

Koska Hadoopilla analysoitava tieto on useimmiten rakenteetonta tietoa, joka on tiedostoihin tallennettu, on MapReduce-ohjelmilla pystyttävä poimimaan tiedostoista ensin merkitykselliset asiat ja vasta sen jälkeen niitä voidaan analysoida. Jos tiedoston rakenne on tiedossa, tämä työ helpottuu, mutta siltikin ohjelmoija joutuu käsin etsimään jokaisesta syötteestä tarpeellisen tiedon ja luomaan siitä avain-arvoparin. Tälläinen ohjelmointi vaatii harjoittelua ja on luonteeltaan varsin erilaista kuin mihin perinteiset data-analyytikot ovat tottuneet tehdessään SQL-tyyppisiä kyselyjä. On selvää, että MapReduce-ohjelmointi asettaa näin haasteita asiantuntevien työntekijöiden löytymiselle yrityksissä. Toisaalta Hadoopin vahvuudet eivät välttämättä tule parhaiten esiin tiedon analysoinnissa, vaan rakenteettoman tiedon muokaamisessa rakenteelliseksi, jota voidaan edelleen ladata tietokantaan analysoitavaksi.

Seuraavassa esitellään Hadoopin tärkeimmät sovellusalueet. Ensimmäinen on ETL (Extract, Transform, Load) -tyyppiset tehtävät, joissa tieto ensin luetaan läpi, sitä muokataan tarpeen mukaan ja lopuksi se ladataan esimerkiksi tietokantaa. Tämän tyyppinen prosessi soveltuu MapReducelle, sillä muokkausvaiheessa yleensä on luonnollista jaoitella tietoa avain-arvo pareihin. Monimutkaiset analyysit on toinen Hadoopin sovellusalue. Jotkut analyysit vaativat, että tieto täytyy käydä läpi use-

aan kertaan, jolloin tälläisen analyysin tekeminen SQL-kyselynä voi olla vaikeaa tai mahdotonta [Sto10]. Hadoopin avulla Map -ja Reduce-prosesseista on mahdollista rakentaa pitkiä suoritusketjuja, joissa edellisen vaiheen tuottamaa muokattua tietoa käytetään seuraavassa vaiheessa.

Kolmas varteenotettava sovellusalue on puoli-rakenteellinen tieto. Puoli-rakenteellinen tieto tarkoittaa tietoa, jolla ei ole valmista skeemaa, mutta tieto on tallennettu tiedostoon esimerkiksi avain-arvo- järjestykseen. Tälläinen avaimiin perustuva tieto voi sisältää paljon erilaisia avaimia, jolloin niiden tallentaminen tietokantaan vastaisi ominaisuuden luomista jokaiselle yksikäsitteiselle avaimelle. Tämä taas vaatisi todella paljon tilaa perinteisestä SQL-tietokannasta. Jotkut sarake-pohjaiset tietokannat soveltuvat kuitenkin myös vaihtelevan määrän ominaisuuksia omaavan tiedon tallentamiseen [HBa].

2.4 Hajautetut tietokannat

Vaikka MapReduce on saanut paljon huomiota viime vuosina Big Datan käsittelyä helpottavana mallina, on tietoa hajautettu usealle koneelle jo kauan, ja erityisesti hajautettuja tietokantoja on ollut kaupallisesti saatavilla jo noin kaksi vuosikymmentä [Sto10]. On mielenkiintoista, että vaikka MapReduce ja hajautetut tietokannat tuntuvat aluksi hyvin erilaisilta, on mikä tahansa rinnakkainen laskenta mahdollista kirjoittaa joko MapReduce-tehtäväksi tai hajautetuksi tietokantakyselyksi [Sto10]. Koska hajautetut tietokannat on erityisesti viritetty nopeaan tiedon hakemiseen suurista tietomääristä, on niiden käyttö analysoinnissa ollut tärkeää. Toisaalta kaupalliset versiot hajautetuista tietokannoista ovat olleet perinteisesti niin kalliita, että vain suurimmilla toimijoilla on ollut niihin varaa. Erityisesti Internetissä liikkuvat tietomäärät ovat niin valtavia, että hajautettujen tietokantojen kustannukset kasvavat liian suuriksi jopa suuryhtiöille [ORS+08]. Koska Big Data on tuonut suuret tietomäärät kaikkien ulottuville ja pienetkin yritykset voivat luoda hyvin paljon erilaista tietoa, on hajautettujen tietokantojen potentiaalinen käyttäjämäärä kasvanut huomattavasti. Lisäksi MapReduce-malli ei suuresta huomiosta huolimatta välttämättä sovellu parhaiten suurien tietomäärien analysointiin, vaan tähän kannattaa käyttää erityisiä tietokantoja [Sto10].

Hajautettujen tietokantojen toimintaperiaate on seuraava: tietokannan taulut on hajautettu horisontaalisti siten, että tietokantataulun rivit on jaettu monelle eri koneelle. Rivien jakamiseen voidaan käyttää erilaisia strategioita, kuten hajautusfunktiota tai Round robin-mallia. Eri koneet eivät jaa mitään keskenään, vaan

jokainen tälläinen solmu on itsenäinen. Kun tietokantaan tehdään SQL-kysely, tuo kysely jaetaan osiin ja suoritetaan erikseen jokaisessa solmussa. Kun kyselyn jokin osakysely on saatu suoritettua, annetaan osakyselyn vastaus eteenpäin SHUFFLE-operaatiolle, joka hoitaa väliaikaisten vastauksien viemisen oikeille solmuille koostettavaksi. Tämä vaihe muistuttaa paljon Hadoopin ositus-vaihetta, jossa saman avaimen omaavat tulosteet viedään samoille Reduce-prosesseille.

Tietokantojen ylläpito-ohjelmisto huolehtii automaattisesti kyselyjen jakamisesta osiin ja väliaikaisten vastauksien uudelleenhajauttamisesta ja koostamisesta. Hadoop-järjestelmän tarkoitus on hyvin pitkälle sama, mutta Hadoop pystyy käyttämään analysointiin mitä tahansa tiedostoa, joka on tallennettu sen tiedostojärjestelmään. Hajautetut tietokannat taas ovat keskittyneet rakenteellisen tiedon analysointiin. Useat kaupalliset järjestelmät pyörittävät tietokantoja, joissa on lukuisia Peta-tavuja tietoa, ja järjestelmät myös skaalautuvat lineaarisesti, kun uusia solmuja lisätään. Nämä kaikki piirteet tekevät hajautetuista tietokannoista merkittävän kilpailijan MapReducelle, jonka suurimmat valtit ovat luultavasti hinta ja yksinkertaisuus.

2.5 Hybridimallit

Koska kumpikaan esitellyistä malleista ei ole sovellu hyvin kaikkiin tiedon analysoinnin tarpeisiin, ovat kaupalliset yritykset tuottaneet järjestelmiä, joissa on yhdistetty Hadoopin ja hajautettujen tietokantojen parhaat puolet. Yleensä näissä järjestelmissä annetaan rajapinta Hadoopiin, jolla voidaan suorittaa monimutkaisia analysointeja. Toinen rajapinta tarjotaan SQL-tyyppiseen tietokantaan, jolla voidaan tehdä kyselyintensiivisiä tehtäviä [Sto10].

3 Big Datan analysointi

Tässä esitellään ensin tiedon analysointia Hadoopissa helpottavia järjestelmiä ja toiseksi esitellään Amazonin tarjoama uusi vaihtoehto Big Datan analysointia varten, joka tuo paljon uusia mahdollisuuksia tehdä monimutkaista analysointia myös pienemmällä budjetilla. Redshiftin on näytetty olevan selvästi halvempi kuin vastaavan Hadoop-klusterin käyttämisen [Fuj13].

3.1 Analysointikielet

Vaikka Hadoopin tarjoama MapReduce-ohjelmointimalli on melko yksinkertainen, on useiden erilaisten kyselyjen tekeminen usein hidasta. Tämä johtuu siitä, että jokaista pientäkin kyselyä varten täytyy aina ohjelmoida oma funktionsa Map - ja Reduce-osia varten. Hadoopin MapReduce-malli voidaan myös ajatella erilaisten tietovuoiden määrittelyksi, joka poikkeaa ajatuksellisesti rakenteellisiin SQL-tietokantoihin perustuvasta hakemisesta [GNC+09].

Tämän ongelman ratkaisemiseksi on Hadoopin päälle kehitetty erilaisia ympäristöjä, joiden avulla käyttäjä voi helpommin määritellä haluamansa kyselyt ja tehtävät ja ne muutetaan alustan toimesta automaattisesti MapReduce -ohjelmaksi. Kaksi lähestymistapaa on tunnistettavissa abstraktioiden rakentamiseksi: tietovuoiden helpompi määrittely ja deklaratiivisen kielen käyttö kyselyjen tekemiselle. Näistä molemmista esitellään esimerkkitapaus: Hive on järjestelmä, joka tarjoaa SQL-kieltä muistuttavan kielen kyselyjen tekemiselle ja Pig taas on tietovuoiden määrittelyyn tarkoitettu järjestelmä.

Ylemmän tason kielet myös nostavat huomattavasti tuottavuutta. Esimerkiksi Facebook on voinut laskea MapReduce-tehtävän määrittelyyn kuluvaa aikaa tunneista ja päivistä minuutteihin ja Yahoo!:lla on saatu vastaavanlaisia tuloksia [TSA+10, ORS+08].

```
SELECT o_orderkey, o_custkey, c_custkey
FROM customer c JOIN
        orders o ON c.c_custkey = o.o_custkey JOIN
        lineitem l ON o.o_orderkey = l.l_orderkey;
```

Kuva 1: Hive esimerkki

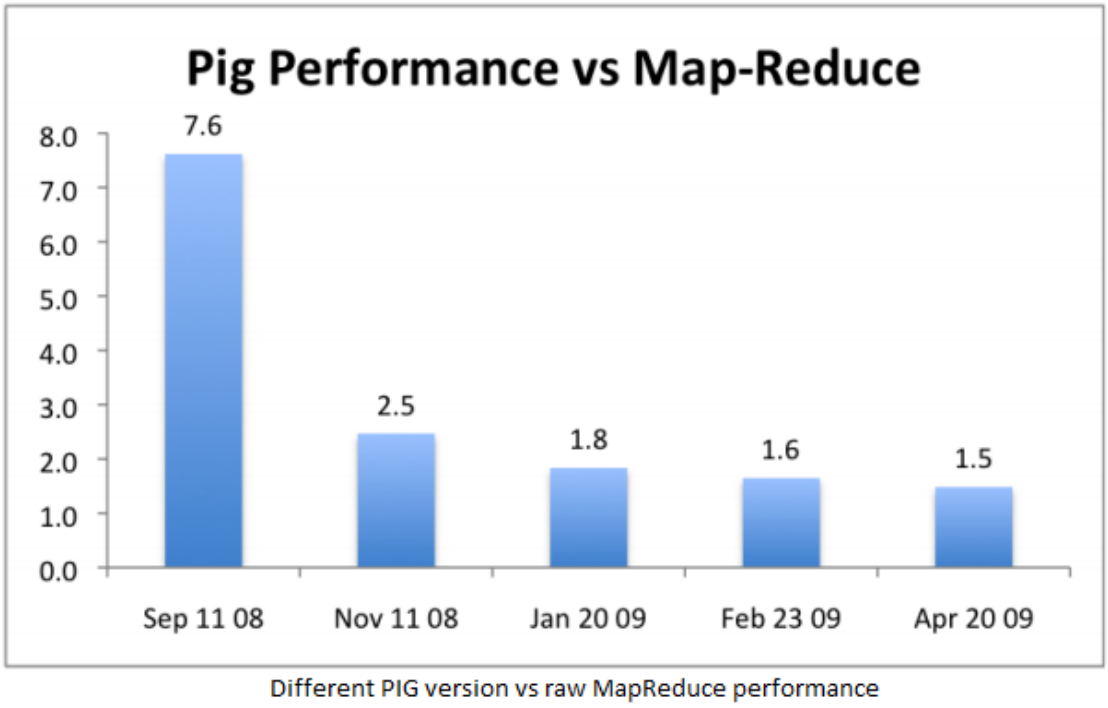
3.1.1 Hive

Hive on Facebookin kehittämä avoimen lähdekoodin järjestelmä, joka on rakennettu Hadoopin päälle [Men12]. Hiven tarkoituksena on tuoda Hadoop-ympäristöön perinteiset relaatiotietokantojen taulut ja sarakkeet, mutta säilyttää Hadoopin joustavuus. Hivessä on oma kyselykieli, HiveQL, joka muistuttaa suuresti SQL:ä. Hivessä tiedosta luodaan virtuaalitauluja, joihin Hadoopin analysoima tieto kartoitetaan. Analysoitavalle tiedolle täytyy siis antaa jonkinlainen rakenne, jotta sitä voidaan analysoida. Analysoitavien tiedostojen olisi hyvä olla puoli-rakenteellisia tiedosto-

ja, kuten csv-tiedostoja, jotta niille on helppo antaa rakenne Hive-tauluun. Kun Hivelle on annettu kartoitus tiedoston kentistä, voidaan tiedoston sisältämä tieto ladata virtuaalitauluun. Kun tieto on virtuaalitaulussa, siihen voidaan kohdistaa kyselyjä. Hive muuttaa kyselyt taustalla MapReduce-prosesseiksi automaattisesti [Sak12]. Kuvassa 1 on esimerkki HiveQL-kyselykielellä tehdystä kyselystä.

Hivessä on myös oma tapansa muokata tauluissa olevaa tietoa. Tietoa voidaan tuoda tauluihin ulkopuolisista lähteistä, mutta tauluissa olevaa tietoa ei voi muokata esimerkiksi UPDATE-käskyllä. Hive sisältää myös metatieto-osan, joka tallentaa tietoa luoduista virtuaalitauluista. Tämä ominaisuus erottaa Hiven esimerkiksi Pigistä.

Facebook käyttää Hiveä monipuolisesti tietovarastoalustassaan, joka perustuu avoimen lähdekoodin teknologioihin. Analysoitavaa tietoa tuodaan Hadoopin HDFS-tiedostojärjestelmään esimerkiksi MySQL-tietokannasta, joka toimii pääasiallisena operatiivisena tietokantana käyttäjille, sekä web-logeista. Myös monimutkaisia analysointeja ja tiedon käsittelyä vaativien peräkkäisten ajojen määrittelyn hankaluus oli yksi syy sille, että Facebook halusi kehittää ylemmän abstraktion MapReduce-tehtävien ohjelmoimiselle [TSA+10].



Kuva 2: Pig vs. MapReduce suorituskyky

3.1.2 Pig

Pig on alunperin Yahoo!:n kehittämä työkalu Hadoopilla tehtävien MapReduce-tehtävien määrittelyyn. Nykyisin Pig on Apachen kehityksen alla [Apa]. Pigissä olennaista ovat tietovuot, joiden määrittely ja hallitseminen on keskeisessä roolissa. Pig sisältää tietovuoiden määrittelykielen Pig Latinin. Pig Latinilla voidaan määritellä mistä ja miten tietoa luetaan, jakaa tietovuo useampaan rinnakkaiseen ja käsitellä jokaista vuota erikseen [Sak12].

Pig Latinin on tarkoitus osua alhaisella tasolla ohjelmoitavien MapReduce-tehtävien ja korkealla tasolla määriteltävien SQL-tyylisten kyselyjen välimaastoon [ORS+08]. Pig on tarkoitettu olemaan luonnollisempi keino ohjelmoijille määritellä MapReduce-tehtäviä kuin SQL, joka taas on tutumpi data-analyytikoille. Tämä ajattelumalli on mielenkiintoinen, sillä Facebook on ottanut hieman päinvastaisen kannan kuin Yahoo! ja sen kehittämä Hive-alusta on paljon lähempänä SQL-kieltä kuin Pig. Tämä voi kertoa jotain Facebookissa ja Yahoo!:lla työskentelevien insinöörien eroista.

Pig Latinilla kirjoitetut kyselyt muodostavat joukon askelia, jotka järjestelmä suorittaa ja muuttaa MapReduce prosessiksi. Jokainen yksittäinen askel määrittelee yhden tietoa muuttavan tapahtuman. Koska askeleiden määrittelyssä käytetään korkeamman tason käskyjä, jotka muistuttava SQL:n käskyjä, voidaan lopullisen ohjelman suorittamisessa käyttää optimointia, eikä askeleita välttämättä suoriteta siinä järjestyksessä kuin ne on kirjoitettu [ORS+08]. Pigissä on mahdollista myös määritellä skripteissä käytettäviä omia funktioita. Näistä on suuri hyöty uudelleenkäytettävyyden kannalta. Esimerkki funktiosta on tiedon parsimiseen web-logista tehty funktio, jota voidaan käyttää aina kun samantyyppisistä logeista on tarvetta hakea tietoa.

Pigillä määriteltyjen MapReduce-ohjelmien suorituskyky on jo vuosia ollut lähellä alunperin MapReduce-tehtäväksi ohjelmoidun ohjelman suorituskykyä [GNC+09]. Kuvassa 2 on esitelty Pigin suorituskykyä verrattuna MapReduceen Apache Hadoopin avoimesta julkaisusta syyskuussa 2008 vuoden 2009 huhtikuuhun. Kuvassa 3 on esimerkki Pig Latinilla kirjoitetusta kyselystä.

3.2 Redshift

Amazon on ollut useita vuosia johtava Internetin IaaS (Infrastructure as a Service) palvelujen tuottaja, eikä laskua ole näkyvissä [Ama13]. Se tarjoaa palveluja yleiskäyttöisistä pilvilaskenta-alustoista DNS-palveluihin ja NoSQL-tietokantoihin.

```
good_urls = FILTER urls BY pagerank > 0.2;
groups = GROUP good_urls BY category;
big_groups = FILTER groups BY COUNT(good_urls)>10^6;
output = FOREACH big_groups GENERATE
            category, AVG(good_urls.pagerank);
```

Kuva 3: Pig Latin esimerkki

Lisäksi palvelut on suunniteltu toimimaan yhteen niin, että esimerkiksi S3-tiedostojen tallennuspalvelusta voidaan hakea ja tuoda tietoa erilaisin Amazonin tarjoamiin tietokantoihin automaattisesti. Varsinkin palvelujen helppo skaalautuminen on houkutellut paljon Internetissä toimivia yrityksiä käyttämään Amazonia palvelujensa pohjana. Esimerkiksi yksi henkilö voi halutessaan rakentaa Amazoniin päiväksi massiivisen tiedon analysointijärjestelmän, käyttää sitä päivän ja ajaa sen alas lopuksi. Lasku tulee vain käytetyistä tunneista. Tämä on ollut aiemmin täysin mahdotonta, kun suurten palvelinympäristöjen ostaminen ja asentaminen ovat jo pelkästään vieneet viikkoja tai kuukausia, puhumattakaan hinnasta. Myös Hadoop-laskentaklusteri on Amazonin palveluissa ollut jo useita vuosia.

Amazon julkaisi uuden tietovarasto-palvelu Redshiftin vuoden 2012 joulukuussa, ja se on muuttamassa Big Datan analysointikenttää merkittävästi. Redshift on hajautettu sarakepohjainen tietokanta, jonne voidaan tallentaa todella suuria määriä tietoa, sadoista Gigatavuista aina Petatavuihin asti. Lisäksi se on paljon edullisempi kuin vastaavat hajautetut tietokannat ovat perinteisesti olleet, tarjoten näin mahdollisuuden myös pienille ja keskisuurille yrityksille analysoida Big Dataa tietokannasta.

Redshift on optimoitu tiedon analysointia varten [Ama13]. Redshift pohjautuu PostgreSQL 8.0.2:n, ja kyselyjä voidaan tehdä perinteiseen SQL-tyyliin. Sarakepohjaisuus mahdollista tiedon paljon tehokkaamman pakkaamisen, sillä samantyyppinen tietoa sijaitsee aina peräkkäin levyllä. Myös koostettujen hakujen tekeminen on paljon nopeampaa sarake-pohjaisissa tietokannoissa kuin perinteisissä rivipohjaisissa tietokannoissa. On tutkittu, että nykyiset arkkitehtuuriratkaisut hyötyvät eniten sarakaepohjaisista tietokannoista, kun tietoa halutaan analysoida nopeasti [HLAM06].

4 Yhteenveto

Tässä seminaarityössä keskityttiin tarkastelemaan Big Dataa analysoinnin näkökulmasta. Ensin esiteltiin tiedon tallennukseen ja hallinnointiin tarkoitettuja järjestelmiä, joiden avulla Big Dataa nykyisin käsitellään. Erityisesti esiteltiin NoSQL-tietokannat lyhyesti sekä paljon esillä ollut MapReduce-malli. Näiden lisäksi hajautetut tietokannat esiteltiin mahdollisuutena käyttää niitä Big Datan analysointiin. Joidenkin tutkimusten mukaan hajautettujen tietokantojen kyselyjen suorituskyky on selvästi parempi kuin Hadoopin, kun tieto on ladattu tietokantaan [Sto10]. Toisaalta on myös esitetty kritiikkiä, jonka mukaan noiden tutkimusten tekemät suorituskykytestit eivät ole kattavia eivätkä anna täysin oikeaa kuvaa Hadoopista [Bor13].

Big Datan analysointia voidaan Hadoopissa tehdä ohjelmoimalla Map ja Reduce-tehtäviä. Hadoopin oletuskieli on Java, mutta myös mitä tahansa skriptikieliä voidaan käyttää. Vaikka MapReduce-malli on hyvin yksinkertainen ja helpottaa paljon ohjelmoijien työtä tekemällä tiedon ja laskennan hajauttamisesta läpinäkyvää, on Hadoopilla myös puutteensa. Koska yksinkertaisimmatkin kyselyt joudutaan aina ohjelmoimaan tekemällä kaksi eri funktiota, jotka lukevat tiedostosta tulevan rivin, on nopeiden kyselyjen tekeminen hankalaa. Tätä ongelmaa helpottamaan ovat useat yritykset kehittäneet omia järjestelmiään, jotka helpottavat MapReduce-prosessien määrittelyä. Erityisesti kaksi eniten käytettyä ovat Pig ja Hive.

Lopuksi esiteltiin melko uusi vaihtoehto Big Datan analysoinnille, joka saattaa ainakin osittain siirtää Big Datan analysoinnin huomiota takaisin hajautettujen tietokantojen suuntaan. Aiemmat hajautettujen tietokantojen ratkaisut ovat olleet kalliita ja vaatineet paljon osaavaa ylläpitoa. Amazon on kuitenkin tuonut markkinoille ratkaisun, joka on automaattisesti hallinnoitu ja skaalautuva sarakepohjainen SQL-tietokanta. Tuotteen nimi on Redshift, ja se lupaa tallennuskapasiteettiä aina Petatavuihin saakka. Suurin ero aiempiin hajautettuihin tietokantaratkaisuihin verrattuna on hinta, joka on ainakin kymmenen kertaa edullisempi kuin perinteiset vastaavat ratkaisut. Koska Amazonilla on tuotevalikoimassaan myös Hadoop-ratkaisu, voidaan haluttaessa Hadoopia käyttää vaikka rakenteettoman tiedon esikäsittelyyn ja lataamiseen Redshiftiin, jota sitten käytetään nopeiden analyysien tekemiseen.

Lähteet

Ama13 Amazon, Capitalize on bigdata and data warehousing with amazon redshift, syyskuu 2013. URL http://www.disys.com/wp-content/uploads/2013/09/WP-Amazon-Redshift1.pdf.

Apa Apache, Pig. URL https://pig.apache.org.

BL07 Berners-Lee, T., Giant global graph, marraskuu 2007. URL http://dig.csail.mit.edu/breadcrumbs/node/215.

Bor13 Borthakur, D., Petabyte scale databases and storage systems at facebook. *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, SIGMOD '13, New York, NY, USA, 2013, ACM, sivut 1267–1268, URL http://doi.acm.org/10.1145/2463676.2463713.

DG04 Dean, J. ja Ghemawat, S., Mapreduce: Simplified data processing on large clusters. *Proceedings of the 6th Conference on Symposium on Opearting Systems Design & Implementation - Volume 6*, OSDI'04, Berkeley, CA, USA, 2004, USENIX Association, sivut 10–10, URL http://dl.acm.org/citation.cfm?id=1251254.1251264.

Fuj13 Fujikawa, K., Redshift is a disruptive price alternative in dwh, huhtikuu 2013. URL https://www.flydata.com/blog/posts/is-amazon-redshift-a-game-changer-for-big-data-analysis.

GNC+09 Gates, A. F., Natkovich, O., Chopra, S., Kamath, P., Narayanamurthy, S. M., Olston, C., Reed, B., Srinivasan, S. ja Srivastava, U., Building a high-level dataflow system on top of map-reduce: The pig experience. *Proc. VLDB Endow.*, 2,2(2009), sivut 1414–1425. URL http://dl.acm.org/citation.cfm?id=1687553.1687568.

Gre Gremlin. URL https://github.com/tinkerpop/gremlin/wiki.

HBa Hbase. URL https://hbase.apache.org/.

HLAM06 Harizopoulos, S., Liang, V., Abadi, D. J. ja Madden, S., Performance tradeoffs in read-optimized databases. *Proceedings of the 32Nd International Conference on Very Large Data Bases*, VLDB '06. VLDB En-

dowment, 2006, sivut 487–498, URL http://dl.acm.org/citation.cfm?id=1182635.1164170.

MCK11 MCKinseyCo, Big data: The next frontier for innovation, competition, and productivity, toukokuu 2011. URL http://www.mckinsey.com/insights/business_technology/big_data_the_next_frontier_for_innovation.

Men12 Menon, A., Big data @ facebook. *Proceedings of the 2012 Workshop on Management of Big Data Systems*, MBDS '12, New York, NY, USA, 2012, ACM, sivut 31–32, URL http://doi.acm.org/10.1145/2378356.2378364.

Neo Neo4j. URL http://neo4j.org.

Ori Orientdb. URL http://orientdb.org.

ORS+08 Olston, C., Reed, B., Srivastava, U., Kumar, R. ja Tomkins, A., Pig latin: A not-so-foreign language for data processing. *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, New York, NY, USA, 2008, ACM, sivut 1099–1110, URL http://doi.acm.org/10.1145/1376616.1376726.

PDCE11 Phithakkitnukoon, S., Dantu, R., Claxton, R. ja Eagle, N., Behavior-based adaptive call predictor. *ACM Trans. Auton. Adapt. Syst.*, 6,3(2011), sivut 21:1–21:28. URL http://doi.acm.org/10.1145/2019583.2019588.

Sak12 Sakr, S., Use sql-like languages for the mapreduce framework, huhtikuu 2012. URL http://www.ibm.com/developerworks/opensource/library/os-mapreducesql/os-mapreducesql-pdf.pdf.

Sto10 Stonebraker, B, maaliskuu 2010. URL http://database.cs.brown.edu/papers/stonebraker-cacm2010.pdf.

TSA+10 Thusoo, A., Shao, Z., Anthony, S., Borthakur, D., Jain, N., Sen Sarma, J., Murthy, R. ja Liu, H., Data warehousing and analytics infrastructure at facebook. *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data*, SIGMOD '10, New York, NY, USA, 2010, ACM, sivut 1013–1020, URL http://doi.acm.org/10.1145/1807167.1807278.