Otto Virtanen

FP-growth-algoritmi

Tietotekniikan kandidaatintutkielma

8. huhtikuuta 2021

Jyväskylän yliopisto

Informaatioteknologian tiedekunta

Tekijä: Otto Virtanen

Yhteystiedot: otto.v.virtanen@student.jyu.fi

Ohjaaja: Antti-Jussi Lakanen

Työn nimi: FP-growth-algoritmi

Title in English: FP-growth algorithm

Työ: Kandidaatintutkielma

Sivumäärä: 19+0

Tiivistelmä: Tässä tutkielmassa tutkitaan FP-growth-algoritmia, joka on yksi tiedonlouhinnan assosiaatio-menetelmän assosiaatiosääntöalgoritmi. Assosiaatiosääntöalgoritmi etsii usein esiintyvät alkiojoukot ja muodostaa niistä assosiaatiosäännöt. Assosiaatiosääntöjä käytetään yleisesti markkinakorianalyysissä, jossa analysoidaan asiakkaiden ostokäyttäytymistä. Lisäksi tutkielmassa verrataan FP-growth-algoritmin nopeutta Apriori-algoritmiin.

Avainsanat: CRM, Koneoppiminen Assosiaatioalgoritmi, FP-growth

Abstract: This bachelor's thesis studies FP-growth-algorithm, which is one of the association rule algorithm of data mining association method. Association rule algorithm finds the frequent itemsets and generates association rules based on those frequent itemsets. Association rules are generally used in the market basket analysis, which analyzes customer's purchasing behavior. In addition in the bachelor's thesis compares FP-growth-algorithm speed to Apriori-algorith.

Keywords: Bachelor's degree, CRM, Data mining

Kuviot

Taulukot

Sisällys

1 Johdanto

Tiedon sähköistyminen ja sen määrän kasvaminen ovat lisänneet tarvetta ottaa käyttöön erilaisia automatisoituja analysointikeinoja. Tiedosta on mahdollista analysoida muun muassa assosiaatioita, jotka kuvaavat tiedossa esiintyviä yhteyksiä. Assosiaatioita kuvaavia assosiaatiosääntöjä etsitään ja muodostetaan hyödyntämällä assosiaatiosääntöalgoritmeja. Assosiaatiosääntöjä voidaan käyttää esimerkiksi markkinakorianalyysissä, jonka avulla pyritään esimerkiksi tunnistamaan asiakkaiden ostokäyttäytymistä (Han, Kamber ja Pei 2011, s. 206–208).

Assosiaatiosääntöalgoritmien tarkoituksena on löytää suuresta datamäärästä usein esiintyvät alkiojoutkot, joiden pohjalta muodostetaan assosiaatiosäännöt. Assosiaatiosääntöalgoritmin toiminta voidaan jakaa kahteen päävaiheeseen. Ensimmäisessä vaiheessa etsitään usein esiintyvät alkiojoukot, ja toisessa vaiheessa muodostetaan itse assosiaatiosäännöt. Ensimmäinen vaihe on tutkimuksellisesti mielenkiintoinen, koska se määrittelee kuinka tehokas tai nopea algoritmi on muodostamaan assosiaatiosäännöt. (Jukić ja Nestorov 2006; Han, Kamber ja Pei 2011, s. 208–210)

Tässä tutkielmassa tutkitaan *frequent pattern growth* -algoritmia (myöhemmin lyhyesti FP-growth), joka hyödyntää binääripuutietorakennetta (myöhemmin FP-puu) usein esiintyvien alkiojoukkojen etsimisessä. FP-growth-algoritmi on verrattain tehokas löytymään usein esiintyvät alkiojoukot suuresta data määrästä verrattuna esimerkiksi Apriori-algoritmiin. (Han, Kamber ja Pei 2011, s. 218–221)

Tutkielman tavoitteena on kirjallisuuskatsauksen keinoin tarkastella FP-growth-algoritmia, ja tutkia mikä on assosiaatiosääntöalgoritmien rooli tiedonlouhinnassa.

2 Tiedonlouhinta ja assosiaatiosäännöt

Tässä luvussa esitellään mitä tiedonlouhinta tarkoittaa ja avataan tarkemmin mitä tiedonlouhinnan assosiaatio-menetelmä pitää sisällään.

2.1 Tiedonlouhinta

Tiedon etsiminen datasta on perinteisesti vaatinut manuaalista datan analysointia ja tulkintaa. Manuaalinen tiedon kerääminen on ollut hidasta, kallista ja subjektiivista. Tiedon määrän lisääntyminen ja datan sähköistyminen ovat sekä vaatineet että mahdollistaneet automaattisten tiedonetsimismenetelmien kehittämisen.

Knowledge discovery in databases (myöhemmin lyhyesti KDD) on prosessi, jonka tavoitteena on etsiä uutta ja hyödyllistä tietoa datasta. Tiedonlouhinta on KDD-prosessin vaihe, jossa tutkittavasta datasta pyritään löytämään säännöllisyydet erilaisia menetelmiä käyttämällä (Han, Kamber ja Pei 2011, s. 16–18). Tiedonlouhinnan menetelmiä ovat muun muassa assosiaatio, klusterointi ja luokittelu (Ngai, Xiu ja Chau 2009; Liao, Chu ja Hsiao 2012; Han, Kamber ja Pei 2011, s. 40).

Vaikkakin tiedonlouhinta on osa KDD-prosessia, tiedonlouhinta-termiä käytetään monesti kuvaamaan koko prosessia (Han, Kamber ja Pei 2011, s. 16-18). Siinä missä tiedonlouhinnan tavoitteena on löytää säännöllisyyksiä on, KDD-prosessin tavoitteena löytää uutta ja hyödyllistä tietoa. Erona löydettyjen säännöllisyyksien ja tiedon välillä on säännöllisyyksien kiinnostavuus. Säännöllisyys on kiinnostava, kun se ylittää tietyn kiinnostavuuskynnyksen, joka on täysin riippuvainen tiedon käyttökohteesta ja -tarkoituksesta. (Fayyad, Piatetsky-Shapiro ja Smyth 1996)

Tässä tutkielmassa keskitytään tutkimaan miten säännöllisyyksiä etsitään tiedonlouhinnan assosiaatio-menetelmän avulla, eikä oteta kantaa ovatko säännöllisyydet tietoa vai ei. Seuraavassa luvussa käsitellään tarkemmin miten säännöllisyyksiä pyritään luomaan assosiaatio-menetelmällä datassa olevien yhteyksien ja korrelaatioiden avulla.

2.2 Assosiaatiosäännöt

Assosiaatiosäännöillä tuodaan esiin yhteyksiä ja korrelaatioita annetusta datasta esimerkiksi jos asiakas on ostanut leipää, hän saattaa ostaa myös voita.

$$\text{Assosiaatiosääntö: } X \Rightarrow Y$$

Usein miten assosiaatiosääntöjä etsitään datasta, joka sisältää ostotapahtumia, joten assosiaatiosääntöjen avulla voidaan tutkia mitkä tuotteet ostetaan usein yhdessä eli toisin sanoen tehdä markkinakorianalyysi (Jukić ja Nestorov 2006). Markkinakorianalyysin tavoitteena on löytää säännöllisyyksiä, joita voidaan hyödyntää esimerkiksi lisämyynnin tai tuotesijoittelun tukena. (Jukić ja Nestorov 2006; Mitra, Pal ja Mitra 2002; Ngai, Xiu ja Chau 2009)

Assosiaatiosäännöt muodostetaan datasta assosiaatiosääntöalgoritmilla, joita on useita erilaisia, tässä tutkielmassa perehdytään FP-growth-algoritmiin sekä sivutaan Apriori-algoritmia. Assosiaatiosääntöjen muodostaminen voidaan jakaa kahteen alaongelmaan (Agrawal, Srikant ym. 1994; Jukić ja Nestorov 2006; Han, Kamber ja Pei 2011, s. 209): Ensimmäinen alaongelma on, että kuinka löydetään sellaiset usein esiintyvät alkiojoukot, jotka ovat kiinnostavia. Toinen alaongelma on, että kuinka näistä usein esiintyvistä alkiojoukoista muodostetaan varsinaiset assosiaatiosäännöt. Alaongelmista ensimmäinen on tutkimuksellisesti mielenkiintoisin, sillä se vaikuttaa merkittävimmin algoritmin tehokkuuteen (Jukić ja Nestorov 2006; Agrawal, Imieliński ja Swami 1993).

Assosiaatiosääntöön liitetään kaksi tunnuslukua: tuki (engl. support) ja luottamus (engl. confidence). Tunnusluvut kertovat onko kyseinen assosiaatiosääntö kiinnostava. Tuki kuvaa, kuinka monta kertaa assosiaatiosääntöä vastaava usein esiintyvä alkiojoukko esiintyy koko tutkittavassa datajoukossa. Luottamus puolestaan kuvaa, kuinka usein alkio X esiintyy alkion Y kanssa. Tyypillisesti assosiaatiosääntö mielletään kiinnostavaksi, kun se tyydyttää ennaltamääritellyt minimituki- ja minimiluottamusarvot. (Jukić ja Nestorov 2006; Han, Kamber ja Pei 2011, s. 208–210) Tuki ja luottamus tunnusluvut voidaan esittää joko absoluuttisina tai suhteellisina.

Otetaan esimerkiksi alla oleva assosiaatiosääntö, joka kuvaa, että pelihiiren ostaneet ostivat myös hiirimaton. Assosiaatiosäännölle on laskettu tuki 3% ja luottamus 50%. Tuki

tarkoittaa, että 3% kaikista ostotapahtumista sisälsi sekä pelihiiren että hiirimaton. Luottamus tarkoittaa, että 50% heistä jotka ostivat pelihiiren ostivat myös hiirimaton. Määritellään minimitukiarvoksi 2% ja minimiluottamusarvoksi 45%. Koska assosiaatiosäännön $pelihiiri \Rightarrow hiirimatto$ tukiarvo tyydyttää minimitukiarvon ja luottamusarvo tyydyttää minimiluottamusarvon voidaan todeta, että assosiaatiosääntö on kiinnostava.

$$\text{Pelihiiri} \Rightarrow \text{Hiirimatto } (tuki = 3\%, luottamus = 50\%)$$

Koska varsinaisten assosiaatiosääntöjen luominen on hyvin suoraviivainen prosessi, keskittyy eri algoritmit ensimmäiseen alaongelmaan eli usein esiintyvien alkiojoukkojen etsimiseen (Agrawal, Imieliński ja Swami 1993). Seuraavaksi luvussa 2.3 esitellään Apriori-algoritmi, joka käyttää luo ja testaa (engl. generate and test) -menetelmää usein esiintyvien alkiojoukkojen etsimisessä ja luvussa 3 FP-growth-algoritmi, joka puolestaan käyttää hajoita ja hallitse (engl. devide and conquer) -menetelmää (Han, Kamber ja Pei 2011, s. 218).

2.3 Apriori-argoritmi

Tässä kappaleessa esitellään lyhyesti Apriori-algoritmin toimintaa. Apriori-algorimi on assosiaatiosääntöalgoritmi, joka etsii usein esiintyvät alkiojoukot luomalla joukon ehdotettuja alkiojoukkoja, joista valitaan usein esiintyvät alkiojoukot eli algoritmi käyttää luo ja testaa -menetelmää. Algoritmin nimi on peräisin siitä, että se käyttää ennakkotietoa (engl. prior knowledge) etsiessään usein esiintyviä alkiojoukkoja. (Agrawal, Srikant ym. 1994; Han, Kamber ja Pei 2011, s. 211–215)

Algoritmi hyödyntää niin sanottua *level-wise* -hakua, jossa k-alkiojoukkoja hyödynnetään etsimään k+1-alkiojoukot. Algoritmi muodostaa ehdotettuja k-alkiojoukkoja, joista usein esiintyvät eli minimitukiarvon tyydyttävät valitaan L_k-joukkoon. L_k-joukkoa puolestaan hyödynnetään seuraavien k+1 ehdotettujen alkiojoukkojen etsimisessä. Jokaisen L_k-joukon etsiminen vaatii koko annetun datan läpikäyntiä. (Han, Kamber ja Pei 2011, s. 211–215)

Käytetään esimerkkinä taulukon 1 datajoukkoa D ja minimitukiarvoa 2.

Algoritmin alussa etsitään usein esiintyvät 1-alkiojoukot, taulukko 2 L_1. Seuraavat L_k joukot, joiden $k >= 2$ etsitään ehdotetuista k-alkiojoukoista, jotka on muodostettu L_{k-1}-joukon

Alkiot
$\{A1, A2, A3\}$
$\{A1, A2, A5\}$
$\{A2, A4\}$
$\{A1, A2, A4\}$
$\{A1, A3\}$
$\{A4, A5\}$

Taulukko 1. Datajoukko D

avulla. Kun ehdotettut k-alkiojoukkot ovat muodostettu, lasketaan jokaiselle k-alkiojoukolle tukiarvo. L_k-joukkoon valitaan ne ehdotetut k-alkiojoukot, jotka tyydyttävät minimitukiarvon. Lopuksi kaikista usein esiintyvistä alkiojoukoista L luodaan assosiaatiosäännöt. Assosiaatiosääntöjen luonti on kuvattuna luvussa 3.2.

Alla olevassa esimerkissä kuvataan kuinka L_1-joukon avulla luodaan 2-alkiojoukot, sekä kuvattuna 2-alkiojoukon tukiarvot. Lisäksi kuvattuna 2-alkiojoukoista karsittu L_2-joukko.

L_1	Tuki
$\{A1\}$	4
$\{A2\}$	4
$\{A3\}$	2
$\{A4\}$	3
$\{A5\}$	2

$\Rightarrow$

2-alkiojoukko	Tuki
$\{A1, A2\}$	3
$\{A1, A3\}$	2
$\{A1, A4\}$	1
$\{A1, A5\}$	1
$\{A2, A3\}$	1
$\{A2, A4\}$	2
$\{A2, A5\}$	1
$\{A3, A4\}$	0
$\{A3, A5\}$	0
$\{A4, A5\}$	1

$\Rightarrow$

L_2	Tuki
$\{A1, A2\}$	3
$\{A1, A3\}$	2
$\{A2, A4\}$	2

Taulukko 2. Ehdotetut alkiojoukot

Apriori-algoritmi kärsii kahdesta ei-triviaalista ongelmasta. Algoritmi joutuu mahdollisesti luomaan todella suuren määrän ehdotettuja alkiojoukkoja ja käymään useasti koko datan

läpi määrittäessään tukiarvoja ehdotetuille alkiojoukoille. Seuraavassa luvussa esitettävä FP-growth-algoritmi pyrkii etsimään usein esiintyvät alkiojoukot ilman ehdotettujen alkiojoukkojen luomista. (Han, Kamber ja Pei 2011, s. 218)

3 FP-growth

FP-growth-algoritmi esiteltiin ensimmäisen kerran artikkelissa Han, Pei ja Yin 2000. FP-growth-algoritmi kehitettiin ratkaisemaan Apriori-algoritmin ongelmat: tarve luoda todella suuri määrä ehdotettuja osajoukkoja ja annetun datan läpikäynti useaan kertaan. Ratkaisuksi esitetään FP-growth-algoritmia, joka hyödyntää binääripuutietorakennetta, jota kutsutaan FP-puuksi (engl. frequent pattern tree) ja hajoita ja hallitse -menetelmää. (Han, Pei ja Yin 2000; Han, Kamber ja Pei 2011, s. 218)

FP-puu kuvaa usein esiintyviä alkiojoukkoja polkuina puussa. FP-puu on järjestetty siten, että usein esiintyvät alkiot voivat jakaa samoja solmuja. Solmulla voi olla yksi vanhempi ja rajaton määrä lapsia. Yksittäinen solmu sisältää tiedon siitä mikä alkio on kyseessä ja kuinka monesti alkio esiintyy kyseisellä paikalla. (Han, Pei ja Yin 2000)

Hajoita ja hallitse -menetelmä pyrkii etsiessään pitkiä usein esiintyviä alkiojoukkoja etsimään lyhyempiä usein esiintyviä alkiojoukkoja joita yhdistelemällä löytämään pitkät usein esiintyvät alkiojoukot. (Han, Pei ja Yin 2000)

FP-growth-algoritmi muodostaa annetusta datasta löytyvistä säännöllisistä alkioista FP-puun, joka säilyttää tiedon alkiojoukon assosiaatioista. Seuraavaksi algoritmi muodostaa FP-puun pohjalta jokaiselle usein esiintyvälle alkiolle ehdolliset joukot, jonka jälkeen ehdollisista joukoista muodostetaan ehdollinen FP-puu. Ehdollisten joukkojen muodostuksessa käydään läpi vain alkiojoukot, joissa alkoi esiintyy. Merkittävästi supistaen läpi käytävien alkiojoukkojen määrää. Lopuksi assosiaatiosäännöt muodostetaan käymällä läpi ehdolliset FP-puut. (Borgelt 2005; Said, Dominic ja Abdullah 2009; Han, Kamber ja Pei 2011, s. 218–221)

3.1 Algoritmin vaiheet

Käytetään taulukon 3 datajoukkoa D. Taulukon 3 datajoukko sisältää alkiojoukkoja, jotka kuvaavat ostotapahtumia, joissa alkiot ovat jotain tuotteita. Määritellään lisäksi minimitukiarvoksi $min\_tuki = 2$ ja minimiluottamusarvoksi $min\_luottamus = 70\%$.

Datajoukon ensimmäisellä läpikäynnillä etsitään 1-alkiojoukot ja lasketaan niiden tukiarvot.

Alkiot
$\{A1,\ A6\}$
$\{A1,\ A2,\ A3\}$
$\{A1,\ A3,\ A5\}$
$\{A1,\ A2,\ A6\}$
$\{A1,\ A2,\ A3\}$
$\{A1,\ A4,\ A7\}$
$\{A1,\ A4,\ A9\}$

Taulukko 3. Datajoukko D

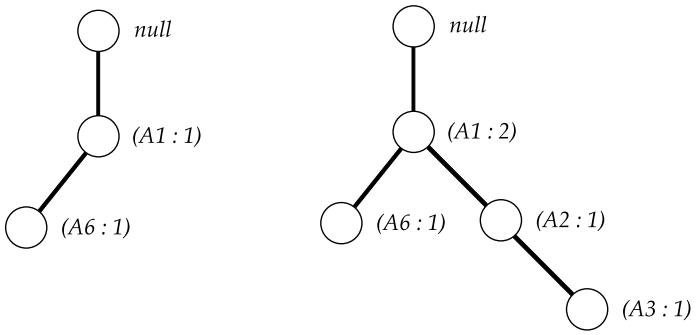
Järjestetään 1-alkiojoukot tukiarvon mukaan laskevaan järjestykseen ja jätetään listasta pois kaikki 1-alkiojoukot, joiden tukiarvo on pienempi kuin *min_tuki*. Esitetään alkiojoukko seuraavasti $\{alkiot : tuki\}$. Muodostuu lista $L = \{\{A1 : 7\},\ \{A2 : 3\},\ \{A3 : 3\},\ \{A4 : 2\},\ \{A6 : 2\}\}$, joka on esitetty taulukossa (4).

1-alkiojoukko	**Tuki**
$\{A1\}$	7
$\{A2\}$	3
$\{A3\}$	3
$\{A4\}$	2
$\{A6\}$	2

Taulukko 4. Datajoukon 1-alkiojoukot

Seuraavaksi luodaan FP-puu. Asetaan puun juuriarvoksi *null* eli tyhjä. Lähdetään käymään datajoukkoa D toisen kerran läpi, siten että jokainen D:n alkiojoukko käydään läpi listan L mukaisessa järjestyksessä toisin sanoen järjestetään läpikäytävä alkiojoukko tukiarvon mukaan laskevasti. Lisätään datajoukon D ensimmäinen alkiojoukko $\{A1,\ A6\}$, joka sisältää alkiot $A1$ ja $A6$ listan L mukaisessa järjestyksessä, FP-puuhun. Kuvataan FP-puun solmuja seuraavasti $(a : m)$, jossa a on datajoukon alkio ja m on sen esiintymien määrä, jota voidaan käsitellä solmun tukiarvona. Luodaan uusi solmu $(A1 : 1)$, jonka vanhempi solmu on juurisolmu sekä luodaan toinen uusi solmu $(A6 : 1)$, joka on linkitetty luotuun $(A1 : 1)$ solmuun. Näin saadaan aikaan kuvion 1 vaiheen 1 mukainen FP-puu.

Jatketaan datajoukon D seuraavan alkiojoukon $\{A1,\ A2,\ A3\}$ käsittelyä. Koska solmu $(A1 : 1)$ on jo olemassa, kasvatetaan sen m arvo yhdellä. Alkioille $A2$ ja $A3$ luodaan uudet solmut $(A2 : 1)$ ja $(A3 : 1)$. Lisätään luodut solmut FP-puuhun siten, että solmu $(A2, 1)$ linkitetään solmuun $(A1 : 2)$ ja $(A3 : 1)$ solmu linkitetään solmun $(A2 : 1)$ lapseksi. FP-puu on nyt kuvion 1 vaiheen 2 mukainen.



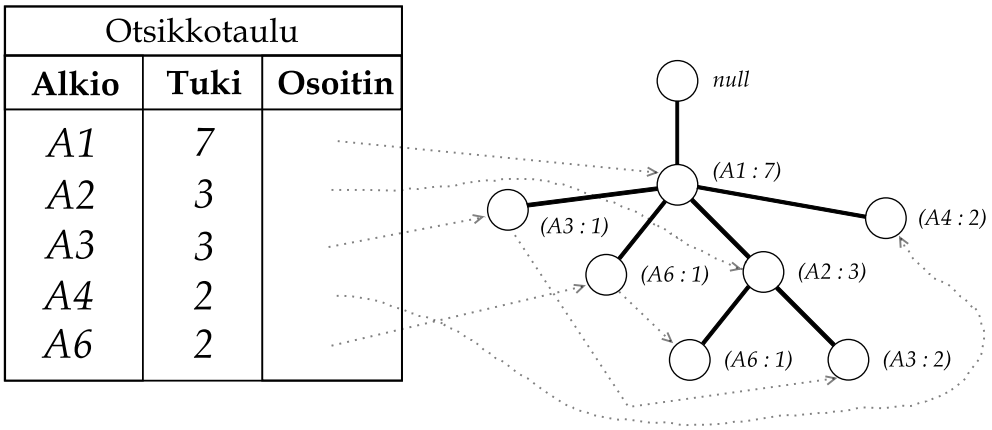
Vaihe 1. Vaihe 2.

Kuvio 1. FP-puun muodostamisen kaksi ensimmäistä vaihetta

Datajoukon D läpi käyntiä jatketaan kunnes kaikki alkiojoukot on lisätty FP-puuhun. Tilanteissa, joissa alkiojoukolla on useita samoja alkioita jo lisätyn alkiojoukon kanssa, jokaista alkiota vastaavan solmun m arvoa kasvatetaan yhdellä ja vain uudet solmut lisätään FP-puuhun. Jos alkiojoukon ensimmäistä alkiota vastaa solmua ei ole jo FP-puussa, voidaan uusi solmu linkittää myös juurisolmuun ja lopuista alkioista muodostetut solmut luodun solmun alle asianmukaisesti.

Valmiin FP-puun läpikäynnin helpottamiseksi luodaan otsikkotaulu 2 , jossa säilytetään tiedot siitä mikä alkio on kyseessä, mikä on alkion tukiarvo ja osoitusketju alkiota vastaavaan solmuun tai solmuihin. Osoitusketju kertoo alkiota vastaavien solmujen sijainnit FP-puussa. Osoitusketjun mahdollistaa, että FP-puuta läpikäydessä ei tarvitse etsiä jokaista alkiota vastaavaa solmua erikseen. Luodun FP-puun ja otsikkotaulun ansiosta säännöllisten joukkojen etsiminen voidaan datajoukon sijaan tehdä FP-puuta tutkimalla (Han, Kamber ja Pei 2011, s. 219).

Seuraavaksi siirrytään tutkimaan FP-puuta. FP-puun tutkiminen aloitetaan listan L viimei-



Kuvio 2. Otsikkotaulu ja valmis FP-puu

Alkio	**Ehdollinen joukko**	**Ehdollinen FP-puu**	**Usein esiintyvät alkiojoukot**
$A6$	$\{A1,\ A2:1\}, \{A1:1\}$	$(A1:2)$	$\{A1,\ A6:2\}$
$A4$	$\{A1:2\}$	$(A1:2)$	$\{A1,A4:2\}$
$A3$	$\{A1,\ A2:2\}, \{A1:1\}$	$(A1:3),\ (A2:2)$	$\{A1,\ A3:2\}, \{A1,\ A2,\ A3:2\}$
$A2$	$\{A1:3\}$	$(A1:3)$	$\{A1,\ A2:3\}$

Taulukko 5. FP-growth algoritmin lopputulos

sestä usein esiintyvästä alkiosta eli $A6$:sta. Alkiolle muodostetaan ehdollinen joukko, joka sisältää FP-puun polut alkiota vastaavasta solmusta $(A6:1)$ juuren. Muodostettu ehdollinen joukko on $\{A1,\ A2:2\}, \{A1:1\}$. Kun ehdollinen joukko on muodostettu, muodostetaan ehdollisen joukon pohjalta alkiolle ehdollinen FP-puu. Puuksi muodostuu $(A1:2)$. Solmu $(A2:1)$ jätetään pois ehdollisesta FP-puusta, koska sen määrä m on alle minimitukiarvon. Ehdollinen FP-puu muodostaa kaikki alkion usein esiintyvät alkiojoukot $\{A1,\ A6:2\}$. Kun koko FP-puu on tutkittu saadaan taulukon 5 mukainen tulos.

3.2 Assosiaatiosääntöjen luominen

Lopuksi luodaan varsinaiset assosiaatiosäännöt, kun usein esiintyvät alkiojoukot on löydetty. Assosiaatiosääntö hyväksytään kun se tyydyttää sekä minimituki että minimiluottamus arvot.

Assosiaatiosäännön luottamus lasketaan seuraavan kaavan mukaan:

$$luottamus(A \Rightarrow B) = \frac{tuki(A \cup B)}{tuki(A)}$$

Assosiaatiosäännöt muodostetaan siten, että jokaisesta usein esiintyvästä alkiojoukosta l luodaan sen ei-tyhjät osajoukot. Luodaan jokaiselle ei-tyhjälle osajoukolle s assosiaatiosääntö "$s \Rightarrow (l - s)$"ja hyväksytään assosiaatiosääntö jos $\frac{tuki(l)}{tuki(s)} \geq min\_luottamus$. (Han, Kamber ja Pei 2011, s. 215–216)

Esimerkiksi usein esiintyvän alkiojoukon {A1, A2, A3 : 2} ei-tyhjät osajoukot ovat {A1}, {A2}, {A3}, {A1, A2}, {A1, A3} ja {A2, A3}. Näistä ei-tyhjistä osajoukoista muodostetaan taulukossa 6 esitetyt ehdotetut assosiaatiosäännöt, joille lasketaan luottamusarvot.

$$\{A1,\ A2\} \Rightarrow \{A3\}, \quad luottamus = 2/3 = 66,6\%$$
$$\{A1,\ A3\} \Rightarrow \{A2\}, \quad luottamus = 2/2 = 100\%$$
$$\{A2,\ A3\} \Rightarrow \{A1\}, \quad luottamus = 2/2 = 100\%$$
$$\{A1\} \Rightarrow \{A2,\ A3\}, \quad luottamus = 2/7 = 28,5\%$$
$$\{A2\} \Rightarrow \{A1,\ A3\}, \quad luottamus = 2/3 = 66,6\%$$
$$\{A3\} \Rightarrow \{A1,\ A2\}, \quad luottamus = 2/3 = 66,6\%$$

Taulukko 6. Ehdotetut assosiaatiosäännöt

Usein esiintyvän alkiojoukon {A1, A2, A3 : 2} ehdotetuista assosiaatiosäännöistä sekä minimituen että minimiluottamuksen tyydyttää assosiaatiosäännöt:

$$\{A1,\ A3\} \Rightarrow \{A2\},\ tuki = 2,\ luottamus = 100\%$$

$$\{A2,\ A3\} \Rightarrow \{A1\},\ tuki = 2,\ luottamus = 100\%$$

Assosiaatiosääntöjen luomisen jälkeen assosiaatiosääntöalgoritmin läpikäynti on saatu päätökseen.

3.3 Algoritmien vertailu

FP-growth-algoritmi etsii usein esiintyvät alkiojoukot käyttämällä hajoita ja hallitse -menetelmää, jossa ei käytetä raskasta ehdotettujen alkiojoukkojen luontia kuten Apriori-algoritmissa. Puo-

lestaan FP-growth-algoritmi hyödyntää kompaktia FP-puu tietorakennetta usein esiintyvien alkiojoukkojen etsimiseen.

Tässä tutkielmassa algoritmeja vertaillaan kirjallisuuskatsauksen keinoin, eikä täten tässä tutkielmassa tuoda esiin tarkkoja suoritusaikoja, koska laitteistot joilla algoritmeja on vertailtu eroaa kirjallisuuden välillä. Suoritusaika kuvaa aikaa, jonka algoritmi käyttää usein esiintyvien alkiojoukkojen etsimiseen. Suoritusajassa ei ole mukana assosiaatiosääntöjen muodostamista, sillä se ei vaikuta merkittävästi algoritmin tehokkuuteen (Jukić ja Nestorov 2006; Agrawal, Imieliński ja Swami 1993).

Vertailuissa, joissa tutkittiin minimitukiarvon vaikutusta FP-growth- ja Apriori-algoritmien suoritusaikaan, havaittiin että, FP-growth-algoritmi on selvästi nopeampi kuin Apriori-algoritmi pienillä tukiarvoilla. Kun tukiarvo on pieni, on usein esiintyviä alkiojoukkoja paljon ja ne ovat pitkiä, jolloin Apriori-algoritmi joutuu käyttämään hyvin paljon aikaa ehdotettujen osajoukkojen luomiseen. (Han, Pei ja Yin 2000; Zhou ja Yau 2007; Borgelt 2005)

Molempien algoritmien suoritusaika kasvaa alkiojoukkojen määrän kasvaessa, mutta FP-growth-algoritmin suoritusajan kasvu on selvästi maltillisempaa kuin Apriori-algoritmin. Tarkoittaen, että FP-growth-algoritmi on selvästi nopeampi kuin Apriori-algoritmi erityisesti suurilla datamäärillä. (Han, Pei ja Yin 2000; Zhou ja Yau 2007; Borgelt 2005)

FP-growth-algoritmi on selvästi nopeampi kuin Apriori-algoritmi usein esiintyvien algoritmien etsimisessä, koska se ei käytä raskasta ehdotettujen osajoukkojen luontia, vaan hyödyntää kompaktia FP-puu tietorakennetta sekä hajota ja hallitse -menetelmää usein esiintyvien alkiojoukkojen etsimiseen. (Han, Pei ja Yin 2000; Zhou ja Yau 2007; Borgelt 2005; Han, Kamber ja Pei 2011, s. 221)

4 Yhteenveto

Tässä tutkielmassa esiteltiin yksi assosiaatiosääntöalgoritmi, FP-growth-algoritmi, kirjallisuuskatsauksen keinoin. Assosiaatiosääntöalgoritmeja käytetään assosiaatiosääntöjen etsimiseen, joita käytetään tiedonlouhinnassa yhtenä keinona kuvaamaan datassa esiintyviä säännöllisyyksiä. Assosiaatiosääntö mielletään kiinnostavaksi, kun se tyydyttää ennalta määritellyt minimituki- ja minimiluottamusarvot. Assosiaatiosääntöjä käytetään tyypillisesti markkinakorianalyysiin, jossa tutkitaan asiakkaiden ostokäyttäytymistä. Markkinakorianalyysiä hyödynnetään esimerkiksi lisämyynnin ja tuotesijoittelun tukena.

Assosiaatiosääntöjen etsiminen ja muodostaminen voidaan jakaa kahteen alaongelmaan: usein esiintyvien alkiojoukkojen etsiminen ja assosiaatiosääntöjen muodostaminen usein esiintyvistä alkiojoukoista. Usein esiintyvien alkiojoukkojen etsiminen vaikuttaa merkittävimmin algoritmin tehokkuuteen. FP-growth-algoritmi koostaa annetusta datasta kompaktin FP-puun, josta muodostetaan useita pienempia ehdollisia FP-puita. Ehdollisista FP-puista saadaan lopuksi muodostettua usein esiintyvät alkiojoukot. FP-growth-algoritmi ei käytä raskasta ehdotettujen osajoukkojen luontia toisin kuin luvussa 2.3 esitelty Apriori-algoritmi. Näin ollen FP-growth-algoritmi on merkittävästi nopeampi kuin Apriori-algoritmi etsimään usein esiintyvät alkiojoukot.

Assosiaatiosääntöjen etsinnässä on viimeisimmässä kirjallisuudessa perehdytty kehittämään algoritmeja, jotka kykenevät muodostamaan säännöllisyyksiä datasta, joka on moniuloitteista. Moniuloitteisella datalla tarkoitetaan data, jossa on esimerkiksi ostotapahtumien lisäksi mukana tieto siitä minkä ikäinen ostaja on ollut. Tällöin assosiaatiosääntö voi olla esimerkiksi että 22-vuotiaat ostivat, jollain todennäköisyydellä luistimet. Erilaisten moniuloitteisia assosiaatiosääntöjä etsivien algoritmien suoritusaika vertailu olisi hyvä jatkotutkimus. (Han, Kamber ja Pei 2011, s. 234–236)

Lähteet

Agrawal, Rakesh, Tomasz Imieliński ja Arun Swami. 1993. “Mining association rules between sets of items in large databases”. Teoksessa *Proceedings of the 1993 ACM SIGMOD international conference on Management of data,* 207–216.

Agrawal, Rakesh, Ramakrishnan Srikant ym. 1994. “Fast algorithms for mining association rules”. Teoksessa *Proc. 20th int. conf. very large data bases, VLDB,* 1215:487–499.

Borgelt, Christian. 2005. “An Implementation of the FP-Growth Algorithm”. Teoksessa *Proceedings of the 1st International Workshop on Open Source Data Mining: Frequent Pattern Mining Implementations,* 1–5. OSDM ’05. Chicago, Illinois: Association for Computing Machinery. ISBN: 1595932100. https://doi.org/10.1145/1133905.1133907. https://doi-org.ezproxy.jyu.fi/10.1145/1133905.1133907.

Fayyad, Usama, Gregory Piatetsky-Shapiro ja Padhraic Smyth. 1996. “From Data Mining to Knowledge Discovery in Databases”. *AI Magazine* 17, numero 3 (maaliskuu): 37. https://doi.org/10.1609/aimag.v17i3.1230. https://ojs.aaai.org/index.php/aimagazine/article/view/1230.

Han, Jiawei, Micheline Kamber ja Jian Pei. 2011. *Data Mining: Concepts and Techniques.* 3rd. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. ISBN: 0123814790.

Han, Jiawei, Jian Pei ja Yiwen Yin. 2000. “Mining frequent patterns without candidate generation”. *ACM sigmod record* 29 (2): 1–12.

Jukić, Nenad, ja Svetlozar Nestorov. 2006. “Comprehensive data warehouse exploration with qualified association-rule mining”. *Decision Support Systems* 42 (2): 859–878. ISSN: 0167-9236. https://doi.org/https://doi.org/10.1016/j.dss.2005.07.009. https://www.sciencedirect.com/science/article/pii/S0167923605001090.

Liao, Shu-Hsien, Pei-Hui Chu ja Pei-Yuan Hsiao. 2012. “Data mining techniques and applications – A decade review from 2000 to 2011”. *Expert Systems with Applications* 39 (12): 11303–11311. ISSN: 0957-4174. https://doi.org/https://doi.org/10.1016/j.eswa.2012.02.063. https://www.sciencedirect.com/science/article/pii/S0957417412003077.

Mitra, S., S. K. Pal ja P. Mitra. 2002. “Data mining in soft computing framework: a survey”. *IEEE Transactions on Neural Networks* 13 (1): 3–14. https://doi.org/10.1109/72.977258.

Ngai, E.W.T., Li Xiu ja D.C.K. Chau. 2009. “Application of data mining techniques in customer relationship management: A literature review and classification”. *Expert Systems with Applications* 36 (2, Part 2): 2592–2602. ISSN: 0957-4174. https://doi.org/https://doi.org/10.1016/j.eswa.2008.02.021. https://www.sciencedirect.com/science/article/pii/S0957417408001243.

Said, Aiman Moyaid, PDD Dominic ja Azween B Abdullah. 2009. “A comparative study of fp-growth variations”. *International journal of computer science and network security* 9 (5): 266–272.

Zhou, Ling, ja Stephen Yau. 2007. “Association Rule and Quantitative Association Rule Mining among Infrequent Items”. Teoksessa *Proceedings of the 8th International Workshop on Multimedia Data Mining: (Associated with the ACM SIGKDD 2007)*. MDM ’07. San Jose, California: Association for Computing Machinery. ISBN: 9781595938374. https://doi.org/10.1145/1341920.1341929. https://doi-org.ezproxy.jyu.fi/10.1145/1341920.1341929.