TRIX-tiedonhakujärjestelmän evaluointi

Timo Aalto

Tampereen Yliopisto
Informaatiotutkimuksen laitos
Pro gradu -tutkielma
maaliskuu 2005

Tampereen yliopisto
Informaatiotutkimuksen laitos
AALTO, TIMO: TRIX-tiedonhakujärjestelmän evaluointi
Pro gradu -tutkielma, 62 s., 3 liitettä
Informaatiotutkimus
Maaliskuu 2005

TIIVISTELMÄ

Tutkimus esittelee Tampereen yliopistossa vuonna 2004 kehitetyn TRIX-tiedonhakujärjestelmän. TRIX on erityisesti XML-dokumenteista tehtävään tiedonhakuun kehitetty osittaistäsmäyttävä, hierarkkiseen tietorakenteeseen perustuvaa indeksointiratkaisua käyttävä tiedonhakujärjestelmän prototyyppi. Tutkimuksen tutkimuskysymyksinä oli selvittää onko järjestelmän kahden eri hakuavainten painojen yhdistämismenetelmän välillä eroa sekä tutkia miten painotuskaavan vakion säätäminen vaikuttaa järjestelmän toimintaan.

Järjestelmää testattiin kansainvälisen INEX-evaluointihankkeen testikokoelmassa, joka koostuu noin 12000:ta XML-merkatusta tieteellisestä artikkelista. Aineistoon tehtiin kyselyjä yhteensä 76 hakuaiheesta ja tuloksia arvioitiin kahdella XML-tiedonhaun evaluointiin kehitetyllä mittarilla. Lisäksi tutkimusta varten kehitettiin oma ohjelma tuloslistojen analysointiin.

Tutkimuksen tuloksina havaittiin, että painojen yhdistämismenetelmien välillä ei ollut merkitsevää eroa sekä että painotuskaavan vakion säätämisellä on voimakas vaikutus järjestelmän palauttamien elementtien kokoon ja vaihteleva vaikutus evaluointimittareiden antamiin tuloksiin.

1 Johdanto

XML-merkkauskielen koko ajan laajeneva käyttö mm. tieteellisten artikkeleiden tallennuksessa, digitaalisissa kirjastoissa ja World Wide Webissä on lisännyt kiinnostusta kieleen myös tiedonhaun tutkimuksen piirissä. XML-tiedonhaussa on tarkoituksena hyödyntää tallennettujen dokumenttien XML-merkkauksella eksplisiittisesti ilmaistua loogista rakennetta ja sen avulla palauttaa niiden relevanteimmat osat vastauksena käyttäjän tiedontarpeeseen koko dokumentin palauttamisen sijaan. XML-tiedonhaku voidaankin mieltää askeleeksi kohti informaation hakua dokumenttien hakemisen sijaan. Tämän entistä fokusoituneemman hakuparadigman myötä tiedonhaun haasteellisuus kasvaa – tiedonhakujärjestelmän täytyy relevantin tiedon löytämisen lisäksi myös päätellä käyttäjälle palautettavan dokumentin osan optimaalinen laajuus. Lisäksi hakuun saattaa kuulua varsinaisen aiheen lisäksi myös dokumentin rakenteeseen liittyviä ehtoja, mitkä järjestelmän kuuluu täyttää.

INEX (INitiative for the Evaluation for XML Retrieval, ks. 2.3) –hanke perustettiin vuonna 2002 luomaan puitteita XML-tiedonhaun evaluoivaan tutkimukseen. Hankkeen puitteissa on koottu laaja XML-dokumenttien testikokoelma hakuaiheineen, dokumenttien rakenteisuuden huomioonottavine relevanssiarvioineen ja evaluointimetriikoineen. Vuonna 2004 Tampereen yliopiston informaatiotutkimuksen laitos päätti osallistua hankkeeseen kehittämällä oman tiedonhakujärjestelmänsä.

Tämä pro gradu –tutkielma esittelee kehitetyn TRIX-tiedonhakujärjestelmän ja evaluoi sen toimintaa tehtäessä tiedonhakuja INEX-testikokoelmasta. Tarkemmin tutkimuskysymykset voidaan ilmaista seuraavasti:

1. Mikä vaikutus järjestelmän käyttämillä hakuavainten painojen yhdistämismenetelmillä on järjestelmän tehokkuuteen ja tuloksina palauttamiin elementteihin, sekä
2. mikä vaikutus järjestelmän painotuskaavan vakioiden manipuloinnilla on järjestelmän tehokkuuteen ja tuloksina palauttamiin elementteihin.

Näihin kysymyksiin pyritään vastaamaan evaluointitutkimuksella. Ensimmäisessä tapauksessa järjestelmän sumean logiikan operaatioon perustuvaa painojen yhdistämismenetelmää verrataan järjestelmän vakiona käyttämään painojen keskiarvon laskemiseen perustuvaan menetelmään. Toisessa tapauksessa verrataan painotuskaavan vakion vaihtoehtoisia arvoja järjestelmän vakioasetukseen.

Ensimmäisessä luvussa käydään läpi yleisiä tiedonhaun käsitteitä. Luku kaksi esittelee XML-merkkauskielen, käsittelee XML-tiedonhaun erityiskysymyksiä, esittelee yleisesti INEX-hankkeen ja referoi aiempaa XML-tiedonhaun tutkimusta. Luku kolme esittelee TRIX-tiedonhakujärjestelmän. Luvussa viisi esitellään tutkimuksen koeympäristö – dokumenttikokoelma, hakuaiheet, relevanssiarviot, evaluointimittarit ja tilastollisen testaamisen menetelmät. Luku viisi käsittelee tutkimuksen tulokset ja luvut kuusi ja seitsemän sisältää keskustelua ja johtopäätökset.

1.1 Käsitteistöä

Tiedonhaku (IR, Information Retrieval) käsittelee laajasti ottaen tiedon esittämistä, tallentamista, organisointia ja hakemista (Baeza-Yates & Ribeiro-Neto 1999, 1). Suppeammin se voidaan käsittää dokumenttien tai niiden viitteiden palauttamista vastauksena tiedonhakupyyntöön (Robertson 1983, 9). Tämä työ kuuluu tiedonhaun tutkimuksen piiriin. Tiedonhaun tutkimuksessa tavoitteena on kehittää käsitteitä, menetelmiä ja järjestelmiä hallitsemaan ja parantamaan tiedonhaun tuloksellisuutta (Järvelin 1995, 25).

Dokumentti on yksi inhimillisen kommunikaation perusmuoto ja talletetun tiedon esittämisen ja välittämisen yksikkö (Järvelin 1995, 9). Tiedonhaussa dokumentti on perinteisesti ollut haettava perusyksikkö (Baeza-Yates & Ribeiro-Neto 1999, 440). Tiedonhakujärjestelmissä dokumentti on yleensä tallennettu elektroniseen muotoon. Kaikki dokumentit koostuvat sisällöstä ja sille ominaisesta loogisesta rakenteesta (Järvelin 1995, 9). Elektroniseen dokumenttiin kuuluu aina sisäinen (tietokoneen käsiteltäväksi tarkoitettu) ja ulkoinen (ihmisaisteille tarkoitettu) esitysmuoto. Kappaleessa 2.1 esiteltävä XML on eräs tapa dokumentin sisäisen muodon määrittämiseen.

Relevanssi on tiedonhaun keskeisimpiä käsitteitä. Relevanssilla katsotaan olevan useita erilaisia ilmentymiä; Saracevic (1996, 12) jakaa relevanssin ilmentymät viiteen kategoriaan: algoritmiseen, aihe-, kognitiiviseen, tilanne- ja affektiiviseen relevanssiin. Tässä työssä relevanssi käsitetään yleisesti Saracevicin määritelmän mukaisena aiherelevanssina (Topical relevance). Aiherelevanssi mittaa kyselyn ilmaiseman ja dokumentin kattaman aiheen välistä suhdetta. Määritelmässä oletetaan kyselyjen ja dokumenttien olevan jostain nimettävästä aiheesta. Relevanssi on luonnostaan moniulotteinen ja –asteinen käsite, mutta perinteisessä tiedonhaun tutkimuksessa se on yleensä typistetty binääriseksi (relevantti/ei relevantti) ja dokumenttikohtaiseksi. Binääristä relevanssia on kritisoitu realismin puutteesta (ks. esim. Sormunen 2002) ja joissain tiedonhaun tutkimushankkeissa onkin siirrytty moniasteisen ja –ulotteisen relevanssin käsitteen käyttämiseen. Koska XML-dokumenteista tehtävän tiedonhaun tuloksena saatavat elementit eivät ole siinä välttämättä itsenäisiä yksikköjä perinteisten dokumenttien tapaan, on sen tutkimuksen yhteydessä siihen kehitetty kaksi ulottuvuutta: kattavuus ja spesifisyys. Näitä selvitetään tarkemmin kappaleessa 4.2.3.

Yksi tiedonhaun tutkimuksen näkökulmista on **evaluoiva tutkimus** jossa ollaan kiinnostuneita tiedonhaun tuloksellisuudesta ja kustannuksista. Evaluoiva tutkimus voidaan karkeasti jaotella tietokantojen, tiedonhakujärjestelmien ja hakujen evaluointiin (Järvelin 1995, 48-55).

Tiedonhakujärjestelmien evaluoinnilla on ollut tiedonhakututkimuksen alkuajoista asti tärkeä osa alan tutkimuksessa. Evaluointien motivaationa on yleensä joko rinnakkaisten järjestelmien vertailu käyttötilannetta vasten, olemassa olevan järjestelmän parantamiseen pyrkiminen tai toimintamallien testaaminen ja vertailu suunniteltaessa uusia tiedonhakujärjestelmiä (Robertson 1983, 10-11).

Robertsonin (1983, 11) ja Hullin (1993, 329) mukaan järjestelmäevaluoinnin suorittamiseen tarvitaan seuraavat osat:

1. Tiedonhakujärjestelmä (tai järjestelmät) - sääntöjen ja menettelytapojen joukko joko ihmisten tai tietokoneiden toteuttamana
2. Vähintään yksi dokumenttikokoelma ja siihen kohdistettavat kyselyt tai tehtävänannot. Dokumenttikokoelmasta on lisäksi tiedettävä mitkä dokumentit

ovat relevantteja mihinkin kyselyyn. Tällaisia kokoelmia kutsutaan yleisesti *testikokoelmiksi* (ks. kappale 4.2).

3. Mittareita jotka osoittavat haun tuloksellisuuden. Mittarit perustuvat relevanttien ja epärelevanttien dokumenttien samankaltaisuuden asteeseen kyselyyn nähden. Yleinen mittari on ns. saanti-tarkkuus – käyrä.
4. Pätevät tilastolliset menetelmät (ks. kappale 4.3) joiden pohjalta järjestelmien tai menetelmien mitatut erot voidaan todistaa tilastollisesti merkittäviksi.

(Robertson 1983,11; Hull 1993, 329.)

Osa yllä olevan listan käsitteistä määritellään tarkemmin kappaleen 4 alaluvuissa.

Tiedonhakujärjestelmällä tarkoitetaan joko ihmisten tai tietokoneiden toteuttamaa sääntöjen ja menettelytapojen joukkoa jotka toteuttavat kaikki tai ainakin osan seuraavista toiminnoista:

- Indeksointi (esityksien muodostaminen dokumenteista keräämällä niistä hakuavaimet)
- Kyselyiden muodostaminen (esityksien muodostaminen tiedontarpeista)
- Täsmäytys (dokumenttien esitysten täsmäyttäminen tiedontarpeiden esityksiin)
- Palaute (em. prosessien toistaminen muokattuina aikaisempien tulosten arvioinnin perusteella)
- Indeksointikielen rakentaminen (esityksien muotosääntöjen tuottaminen)

(Robertson 1983, 9.)

Nykyiset tiedonhakujärjestelmät ovat automatisoituja, useimmiten käänteisrakenteeseen (inverted file structure) pohjautuvia, osittais- tai täystäsmäyttäviin hakuperiaatteisiin perustuvia tietokoneohjelmistoja.

Käänteisrakennetta käyttävä järjestelmä sisältää käänteistiedoston (inverted file), johon on poimittu kaikki haettavista dokumenteista indeksoidut hakuavaimet ja niiden kaikki sijainnit. Käänteistiedosto mahdollistaa huomattavasti nopeamman haun kuin avainten etsiminen alkuperäismuotoisesta tiedostosta. Käänteistiedoston hakemistona käytetään

sanakirjatiedostoa (dictionary file) joka sisältää hakuavaimet ja niiden esiintymien määrän.(Järvelin 1995, 96-98.)

Täsmäyttämisellä tarkoitetaan kyselyiden (tiedontarpeiden esitysten) ja dokumenttien esitysten samankaltaisuuden (similarity) tutkimista. Tämä prosessi määrittelee mitkä dokumentit ovat relevantteja kyselyn kannalta. Täsmäyttämismenetelmät voidaan jakaa ensi tasolla täystäsmäyttäviin menetelmiin ja osittaistäsmäyttäviin menetelmiin. Boolen logiikkaan perustuvat ratkaisut ovat täystäsmäyttäviä ja osittaistäsmäyttävistä menetelmistä yleisimpiä on ns. vektorimalli (ks. 2.4.2.1) ja erilaiset todennäköisyyslaskentaan perustuvat mallit (ks. 2.4.2.2).

Lähes kaikki osittaistäsmäyttävät menetelmät painottavat dokumenttikokoelmassa esiintyviä termejä sen perusteella miten hyvin niiden oletetaan kuvaavan dokumentin sisältöä. Yleinen tapa määrittää hakuavainten painot on ns. tf*idf –menetelmä (Term Frequency * Inverted Document Frequency). Menetelmässä termin paino määräytyy kahden tekijän pohjalta: kuinka usein termi *j* esiintyy dokumentissa *i* (termin frekvenssi $tf_{i,j}$) ja kuinka usein *j* esiintyy koko dokumenttikokoelmassa. (dokumenttifrekvenssi df_j) Termin paino saadaan kaavalla:

$$w_{i,j} = tf_{i,j} * idf_{i,j} = tf_{i,j} * \log N / df_j$$

jossa *N* on dokumenttien määrä koko dokumenttikokoelmassa ja *idf* tarkoittaa käänteistä dokumenttifrekvenssiä. Menetelmä antaa korkean painon termeille jotka esiintyvät usein pienessä osassa dokumenttikokoelmaa. (Lee, Chuang & Seamons 1997, 68-69.)

Kyselyssä annettujen hakuavainten lasketut painot täytyy vielä yhdistää jollain matemaattisella menetelmällä lopullisen dokumentin relevanssilajitteluarvon saamiseksi. Yksinkertainen tapa suorittaa yhdistäminen on laskea kaikkien kyselyn avainten painojen keskiarvo.

Evaluointitutkimuksen suorittaminen ilman järjestelmän tehokkuutta ilmaisevia **mittareita** olisi hyödytöntä. Useimmiten järjestelmien tehokkuutta evaluoidessa käytetään mittaristoja jotka perustuvat kahteen tiedonhaun peruskäsitteeseen: saantiin ja tarkkuuteen.

Saanti (Recall) voidaan määritellä palautettujen relevanttien dokumenttien määränä verrattuna kokoelman kaikkiin relevantteihin dokumentteihin (Robertson 1983, 19).

Tarkkuus (Precision) määritetään taas relevanttien dokumenttien määränä palautetuista dokumenteista (Robertson 1983, 19).

Yhdessä saanti ja tarkkuus ovat tiedonhaun onnistuneisuuden konkreettisia mittareita. Ne voivat mitata sekä käyttäjän saaman tiedon määrää ja laatua että järjestelmän toiminnan laatua (Järvelin 1995, 56). Saantia ja tarkkuutta voidaan käyttää useilla tavoilla evaluoinnin mittareina.

Hakujen tuloksellisuutta kuvataan yleensä saanti-tarkkuuskäyrällä. Yleisin tapa muodostaa saanti-tarkkuuskäyrä on mitata tarkkuutta kiinteillä saantitasoilla jokaiselle kyselylle ja muodostaa keskimääräinen saanti-tarkkuuskäyrä käyttämällä keskimääräistä tarkkuutta eri saantitasoilla. Toinen tapa on käyttää ns. DCV (Document Cutoff Value) –menetelmää jossa lasketaan keskimääräinen saanti ja tarkkuus ennalta määritetylle määrälle dokumentteja. (Hull 1993, 329-331).

2 XML-tiedonhaku

2.1 XML-merkkauskieli

eXtensible Markup Language (XML) (Bray, Paoli, Sperberg-McQueen, Maler & Yergeau 2004) -spesifikaatio määrittelee XML-dokumentteina tunnettujen tietokoneen luettavaksi tarkoitettujen tekstidokumenttien joukon. XML on SGML (Structured Generalized Markup Language, ISO 8879:1986) –merkkauskielen yksinkertaistettu johdannainen. (Bray ym. 2004). XML on SGML:n tavoin ns. *metakieli;* sitä käytetään määrittämään merkkauskieliä standardoidun kaavan mukaisesti. XML sisältää ainoastaan tiedon tallennuksen muotosäännöt jättäen merkkaussanaston sovellusyhteyden mukaan määriteltäväksi. Tämä mahdollistaa XML:n yleiskäyttöisyyden; sillä voidaan luoda sovelluskohtaisia merkkauskieliä monentyyppisten dokumenttien kuvaamiseen. (Klein 2001, 26.)

Alun perin XML suunniteltiin elektronisen julkaisemisen tarpeisiin, mutta se on saavuttanut tärkeän roolin hyvin monimuotoisen informaation välityksessä webissä ja muissa ympäristöissä (Extensible Markup Language 2004). Perinteisten tekstidokumenttien tallentamisen lisäksi niitä käytetään monenmuotoisen tiedon tallentamiseen, kuten esimerkiksi vektorigrafiikan, elektronisen kaupankäynnin suoritusten ja matemaattisten kaavojen tallentamiseen (Walsh 1998). Tämän lisäksi useita standardoituja merkkauskieliä on määritelty XML:nä esimerkkinä XHTML, joka on HTML 4.0 –standardin uudelleenmäärittely ns. *xml-sovelluksena* (XML application) (Klein 2001, 26.)

2.1.1 Dokumentin fyysinen ja looginen rakenne

Merkkauksella (markup) tarkoitetaan sitä, että tietyt merkkijonot dokumentissa ovat dokumentin varsinaista sisältöä kuvaavaa informaatiota. Merkkauksella kuvataan dokumentin tieto- ja loogista rakennetta sekä kuvaillaan sen sisältämää tietoa. Merkkaus esitetään dokumenteissa yleensä sanoina kulmasulkujen (<>) välissä, kuten esimerkiksi **<nimi>** tai **<h1>.** Tällaisia merkkausyksikköjä kutsutaan yleisesti *tunnisteiksi (tags).* (Klein 2001, 26.)

XML:n yleisin merkkausyksikkö on *elementti (element)*, josta puhutaan usein myös *komponenttina (component)* tai *solmuna (node)*. Elementti koostuu pakollisista aloitus- ja lopetustunnisteista, kuten esimerkiksi **<henkilö>** ja **</henkilö>**. Jos elementillä ei ole sisältöä, alku- ja lopputunnisteet voidaan yhdistää: <**henkilö />.** Elementti voi sisältää muita elementtejä tai tekstiä. Tämän lisäksi elementti voi omata tarkentavia attribuutteja jotka koostuvat ominaisuudesta ja sen arvosta; esimerkiksi <**henkilö nimi=”jussi”>.** Elementin sisällä olevaa toista elementtiä kutsutaan ulomman elementin ali- tai lapsielementiksi. XML-dokumenteissa elementtien täytyy olla oikeaoppisesti sisäkkäisiä; lapsielementtien alku- ja lopputunnisteiden täytyy sijaita yläelementtinsä alku- ja lopputunnisteiden sisällä. (Klein 2001, 26-27.)

Elementtien oikean järjestyksen lisäksi XML-dokumenteilla on muutamia muita rakenne-ehtoja. Dokumentilla saa olla vain yksi juurielementti – elementti jonka suhteen kaikki muut dokumentin elementit ovat alielementtejä. Elementtien nimien täytyy vastata täsmälleen toisiaan aloitus- ja lopetustunnisteissa. Lisäksi elementtien nimeämisessä, vaikka se onkin äärimmäisen vapaata, täytyy noudattaa tiettyjä nimeämiskonventioita. Lisäksi on otettava huomioon että XML säilyttää tyhjän tilan tekstissä. Kun dokumentti täyttää nämä ehdot, sen voidaan sanoa olevan oikein muodostettu (well-formed). (North & Hermans 2000, 67-68.)

Edellä selitetyn havainnollistamiseksi Kuvio 1:ssa on yksinkertainen esimerkki XML-dokumentista:

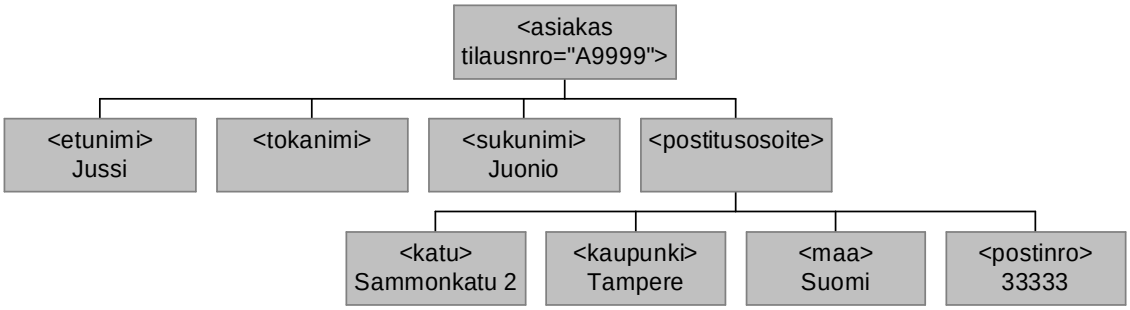
```
<?xml version=”1.0”?>
<asiakas tilausnro=”A9999”>
<etunimi>Jussi</etunimi >
<tokanimi/>
<sukunimi>Juonio</sukunimi >
<postitusosoite>
            <katu>Sammonkatu 2</katu >
            <kaupunki>Tampere </kaupunki>
            <maa>Suomi</maa>
            <postinro>33333</postinro >
</postitusosoite >
</asiakas>
```

Kuvio 1. Esimerkki XML-dokumentista

XML-dokumentti voidaan esittää puumaisena hierarkkisena rakenteena, joka koostuu elementeistä, joilla on alielementtejä ja arvoja (Klein 2001, 27; Jaideep & Anupama 2000, 33). Puurakenne on moniin tarkoituksiin riittävän tehokas monimutkaisenkin

tiedon mallintamiseen. Lisäksi tällaisen rakenteen käsittelyyn on helppo tehdä ohjelmia (Jaideep ja Anupama 2000, 33).

Kuviossa 2 on kuvion 1 XML-dokumentti esitettynä puumaisena rakenteena.



Kuvio 2. XML-dokumentti esitettynä puumaisena hierarkiana.

2.1.2 DTD

Metakielenä XML ei itsessään ota kantaa sen sisältämän tiedon käyttöön tai merkityksiin, vaan XML-sovelluksen sanasto käyttöineen ja merkityksineen täytyy määritellä sovelluskohtaisesti. Määrittely voidaan toteuttaa joko DTD:tä (Document Type Definition) tai uudempaa XML-kaaviota (XML Schema, ks. Fallside & Walmsley 2001) käyttäen. Kumpikaan tekniikoista ei määrittele tiedon merkityksiä, mutta ne pystyvät määrittelemään käytettävissä olevat elementit attribuutteineen ja sallittuine käyttötapoineen dokumenttien rakenteissa. (Klein 2001, 27.) Koska INEX-testikokoelman aineisto käyttää DTD:tä keskitytään tässä työssä sen esittelyyn.

Verrattuna XML-kaavioon DTD mahdollistaa vain suhteellisen yksinkertaisten rakenne-ehtojen määrittelemisen. DTD:llä voidaan määritellä elementtien sallitut lapsielementit, elementtien sallitut attribuutit ja paikat missä dokumentin sisältöteksti voi sijaita. (Klein 2001, 27.) Kuvion 1 dokumenttia vastaava DTD löytyy esimerkkinä kuviosta 3:

```
<!ELEMENT asiakas (etunimi, tokanimi,
  sukunimi, postitusosoite)>
<!ATTLIST asiakas tilausnro ID #REQUIRED>
<!ELEMENT etunimi (#PCDATA)>
<!ELEMENT tokanimi (#PCDATA)>
<!ELEMENT sukunimi (#PCDATA)>
<!ELEMENT postitusosoite
 (katu, kaupunki, maa, postinro)>
<!ELEMENT katu (#PCDATA)>
<!ELEMENT kaupunki (#PCDATA)>
<!ELEMENT maa (#PCDATA)>
<!ELEMENT postinro (#PCDATA)>
```

Kuvio 3. Esimerkki-DTD kuvion 1 dokumentille

Kuviossa 3 oleva DTD määrittää elementille ”asiakas” neljä mahdollista lapsielementtiä ja pakollisen (#REQUIRED) attribuutin ”asiakasnro”. Kaikille elementin ”asiakas” lapsielementeille, poislukien ”postitusosoite” jolla on omia lapsielementtejä, on määritelty että niiden täytyy sisältää tekstiä. (#PCDATA).

Aiemmin mainitun oikeamuotoisuuden (well-formedness) lisäksi XML-dokumentti joka omaa kunnollisen DTD:n ja noudattaa sen sääntöjä, on ns. *validi XML-dokumentti*. XML-standardi ei vaadi DTD:n (tai XML-kaavion) käyttöä, mutta sen käyttö mahdollistaa dokumenttien validisuuden tarkastamisen sekä dokumenttikokoelmien standardoinnin. (Jaideep & Anupama 2000, 33.)

2.2 XML-tiedonhaun erityispiirteitä

Perinteisessä tekstitiedonhaussa dokumenttia on pidetty tiedonhaun perusyksikkönä (Govert, Kazai, Fuhr & Lalmas 2003, 1). Niiden ei ole oletettu sisältävän hierarkkisia rakenteita eikä niiden indeksointirakenteissa oleteta olevan muuta kuin kokoteksti sekä mahdollisia ennaltamääriteltyjä kenttiä (Baeza-Yates ym. 2002, 53). Tiedonhaussa XML-dokumenteista on kuitenkin mahdollista hyötyä niiden luontaisesta rakenteisuudesta; kokonaisten dokumenttien palauttamisen sijaan voidaan käyttäjälle palauttaa dokumenttien osia (elementtejä) jotka parhaiten vastaavat hänen tiedontarpeeseensa (Govert ym. 2003, 1). FERMI-multimediatiedonhakumallin (Chiamarella, Mulhem & Fourel 1996) mukaan palautettavien elementtien pitää olla syvimmältä mahdolliselta dokumentin hierarkian tasolta, ts. olla mahdollisimman spesifejä, ja samalla pystyä kattavasti vastaamaan tiedontarpeeseen (Fuhr & Grossjohann 2001, 172).

Dokumenttien rakenteisuus mahdollistaa myös uusien toiminnallisuuksien lisäämisen tiedonhakujärjestelmiin. Dokumenttien rakennetta voidaan käyttää tarkentamaan hakuja lisäämällä niihin rakenne-ehtoja; käyttäjät voivat esimerkiksi hakea dokumentteja joissa tietyt avainsanat esiintyvät halutussa kontekstissa. Lisäksi XML-elementeissä voidaan esittää muitakin tietotyyppejä kuin pelkkää tekstiä, esimerkiksi numeroita tai päivämääriä. Ei-tekstuaalisten tietotyyppien hyödyntäminen vaatii kuitenkin tekstin täsmäyttämisen lisäksi erillisiä prosessointimenetelmiä. (Liu, Zou & Chu. 2004, 88.)

Elementtien palauttaminen hakutuloksina kokonaisten dokumenttien sijaan tuottaa myös muutamia ongelmia. Toisin kuin kokonaiset dokumentit, XML-elementit eivät ole itsenäisiä kokonaisuuksia, vaan ne liittyvät juurielementtiinsä rakenteensa ja sisältönsä osalta. Palautetut elementit voivat myös olla päällekkäisiä; järjestelmä voi esimerkiksi palauttaa luvun (section) dokumentista ja sen jälkeen kappaleen (paragraph) jo palautetusta luvusta. Palautettavan elementin koko on myös yksi ongelma; elementin pitäisi periaatteessa vastata kyselyyn mahdollisimman kattavasti. Lisäksi tulosten esittäminen perinteisenä lineaarisena tuloslistana on ongelmallista yksittäisten elementtien kontekstiriippuvuuden takia, jatkuvat kontekstimuutokset tuloslistassa voivat aiheuttaa käyttäjälle turhaa hämmennystä. Tämän vuoksi olisikin suotavaa rypästää samasta dokumentista peräisin olevat elementit tuloslistassa. (Gövert ym. 2003, 3.)

2.3 INEX-hanke

The Initiative for the Evaluation of XML-retrieval (INEX)-hanke perustettiin vuonna 2002 tarkoituksena kehittää perusrakenteet ja menetelmät sisältöpainotteisten XML-dokumenttien haun evaluointiin. Näihin kuuluvat laaja XML-dokumenttien testikokoelma ja tarvittavat menetelmät hakutulosten evaluointiin. INEX- testikokoelma koostuu tiedonhaun testikokoelmille tyypillisesti kolmesta osasta: dokumenttijoukosta, aiheista (topics) ja relevanssiarvoista (Gövert & Kazai 2003, 1-4). INEX-testikokoelma on kuvattu tarkemmin kappaleessa 4.

Ensimmäisenä vuonna (2002) hankkeeseen osallistui lopulta 36 organisaatiota jotka lähestyivät aihepiiriä hyvin erilaisista lähtökohdista. Osallistujien lähestymistavat voitiin karkeasti luokitella kolmeen kategoriaan:

- Tiedonhaku-orientoituneet ryhmät, jotka keskittyivät erilaisten tiedonhakumallien laajentamiseen XML-dokumenteille sopiviksi
- Tietokantaorientoituneet ryhmät, jotka pyrkivät laajentamaan erilaisten tietokantahallintajärjestelmien (DBMS) toimintoja rakenteisten dokumenttien käsittelyyn
- XML-spesifit ryhmät, jotka pyrkivät kehittämään puhtaasti XML-käyttöön tarkoitettuja malleja ja järjestelmiä yleisesti perustaen työnsä XML-standardeihin (XSL, XPath, XQuery)

Muutamat ryhmistä myös yhdistelivät elementtejä eri kategorioista. (Gövert & Kazai 2003, 2.)

Vuonna 2004 INEX- hankkeeseen ilmoittautui 59 organisaatiota ja hankkeen puitteissa tehdään tutkimusta puhtaan järjestelmäevaluoinnin, ns. ”ad-hoc –tiedonhaun” (johon tämä työ liittyy) lisäksi neljällä muulla linjalla (Track):

- Relevanssipalautelinjassa (Relevance Feedback Track) tutkitaan relevanssipalautemenetelmien toimintaa XML-tiedonhaussa.
- Heterogeenisen kokoelman linjassa (Heterogeneous Collection Track) haetaan metodeja erilaisia DTD:itä sisältävien dokumenttien hakuun.
- Luonnollisen kielen käsittely -linjassa (Natural Language Processing Track) pyritään soveltamaan luonnollisen kielen käsittelytekniikoita XML-tiedonhakuun.
- Interaktiivisen tiedonhaun linjassa (Interactive Track) tutkitaan käyttäjien toimintaa heidän saadessaan hakutuloksena XML-dokumenttien elementtejä, ja toisaalta pyritään kehittämään käyttäjäkeskeisiä tekniikoita XML-tiedonhakuun.

(INitiative for the Evaluation of… 2004.)

2.4 XML-tiedonhaun aikaisempaa tutkimusta

XML:n käytössä ja tutkimuksessa on ollut havaittavissa kaksi erilaista näkökulmaa:

- *Dokumenttikeskeinen* näkökulma keskittyy XML-sovelluksiin jotka käsittelevät dokumentteja sen perinteisessä merkityksessä, jossa rakennetta käytetään pelkästään esittämään dokumenttien loogista rakennetta
- *Datakeskeinen* näkökulma näkee XML:n keinona rakenteisen tiedon (tai datan) siirrossa, kuten tapahtuu elektronisessa kaupankäynnissä, taulukkolaskennassa tai tietokannoissa.

(Fuhr & Grossjohann 2001, 172.)

Näkökulmia vertailtaessa on melko itsestään selvää, että XML-tiedonhaun tutkimus kuuluu ensin mainitun näkökulman piiriin. Tiedonhaku rakenteisista dokumenteista voidaan vielä jakaa karkeasti kahteen eri lähestymistapaan:

- *Rakenteellinen (Structural)* lähestymistapa pyrkii rikastamaan hakua dokumenttirakenteen käyttämisellä hakuehtoina; esimerkiksi hakuavaimen pitää esiintyä tietyissä dokumenttirakenteen osissa.
- *Sisältöpohjaiset (Content-based*) suuntaukset tähtäävät dokumenttien kyselyyn nähden relevanteimpien osien palauttamiseen.

(Fuhr & Grossjohann 2001, 172.)

Tämä tutkimus keskittyy sisältöpohjaiseen tiedonhakuun. Seuraavissa alaluvuissa keskitytään XML-dokumenttien indeksoinnin ja täsmäyttämisen tutkimuksen esittelyyn.

2.4.1 XML:n indeksointiratkaisuja

Käyttötarkoituksesta riippuen XML-dokumentteja voidaan indeksoida useilla eri tavoilla. Tässä työssä indeksointiratkaisut on jaoteltu Lukin ym. (2002, 416) kartoituksen pohjalta kolmeen kategoriaan: rakenteettomiin (Flat-File Indexing), puolirakenteisiin (Semistructured Indexing) ja rakenteisiin indekseihin (Structured Indexing). Jokaisesta kategoriasta esitellään yksi tai useampia esimerkkejä. Esimerkkien

pääpaino on IR-puolen ratkaisuissa; tietokantaparadigmoihin tukeutuvia menetelmiä ei tässä työssä käsitellä.

2.4.1.1 Rakenteettomat indeksit

Yksinkertaisin tapa indeksoida XML-dokumentteja on käsitellä niitä tavallisina rakenteettomina tekstidokumentteina (flat file). Yksinkertaisimmillaan tämä tapahtuu poistamalla XML-tunnisteet kokonaan ja indeksoimalla ainoastaan dokumentin sisältö. Tällöin tosin menetetään mahdollisuus käyttää tunnisteita apuna hakuprosessissa. Tämäntyyppistä indeksointimenetelmää käytetään eräissä internet-hakukoneissa esimerkiksi MS-Word ja Adobe PDF-dokumenttien indeksointiin. (Luk ym. 2002, 416.)

Vaihtoehtoisena menetelmänä on elementtien poistamisen sijaan käyttää niitä indeksointitermeinä joko poistamalla kulmasulut tunnisteista ja käyttämällä niitä tavallisina termeinä tai vaihtoehtoisesti säilyttämällä kulmasulut ja käyttämällä niitä erikseen rakenteen indeksointiin (Luk ym. 2002, 416). Elementtien käytössä indeksointitermeinä tosin joudutaan olettamaan, että niiden nimillä on merkityssuhde sisältöön.

Yksi rakenteeton indeksointitapa on käsitellä kaikkia XML-dokumentin elementtejä itsenäisinä dokumentteina ja käsitellä niitä tavallisilla IR-menetelmillä. Ratkaisun ongelmana on kuitenkin käänteistiedoston valtava koko ja redundanssi (käänteistiedosto termille sisältää sekä sen sijaintielementit että niiden kaikki yläelementit) (Guo, Shao, Botev & Shanmugasundaram 2003, 20-21). Mass ja Mandelbrod (2004) käyttivät tämänsukuista lähestymistapaa INEX 2003:ssa; he valitsivat INEX 2002 –aineiston perusteella joukon elementtejä joissa valtaosa hyvin relevanteista tuloksista sijaitsi, erottelivat ne itsenäisiksi dokumenteiksi ja indeksoivat niiden sisällöt omiin elementtikohtaisiin käänteistiedostoihinsa (Mass & Mandelbrod 2004, 54; Carmel, Maarek, Mandelbrod & Soffer 2003, 154).

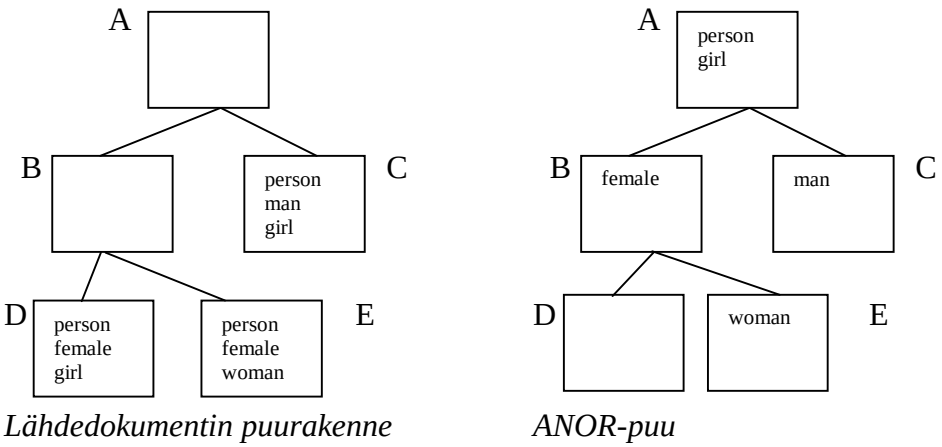
2.4.1.2 Puolirakenteiset indeksit

Puolirakenteisissa indekseissä ei käytetä välttämättä kaikkea rakenneinformaatiota indeksoinnissa. Joissain ratkaisuissa rakenne määritetään ennalta käsin ja informaatio

syötetään valmiiseen rakenteeseen. Toisissa ratkaisuissa dokumentit voivat saada omantyyppisensä (kuten puumaisen) rakenteen. (Luk ym. 2002, 416-417.)

Yksinkertaisin puolirakenteinen indeksointitapa on kenttärakenteen käyttö; XML-dokumentti esitetään joukkona erilaisia kenttiä. Dokumentti voi esimerkiksi sisältää kentät tekijälle, otsikolle ja kustantajalle. Dokumenttien sisältö sen jälkeen indeksoidaan omiin kenttiinsä. Kenttärakenne-indeksointi mahdollistaa perinteisten hakukoneiden suhteellisen helpon laajennettavuuden rakenteisten dokumenttien käsittelyyn, jos indeksoitava aineisto ei ole kovin monimutkainen rakenteeltaan. (Luk ym. 2002, 417.) Kenttärakenne-indeksointia on käytetty mm. Berkeleyn yliopiston Cheshire II –hakujärjestelmän (Larson 2004, 38-45) INEX-kokoelman indeksointiratkaisuna.

Puumallisista ratkaisuista Lee, Yoo, Yoon ja Berra (1996) tutkivat useita vaihtoehtoja rakenteisen tiedon indeksointiin ja päätyivät johtopäätökseen, että käänteistiedosto kaikista elementeistä ilman toistoa (Inverted Index for All Nodes Without Replication, ANOR) tuotti parhaat tulokset hakuaikojen ja käänteistiedoston koon suhteen (Lee ym. 1996, 98). Menetelmä toimii siten että kokoelma rakenteisia dokumentteja jäsennetään yhdeksi virtuaaliseksi (osa elementeistä ei todellisuudessa ole olemassa) dokumenttipuuksi jonka kaikki elementit saavat ainutkertaisen tunnisteen. Jokainen Indeksointitermi talletetaan sen jälkeen vuorollaan elementtiin, joka on ko. termin kaikkien esiintymien alin yhteinen yläelementti. Tämän myötä jokainen termi on tallennettu ainoastaan yhteen elementtiin koko puussa. (Trotman 2004, 621.)



Lähdedokumentin puurakenne *ANOR-puu*

Kuvio 4. ANOR-puu (Trotman 2004, 621)

ANOR-rakenteen ongelmana on se, että termien sijainnit indeksissä eivät ole samat kuin niiden sijainnit alkuperäisissä dokumenteissa, vaan ne löytyvät esiintymien alimmista yhteisistä yläelementeistä. Esimerkki tämän tuottamasta ongelmatilanteesta löytyy kuviosta 4, jossa termi ”girl” on ANOR-puussa väärin kohotettu elementtiin A. Tämä virhe mahdollistaa tietyissä tapauksissa termin ”girl” tulevan hakutulokseen paikasta missä se ei todellisuudessa sijaitse. (Trotman 2004, 621.)

2.4.1.3 Rakenteiset indeksit

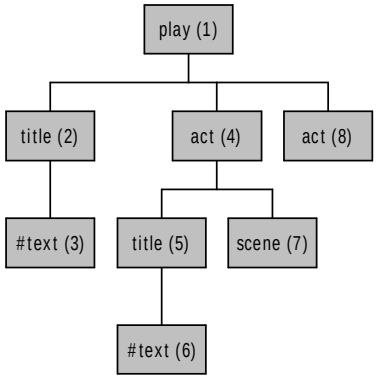
Rakenteisia indeksejä voidaan soveltaa aineistoihin jotka omaavat hyvin määritellyn säännöllisen rakenteen (vrt. luku 2.1.1). Rakenteisten indeksien käytössä on selkeitä etuja verrattuna muihin ratkaisuihin. Haku rakenteisesta indeksistä on nopeaa ja hyvin optimoitavissa. Hakutulokset ovat myös hyvin tarkkoja verrattuna esimerkiksi ANOR-indeksointiin. Näiden syiden takia rakenteiset indeksit ovat erityisen soveltuvia rakenteisen tiedon (kuten XML-dokumenttien) käsittelyyn. (Luk ym. 2002, 418.)

Eräs tapa rakenteiseen indeksointiin on polkuindeksointi. Geva ja Leo-Spork (2004) käyttivät XML-dokumenttien osiin viittaamiseen käytettävää XPath-teknologiaa (ks. Clark & DeRose 1999) XML-käänteistiedoston luomiseen. Heidän toteutuksessaan jokainen dokumentissa esiintyvä termi tunnistetaan kolmen osan avulla:

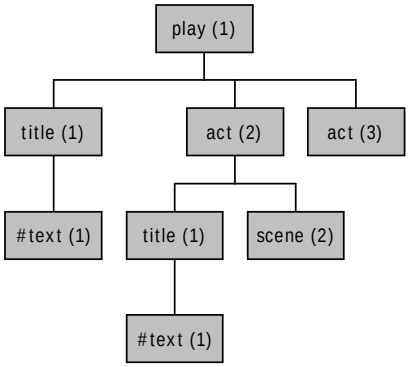
- Tiedostopolku, kuten C:/INEX/ex/2001/x0321.xml
- Absoluuttinen XPath-konteksti, kuten /article[1]/bdy[1]/sec[5]/p[3]
- Termin sijaintipaikka edellä määritellyn elementin sisällä.

XPath-indeksointi mahdollistaa termien sijainnin äärimmäisen tarkan indeksoinnin, mutta sen ongelmana on valtava redundanssi ja siitä seuraava käänteistiedoston suuri koko. Redundanssia syntyy absoluuttisten XPath-kontekstien identtisten osien monikertaisissa esiintymissä (esimerkiksi pelkästään kontekstin viimeinen osa vaihtuu), samassa elementissä sijaitsevien useiden avainten vaatiessa omat, mutta identtiset kontekstit sekä monien kontekstien löytyessä lähes kaikista dokumenteista. (Geva & Leo-Spork 2004, 110-111.)

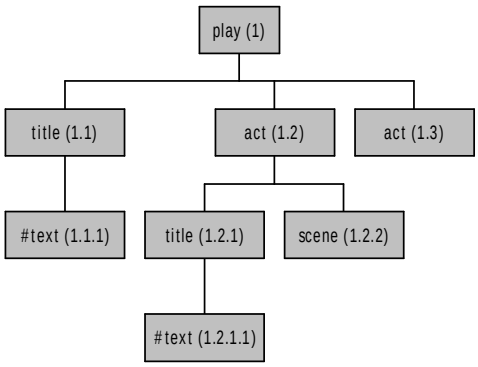
Tatarinov ym. (2002) esittivät kolme eri menetelmää XML-dokumenttien rakenteen indeksointiin – globaalin järjestysluvun (Global order encoding), paikallisen järjestysluvun (Local order encoding) ja Dewey-järjestysluvun (Dewey order encoding). Globaalissa järjestyksessä jokaiselle elementille annetaan järjestysnumero joka ilmaisee elementin absoluuttisen sijainnin dokumentissa. Paikallisessa järjestyksessä elementeille annetaan järjestysluku, joka ilmaisee niiden sijainnin suhteessa sisarelementteihinsä. Dewey-järjestys perustuu Deweyn luokitusmenetelmään. Siinä elementille annetaan tunnisteeksi vektori, joka ilmaisee polun dokumentin juurielementistä kohde-elementtiin. Jokainen elementti sisältää tunnisteessaan tiedon vanhempansa ja itsensä paikallisesta järjestyksestä ja sen myötä määrittää täsmällisesti sijaintinsa dokumentissa. Ominaisuuksiltaan Dewey-järjestys on yhdistelmä globaalin ja paikallisen järjestyksen ominaisuuksista. (Tatarinov ym. 2002, 3-4.)



Globaali järjestys



Paikallinen järjestys



Dewey-järjestys

Kuvio 5. Globaali-, paikallinen ja Dewey-järjestys.

Guo ym. (2003) käyttivät Dewey-järjestystä XRANK-hakukoneensa käänteistiedoston luomisessa. He kokeilivat kahta eri järjestämistapaa käänteistiedoston luonnissa; ensimmäisessä avaimen esiintymät järjestettiin tunnisteen mukaan nousevaan järjestykseen (Dewey Inverted List, DIL) ja toisessa sanan esiintymistä lasketun

ElemRank-painoarvon mukaan laskevaan järjestykseen (Ranked Dewey Inverted List, RDIL). Molemmissa järjestämistavoissa oli puutteensa – DIL oli tehoton yleisten sanojen tai suurten aineistojen käsittelyssä ja RDIL taas toimi heikosti kun suhteellisen yleiset sanat olivat hajallaan ympäri kokoelmaa. Ratkaisuksi ongelmaan tutkijat kehittivät hybridin DIL:stä ja RDIL:stä, HDIL:in (Hybrid Dewey Inverted List). HDIL sisältää täysimittaisen DIL-käänteistiedoston, pienen RDIL-käänteistiedoston sekä ennakointialgoritmin, mikä arvioi milloin kannattaa siirtyä ensin läpikäytävästä RDIL:stä DIL-käänteistiedostosta hakemiseen. (Guo ym. 2003, 21-24.)

2.4.2 XML-elementtien täsmäytys sisältöhauissa

XML-dokumenttien rakenteisuus aiheuttaa uusia haasteita myös täsmäytyksen osalta. Koska kyselyn tuloksena voidaan palauttaa myös dokumenttien osia kokonaisen dokumentin sijaan, joitain perinteisten tiedonhakumallien perusoletuksia joudutaan miettimään uudelleen.

Suuri kysymys XML-elementtien täsmäytyksessä, käytettävästä menetelmästä riippumatta, on avainten frekvenssien laskeminen. Perusoletukset avaimen esiintymisjakaumasta kokonaisessa dokumentissa eivät välttämättä pidä paikkaansa elementtitasolla, joten frekvenssin merkitystä elementtitasolla voidaan joutua harkitsemaan uudelleen. Samalla joudutaan myös miettimään uudelleen perinteisessä tf*idf –painonlaskennassa käytettävän käänteisen kokoelmafrekvenssin (idf) merkitystä. idf on perinteisesti laskettu dokumenttitasolta, mutta palautettavan yksikön pienentyessä on myös sen määritelmä arvioitava uudelleen, esimerkiksi elementtitasolle. (Luk ym. 2002, 424.)

Suurin osa moderneista (ja INEX-hankkeessa käytetyistä) tiedonhakujärjestelmistä ovat osittaistäsmäyttäviä ja perustuvat siinä joko vektorimalliin tai todennäköisyyslaskentaan. Seuraavassa esitellään muutamia em. menetelmiin perustuvia ratkaisuja. Ratkaisut on valittu vuoden 2003 INEX-hankeessa parhaiten menestyneiden joukosta.

2.4.2.1 Vektorimalliin perustuvia ratkaisuja

Gerald Saltonin (1989) kehittämä vektorimalli (Vector Space Model) on kuuluisin ja ehkä eniten tiedonhakujärjestelmien kehitykseen vaikuttanut osittaistäsmäytysmalli. Vektorimallissa sekä kyselyt että dokumentit esitetään avainten vektoreina:

$$\boldsymbol{D_i} = (a_{i1}, a_{i2},...a_{it})$$
$$\boldsymbol{Q_j} = (q_{j1}, q_{j2},...q_{jt})$$

joissa kertoimet a_{ik} ja q_{jk} osoittavat avaimen k arvoa joko dokumentissa D_i tai kyselyssä $Q_{i.}$ a_{ik} ja q_{jk} voidaan arvottaa yhdeksi jos avain esiintyy dokumentissa D_i tai kyselyssä Q_i tai nollaksi avaimen puuttuessa. Vaihtoehtoisesti kertoimille voidaan antaa numeeriset painoarvot välillä [0,1] riippuen avainten tärkeydestä kyseessä olevassa dokumentissa tai kyselyssä.(Salton 1989, 313-314.)

Kun termien painoarvot on saatu määriteltyä, tarvitaan menetelmä mittaamaan kyselyvektorin ja dokumenttivektoreiden samankaltaisuutta. Yleensä mittaamiseen käytetään kosinifunktiota. Funktion tulos on vektoreiden välisen kulman kosini arvoltaan aina välillä [0,1]. Arvo 1 edustaa täydellistä samankaltaisuutta jota aiemminmainittu Boolen logiikka vaatisi. Arvo nolla taas edustaa vektorien täydellistä erilaisuutta. Kyselyn ja dokumenttien välille voidaan määrittää kynnysarvo jonka sen on ylitettävä tullakseen lasketuksi mukaan palautettavaan tulosjoukkoon. (Järvelin 1995, 123-125.)

Mass ja Mandelbrod (2004) pyrkivät kokeissaan laajentamaan klassista vektorimallia ottamaan huomioon elementtitason frekvenssit. Heidän ratkaisunsa elementtien sisäkkäisyyden tuomaan ongelmaan oli jakaa kokoelma kuuteen eri käänteishakemistoon elementtien koon mukaan (ks. 2.4.1.1) ja täsmäyttää ne erikseen käyttäen alla olevan järjestelmän vakiotäsmäytyskaavaa. Tämän jälkeen kaikista indekseistä saadut tuloslistat normalisoitiin täsmäyttämällä annettu kysely itseään vasten kuin kysely olisi dokumentti kokoelmassa. Tästä saadulla tuloksella normalisoituna eri indekseistä saatavat tulokset olivat verrannollisia keskenään ja niistä voitiin muodostaa yksi yhdistetty tuloslista. (Mass & Mandelbrod 2004, 53-55.)

Massin ja Mandelbrodin lähestymistapa on hyvin yksinkertainen ja helposti olemassa olevien järjestelmien päälle laajennettavissa, mutta siinä on ongelmansa; potentiaaliset palautettavat elementit on tiedettävä etukäteen ja useat päällekkäiset käänteistiedostot käsittelyineen vievät sekä tilaa että prosessointiaikaa. (Mass & Mandelbrod 2004, 58.)

Crouch, Apte ja Bapat (2004) käyttivät kokeissaan Smart-tiedonhakujärjestelmää ja erityisesti sen laajennetun vektorimallin (extended vector space model) sovellusta. Laajennetussa vektorimallissa dokumentin sisältämä informaatio pystytään jakamaan erillisiin luokkiin; esimerkiksi tekijän nimi ja julkaisupäivämäärä varsinaisen sisällön lisäksi. Laajennetussa vektorimallissa dokumenttivektori koostuu alivektoreista joista kukin edustaa omaa luokkaansa. Crouch ym. järjestivät INEX-kokoelman 18 eri luokkaan.

CO-kyselyiden käsittelyssä 18 luokasta käytettiin kahdeksaa jotka sisälsivät artikkelien tekstiosat (*abs, ack, atl, bibl_atl, bibl_ti, ed-intro, kwd, bdy*). Kyselyiden avaimet täsmäytettiin kaikkia näitä alivektoreita vasten. Avainten painoja laskettaessa tutkijat kokeilivat sekä *Lnu/Itu* –painotuksen (ks. Singhal, Buckley & Mitra 1996) varianttia sekä tf*idf-varianttia, ensin mainitun toimiessa paremmin.

SMART tuotti varsin hyviä tuloksia INEX 2003:ssa siitä huolimatta että se ei, perinteisenä hakukoneena, pystynyt käytännössä ollenkaan käsittelemään dokumenttien rakenteisuutta. Laajennetun vektorimallin kyky toteuttaa rakenteisuuden mahdollistamia lisäominaisuuksia jäi tutkimuksessa kyseenalaiseksi. (Crouch, Apte & Bapat. 2004, 89-93.)

2.4.2.2 Todennäköisyyslaskentaan perustuvia ratkaisuja

Maronin ja Kuhnsin (1960, 216-224) ensimmäisenä esittämä todennäköisyyslaskentaan perustuva osittaistäsmäytysmalli on, vektorimallin ohella, paljon käytetty ja tutkittu täsmäytysmenetelmäryhmä. Todennäköisyysmalleja on kehitetty useita erilaisia, mutta niiden perusidea voidaan kiteyttää kysymykseen: ”Mikä on todennäköisyys sille, että *tämä* dokumentti on relevantti suhteessa *tähän* kyselyyn?” (Sparck-Jones, Walker & Robertson 2000a, 780; 783.)

Yleisesti ottaen todennäköisyysmalleissa on tapahtuma-avaruus *Q x D*, jossa *Q* edustaa kaikkia mahdollisia kyselyitä ja *D* kokoelman kaikkia dokumentteja. Erilaisten todennäköisyysmallien erot ovat lähinnä niiden tavassa esittää ja kuvailla kyselyitä ja dokumentteja. (Crestani, Lalmas & van Rijsbergen. 1998, 530.)

Eräs hyvin tunnettu todennäköisyyslaskentaan perustuva tiedonhakujärjestelmä on Okapi (Robertson & Walker, 1999). Sen BM25 –painotusfunktiota on sovellettu useisiin erilaisiin tiedonhakujärjestelmiin. BM25 -funktio voidaan esittää seuraavasti:

$$CW = \frac{tf_i(k_1+1)}{k_1*((1-b)+b\frac{dl}{avdl}+tf_i} * log\frac{N}{n}$$

missä

CW = yhdistetty paino

N = dokumenttien määrä kokoelmassa

n = avaimen sisältävien dokumenttien määrä kokoelmassa

k1 ja *b* ovat vakioita joiden arvot riippuvat käytettävien kyselyiden ja mahdollisesti dokumenttikokoelman ominaisuuksista.

tf_i = termin frekvenssi dokumentissa

dl ja *avdl* ovat dokumentin pituus ja dokumenttien keskimääräinen pituus kokoelmassa. (Robertson & Walker 1999, 2; Sparck-Jones, Walker & Robertson 2000b, 811-814.)

BM25:n pituusnormalisoinnissa käytettävät vakiot, varsinkin *b*, mahdollistavat kaavan toiminnan säätelemisen. TREC-kokeilut ovat osoittaneet että *b*:n arvon pitäisi olla noin 0,75 (Sparck-Jones, Walker & Robertson 2000b, 813)

HyREX- hakukoneessa (Abolhassani, Fuhr & Malik 2004) käytettiin vuoden 2003 INEX-kierroksella aikaisemman BM25 –pohjaisen ratkaisun sijaan Amatin ja Van Rijsbergenin (2002, 362-373) DFR–mallia jossa avainten painot johdetaan mittaamalla poikkeamaa varsinaisesta aineistosta mitatun avainten jakautuman ja satunnaisjakauman välillä. Tutkijat johtivat DFR-mallista neljä erilaista painojenlaskemistapaa.

Suoraan sovellettuna DFR-malli toimi BM25-ratkaisua huonommin, mutta kun siihen lisättiin pituus- ja hierarkiatasonormalisoinnit tulokset parantuivat huomattavasti.

Tulokset olivat tarpeeksi hyviä oikeuttamaan DFR-mallin jatkokehityksen. (Abolhassani, Fuhr & Malik 2004, 27-29, 31-32.)

Sigurbjörnsson, Kamps ja de Rijke. (2004) käyttivät FlexIR –hakujärjestelmässään Todennäköisyyslaskentaan perustuviin menetelmiin kuuluvaa monijäsenistä kielimallia (multinomial language model) ns. Jelinek-Mercer –tasoituksella (ks. Hiemstra 2001). Tutkijat muuntelivat tasoituksen voimakkuutta ja pyrkivät sitä kautta vaikuttamaan järjestelmän palauttamien elementtien kokoon. Tasoitusparametrin arvolla 0.9 tasoituksen vaikutus oli hyvin vähäistä ja järjestelmä vaikutti suosivan suuria, artikkelitason elementtejä. Arvolla 0.1 järjestelmä palautti lähinnä pieniä elementtejä.

Tutkijat huomasivat että tasoituksen astetta muuttamalla voidaan vaikuttaa dramaattisesti palautettavien elementtien keskimääräiseen kokoon. He myös kokeilivat elementin kontekstin huomioon ottamista niiden relevanssin arvioinnissa yhdistämällä elementistä saadun todisteen artikkelitason todisteen kanssa. (Sigurbjörnsson, Kamps & de Rijke 2004, 20-21.)

3 TRIX (Tampere Retrieval and Indexing engine for XML)

TRIX on Tampereen yliopiston informaatiotutkimuksen laitoksella kesällä 2004 tutkimuskäyttöön kehitetty, erityisesti XML-dokumenttien indeksointiin ja hakuun suunniteltu tiedonhakujärjestelmän prototyyppi. TRIX perustuu indeksointijärjestelmässään hierarkkiseen tietorakenteeseen ja XML-tiedonhaun erikoisvaatimukset huomioonottavaan tf*idf-pohjaiseen osittaistäsmäytykseen. Järjestelmä toimii PC/Windows–alustalla ja on ohjelmoitu kokonaisuudessaan C++ -ohjelmointikielellä tehokkuuden maksimoimiseksi. TRIXin tässä työssä käytettävä versio pystyy suorittamaan sisältöpohjaisia hakuja ilman rajoituksia ja rajallisesti rakenne-ehtoja sisältäviä hakuja.

TRIXin kehittämisen motivaationa oli tuottaa INEX-hankkeen vuoden 2004 kierrokselle hakujärjestelmä rakenteisille dokumenteille ja tutkia hierarkkiseen tietorakenteeseen perustuvan indeksointijärjestelmän toimivuutta XML-tiedonhaussa. (Kekäläinen, Junkkari, Arvola & Aalto 2005, 72-75.)

TRIX toimii joko vuorovaikutteisesti ottamalla vastaan käyttäjän hakutilanteessa kirjoittamaan kyselyn tai eräajomoodissa ottamalla vastaan kyselysarjan erillisestä tekstitiedostosta. Tämän lisäksi käyttäjä voi ohjelman käynnistyksen yhteydessä määritellä painotuskaavan käyttämät vakiot ja valita kahdesta eri hakuavainten painojen yhdistämismenetelmästä sekä kolmesta erilaisesta päällekkäisten elementtien käsittelytavasta:

- Päällekkäisiä elementtejä ei hyväksytä lainkaan (oletus)
- Osittainen päällekkäisyys sallitaan; suorat ylä- ja alielementit jotka saavat palautettavaa elementtiä alemman painon hylätään.
- Kaikki päällekkäiset elementit hyväksytään.

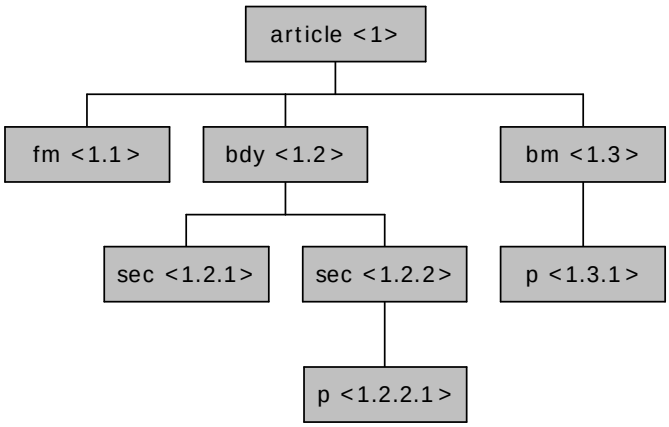
3.1 Indeksointimenetelmä

TRIXin indeksointimenetelmä perustuu puumaiseen hierarkkiseen tietorakenteeseen. Indeksointimenetelmä perustuu Niemen (1983, 151-157) kehittämään hierarkkisten tietorakenteiden indeksointimenetelmään. Vastaavia menetelmiä on sovellettu mm. osa-kokonaisuussuhteiden mallinnukseen oliotietokannoissa (Junkkari 2004) sekä ns. NF^2

(Non-First-Formal-Form) relaatioiden indeksointiin (ks. esim. Järvelin ja Niemi 1995, 102-110, Niemi ja Järvelin 1995, 215-231). Indeksointimenetelmä muistuttaa myös Dewey-luokitusjärjestelmän notaatiota (vrt. Tatarinov ym. 2002, 3-4). TRIX-hakukoneessa indeksointimenetelmää sovelletaan ns. D- ja I-puissa.

D-puut on järjestelmän esitys dokumenttien rakenteesta. D-puissa elementti saa numerotunnisteen joka ilmaisee sen suhteellisen sijainnin verrattuna dokumentin muihin elementteihin. Elementin numerotunniste koostuu sen äitielementin tunnisteesta sekä sen järjestysnumerosta sisarelementtiensä kesken. Tunnisteen avulla jokainen elementti voidaan tunnistaa yksiselitteisesti seuraamalla tunnistenumeron ilmaisemaa vektoria. Samalla saadaan myös selville elementtien väliset vanhemmuus-jälkeläissyyssuhteet.

Esimerkkinä kuviossa 6 juurielementti *article* saa juurielementtinä tunnisteen 1. Articlen lapsielementit saavat tunnisteet 1.1, 1.2 ja 1.3. *bdy*:n (1.2) lapsielementit saavat vuorostaan arvot 1.2.1 ja 1.2.2 jne.

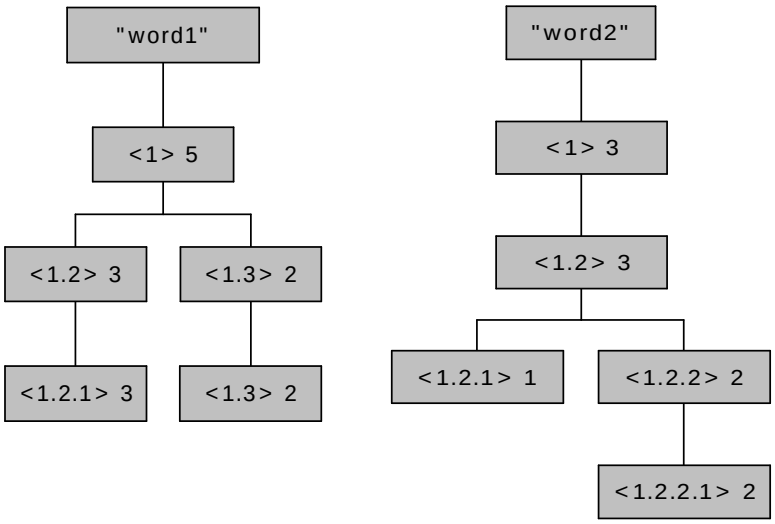


Kuvio 6. Esimerkki D-puun periaatteesta

Ohjelmassa D-puut sisältävät pelkästään dokumentin rakenteen; kaikki sisältö on korvattu elementin sisältämien indeksoitujen sanojen määrällä. D-puista saadaan täsmäyttämistä varten kaikkien kontekstielementtien määrä ja kaikkien elementtien nimet.

I-puita voidaan pitää järjestelmän varsinaisena käänteistiedostona. I-puut sisältävät indeksoitujen avainten lukumäärän ja sijaintipaikat D-puissa sijaintielementin tarkkuudella. I-puista saadaan täsmäyttämistä varten myös avaimen frekvenssi

elementeissä ja avaimen sisältävien kontekstielementtien määrä. Kuviossa 7 on esimerkki I-puista kahdelle avaimelle. Esimerkissä avainten sijaintipaikat ovat kulmasuluissa ja numero on sen frekvenssi elementissä.



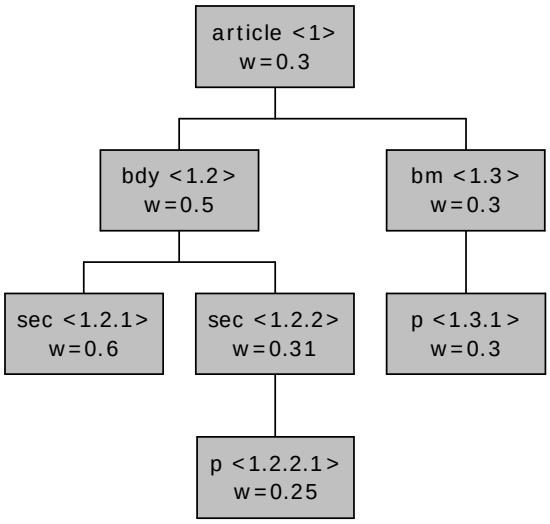
Kuvio 7. I-puita.

Indeksoitavasta aineistosta on poistettu ennen indeksointia kaikki tyylimäärittelyelementit, matemaattisia kaavoja kuvaavien elementtien sisällöt sekä muita järjestelmän toiminnan kannalta epäolennaisia elementtejä ja/tai niiden sisältöjä. Indeksoitaessa aineistosta on poistettu sulkusanat ja avaimet on ajettu Porter-sanastemmerin läpi.

3.2 Haun suoritus ja tulosten lajittelu

Kyselyn vastaanotettuaan TRIX alkaa käydä läpi dokumenttikokoelmaa. Löydettyään ensimmäisen dokumentin joka sisältää jonkun kyselyn avaimista (paitsi ei-toivotuksi ilmaistun avaimen) järjestelmä lataa ko. dokumentin D-puun ja muodostaa siitä tyhjän ns. Q-puun. Tämän jälkeen järjestelmä hakee kyselyn avainten I-puiden käsittelyssä olevaa dokumenttia vastaavat osat ja liittää ne yksitellen Q-puuhun laskien samalla painoja avaimille dokumentin elementeissä. Avainten painojen laskeminen ja yhdistäminen kuvataan myöhemmin tässä kappaleessa.

Kuviossa 8 on kuvion 5 D-puusta muodostettu Q-puu laskettuine painoineen.



Kuvio 8. Q-puu.

Kun käsittelyssä olevan dokumentin kaikki avaimet on käsitelty, palautetaan siitä suurimman painon saanut elementti joka lisätään tuloslistaan. Palauttamista jatketaan kunnes kaikki dokumentin elementit on joko palautettu, hylätty päällekkäisinä tai hylätty painoltaan 1500 parhaan elementin joukkoon kuulumattomana. Tämän jälkeen järjestelmä siirtyy käsittelemään seuraavaa dokumenttia.

Q-puuhun laskettavien painojen saamiseksi tarvitaan D-puista kaikkien kontekstielementtien lukumäärä *N* ja I-puista avainten frekvenssit ja avaimen sisältävien kontekstielementtien määrät. Avainten paino w_k lasketaan tf*idf -pohjaisella kaavalla joka on johdettu Hawkingin, Thistlewaiten ja Craswellin (1998, 1) TREC-6 -kokeissa käyttämästä Okapi BM25-painotusfunktion Cornell-variantista:

$$w_k = \frac{kf}{kf + v*\left(a + b*\frac{ef_c}{\sqrt{ef_k}}\right)} * \frac{\log\left(\frac{N}{n}\right)}{\log N}$$

jossa

kf = hakuavaimen frekvenssi elementissä

ef_k = elementin lapsikontekstielementtien, joissa avain k, määrä

ef_c = elementin kaikkien lapsikontekstielementtien määrä

v = vakio, oletusarvo 2

a = vakio, oletusarvo 0.6

b = vakio, arvoltaan 1-*a*

N= kaikkien kontekstielementtien määrä
n= avaimen sisältävien kontekstielementtien määrä

Kaava tuottaa avaimelle [0..1] –asteikolle sijoittuvan painoarvon.

TRIX:ssä käytetyssä laskentakaavassa tärkeimpänä innovaationa on pituusnormalisoinnin suoritustapa; se perustuu lapsikontekstielementtien määrän käyttämiseen elementtien sisällön pituuden sijaan.

Hakuavainten esiintymistä voidaan kontrolloida seuraavasti:
Jos hakuavaimen eteen on asetettu ei-toivottavuutta kuvaava “-” -operaattori, otetaan avaimen painosta vastaluku:

$$w_k^- = -w_k$$

Operaatio laskee avaimen painoa voimakkaasti ja siten pudottaa avaimen sisältämän elementin sijoitusta tuloslistassa.

Jos hakuavaimen eteen on asetettu toivottavuutta kuvaava ”+” -operaattori, otetaan avaimen painosta neliöjuuri:

$$w_k^+ = \sqrt{w_k}$$

Operaatio suurentaa avaimen painoa voimakkaasti, johtuen neliöjuuren käyttäytymisestä arvoilla jotka ovat pienempiä kuin 1 mutta suurempia kuin 0, ja siten parantaa avaimen sisältämän elementin sijoitusta tuloslistassa.

Hakuavaimen ollessa fraasi vaaditaan sen kaikkien osien esiintyvän samassa kontekstielementissä. Fraasin painoa laskettaessa käytetään sen pienimmän frekvenssin omaavan osan frekvenssiä. TRIX ei tue tarkempaa avainten etäisyyshakua.

Painotuskaavassa on kolme vakiota (*v,a,b)* joiden arvoja muuttamalla säädetään pituusnormalisoinnin vaikutusta kaavan tf-osuudessa. Suurin vaikutus on vakioiden *a* ja *b* suhteella. Painoarvoja muuttamalla voidaan vaikuttaa palautettavien elementtien kokoon ja siten optimoida järjestelmän toimintaa käytössä olevassa kokoelmassa.

Täsmäytettävän elementin lopullinen paino saadaan yhdistämällä kaikkien siinä esiintyvien avainten painot. TRIX:ssä käytetään kahta vaihtoehtoista painojen yhdistämismenetelmää:

1) Keskiarvon laskeminen kaikkien kyselyn avainten painosta
2) Yhdistäminen ns. Einsteinin summana (ks. Mattila 1998) tunnettua assosiatiivista sumean logiikan operaatiota käyttäen:

$$w_{1,2} = \frac{w_1 + w_2}{1 + w_1 \cdot w_2}$$

4 Koeasetelman kuvaus

Tässä kappaleessa tarkastellaan tutkimuksen evaluointiympäristöä kappaleessa 1.2 esitetyn listan mukaisesti.

4.1 Tiedonhakujärjestelmä

Testattavana hakujärjestelmänä tässä tutkimuksessa toimi kappaleessa 3 esitelty TRIX. Tutkimuksessa testattiin hakujärjestelmän tehokkuutta sisältöpohjaisten kyselyjen suorittamisessa käyttäen avainten yhdistämisessä sekä keskiarvoon että Einsteinin summaan perustuvia menetelmiä. Lisäksi tutkittiin painotuskaavan vakioiden muuttamisen vaikutuksia järjestelmän tehokkuuteen.

Tutkimusta varten järjestelmällä ajettiin seuraavat kyselysarjat:

- Vuoden 2003 kyselyt, vakion *a* arvot {0.2, 0.4, 0.6, 0.8, 0.9}, yhdistäminen keskiarvolla
- Vuoden 2004 kyselyt, vakion *a* arvot {0.2, 0.4, 0.6, 0.8, 0.9}, yhdistäminen keskiarvolla
- Vuoden 2003 kyselyt, vakion *a* arvot {0.2, 0.4, 0.6, 0.8, 0.9}, yhdistäminen Einsteinin summalla
- Vuoden 2004 kyselyt, vakion *a* arvot {0.2, 0.4, 0.6, 0.8, 0.9}, yhdistäminen Einsteinin summalla.

4.2 Testikokoelma

Tiedonhaun testikokoelmat ovat operationaalisten hakuympäristöjen abstraktioita jotka mahdollistavat erilaisten hakustrategioiden ja – menetelmien kokeilemista kontrolloiduissa laboratorio-olosuhteissa (Voorhees 2002, 3). Testikokoelmat ovat olleet olennainen osa tiedonhaun tutkimusta sen alkuvaiheista asti. Alkuaikojen kuuluisin testikokoelma kehitettiin ns. Cranfieldin kokeiden yhteydessä (ks. Sparck-Jones 1983, 256-284). Kuuluisimpia nykyisistä kokoelmista on TREC (The Text REtrieval Conference, http://trec.nist.gov) -konferenssien yhteydessä rakennetut kokoelmat. Testikokoelmat koostuvat kolmesta osasta: **Dokumenttijoukosta**,

tiedontarpeiden tai **aiheiden (topics) joukosta** ja **relevanssiarvioista**, jotka osoittavat mitkä dokumentit pitäisi palauttaa mihinkin aiheeseen liittyen. (Voorhees 2002, 3.)

INEX –testikokoelma muodostuu normaalien testikokoelmien tapaan kolmesta osasta: dokumenteista, aiheista ja relevanssiarvioista. XML-testikokoelmat poikkeavat kuitenkin usealla tavalla perinteisistä tiedonhaun testikokoelmista. Perinteisissä testikokoelmissa dokumentteja pidetään rakenteettomina yksikköinä, kyselyjä pidetään yleensä termien tai fraasien kokoelmina ja relevanssiarviot määrittävät onko koko dokumentti relevantti vai ei. XML-dokumenttien sisältö on sen sijaan jaoteltu pienempiin, sisäkkäisiin elementteihin joista jokainen voi olla haettava kohde. Kyselyissä XML-kyselykielet mahdollistavat sekä sisällöllisten että rakenteellisten ehtojen asettamisen. Myös relevanssiarvioissa on otettava huomioon dokumenttien rakenteisuus. (Sigurbjörnsson, Larsen, Lalmas & Malik 2004, 1.)

4.2.1 Dokumentit

Dokumenttikokoelma saatiin lahjoituksena IEEE Computer Societylta. Se sisältää XML-merkatut kokotekstit 12107 artikkelista jotka ovat peräisin 18 tieteellisestä julkaisusta joita IEEE Computer Society on julkaissut vuosina 1995-2002. Kokoelman koko on yhteensä 494 MB. Kokoelman rakenne on todettu riittävän monimutkaiseksi toimiakseen testikokoelmana; sen DTD määrittää 192 erilaista elementtiä. Keskimääräinen kokoelman artikkeli sisältää 1532 XML-elementtiä. (Gövert ja Kazai 2003, 4.)

4.2.2 Aiheet ja kyselyt

Hakuaiheet ovat kirjallisia esityksiä hakutehtävää motivoivasta tiedontarpeesta. Ne koostuvat yleensä tunnistetietojen lisäksi tehtävänannosta, kuvauksesta joka täsmentää tehtävää ja ohjeista mitkä ohjaavat relevanttien dokumenttien tunnistamisessa. Hakuaiheet ovat INEX-hankkeeseen osallistuvien ryhmien kehittämiä; jokaisen osallistuvan organisaation odotetaan toimittavan kuusi hakuaihetta kyseessä olevan vuoden kierrokselle (Sigurbjörnsson ym. 2004, 4).

XML- hakujärjestelmillä on mahdollista suorittaa monimutkaisempia kyselyitä kuin perinteisillä rakenteettoman tiedon hakujärjestelmillä; hauissa voidaan esimerkiksi käyttää hyväksi tiedon rakenteellisuutta ja siten rajata haku vain tiettyihin elementteihin kokoelmassa. Toisaalta, koska dokumenttien rakenne ei ole välttämättä tiedossa, järjestelmien täytyy myös pystyä suorittamaan kyselyjä joissa ei ilmaista rakenteellisia ehtoja. Tämän vuoksi INEX–hankkeessa päädyttiin käyttämään kahta eri aihetyyppiä:

Pelkkä sisältö (CO, Content Only) – kyselyissä dokumenttien rakenne jätetään huomiotta. Niitä voidaan pitää jossain määrin perinteisinä tiedonhakuaiheina. Niiden tuoma haaste XML-tiedonhakuun on hakutuloksina saatavien elementtien vaihteleva koko ja mahdollinen päällekkäisyys

Sisältö ja rakenne (CAS, Content-and-Structure) – kyselyissä aiheiden kuvauksessa on suora viittaus XML-rakenteeseen joko ilmaisemassa haluttua kohdetta tai ilmaisemassa haluttua hakuavainten sijaintia. (Gövert & Kazai 2002, 2.)

Tässä työssä keskitytään pelkän sisällön (CO) aihetyypin hakujen tutkimiseen johtuen testattavan järjestelmän CAS-aiheiden käsittelyn keskeneräisyydestä.

INEX –kokoelman aiheiden formaatti ja kehitysmenettely perustuu TREC-testikokoelman ohjeisiin aiheiden teosta muutettuna ottamaan huomioon kaksi erityyppistä aiheryhmää (CO ja CAS). Aiheiden formaattiin lisättiin mahdollisuus osoittaa sisältyvyysehtoja siitä mitkä elementit pitää palauttaa kyselyssä. (Gövert & Kazai 2002, 4.)

Sekä CO- että CAS- aiheet koostuvat neljästä osasta: nimeke (title), kuvaus (description), kerronta (narrative) ja avainsanat (keywords). Näistä nimeke-osa eroaa CO-ja CAS-kyselyjen välillä. CO-aiheen nimeke on lyhyt sisältäen 2-5 termiä jotka kuvaavat aihetta. Fraasit asetetaan lainausmerkkien (“ ”) sisään. Termien haluttavuutta tai ei-toivottavuutta voi lisäksi ilmaista plus- ja miinusmerkeillä. Kuviossa 9 on esimerkki CO-tyypin aiheesta vuodelta 2003.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE inex_topic SYSTEM "topic.dtd">
<inex_topic topic_id="126" query_type="CO" ct_no="25">
<title>Open standards for digital video in distance learning</title>
```

```
<description>Open technologies behind media streaming in distance
learning projects</description>
<narrative> I am looking for articles/components discussing
methodologies of digital video production and distribution that respect
free access to media content through internet or via CD-Roms or DVDs in
connection to the learning process. Discussions of open versus
proprietary standards of storing and sending digital video will be
appreciated. </narrative>
<keywords>media streaming,video streaming,audio streaming,
digital video,distance learning,open standards,free
access</keywords>
</inex_topic>
```

Kuvio 9. vuoden 2003 CO-aihe

Kysely on tiedonhakujärjestelmän ymmärtämään muotoon muokattu esitys tiedontarpeesta. Kysely on rakenteeton jos se koostuu pelkästään hakuavaimista, tai rakenteellinen jos siinä käytetään esimerkiksi boolen operaattoreita ilmaisemaan avainten välisiä suhteita. Tässä työssä kaikki kyselyt ovat rakenteettomia.

Aikaisempina vuosina osallistujat ovat voineet käyttää kyselyitä muodostaessaan tietoja title-, description- ja keywords-kentistä. Vuonna 2004 sääntöjä muutettiin siten, että ainoastaan title-kentän tietoja sai käyttää kyselyissä. Muutoksen motivaationa oli saada kyselyistä lyhyempiä ja enemmän todellisten käyttäjien tekemien kyselyjen kaltaisia (Lalmas 2004). Kyselyt on mahdollista muodostaa joko manuaalisesti (ihminen muodostaa kyselyt) tai automaattisesti (tietokone suodattaa kyselyt aiheista), mutta evaluoitavaksi lähetettävistä ajoista vähintään yksi täytyy suorittaa automaattisesti muodostettuja kyselyitä käyttäen. Kaikki Tampereen yliopiston vuonna 2004 evaluoitavaksi lähettämät ajot ja kaikki tässä tutkimuksessa suoritettavat ajot ovat automaattisia. Kyselyt muodostettiin automaattisuutta simuloiden käyttämällä seuraavaa proseduuria:

1. Otetaan kaikki sanat hakuaiheiden title-kentästä
2. Jos fraasiksi merkatun (” ”) hakuavaimen edessä ei ole ”-” merkkiä, fraasin sisältämät sanat toistetaan kyselyn perään ilman lainausmerkkejä. Jos sana esiintyy useammassa fraasissa, se toistetaan vain kerran
3. Etsitään ja poistetaan mahdolliset välilyönnit operaattorien ja avainten välistä
4. Jos avaimessa on väliviiva, viiva poistetaan ja sen ympärillä olevat sanat muutetaan fraasiksi lisäämällä lainausmerkit
5. Rivinvaihdot poistetaan muualta kuin viimeisen avaimen perästä

6. Kohdassa 4 muodostuneet mahdolliset sisäkkäiset lainausmerkit poistetaan.

Tässä tutkimuksessa tehtyjen ajojen kyselylistat löytyvät liitteistä 1 ja 2.

4.2.3 Relevanssiarviot

INEX-hankkeessa hakuaiheen kehittäjä arvioi palautettujen dokumenttien relevanssin. Arvioissa ei käytetä konsensusmenettelyä vaan arvioitsijan tulee suorittaa arviointi itsenäisesti. INEX-kokoelman relevanssiarvioissa käytetään elementtien arviointiin kahta relevanssin dimensiota:

Kattavuus (Exhaustivity, E), joka kuvaa aiheen käsittelyn laajuutta elementissä.
Spesifisyys (Specificity, S), joka kuvaa aiheen käsittelyyn keskittymistä elementissä

Kattavuutta mitataan seuraavalla neliportaisella asteikolla:
Ei katetta (E0): elementti ei käsittele aihetta lainkaan.
Marginaalinen kattavuus (E1): Elementti käsittelee vain muutamaa näkökulmaa aiheesta
Kohtuullinen kattavuus (E2): Elementti käsittelee useita aiheen näkökulmia.
Korkea kattavuus (E3): Elementti käsittelee (lähes) kaikkia aiheen näkökulmia.

Spesifisyyttä mitataan seuraavalla neliportaisella asteikolla:
Ei spesifiyyttä (S0): aihetta ei käsitellä evaluoitavassa elementissä.
Marginaalisesti spesifi (S1): aihe on ainoastaan sivuteema elementissä.
Kohtuullisen spesifi (S2): aihe on elementin pääteema.
Erittäin spesifi (S3): elementti käsittelee ainoastaan aihetta.

Dimensiot ovat yleisesti ottaen riippumattomia toisistaan. Ainoastaan siinä tapauksessa, että toinen dimensio on arvoltaan nolla täytyy toisenkin dimension saada arvokseen nollan. (Kazai, Lalmas & Piwowarski 2004, 1.)

4.2.4 Mittaristot

XML–tiedonhaun erityispiirteistä johtuen perinteisten evaluointihankkeiden (kuten TREC) mittareita ei ole mahdollista soveltaa INEXiin ilman muutoksia (Gövert & Kazai 2003, 9). INEX –hankkeessa on tuotettu kaksi mittaristoa; inex_eval (ts. inex-

2002), joka toimii laajennettuna myös INEX 2004:n ”virallisena” mittaristona (de Vries, Kazai & Lalmas 2004, 249) sekä inex_eval_ng (inex-2003). inex_eval_ng kehitettiin korjaamaan inex_eval:in suurinta puutetta; inex_eval ei rankaise mitenkään päällekkäisistä tuloksista vaan jopa suosii järjestelmiä jotka palauttavat päällekkäisiä elementtejä tuloksina (Kazai 2004, 185). Molemmat mittarit tuottavat tuloksenaan tavanomaisen saanti-tarkkuuskäyrän, kyselykohtaiset keskitarkkuudet (average precision) ja niistä lasketun keskitarkkuuden. Mittarit esitellään tarkemmin seuraavissa kappaleissa.

TRIXin kehitystyön yhteydessä päätettiin käyttää evaluointityökaluna inex_eval_ng:tä koska TRIX ei normaaliasetuksillaan palauta ollenkaan päällekkäisiä elementtejä. Tässä työssä esitellään INEXissä käytettävät mittarit niiltä osin kuin ne ovat yhteisiä vuosien 2003 ja 2004 evaluointikierroksilla. Tämän työn empiirisessä osassa käytetään molempia mittaristoja rajoitetusti yleistetyillä (generalized quantization) ja kattavasti tiukoilla relevanssikriteereillä (strict quantization) sekä inex_eval_ng:n osalta tulosten päällekkäisyys huomioon ottaen (overlapping considered). Nämä rajaukset ovat perusteltavissa sillä, että käytössä olevat kvantifiointimenetelmät tuottavat hyvin samanlaisia tuloksia (Fuhr, Malik & Lalmas 2004, 7) sekä ajojen määrän pitämisellä kohtuullisena.

INEXin virallisten mittareiden lisäksi tässä työssä käytetään tutkijan kirjoittamaa analysoi_mm –ohjelmaa, joka tunnistaa TRIXin tuottamista tuloslistoista palautetut elementit ja lajittelee ne seuraaviin kategorioihin:

- Artikkelitason elementit {article, bdy}
- Tiivistelmät {abs}
- Lukutason elementit {sec, ss1,ss2}
- Kappaletason elementit {p, ip1}
- Muut

Kategorisointi perustuu Massin ja Mandelbrodin (2004, 54) INEX 2002 –aineistosta tekemään havaintoon että CO-aiheissa 1296 1394:stä korkean relevanssiarvon saaneesta elementistä kuului em. kategorioihin. Massin ja Mandelbrodin jaon lisäksi kategorioihin

on lisätty NEXI-spesifikaatiossa (Trotman & Sigurbjörnsson 2004) esitetyn elementtien synononymialistan mukaiset elementit. Ohjelma toimii ottamalla parametrinä analysoitavan tiedoston nimen, sen sisältämän hakuaiheiden määrän prosenttiosuuksien laskentaa varten sekä haluttavan katkaisuarvon joka määrittää kuinka monta kärkitulosta jokaisen hakuaiheen tuloslistasta analysoidaan. Tässä työssä katkaisuarvona (Document Cutoff Value, DCV) käytetään arvoa 100, joka on INEX-relevanssipooliin mukaan otettava määrä. Määrän lisääminen esimerkiksi 1500:n ei vaikuta ratkaisevasti jakaumaan. Ohjelman perl-kielinen lähdekoodi on liitteessä 3.

Kategorisoitaessa elementtejä herää kysymys mahdollisesta optimaalisesta elementtijakaumasta. Jakauman selvittämistä vaikeuttaa se, että relevanssiarvioista ei ole tiedettävästi saatavissa kattavia tilastotietoja. Kamps, Marx, de Rijke ja Sigurbjörnsson (2003, 2) tekivät INEX 2002 –relevanssiarvioista analyysin jonka mukaan, tämän tutkimuksen kategorioihin sovellettuna, tiukan relevanssimäärittelyn (strict quantization, ks. 4.2.4.1) mukaisesti artikkelitason elementtejä oli 30 %, tiivistelmiä 1,7 %, lukutason elementtejä 32,4 %, kappaletason elementtejä 33,4 % ja 2,6 % muita elementtejä. Kampsin ym. analyysin pätevyyttä tässä tapauksessa tosin kyseenalaistaa aineiston vähyys nykytilaan verrattuna ja relevanssimäärittelyn osittainen muuttaminen vuoden 2002 jälkeisille kierroksille.

4.2.4.1 inex_eval (inex-2002)

inex_eval -mittaristo kehitettiin INEX 2002 yhteydessä ja sitä on myöhemmin muokattu käyttämään INEX 2003:n mukaisia relevanssin dimensioita (ks. luku 4.1.3). Se käyttää evaluointimittana relevanssin todennäköisyyttä (precall, kts. Raghavan ym. 1989) elementeille ja laskee todennäköisyyden P (rel|retr) sille onko käyttäjän tarkastelema elementti relevantti:

$$P(rel \mid retr)(x) := \frac{x * n}{x * n + esl_{x*n}}$$

jossa esl_{x*n} merkitsee odotettua haun pituutta eli arvioitua epärelevanttien elementtien määrää joka on haettu ennen kuin määrätty saantitaso x on tavoitettu, ja n on aiheen kaikkien relevanttien elementtien määrä.

Mittariston käyttöä varten kappaleessa 4.1.3 esitetyt relevanssin dimensiot täytyy johtaa yhteen relevanssiarvoon käyttämällä kvantifiointia $f_{quant}(e,s): ES \rightarrow [0,1]$, jossa *ES* ilmaisee mahdollisten relevanssiparien joukkoa *(e,s)* jossa *e* on kattavuus (exhaustivity), ja *s* spesifisyys (specificity) (vrt. kappale 4.2.4):

$$ES = \{(0,0), (1,1), (1,2), (1,3), (2,1), (2,2), (2,3), (3,1), (3,2), (3,3)\}$$

Kvantifioinnissa voidaan käyttää kahta funktiota käyttötarpeen mukaan. $\mathbf{f}_{strict}$ on tarkoitettu käytettäväksi silloin kun halutaan tutkia menetelmien toimintaa korkean kattavuuden ja spesifiyden omaavien elementtien haussa:

$$\mathbf{f}strict(e,s) := \begin{cases} 1 \ jos \ e = 3 \ ja \ s = 3 \\ 0 \ muussa \ tapauksessa \end{cases}$$

$\mathbf{f}_{generalized}$ puolestaan arvottaa elementtejä niiden relevanssin asteen mukaan seuraavasti:

$$\mathbf{f}gen(e,s) := \begin{cases} 1 & jos \ (e,s) = (3,3), \\ 0{,}75 & jos \ (e,s) \in \{(2,3),(3,\{2,1\})\}, \\ 0.5 & jos \ (e,s) \in \{(1,3),(2,\{2,1\})\}, \\ 0{,}25 & jos \ (e,s) \in \{(1,2),(1,1)\}, \\ 0 & jos \ (e,s) = (0,0). \end{cases}$$

(Kazai 2003, 185.)

4.2.4.2 inex_eval_ng (inex-2003)

inex_eval_ng kehitettiin paikkaamaan inex_eval:in suurinta ongelmatekijää, päällekkäisistä tuloksista palkitsemista. inex_eval_ng pyrkii ratkaisemaan ongelman lisäämällä elementin koon ja päällekkäisyyden saannin ja tarkkuuden määrityksiin. Sen sijaan että saanti ja tarkkuus laskettaisiin tietyn haetun elementtimäärän jälkeen, laskennassa käytetään perusparametrina haettujen elementtien kokoa. Päällekkäisyyttä laskettaessa otetaan huomioon vain lisäys jo nähtyihin elementteihin. Laskennassa oletetaan että relevantti informaatio on levittynyt tasaisesti koko elementissä. Saanti lasketaan kaavalla:

$$recall_o = \frac{\sum_{i=1}^{k} e(c_i) \cdot \frac{|c_i'|}{c_i}}{\sum_{i=1}^{N} e(c_i)}$$

ja tarkkuus kaavalla:

$$precision_o = \frac{\sum_{i=1}^{k} s(c_i) \cdot |c_i'|}{\sum_{i=1}^{k} |c_i'|}$$

Elementit $c_1,...,c_k$ em. kaavoissa muodostavat järjestetyn tuloslistan. *N* on elementtien kokonaismäärä kokoelmassa. *e(c$_i$)* ja *s(c$_i$)* ilmaisevat kvantifioitua relevanssiarvoa elementille c_i. Elementin koko on ilmaistu $|c_i|$:llä ja $|c_i'|$ on käyttäjän aikaisemmin näkemättömän elementin koko. $|c_i'|$ voidaan laskea seuraavasti:

$$|c_i'| = \left| c - \bigcup_{c \in C[1,n-1]} (c) \right|$$

missä *n* on c_i:n paikka tuloslistassa ja *C*[1, *n*-1] on elementtien joukko mikä on haettu paikkojen [1, *n*-1] välissä.

inex-2003:ssa relevanssin dimensiot ovat kvantifioitu suunnilleen samoin kuin inex-2002:en nykyisessä versiossa: $f'_{quant}(e,s) : ES \rightarrow [0,1]$, jossa *E* ja *S* = {0,1,2,3}. **f'***strict* kvantifioidaan 1:si jos *e*=3 ja *s*=3, ja kaikissa muissa tapauksessa nollaksi. **f'***generalized* kvantifioidaan puolestaan seuraavasti: **f'***gen(e)* =*e*/3 ja **f'***gen(s)* =*s*/3. (Kazai 2003, 185-186.)

4.3 Tulosten tilastollinen testaaminen

Evaluointitutkimuksessa tarvitaan tilastollista testaamista tulosten välisten erojen merkitsevyyden testaamiseen. Heikkilän (1998, 182) mukaan tilastollinen testausprosessi koostuu kuudesta vaiheesta:

- hypoteesin asettaminen
- otoksen poimiminen
- tilastollisen testin valinta
- testin suorittaminen

- tuloksen tulkinta
- johtopäätöksen tekeminen

Hypoteesien asettamisen lähtökohta on nollahypoteesi H_0 joka väittää että menetelmien välillä ei ole eroa. Tämän lisäksi asetetaan vaihtoehtoinen hypoteesi H_1 joka väittää menetelmien välillä olevan eroa. Merkitsevyystestit mittaavat todennäköisyyden *p* sille, että menetelmien välillä havaitut erot ovat sattumanvaraisia; pienet erot eivät riitä nollahypoteesin kumoamiseen ja vaihtoehtoisen hypoteesin hyväksymiseen. Ennen testin suorittamista valitaan riski- tai merkitsevyystaso α ja jos *p* on arvoltaan pienempi kuin α voidaan todeta menetelmien olevan merkitsevästi eroavia. (Heikkilä 1998, 182; Hull 1993, 333.)

Merkitsevyystasoina käytetään yleisimmin arvoja 0,05 (5 %), 0,01 (1 %) ja 0,001 (0,1 %). Testatun eron sanotaan olevan tilastollisesti erittäin merkitsevä, jos $p \leq 0{,}001$, merkitsevä jos $0{,}001 < p \leq 0{,}01$, melkein merkitsevä jos $0{,}01 < p \leq 0{,}05$ ja suuntaa antava jos $0{,}05 < p \leq 0{,}1$. Useimmiten puhutaan tosin vain tilastollisesta merkitsevyydestä jollain annetulla merkitsevyystasolla. (Heikkilä 198, 185-186.)

Merkitsevyystestit voidaan jakaa useisiin eri kategorioihin. Yleisimmät metodit jaetaan parametrisiin ja ei-parametrisiin. Van Rijsbergenin (1979, 178) mukaan parametriset testit ovat kyseenalaisia tiedonhaun tutkimuksessa koska saanti ja tarkkuus eivät suureina seuraa normaalijakaumaa. Hull (1993, 333) kuitenkin väittää että parametrisiä testejä voitaisiin käyttää riittävän suurten aineistojen tilastolliseen tarkasteluun.

Testit voidaan jakaa myös kahteen kategoriaan sen mukaan testataanko niillä kahta vai useampaa menetelmää (Hull 1993, 333). Tässä työssä käytetään molempiin kategorioihin kuuluvia testejä, Wilcoxonin merkkitestiä (Wilcoxon signed-rank test) tutkittaessa painojen yhdistämismenetelmän vaikutusta keskitarkkuuteen ja Friedmanin testiä tutkittaessa painotuskaavan vakion *a* vaikutusta keskitarkkuuteen.

Wilcoxonin merkkitesti on ei-parametrinen testi joka testaa kahden rinnakkaisen aineiston mediaanien eroa (Conover 1980, 280). Testi suoritetaan laskemalla aineiston jokaisen tapauksen (tässä tapauksessa hakuaiheen) eri menetelmillä saadun tuloksen

erotus. Sen jälkeen erotukset asetetaan suuruusjärjestykseen itseisarvonsa mukaan siten että pienin erotus saa sijan 1, seuraavaksi pienin sijan 2 jne. Tämän jälkeen sijoitukset kerrotaan erotuksensa etumerkillä, lasketaan positiivisten erotuksien sijoitusten summa ja negatiivisten erotuksien sijoitusten summa. Kumpaakin sijoitusten summaa verrataan sen jälkeen odotettuun arvoonsa joka olettaa että molempien menetelmien sijoituksien summa on sama. (Hull 1993, 333.)

Olkoon X_i ja Y_i hakumenetelmän X ja Y tulokset kyselylle i, jossa i= 1...n ja määritellään että $D_i = Y_i - X_i$. Testisuure saadaan kaavalla

$$T = \frac{\sum R_i}{\sqrt{\sum R_i^2}},$$

jossa R_i= +/- (D_i)* sija$|D_i|$. (Hull 1993, 333.)

Friedmanin testi on Wilcoxonin tavoin ei-parametrinen testi mutta se on tarkoitettu useamman kuin kahden menetelmän vertailuun. Tulokset sijoitetaan taulukkoon joka koostuu riveistä b ja sarakkeista k rivien edustaessa kyselyjä ja sarakkeiden vertailtavia menetelmiä. Jokaisen menetelmän tulos on järjestysluku väliltä 1-k rivin pienimmän arvon saadessa sijoituksen 1, seuraavan saadessa sijoituksen 2 jne. Testin testisuureen arvo T_2 saadaan kaavalla

$$T_2 = \frac{(k-1)\left[B_2 - bk(k+1)^2/4\right]}{A_2 - B_2}$$

jossa

$$A_2 = \sum_{i=1}^{b}\sum_{j=1}^{k}\left[R(X_{ij})\right]^2$$

ja

$$B_2 = \frac{1}{b}\sum_{j=1}^{k} R_j^2$$

jossa b on rivien määrä, k on sarakkeiden määrä, $R(X_{ij})$ on solun järjestysluku rivillä i sarakkeessa j ja R_j on järjestyslukujen summa sarakkeessa j. (Conover 1980, 295-296; 299-300.)

Testisuureen arvoa verrataan F-jakaumaan ja jos se on merkitsevä (sattuman todennäköisyys *p* on alle merkitsevyystason α) nollahypoteesi H_0 voidaan hylätä. Nollahypoteesin tultua hylätyksi voidaan menetelmiä *i* ja j vertailla keskenään:

$$\left|R_j - R_i\right| > t_{1-\alpha/2}\left[\frac{2b(A_2 - B_2)}{(b-1)(k-1)}\right]^{½}$$

jossa R_i, R_j, A_2 ja B_2 on annettu edellä ja $t_{1-\alpha/2}$ määrittää kriittisen arvon merkitsevyystasolla α. (Conover 1980, 300.)

Tilastollisen tarkastelun yhteydessä voidaan huomauttaa että tilastollinen merkitsevyys ei välttämättä tarkoita merkitsevyyttä käytännössä. Sparck-Jones (1974) esittää peukalosääntönä että jos ero menetelmien keskituloksissa on olennainen jos ero on yli 10 %, merkittävä jos 5 % -10 % ja merkityksetön jos ero on alle 5 %. (Keen 1992, 497.)

5 Tulokset

Koetulosten tarkastelu on jaettu neljään alalukuun. Ensimmäisessä luvussa tarkastellaan painojen yhdistämismenetelmän vaikutusta muuten oletusparametreillään toimivan järjestelmän tehokkuuteen inex_eval- ja inex_eval_ng -mittareilla mitattuna. Toinen luku tarkastelee vakion *a* arvon muuttamisen vaikutusta järjestelmän tehokkuuteen. Kolmannessa luvussa tutkitaan painojen yhdistämismenetelmien vaikutusta järjestelmän palauttamien elementtejen kokoon. Lopuksi tutkitaan vakion *a* arvon muuttamisen vaikutusta järjestelmän palauttamien elementtejen kokoon.

Kokeita varten ajettiin sekä vuoden 2003 että 2004 kyselysarjat, mutta evaluointiohjelmiston toimimattomuuden vuoksi vuoden 2004 tulokset voitiin evaluoida ainoastaan inex_eval -mittareilla. Tästä syystä vuoden 2004 kyselysarjan tulokset esitellään vain em. mittareilla analysoituna.

5.1 Painojen yhdistämismenetelmän vaikutus keskitarkkuuteen

Tiedonhakujärjestelmän tukemille kahdelle avainten painojen yhdistämismenetelmille laskettiin kyselyjen keskiarvoiset tarkkuudet käyttäen kahta eri evaluointityökalua (inex_eval ja inex_eval_ng) sekä yleistetyillä että tiukoilla relevanssikriteereillä. Taulukko 1 havainnollistaa keskiarvoisina tarkkuuksina Einsteinin summa -yhdistämismenetelmän tehokkuutta vuosien 2003 ja 2004 kyselyillä verrattuna järjestelmän vakiona käyttämään keskiarvon laskemiseen perustuvaan yhdistämismenetelmään. Taulukkoon on merkitty menetelmien tuottamien keskitarkkuuksien lisäksi menetelmien välinen ero prosentuaalisesti sekä Wilcoxonin testin tuottama *p*-arvo. Vuoden 2004 tulokset inex_eval_ng –mittarilla puuttuvat evaluointityökalun toimimattomuuden vuoksi.

Vuosi	Evaluointimittari	keskiarvo	Einsteinin summa	muutos %	*p*
2003	**inex_eval, yleistetty**	0,0269	0,0264	-1,8 %	0,52
	inex_eval, tiukka	0,0358	0,0356	-0,5 %	0,24
	inex_eval_ng, yleistetty	0,0994	0,0986	-0,8 %	0,13
	inex_eval_ng, tiukka	0,1348	0,1368	+1,5 %	0,12
2004	**inex_eval, yleistetty**	0,0162	0,0161	-2,4%	0,13
	inex_eval, tiukka	0,0221	0,0234	+14,3%	0,60

Taulukko 1. Keskiarvoiset tarkkuudet vuosien 2003 ja 2004 kyselyistä keskiarvo- ja Einsteinin summa –yhdistämismenetelmillä. Keskitarkkuudet on laskettu kahdella eri evaluointimittarilla käyttäen sekä yleistettyä että tiukkaa relevanssikriteeristöä.

Kummankaan vuoden kyselyillä yhdistämismenetelmien välillä ei ollut tilastollisesti merkitseviä eroja (kaikissa tapauksissa $p>0,1$). Menetelmien väliset erot ovat merkityksettömiä (<5 %) myös käyttäen Sparck-Jonesin peukalosääntöä.

5.2 Vakion *a* säätämisen vaikutus keskitarkkuuteen

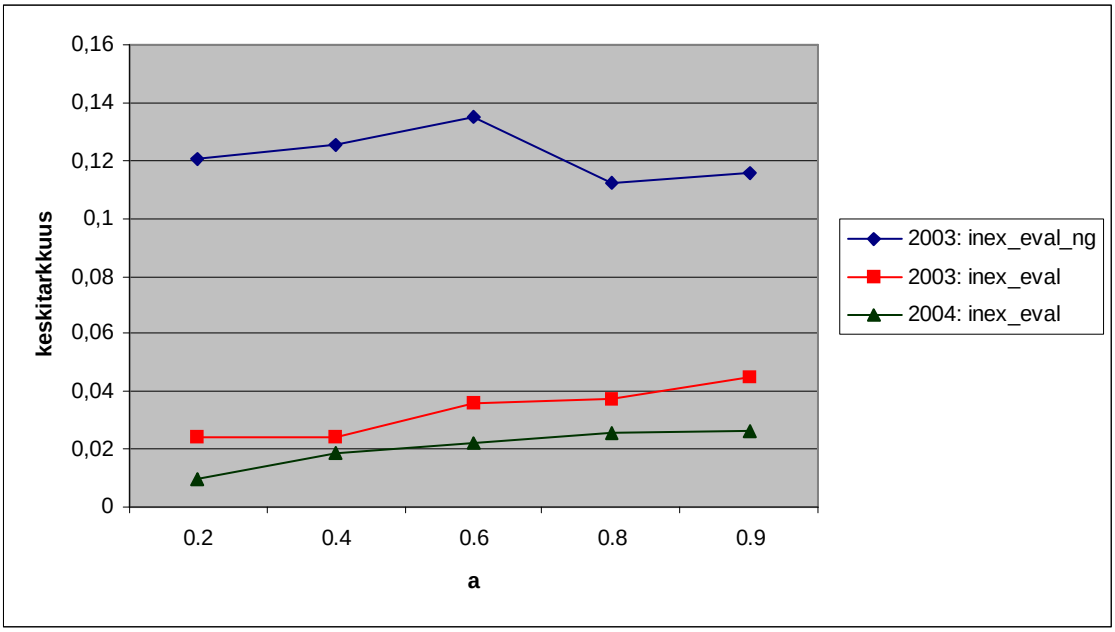
TRIX mahdollistaa järjestelmän ”virittämisen” manipuloimalla painotuskaavan vakioita; olennaisin vaikutus on vakion a (1-b) arvon muuttamisella. Järjestelmällä suoritettiin sarja ajoja käyttäen a:n arvoja 0.2, 0.4, 0.6, 0.8 ja 0.9. Ajoille laskettiin kyselyjen keskiarvoiset tarkkuudet käyttäen inex_eval ja inex_eval_ng mittareita. Molemmilla mittareilla käytettiin tiukkoja relevanssikriteereitä. Taulukko 2 havainnollistaa keskiarvoisina tarkkuuksina vakion a muuttamisen vaikutusta verrattuna järjestelmän oletuksena käyttämään a:n arvoon 0.6. Taulukkoon on myös merkitty eri a:n arvojen ja vakioarvon 0.6 välille saadut merkitsevyyserot p eri evaluointimittareilla. Merkinnässä on käytetty tähtiä indikoimaan merkitsevyyttä.

Vuosi	Mittari	*a*=0.2	*a*=0.4	*a*=0.6	*a*=0.8	*a*=0.9
2003	**inex_eval**	0,024 (-32,6 %)*	0,025 (-31,4 %)	0,036	0,037(+3,6 %)	0,045 (+26,1 %)*
	inex_eval_ng	0,121(-10,5 %)	0,126 (-6,8 %)	0,135	0,112 (-16,7 %)	0,116 (-14,1 %)
2004	**inex_eval**	0,010 (-55,1 %)***	0,018 (-16,8 %)**	0,022	0,026 (+16,8 %)	0,027 (+20,6 %)

Taulukko 2. Keskiarvoiset tarkkuudet vuoden 2003 ja 2004 kyselyistä muutoksineen vakion *a* arvoilla {0.2, 0.4, 0.6, 0.8, 0.9}. Keskitarkkuudet on laskettu kahdella eri evaluointimittarilla käyttäen tiukkaa relevanssikriteeristöä (2004 vain inex_eval). Tähdet osoittavat tilastollisia merkitsevyyksiä seuraavasti: *:melkein (0,01<p≤0,05) merkitsevä, **: merkitsevä (0,001<p≤0,01), *: erittäin merkitsevä (p≤ 0,001).**

Vuoden 2003 kyselyillä inex_eval –mittarilla kokeiltujen arvojen ääripäät 0,2 ja 0,9 tuottivat vakioon verrattuna melkein merkitsevän eron (p<0,05). inex_eval_ng –mittarilla erot kaikkien arvojen välillä olivat riittämättömiä tilastollisen merkitsevyyden saavuttamiseksi. Vuoden 2004 kyselyillä arvojen 0.2 ja 0.4 erot vakioon olivat ensimmäisessä tapauksessa erittäin merkitseviä ja jälkimmäisessä merkitseviä. Sparck-Jonesin peukalosääntöä käyttäen erot eivät ole kuitenkaan merkittäviä yhdessäkään tapauksessa.

Keskitarkkuuksista voidaan kuitenkin nähdä, että inex_eval –mittarilla järjestelmän tehokkuus paranee *a*:n arvon kasvaessa. Erityisesti huomioitavaa on vuoden 2003 kyselyillä tehokkuuden paraneminen välillä 0.8-0.9, jossa parannusta on 22,5 % huolimatta siitä että arvojen väli on vain 0,1 normaalin 0,2 sijasta. inex_eval_ng sen sijaan saavuttaa parhaan mitatun tehokkuutensa vakioarvolla 0.6. Kuvio 10 havainnollistaa vakion muuttamisen vaikutusta järjestelmän tehokkuuteen.



Kuvio 10. Vakion *a* säätämisen vaikutus keskitarkkuuteen inex_eval ja inex_eval_ng –mittareilla käyttäen tiukkaa relevanssikriteeristöä.

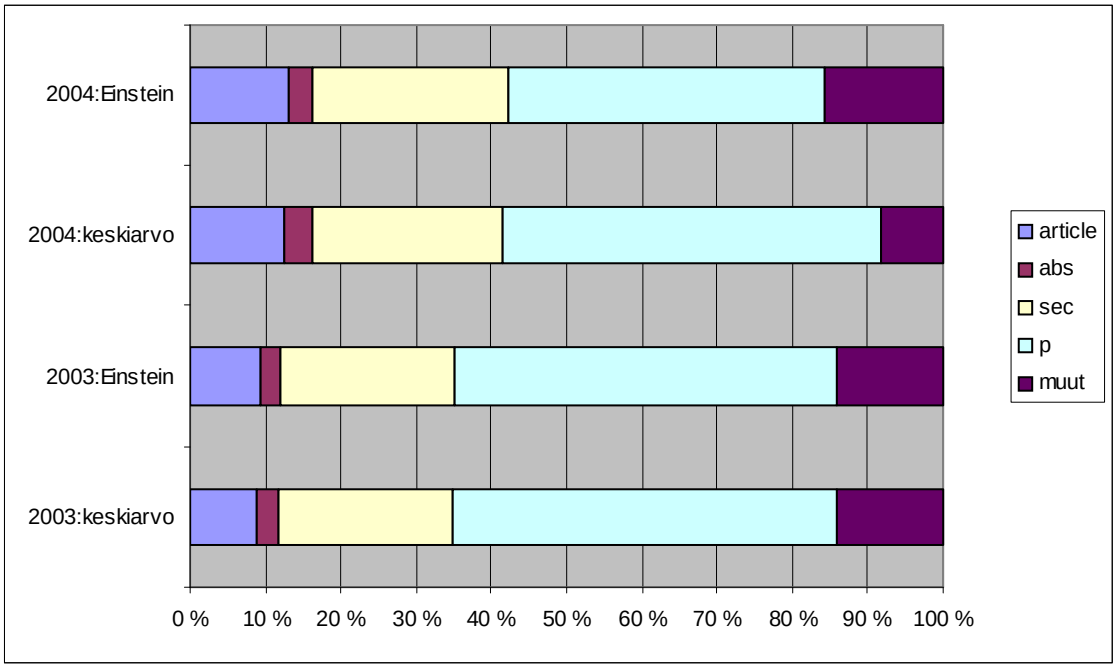
5.3 Painojen yhdistämismenetelmän vaikutus palautettavien elementtien kokoon

Keskitarkkuuden lisäksi painojen yhdistämismenetelmiä verrattaessa tutkittiin onko yhdistämismenetelmällä vaikutusta palautettavien elementtien kokojakaumaan. Kokojakaumaa tutkittiin analysoimalla tuloslistoja kappaleessa 4.2.4 esitellyllä analysointiohjelmalla käyttäen katkaisuarvona 100 elementtiä. Taulukko 3 havainnollistaa menetelmien palauttamat elementtijakaumat ja niiden väliset muutosprosentit.

vuosi: 2003				**vuosi: 2004**			
elementtiryhmä	**keskiarvo**	**Einstein**	**muutos**	**elementtiryhmä**	**keskiarvo**	**Einstein**	**muutos**
article	319	331	+3,8 %	**article**	505	523	+3,6 %
abs	105	103	-1,9 %	**abs**	142	129	-9,2 %
sec	831	826	-0,6 %	**sec**	1010	1036	+2,6 %
p	1839	1833	-0,3 %	**p**	2015	1688	-16,2 %
muut	506	507	+0,2 %	**muut**	328	624	+90,2 %

Taulukko 3. Palautettujen elementtien kokojakauma verrattaessa kahta eri painojen yhdistämismenetelmää. Elementtejä yhteensä 3600 (2003) / 4000 (2004).

Taulukosta 3 nähdään että vuoden 2003 kyselyillä Einsteinin summalla yhdistäminen palauttaa hieman (3,8 %) enemmän artikkelitason elementtejä pienempien elementtien kustannuksella, ei eroja voida pitää mitenkään merkitsevinä. Vuoden 2004 kyselyillä erot ovat huomattavampia; erityisesti kategoriaan ”muut” kuuluvien elementtien määrä kasvoi rajusti käytettäessä Einsteinin summaa. Huomattavia muutoksia oli myös kappaletason elementtien ja tiivistelmien määrissä. Kuvio 11 havainnollistaa elementtiryhmien keskinäistä jakautumaa.



Kuvio 11. Palautettujen elementtiryhmien prosentuaaliset osuudet keskiarvolla ja Einsteinin summalla yhdistettäessä.

5.4 Vakion *a* säätämisen vaikutus palautettavien elementtien kokoon

Elementtien kokojakauman analysointia käytettiin myös vakion *a* säätämisen vaikutuksen tutkimiseen. Kokojakauman tutkiminen suoritettiin samalla tavoin kuin kappaleessa 5.3. Taulukoista 4a ja 4b voidaan nähdä vakion säätämisen vaikutus prosentuaalisesti verrattuna järjestelmän vakioasetukseen a=0.6.

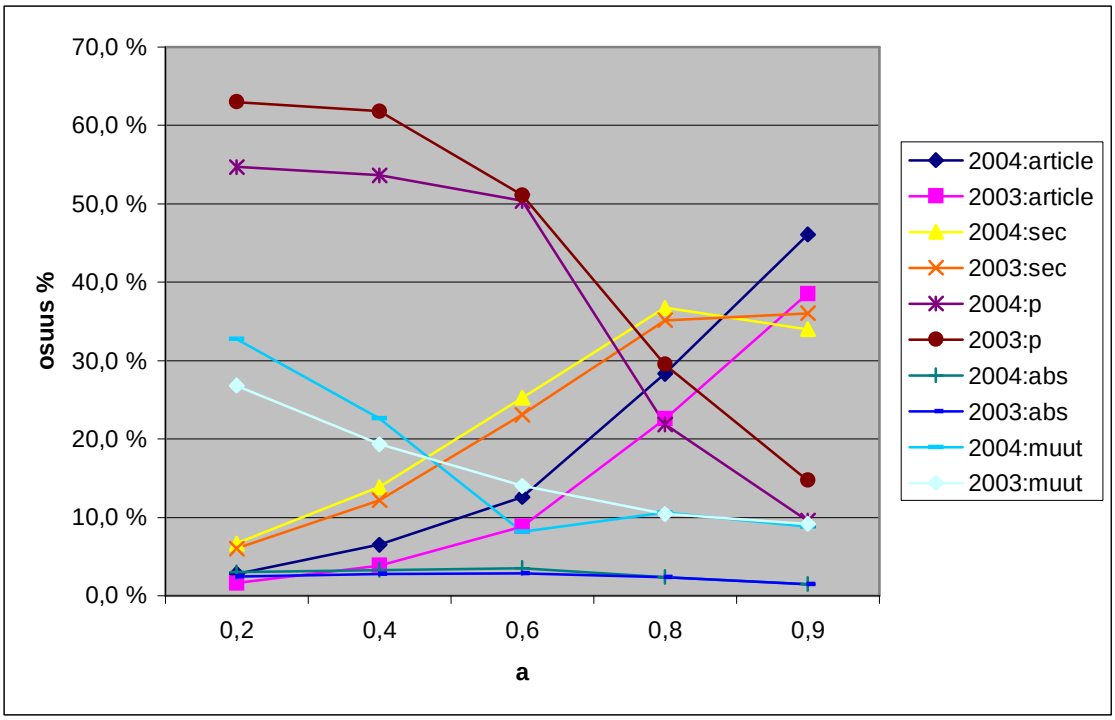
elementtiryhmä	*a*=0.2	*a*=0.4	*a*=0.6	*a*=0.8	*a*=0.9
article	59 (-82 %)	139 (-56 %)	319	812 (+155 %)	1387 (+334,8 %)
abs	89 (-15 %)	101 (-4 %)	105	85 (-19 %)	55 (-47,6 %)
sec	218 (-74 %)	440 (-47 %)	831	1265 (+52 %)	1297 (+56,1 %)
p	2268 (+23 %)	2224 (+21 %)	1839	1062 (-42,3 %)	531 (-71,1 %)
muut	966(+91 %)	696 (+38 %)	506	376 (-25,7 %)	330 (-34,1 %)

Taulukko 4a. Palautettujen elementtien kokojakauman muutos verrattuna vakioarvoon *a*=0.6. Vuoden 2003 kyselyt, elementtejä yhteensä 3600.

elementtiryhmä	*a*=0.2	*a*=0.4	*a*=0.6	*a*=0.8	*a*=0.9
article	113 (-78 %)	262 (-48 %)	505	1133 (+124 %)	1844 (+265 %)
abs	121 (-15 %)	133 (-6 %)	142	97 (-32 %)	61 (-57 %)
sec	268 (-73 %)	555 (-45 %)	1010	1470 (+46 %)	1360 (+35 %)
p	2187 (+9 %)	2145 (+6 %)	2015	875 (-57 %)	383 (-81 %)
muut	1311 (+300 %)	905 (+176 %)	328	425 (+30 %)	352 (+7)

Taulukko 4b. Palautettujen elementtien kokojakauman muutos verrattuna vakioarvoon *a*=0.6. Vuoden 2004 kyselyt, elementtejä yhteensä 4000.

Vakion *a* arvon kasvaessa palautettavien elementtien koko kasvaa radikaalisti lähinnä kappaletason elementtien kustannuksella. Artikkelitason elementtien määrän kasvu kiihtyy arvon lähestyessä yhtä lukutason elementtien osuuden kasvun hiipuessa. Kuvio 12 havainnollistaa elementtiryhmien jakaumien muutosta vakion *a* arvon kasvaessa.



Kuvio 12. Vakion *a* muutoksen vaikutus elementtiryhmien jakaumaan prosentuaalisesti.

Kuviosta 12 voidaan nähdä olennaisimpien elementtiryhmien (article, sec, p) jakaumien leikkaavan *a*:n arvojen välillä 0.7-0.85. Tämä korreloi jossain määrin Hawkingin ym. (1998, 1) painotuskaavassa käyttämän *a*:n arvon 0.75 kanssa. Järjestelmän vakioasetuksella *a*=0.6 järjestelmä suosii kappaletason elementtejä, palauttaen niitä yli 50 %. Elementtiryhmän ”muut” suuri osuus pienillä *a*:n arvoilla selittyy sen koostumisesta suurimmalta osin artikkeleiden kirjoittajien elämänkertatietoja sisältävistä vt-elementeistä; nämä elementit sisältävät yhden p-elementin ja TRIXin suunnittelussa tehdyn valinnan mukaan ne palautetaan sisältämänsä elementin sijaan.

6 Keskustelua

TRIX on osoittautunut, käytetyistä metriikoista ja/tai asetuksista riippuen, kilpailukykyiseksi tiedonhakujärjestelmäksi. Kazai ym. (2004, 39-40) raportoi TRIXin saavuttavan kaksoisvoiton vuoden 2004 INEX-ajoissa käytettäessä uutta XCG-evaluointimetriikkaa. Kekäläinen ym. (2004, 74) osoittaa järjestelmän pystyvän tuottamaan kilpailukykyisiä tuloksia vuoden 2004 virallisella inex_eval –metriikalla käytettäessä metriikalle suotuisia asetuksia.

Tässä tutkimuksessa selvitettiin neljää asiaa – onko TRIX-tiedonhakujärjestelmän avainten painojen eri yhdistämismenetelmillä vaikutusta järjestelmän tehokkuuteen tai sen palauttamiin elementteihin ja mikä vaikutus järjestelmän painotuskaavan vakioiden manipuloinnilla on järjestelmän tehokkuuteen ja palauttamiin elementteihin. Tehokkuutta mitattiin kahdella eri evaluointimetriikalla käyttäen yhdistämismenetelmien yhteydessä sekä tiukkaa että yleistä relevanssikriteeristöä ja vakioiden manipuloinnin yhteydessä tiukkaa relevanssikriteeristöä. Palautettujen elementtien analysointiin käytettiin varta vasten tätä tutkimusta varten kirjoitettua ohjelmaa.

Avainten painojen yhdistämismenetelmien välillä ei ollut merkitseviä eroja. Einsteinin summalla painojen yhdistäminen tuotti ainoastaan yhdessä tapauksessa keskiarvolla yhdistämistä paremman tuloksen Sparck-Jonesin peukalosäännöllä arvioituna, muissa tapauksissa lähes yhtä nimellisesti. Vertailtaessa yhdistämismenetelmien palauttamia tuloslistoja ei elementtijakaumissakaan ole havaittavissa kovin olennaisia eroja menetelmien välillä – Einsteinin summalla yhdistäminen tosin hieman suosii artikkelitason elementtejä.

Painotuskaavan vakion *a* arvon muuttamisella näytti olevan yhdistämismenetelmää enemmän vaikutusta järjestelmään. *a*:n arvon suurentuessa järjestelmä palautti kiihtyvissä määrin artikkelitason elementtejä pienempien kustannuksella. Tämä oli odotettavissa, koska *a*:n kasvaessa painotuskaavan pituusnormalisoinnin vaikutus vähenee ja termin frekvenssin vaikutus kasvaa.

inex_eval –mittarilla mitattuna järjestelmän tehokkuus näytti kasvavan *a*:n arvon – ja samalla suurten elementtien osuuden – kasvaessa. Toinen mittari, inex_eval_ng, taas tuotti parhaan tuloksen järjestelmän oletusarvolla; *a*:n arvon kasvaessa vakiosta keskitarkkuus heikentyi huomattavasti. Jos inex_eval näyttää suosivan suuria elementtejä tai jopa kokonaisia dokumentteja palauttavia järjestelmiä asettuu sen käyttäminen ainakin CO-ajojen evaluoinnissa jokseenkin omituiseen valoon.

Tutkimuksesta on myös todettavissa se, että TRIX tuottaa varsin huonoja tuloksia inex_eval -metriikalla; tämä voidaan suoraan johtaa metriikan tunnettuun taipumukseen suosia järjestelmiä jotka palauttavat päällekkäisiä tuloksia (vrt. 4.2.4). Kekäläinen ym. (2004, 74) raportoikin TRIXin inex_evalilla mitatun tehokkuuden paranevan merkittävästi kun päällekkäisten elementtien palauttaminen sallitaan.

7 Johtopäätökset

Tässä tutkimuksessa esiteltiin TRIX-tiedonhakujärjestelmä, joka soveltaa rakenteettomaan dokumenttiympäristöön kehitettyä Okapi BM25 –pohjaista hakuavainten painonlaskentaa rakenteisiin dokumentteihin. Painonlaskenta on toteutettu käyttämällä elementtikohtaisia avainfrekvenssejä ja elementteihin perustuvaa pituusnormalisointia.

Tutkimuksen empiirisessä osassa tutkittiin evaluointitutkimuksen keinoin kahta järjestelmän toimintaan vaikuttavaa tekijää. Ensimmäisenä tutkittiin onko järjestelmään toteutettujen kahden eri painojen yhdistämismenetelmän välillä eroa. Tutkimus osoitti, että Einsteinin summaan ja keskiarvon laskentaan perustuvien yhdistämismenetelmien välillä ei ollut merkitseviä eroja. Toisena tekijänä tutkittiin sitä, miten painotuskaavan vakion arvoa säätämällä voidaan vaikuttaa palautettavien elementtien kokoon. Vakion arvon säätämisellä oli voimakas vaikutus palautettavien elementtien kokoon. Järjestelmän tehokkuuteen keskitarkkuuksilla mitattuna vakion säätämisellä oli vaihteleva vaikutus riippuen käytetystä evaluointimetriikasta.

Tämä työ on tiettävästi ensimmäinen laajempi TRIXiä käsittelevä tutkimus ja siten se voi ehkä toimia päänavauksena laajemmallekin jatkotutkimukselle. Mielenkiintoisena jatkotutkimuksen mahdollisuutena voisi mainita esimerkiksi tulosjoukkojen päällekkäisyyksien tutkimisen.

8 Lähdeluettelo

Abolhassani, M; Fuhr, N; Malik, S. (2004). HyREX at INEX 2003. Teoksessa Fuhr, N ym. (toim). Initiative for the Evaluation of XML Retrieval (INEX). Proceedings of the Second INEX Workshop. December 15-17, 2003 Schloss Dagstuhl, 27-32.

Amati, G; Van Rijsbergen, C. (2002). Probabililistic Models of Information Retrieval Based on Measuring the Divergence from Randomness. ACM Transactions on Information Systems, Vol 20, No.4, October 2002, 357-389.

Baeza-Yates, R; Ribeiro-Neto, B. (1999). Modern Information Retrieval. Addison-Wesley.

Baeza-Yates, R; Fuhr, N; Maarek, Y (toim.). (2002). Second Edition of the ”XML and Information Retrieval” Workshop Held at SIGIR 2002, Tampere, Finland, Aug 15th, 2002. SIGIR Forum Fall 2002, Vol 36, No. 2.

Bray, T; Paoli, J; Sperberg-McQueen, C; Maler, E; Yergeau, F. (2004). Extensible Markup Language (XML) 1.0 (Third Edition). W3C Recommendation 04 February 2004
Saatavilla www-muodossa: <http://www.w3.org/TR/2004/REC-xml-20040204/>. Viitattu 2.3.2005.

Carmel, D; Maarek, Y; Mandelbrod, M; Mass, Y; Soffer, A. (2003). Searching XML Documents via XML Fragments. Teoksessa Proceedings of SIGIR ’03 July 28- August 1, 2003, Toronto, CANADA. New York, Ny: ACM, 151-158.

Chiamarella, Y; Mulhem, P; Fourel, F. (1996). A Model for Multimedia Information Retrieval. Technical Report, FERMI ESPRIT BRA 8134, University of Glasgow. April 1996. Saatavilla www-muodossa:
<http://www.dcs.gla.ac.uk/fermi/tech_reports/reports/fermi96-4.ps.gz>. Viitattu 15.2.2005.

Clark, J; DeRose, S. (1999). XML Path Language (XPath) Version 1.0. Saatavana: <http://www.w3.org/TR/xpath>. Viitattu 5.3.2005.

Conover, W. (1980). Practical nonparametric statistics (2nd ed.) New York: John Wiley & Sons.

Crestani, F; Lalmas, M; Van Rijsbergen, C; Campbell; I. (1998). ”Is this document relevant? ...probably”: A Survey of Probabilistic Models in Information Retrieval. December 1998 ACM Computing Surveys (CSUR), Volume 30 Issue 4.

Crouch, C; Apte, S; Bapat, H. (2004). An Approach to Structured Retrieval Based on the Extended Vector Model. Teoksessa Fuhr, N ym. (toim). Initiative for the Evaluation of XML Retrieval (INEX). Proceedings of the Second INEX Workshop. December 15-17, 2003 Schloss Dagstuhl, 89-93.

de Vries, A; Kazai, G; Lalmas, M. (2004). Evaluation Metrics 2004. Teoksessa Fuhr, N ym. (toim.) INEX 2004 Workshop Pre-Proceedings. December 6-8, 2004. Schloss Dagstuhl, 249-250.

Extensible Markup Language (XML). Saatavilla www-muodossa: <http://www.w3.org/XML/>. Viitattu 5.3.2005.

Fallside, D; Walmsley, P. (2004). XML Schema Part 0: Primer Second Edition. W3C Recommendation 28 October 2004. Saatavilla www-muodossa: <http://www.w3.org/TR/xmlschema-0/>. Viitattu 5.3.2005.

Fuhr, N; Grossjohann, K. (2001). XIRQL: A Query Language for Information Retrieval in XML Documents. Proceedings of SIGIR ’01 September 9-12, 2001, New Orleans, Louisiana, USA. New York, Ny: ACM, 172-180.

Fuhr, N; Malik, S; Lalmas; M. (2004). Overview of the Iniative for the Evaluation of XML Retrieval (INEX) 2003. Teoksessa Fuhr, N ym. (toim). Initiative for the Evaluation of XML Retrieval (INEX). Proceedings of the Second INEX Workshop. December 15-17, 2003 Schloss Dagstuhl, 1-11.

Geva, S; Leo-Spork, M. (2004). Xpath Inverted File for Information Retrieval. Teoksessa Fuhr, N ym. (toim). Initiative for the Evaluation of XML Retrieval (INEX). Proceedings of the Second INEX Workshop. December 15-17, 2003 Schloss Dagstuhl, 110-117.

Guo, L; Shao, F; Botev, C; Shanmugasundaram, J. (2003). XRANK: Ranked Keyword Search over XML Documents. Teoksessa Proceedings of SIGMOD 2003, June 9-12, 2003, San Diego, CA. New York, Ny: ACM, 16-27.

Gövert, N., Kazai, G., Fuhr, N. & Lalmas, M. (2003). Evaluating the effectiveness of content-oriented XML-retrieval. Saatavilla www-muodossa: <http://www.is.informatik.uni-duisburg.de/bib/fulltext/ir/Goevert_etal:03a.pdf>. Viitattu 5.3.2005.

Gövert, N; Kazai, G. (2003). Overview of the INitiative for the Evaluation of XML retrieval (INEX) 2002. teoksessa Fuhr, N ym. (toim.) Proceedings of the First Workshop of the INitiative for the Evaluation of XML retreival (INEX). December 9-11, 2002, Schloss Dagstuhl, 1-17.

Hawking, D.; Thistlewaite, P; Craswell, N. (1998). ANU/ACSys TREC-6 Experiments. Teoksessa Voorhees, D; Harman, E (toim.). NIST Special Publication 500-240: The Sixth Text REtrieval Conference (TREC 6). Page 275. Saatavilla www-muodossa: <http://trec.nist.gov/pubs/trec6/papers/anu.ps.gz>. Viitattu 5.3.2005.

Heikkilä, T. (1998). Tilastollinen tutkimus. Helsinki: Oy Edita Ab.

Hiemstra, D. (2001). Using Language Models for Information Retrieval. PhD Thesis, University of Twente.

Hull, D. (1993). Using statistical testing in the evaluation of retrieval experiments. Teoksessa Korfhage, R ym. (toim.) Proceedings of the 16th international conference on research and development in information retrieval. New York, NY. ACM, 329-338.

INitiative for the Evaluation of XML Retrieval. Saatavilla www-muodossa: <http://inex.is.informatik.uni-duisburg.de:2004/>. Viitattu 5.3.2005.

Jaideep, R; Anupama, R. (2000). XML: Data's Universal Language. IEEE IT Professional Vol.2, No.3, 32-36.

Järvelin, K. (1995). Tekstitiedonhaku tietokannoista. Espoo: Suomen ATK-kustannus Oy.

Järvelin, K; Niemi, T. (1995). An NF^2 relational interface for document retrieval, restructuring and aggregation. Teoksessa Fox, E; Ingwersen, P; Fidel, R (1995). The 18th International Conference on Research and Development in Information Retrieval (ACM SIGIR '95) Seattle, Wa, July 9-12, 1995. New York, Ny: ACM, 102-110.

Junkkari, M. (2004). An Object-Oriented Representation for Modeling and Managing Part-of Relationships. Journal of Intelligent Information Systems, forthcoming issue.

Kamps, J; Marx, M; de Rijke M; Sigurbjörnsson, B (2003). XML Retrieval: What to Retrieve? Teoksessa Proceedings of SIGIR '03 July 28- August 1, 2003, Toronto, Canada. New York, Ny: ACM.

Kazai, G. (2004). Report of the INEX 2003 Metrics working group. Teoksessa Fuhr, N ym. (toim). Initiative for the Evaluation of XML Retrieval (INEX). Proceedings of the Second INEX Workshop. December 15-17, 2003 Schloss Dagstuhl. 184-191.

Kazai, G; Lalmas, M; Piwowarski, B. (2004). INEX 2004 Relevance Assessment Guide. Saatavilla www-muodossa (vaatii salasanan): <http://inex.is.informatik.uni-duisburg.de:2004/internal/pdf/INEX04_Retrieval_Task.pdf>. Viitattu 5.3.2005.

Kazai, G; Lalmas, M; de Vries, A. (2004). Reliability Tests for the XCG and inex-2002 Metrics. Teoksessa Fuhr, N ym. (toim.) INEX 2004 Workshop Pre-Proceedings. December 6-8, 2004. Schloss Dagstuhl, 33-40.

Keen, E. (1992). Presenting results in experimental retrieval comparisons. Information Processing & Management 28(4), 491-501.

Kekäläinen, J; Junkkari, M; Arvola, P; Aalto, T. (2004). TRIX 2004 – Struggling with the overlap. Teoksessa Fuhr, N ym. (toim.) INEX 2004 Workshop Pre-Proceedings. December 6-8, 2004. Schloss Dagstuhl, 72-75.

Klein, M. (2001). XML, RDF and Relatives. IEEE Intelligent systems Vol. 16, No. 2, 26-28.

Lalmas, M. (2004). Sähköpostiviesti Inex04-participants –postituslistalle 19.7.2004. Aihe: [Inex04-participants] Retrieval Tasks and Submission Runs guidelines.

Larson, R. (2004). Chesire II at INEX ’03: Component and Algorithm Fusion for XML Retrieval. Teoksessa Fuhr, N ym. (toim). Initiative for the Evaluation of XML Retrieval (INEX). Proceedings of the Second INEX Workshop. December 15-17, 2003 Schloss Dagstuhl, 38-45.

Lee, Y; Yoo, S-J; Yoon, K; Berra, P. (1996). Index Structures for Structured Documents. ACM First International Conference on Digital Libraries, Bethesda, Maryland, March 1996, 91-99.

Lee, D; Chuang, H; Seamons, K. (1997). Document Ranking and the Vector-Space Model. IEEE Software March/April 1997, 67-75.

Liu, S; Zou, Q; Chu, W. (2004). Configurable Indexing and Ranking for XML Information Retrieval. Proceedings of SIGIR 2004 July 25-29, Sheffield, South Yorkshire, UK. New York, Ny: ACM, 88-95.

Luk, R; Leong, H; Dillon, T; Chan, A; Croft, B; Allan, J. (2002). A Survey in Indexing and Searching XML Documents. JASIST 53(6), 2002, 415-437.

Maron, M; Kuhns, J.(1960). On Relevance, Probabilistic Indexing and Retrieval. J.ACM 7, 216-244.

Mass, Y; Mandelbrod, M. (2004). Retrieving the most relevant XML Components. Teoksessa Fuhr, N ym. (toim). Initiative for the Evaluation of XML Retrieval (INEX). Proceedings of the Second INEX Workshop. December 15-17, 2003 Schloss Dagstuhl, 53-58.

Mattila, J. (1998). Sumean logiikan oppikirja: Johdatusta sumeaan matematiikkaan. Helsinki:Art House.

Niemi, T. (1983). A seven-tuple representation for hierarchical data structures. Information Systems Vol. 8 No. 3, 1983, 151-157.

Niemi, T; Järvelin, K. (1995). A Straightforward NF^2 relational interface with applications in information retrieval. Information Processing&Management, 31(2), 215-231.

North, S; Hermans, P. (2000). XML Trainer Pro. Helsinki:Oy Edita ab IT Press.

Raghavan, V; Bollman, P; Jung, G. (1989). A critical investigation of recall and precision. ACM Transactions on Information Systems, 7(3), 1989, 205-229.

Robertson, S. (1983). The methodology of information retrieval experiment. Teoksessa Sparck-Jones, K. (toim.). The Information Retrieval Experiment, 9-31. Saatavana www-muodossa: <http://www.itl.nist.gov/iad/894.02/projects/irlib/pubs/ire/ire_index/ire.html>. Viitattu 5.3.2005.

Robertson, S; Walker, S. (1999). Okapi/Keenbow at TREC 8. Saatavilla www-muodossa: <http://trec.nist.gov/pubs/trec8/papers/okapi.pdf>. Viitattu 5.3.2005.

Salton, G. (1989). Automatic text processing: the transformation, analysis and retrieval of information by computer. Reading, Mass.:Addison-Wesley.

Saracevic, T. (1996). Relevance Reconsidered. Teoksessa Information Science: Integration in perspectives. Proceedings of the Second Conference on Conceptions of Library and Information Science (CoLIS 2). Copenhagen (Denmark), 201-218. Saatavana www-muodossa: <http://www.scils.rutgers.edu/~tefko/CoLIS2_1996.doc>. Viitattu 5.3.2005.

Sigurbjörnsson, B ; Kamps. J ; de Rijke, M. (2004). An Element-based Approachto XML Retrieval. Teoksessa Fuhr, N ym. (toim). Initiative for the Evaluation of XML Retrieval (INEX). Proceedings of the Second INEX Workshop. December 15-17, 2003 Schloss Dagstuhl, 19-26.

Sigurbjörnsson, B; Larsen, B; Lalmas, M; Malik, S. (2004). INEX '04 Guidelines for Topic Development. Saatavana www-muodossa (vaatii salasanan) : <http://inex.is.informatik.uni-duisburg.de:2004/internal/pdf/INEX04TopicDevGuide.pdf>. Viitattu 5.3.2005.

Singhal, A; Buckley, C; Mitra, M. (1996). Pivoted Document Length Normalization. Teoksessa Proceedings of the 19th Annual International ACM SIGIR Conference, (Zürich, 1996), 21-29.

Sormunen, E. (2002). Liberal Relevance Criteria of TREC - Counting on Negligible Documents? Beaulieu, M. ym. (toim.): Proceedings of the Twenty-Fifth Annual ACM SIGIR Conference on Research and Development in Information Retrieval. August 11-15, 2002, Tampere, Finland. Special Issue of SIGIR Forum 36, 324-330. Saatavana www-muodossa: <http://www.info.uta.fi/tutkimus/fire/archive/298-sormunen.pdf>. Viitattu 5.3.2005.

Sparck-Jones, K. (1974). Automatic indexing. Journal of Documentation 30(4), 393-432.

Sparck-Jones, K. (1983). The Cranfield tests. Teoksessa Sparck-Jones, K (toim.). The Information Retrieval Experiment, 256-284. Saatavana www-muodossa :

<http://www.itl.nist.gov/iad/894.02/projects/irlib/pubs/ire/ire_index/ire.html>. Viitattu 5.3.2005.

Sparck-Jones, K; Walker, S; Robertson, S. (2000). A Probabilistic model of information retrieval: development and comparative experiments Part 1. Information Processing and Management 36 (2000), 779-808.

Sparck-Jones, K; Walker, S; Robertson, S. (2000). A Probabilistic model of information retrieval: development and comparative experiments Part 2. Information Processing and Management 36 (2000), 809-840.

Tatarinov, I; Viglas, S; Beyer, K; Shanmugasundaram, J; Shekita, E; Zhang, C. (2002). Storing and Querying Ordered XML Using a Relational Database System. teoksessa Proceedings of SIGMOD '2002, June 4-6, Madison, Wisconsin, USA. New York, Ny: ACM, 204-215. Saatavilla www-muodossa: <http://www.csd.uch.gr/~hy561/Papers/OrderedXML-sigmod02.pdf>. Viitattu 5.3.2005.

Trotman, A. (2004). Searching structured documents. Information Processing and Management Volume 40, Number 4, July 2004, 619-632.

Trotman, A; Sigurbjörnsson, B. (2004). Narrowed Extended XPath I. Saatavana www-muodossa (vaatii salasanan) : <http://inex.is.informatik.uni-duisburg.de:2004/internal/pdf/NEXI.pdf>. Viitattu 5.3.2005.

van Rijsbergen, C. (1979). Information Retrieval (2nd ed.) London:Butterworths.

Voorhees, E. (2002). Overview of TREC 2002. Saatavana www-muodossa : <http://trec.nist.gov/pubs/trec11/papers/OVERVIEW.11.pdf>. Viitattu 5.3.2005.

Walsh, N. (1998). A Technical Introduction to XML. O'Reilly XML.com. Saatavilla www-muodossa: <http://www.xml.com/pub/a/98/10/guide0.html>. Viitattu 5.3.2005.

Liite 1

Käytetyt vuoden 2003 CO-kyselyt

91. Internet traffic
92. "query tightening" query tightening "narrow search" narrow search "incremental query answering" incremental query answering
93. "Charles Babbage" Charles Babbage -institute -inst
94. "hyperlink analysis" hyperlink analysis +"topic distillation" topic distillation
95. +"face recognition" face recognition approach
96. +"software cost estimation" software cost estimation
97. Converting Fortran source code
98. "Information Exchange" Information Exchange +XML "Information Integration" Information Integration
99. perl features
100. +association +mining +rule +medical
101. +"t test" +information
102. distributed storage systems for grid computing
103. UML formal logic
104. Toy Story
105. +categorization "textual document" textual document learning evaluation
106. Content protection schemes
107. "artificial intelligence" artificial intelligence AI practical application industry "real world" real world
108. ontology ontologies overview practical example
109. "CPU cooling" CPU cooling "cooling fan design" cooling fan design "heatsink design" heatsink design "heat dissipation" heat dissipation airflow casing
110. "stream delivery" stream delivery "stream synchronization" stream synchronization audio video streaming applications
111. "natural language processing" natural language processing -"programming language" -"modeling language" +"human language" human language
112. +"Cascading Style Sheets" Cascading Style Sheets -"Content Scrambling System"
113. "Markov models" Markov models "user behaviour" user behaviour
114. +women "history computing" history of computing
115. +"IP telephony" IP telephony +challenges
116. "computer assisted art" computer assisted art "computer generated art" computer generated art
117. Patricia Tries
118. shared shared nothing database
119. Optimizing joins in relational databases
120. information retrieval models
121. Real Time Operating Systems
122. Lossy Compression Algorithm
123. multidimensional index "nearest neighbour search" nearest neighbour search
124. application algorithm +clustering +"k means" +"c means" "vector quantization" vector quantization "speech compression" speech compression "image compression" image compression "video compression" video compression
125. +wearable ubiquitous mobile computing devices
126. Open standards for digital video in distance learning

Liite 2

Käytetyt vuoden 2004 CO-kyselyt

162 Text and Index Compression Algorithms
163 +"multi agents" +networks +Internet
164 Knowledge management frameworks and organizational memories
165 technology disabled handicapped people
166 +"tree edit distance" +XML -image tree edit distance
167 query processing spatial data -multimedia -mining
168 New Zealand Digital Library Project
169 +"Query expansion" +"relevance feedback" +web query expansion relevance feedback
170 "topic maps" +web topic maps
171 problems integration +XML technologies +"relational databases" relational databases solutions
172 "retrieval techniques" "textual annotations" "MPEG 7 videos" -image retrieval techniques textual annotations MPEG 7 videos
173 content based music retrieval
174 Internet web page +prefetching algorithms -CPU -memory -disk
175 +"Association Rules" +"Data Mining" +"Large Databases" Association Rules Data Mining Large Databases
176 Secure Web cookies using authentication integrity and confidentiality
177 Difference of two unordered labeled trees or change detection in unordered labeled trees
178 +"Multimedia Information" +Retrieval Multimedia Information
179 +"LANs interconnection" +ATM LANs interconnection
180 +"Electronic books" +"copy protection" Electronic books copy protection
181 +keyword search +structured +XML document
182 +type +implementation +workflow +system +web
183 +software +quality
184 statistical approaches pattern recognition
185 gesture recognition
186 electronic business data mining
187 LSI +"dimension reduction" "latent semantics" "random projection" -stoplist "curse of dimensionality" dimension reduction latent semantics random projection curse of dimensionality
188 +new +fortran +90 +compiler
189 +"machine learning" +clustering +"large scale" +algorithm machine learning large scale
190 "e commerce" "e business" "data warehouse" e commerce business data warehouse
191 technical "MPEG 4"
192 cybersickness nausea "virtual reality" virtual reality
193 Good Turing estimate smoothing
194 "multi layer perceptron" "radial basis functions" comparison radial basis functions
195 +crawler policy +WWW "search engine" search engine
196 "multimedia e learning" training Internet Web +"education problem" "communication problem" "time constraints" computer -devices multimedia e learning education problem communication time constraints
197 "data compression" +"information retrieval" data compression information retrieval

198 application development +python +java comparison
199 +mobile distributed wireless +"peer to peer" network peer to peer
200 reusing +metadata for +media on +mobile devices
201 +web WWW "relevance score" +ranking relevance score

Liite 3

Analysoi_mm –ohjelman perl-kielinen lähdekoodi

```
#!/usr/bin/perl

#
#analysoi_mm.pl -ohjelma TRIX:n tuloslistojen tutkimiseen
#tekijä Timo Aalto (etunimi.sukunimi(at)uta.fi)
#v 1.0 29.11.2004
#käyttö:perl analysoi_mm.pl [tiedosto, esim. result.xml] [topikkeja
tuloslistassa, esim. 36] [analysoitava top n, esim. 100]
#
#
use strict;

open (TULOS,$ARGV[0]) or die "syntaksi: analysoi_mm.pl [tiedosto,
esim. result.xml] [topikkeja tuloslistassa, esim. 36] [analysoitava
top n, esim. 100]\n";


my $counter = 0;
my $rivi;
my $article=0;
my $sec=0;
my $para=0;
my $list=0;
my $abs=0;
my $head=0;
my $muut=0;

my $topikkeja=$ARGV[1]; #kuinka monta topikkia tuloslistassa on,
vaikuttaa prosenttiosuuksien laskentaan
my $raja=$ARGV[2]; #kuinka monta ekaa tulosta katotaan/topikki.
1500=kaikki
my $summa=$raja*$topikkeja;
while ($rivi = <TULOS>){
  if ($rivi =~m/\/(\w+\-*\w*\[\d+\])<\/path><rank>(\d+)<\/rank>/){
#napataan viimeisen elem. tunniste $1 ja rank $2

      $counter++;
      my $temp=$1;
      if ($2<$raja+1){ #rajaus, niin kauan analysoidaan kun on alle
rajan
      if ($temp=~m/^(article\[|bdy\[)/){$article++;} #mikä elementti,
lisätään oikeaan muuttujaan.
      elsif ($temp=~m/^(abs\[)/){$abs++;}
      elsif ($temp=~m/^(sec\[|ss1\[|ss2\[|ss3\[)/){$sec++;}
      elsif ($temp=~m/^(p\[|ip1\[|ip2\[|ilrj\[|ip3\[|ip4\[|ip5\[|item-
none\[|p1\[|p2\[|p3\[)/){$para++;}
      #elsif
#($temp=~m/^(dl\[|l1\[|l2\[|l3\[|l4\[|l5\[|l6\[|l7\[|l8\[|9l\[la\[|lb\
[|lc\[|ld\[|le\[|list\[|numeric-list\[|numeric-rbrace\[|bullet-
list\[)/)#{$muut++;}
      #elsif
($temp=~m/^(h\[|h1\[|h1a\[|h2\[|h2a\[|h3\[|h4\[)/){$muut++;}
      else{$muut++;}
}
else{}
}
```

```
else{}
}

print "article|bdy\n".$article."\t".$article/$summa."\n";
print "abs\n".$abs."\t".$abs/$summa."\n";
print "sec|ss1|ss2\n".$sec."\t".$sec/$summa."\n";
print "p|ip1\n".$para."\t".$para/$summa."\n";
#print "list\n".$list."\t".$list/$summa."\n";
#print "head\n".$head."\t".$head/$summa."\n";
print "muut\n".$muut."\t".$muut/$summa."\n";
print "analysoituja tulosia ".$summa." (".$raja." X
".$topikkeja.")\n";
print "tuloksia listassa yhteensä ".$counter."\n";
close TULOS;

exit 0;
```