Normalisoitu pakkausetäisyys: sovelluksia ja variaatioita

Timo Sand

Kandidaatintutkielma
HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

Helsinki, 8. joulukuuta 2013

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

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

Tekijä — Författare — Author
Timo Sand

Työn nimi — Arbetets titel — Title

Normalisoitu pakkausetäisyys:sovelluksia ja variaatioita

Oppiaine — Läroämne — Subject
Tietojenkäsittelytiede

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
Kandidaatintutkielma	8. joulukuuta 2013	22

Tiivistelmä — Referat — Abstract

Esittelemme metodin pakkauspohjaiseen klusterointiin. Tämä metodi ei ole riippuvainen kohteen ominaisuuksista tai taustatiedoista. Metodi toimii seuraavanlaisesti: Ensin määritämme parametrittoman, universaalin, samankaltaisuusetäisyyden normalisoitu pakkausetäisyys (NCD), joka lasketaan pakatuista tiedostoista, yksittäin ja pareittain katenoituna. Seuraavaksi sovellamme hierarkkista klusterointimenetelmää. NCD ei ole rajattu tiettyyn sovellusalueeseen ja sitä voi käyttää useaan alueeseen kerralla. Esittelemme Kolmogorov-kompleksisuutta ja tähän perustuvan normalisoidun informaatioetäisyyden, jotka muodostavat NCD:n pohjan. Tämän lisäksi käymme läpi NCD:n kohinansietokykyä ja rajoitteita pakkaajan valinnassa NCD:n käyttöä varten. Seuraavaksi esitellään Google-samankaltaisuusetäisyys, joka perustuu samaan toimintaperiaatteeseen kuin NCD, mutta käyttää internethakutuloksien lukumäärää samankaltaisuuden mittarina.

ACM Computing Classification System (CCS):
Information systems - Similarity measures
Theory of computation - Unsupervised learning and clustering
Data mining - Clustering

Avainsanat — Nyckelord — Keywords
Samankaltaisuus, Klusterointi, Valvomaton oppiminen, Tiedonlouhinta

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — Övriga uppgifter — Additional information

Sisältö

1 Johdanto

Kaikki data on luotu samanveroiseksi, mutta jotkut datat ovat samankaltaisempia kuin toiset. Esitämme tavan, jolla ilmaista tämä samankaltaisuus, käyttäen Cilibrasin ja Vitányin esittelemää samankaltaisuuden metriikkaa *(engl. similarity metric)*, joka perustuu tiedoston pakkaamiseen. [CV05] Metriikka on parametriton, eli se ei käytä datan ominaisuuksia tai taustatietoja, ja sitä voi soveltaa eri aloihin ilman muunnoksia. Metriikka on universaali, joten se kykenee approksimoimaan objektiparin hallitsevan piirteen *(engl. dominant feature)*, jonka perusteella parin samankaltaisuus arvioidaan. Se on vakaa siinä mielessä, että sen tulokset ovat riippumattomia käytetystä pakkaajasta [CV05]. Pakkaajalla tarkoitetaan pakkausohjelmaa kuten *gzip, ppmz, bzip2*.

Pakkaukseen perustuva samankaltaisuus *(engl. Compression-Based Similarity)* on universaali metriikka, jonka kehittivät Cilibrasi ja Vitányi [CV05]. Yksinkertaistettuna tämä tarkoitta, että kaksi objektia ovat lähellä toisiaan, jos voimme 'pakata' yhden objektin huomattavasti tiiviimmin toisen objektin datalla. Abstraktina ideana toimii se, että voimme kuvailla ytimekkäämmin yhden objektin toisen avulla, mikäli objektit ovat samankaltaisia. Tämän esittelemme luvussa 2 ja samalla käymme läpi mihin teoriaan algoritmi perustuu sekä miten se toimii. Edellä mainitun vakauden esittämiseen voimme käyttää useaa tosielämän pakkausalgoritmia: tilastollista (PPMZ), Lempel-Ziv -algoritmiin pohjautuvaa hakemistollista (gzip), lohkoperusteista (bzip2) tai erityistarkoitusta varten suunniteltua (Gencompress).

Käytetyt yhtälöt pätevät aina lähes täsmälleen, koska siirrytään Kolmogorov-kompleksisuudesta laskettavaan approksimaatioon. Approksimaation virhe on hyvin pieni: $O(\log n)$ (n on esimerkiksi tiedoston pituus).

Tarkoituksemme on koota yhteen samankaltaisuuden metriikkaan kaikki tehokkaat etäisyydet *(engl. effective distances)*: tehokkaat versiot Hammingin etäisyydestä, Euklidisesta etäisyydestä, Lempel-Ziv etäisyydestä ja niin edelleen. Tämän metriikan tulee olla niin yleinen, että se toimii yhtäläisesti ja samanaikaisesti kaikilla aloilla: musiikilla, tekstillä, kirjallisuudella, ohjelmilla,

genomeilla, luonnollisen kielen määrityksillä. Sen pitäisi pystyä samanaikaisesti havaitsemaan kaikki samankaltaisuudet objektien välillä, joita muut etäisyydet havaitsevat erikseen.

Kun määrittelemme ryhmän sallittavia etäisyyksiä *(engl. admissible distances)* haluamme sulkea pois epärealistiset etäisyydet, kuten $f(x, y) = \frac{1}{2}$ jokaiselle $x \neq y$. Saavutamme tämän rajoittamalla objektien määrää annetussa etäisyydessä objektista. Teemme tämän huomioimalla vain todellisia etäisyyksiä seuraavasti: oletamme kiinnitetyn ohjelmointikielen, joka toimii viitekielenä. Tämä ohjelmointikieli voi olla yleinen kieli kuten LISP tai Ruby, mutta se voi myös olla määrätty universaali Turingin kone. [CV05, CV07] Valinnalla ei kuitenkaan ole merkitystä, sillä teoria on invariantti ohjelmointikielen muutoksille, kunhan pysytään tehdyssä valinnassa.

Jotta voimme soveltaa tätä ideaalia ja tarkkaa matemaattista teoriaa käytäntöön, pitää meidän korvata ei laskettavissa oleva Kolmogorov-kompleksisuus approksimaatiolla käyttäen standardia pakkaajaa.

Luvussa 3 esittelemme algoritmin käyttöä klusteroinnissa. Aloitamme siitä, miten yleisesti *Normalisoidun pakkausetäisyyden* (*NCD*) avulla pystymme klusteroimaan tuloksia eri kategorioihin; miten musiikkikappaleet klusteroituvat saman genren tai artistin alle, miten erityyppiset tiedostot kategorisoituvat oikeisiin ryhmiin.

Luvussa 4 esitellään NCD:n ominaisuuksia ja ongelmia. Ensiksi esitellään NCD:n kohinansietokykyä, eli katsotaan mitä tapahtuu kun lisätään vähitellen kohinaa toiseen tiedostoista, jota pakataan, ja mittaamalla samankaltaisuutta tämän jälkeen [CAO07]. Saamme nähdä miten paljon kohina vaikuttaa NCD:n laskemiin etäisyyksiin ja huonontaako se klusteroinnin tuloksia.

Mikään algoritmi ei ole täydellinen, ja NCD-algoritmillakin on ongelmansa. Algoritmissa itsessään ei ole selvää heikkoutta, mutta sen käytössä on otettava pakkaajan valinta huomioon, koska monet suosituista pakkausalgoritmeista ovat optimoituja tietyn kokoisille tiedostoille. Näissä algoritmeissa on jokin rajoittava tekijä, joka määrittelee mikä tiedostokoko on sopiva [CAO05]. Jos tiedostokoko on pienempi kuin rajoitus, on pakkaus tehokasta, mutta kun koko

on liian suuri, pakkauksesta tulee huomattavasti tehottomampaa. Esittelemme tuloksia eri pakkausalgoritmien vertailuista ja mikä näistä algoritmeista on havaittu parhaimmaksi NCD:n kanssa käytettäväksi.

NCD ei ole ainut metriikka, jolla voidaan mitata samankaltaisuutta. Internetiä hyödyntäen on tehty metriikka, joka käyttää hakukoneita samankaltaisuuden tutkimiseen; tämä on nimetty Google-samankaltaisuusetäisyydeksi *(engl. Google Similarity Distance)* [CV07]. Tämä toimii myös muilla hakukoneilla kuten Bingillä. Luvussa 5 esittelemme tämän.

2 Normalisoitu pakkausetäisyys

2.1 Kolmogorov-kompleksisuus

Merkkijonon x *Kolmogorov-kompleksisuus* on pituus bitteinä, lyhimmästä tietokoneohjelmasta joka ilman syötettä palauttaa merkkijonon x; tämä merkitään $K(x) = K(x|\lambda)$, λ esittää tyhjää syötettä. Lyhimmän tietokoneohjelman pituus, joka palauttaa x syötteellä y, on Kolmogorov-kompleksisuus x:stä syötteellä y; tämä merkitään $K(x|y)$. Yksi tapa hahmottaa Kolmogorov-kompleksisuutta $K(x)$ on ajatella se pituutena bitteinä x:n tiiviimmin pakatussa muodossa, josta voidaan purkaa x pakkauksenpurkuohjelmalla.

2.2 Normalisoitu informaatioetäisyys

Artikkelissa [CV05] on esitelty *informaatioetäisyys* $E(x, y)$, joka on määritelty lyhimpänä binääriohjelman pituutena, joka syötteellä x laskee y:n ja syötteellä y laskee x:n. Tämä lasketaan seuraavasti:

$$E(x, y) = max\{K(x|y), K(y|x)\}. \tag{1}$$

Normalisoitu versio informaatioetäisyydestä $E(x, y)$, jota kutsutaan *normali-*

soiduksi informaatioetäisyydeksi, on määritelty seuraavasti

$$NID(x,y) = \frac{max\{K(x|y), K(y|x)\}}{max\{K(x), K(y)\}}. \quad (2)$$

Tämä on paras mahdollinen *samankaltaisuuden metriikka*, ja sen on osoitettu [CV05] täyttävän vaatimukset etäisyyden metriikaksi. *NID* ei kuitenkaan ole laskettava tai edes semi-laskettava, koska Turingin määritelmän mukaan Kolmogorov-kompleksisuus ei ole laskettava [CV05]. Nimittäjän approksimointi annetulla pakkaajalla C on $max\{C(x), C(y)\}$. Osoittajan paras approksimaatio on $max\{C(xy), C(yx)\} - \min\{C(x), C(y)\}$ [CV05]. Kun *NID* approksimoidaan oikealla pakkaajalla, saadaan tulos jota kutsutaan *normalisoiduksi pakkausetäisyydeksi*. Tämä esitellään formaalisti myöhemmin.

2.3 Normaali pakkaaja

Seuraavaksi esitämme aksioomia, jotka määrittelevät laajan joukon pakkaajia ja samalla varmistavat *normalisoidussa pakkausetäisyydessä* halutut ominaisuudet. Näihin pakkaajiin kuuluvat monet tosielämän pakkaajat.

Pakkaaja C on *normaali* jos se täyttää seuraavat aksioomat:

1. *Idempotenssi*: $C(xx) = C(x)$ ja $C(\lambda) = 0$, jossa λ on tyhjä merkkijono,
2. *Monotonisuus*: $C(xy) \geq C(x)$,
3. *Symmetrisyys*: $C(xy) = C(yx)$ ja
4. *Distributiivisuus*: $C(xy) + C(z) \leq C(xz) + C(yz)$.

2.4 Normalisoitu Pakkausetäisyys

Jotta voimme soveltaa tätä ideaalia ja tarkkaa matemaattista teoriaa käytäntöön, pitää meidän korvata ei laskettavissa oleva Kolmogorov-kompleksisuus approksimaatiolla käyttäen standardia pakkaajaa.

Normalisoitua versiota *hyväksyttävästä etäisyydestä* $E_c(x, y)$, joka on pakkaajaan C pohjautuva approksimaatio normalisoidusta informaatioetäisyydestä, kutsutaan nimellä *Normalisoitu Pakkausetäisyys (NCD)* [CV05]. Tämä lasketaan seuraavasti

$$NCD(x, y) = \frac{C(xy) - min\{C(x), C(y)\}}{max\{C(x), C(y)\}}. \tag{3}$$

NCD on funktioiden joukko, joka ottaa argumenteiksi kaksi objektia (esimerkiksi tiedostoja tai Googlen hakusanoja) ja tiivistää nämä, erillisinä ja yhdistettyinä. Tämä funktioiden joukko on parametrisoitu käytetyn pakkaajan C mukaan.

Käytännössä *NCD*:n tulos on välillä $0 \leq r \leq 1 + \epsilon$, mikä vastaa kahden tiedoston eroa toisistaan; mitä pienempi luku, sitä enemmän tiedostot ovat samankaltaisia. Tosielämässä pakkausalgoritmit eivät ole yhtä tehokkaita kuin teoreettiset mallit, joten virhemarginaali ϵ on lisätty ylärajaan. Suurimmalle osalle näistä algoritmeista on epätodennäköistä että $\epsilon > 0.1$.

Luonnollinen tulkinta *NCD*:stä, jos oletetaan $C(y) \geq C(x)$, on

$$NCD(x, y) = \frac{C(xy) - C(x)}{C(y)}. \tag{4}$$

Eli etäisyys x:n ja y:n välillä on suhde y:n parannuksesta, kun y pakataan käyttäen x:ää, ja y:n pakkauksesta yksinään; suhde ilmaistaan etäisyytenä bittien lukumääränä kummankin pakatun version välillä.

Kun pakkaaja on normaali, niin *NCD* on normalisoitu hyväksyttävä etäisyys, joka täyttää metriikan yhtälöt, eli se on samankaltaisuuden metriikka.

Taulukossa 1 esitellään muutama arvo joilla NCD lasketaan ja mikä tulos näistä saadaan.

Arvot	Laskutoimitus	NCD:n arvo
$C(xx) = 26, C(x) = 17$	$\frac{26-17}{17}$	0.529
$C(xx) = 33, C(x) = 22$	$\frac{33-22}{22}$	0.5
$C(xx) = 30, C(x) = 17$	$\frac{30-17}{17}$	0.765
$C(xx) = 20, C(x) = 14$	$\frac{20-14}{14}$	0.428
$C(xx) = 16, C(x) = 14$	$\frac{16-14}{14}$	0.143
$C(xx) = 28, C(x) = 14$	$\frac{28-14}{14}$	1
$C(xy) = 37, C(x) = 28, C(y) = 30$	$\frac{37-28}{30}$	0.3

Taulukko 1: *Normalisoituja pakkausetäisyyksiä eri syötteille.*

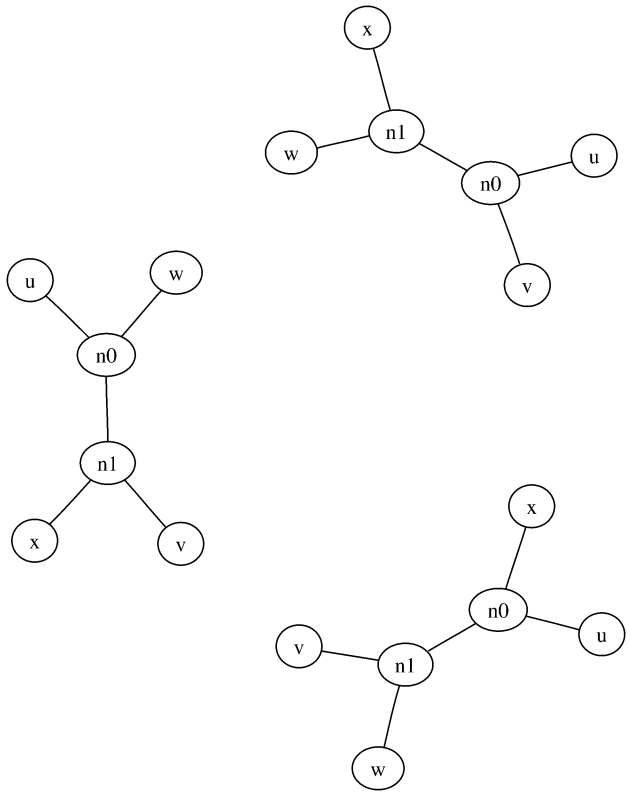
3 Klusterointi

Yksi NCD:n käyttötarkoituksista on objektien klusterointi ilman, että datasta tarvitsee etukäteen kerätä ominaisuuksia. Täten samaa menetelmää voidaan hyödyntää klusteroimaan täysin eri tyyppisiä datajoukkoja. Klusterit ovat ryhmä objekteja, jotka ovat samankaltaisia valitun metriikan mukaan. Tässä tarkoitus on analysoida dataa, joka on tunnisteeton ja siitä ei tiedetä etukäteen klustereiden lukumäärää. NCD:n kanssa käytetään yleisesti hierarkkista klusterointia, koska sitä pidetään yhtenä parhaista valvomattoman oppimisen *(engl. unsupervised learning)* metodeina. Yksi luonnollisimmista tavoista esittää objektien suhteet on dendrogrammi, joka on yleensä suunnattu binääripuu tai suuntaamaton ternääripuu [CVDW04]. Rakentaaksemme tällainen puu etäisyysmatriisista käytämme nelikkomenetelmää.

3.1 Nelikkomenetelmä

Meillä on määrätty joukko objekteja ja näiden pareittaisista NCD:n arvoista muodostuvat etäisyysmatriisin alkiot, jotka edustavat etäisyyksiä kaikkien objektiparien välillä. Tässä matriisissa pareittaiset suhteet ovat raa'assa muodossa ja täten tietoa on vaikea hahmottaa. Jotta voimme klusteroida objektit hierarkkisesti määritämme ternääripuun *(engl. ternary tree)*, joka vastaa etäisyysmatriisin tietosisältöä jonkin valitun kustannusmittarin mukaan.

Käsittelemme jokaista neljän objektin ryhmää n kokoisesta joukosta. Näiden



Kuva 1: *Kolme mahdollista nelikkotopologiaa* [CV05]

ryhmien lukumäärä on $\binom{n}{4}$. Jokaisesta ryhmästä $\{u, v, w, x\}$ rakennamme puun, jossa jokaisella sisäsolmulla on 3 naapuria, eli puu koostuu kahdesta alipuusta, joissa on kummassakin kaksi lapsisolmua. Tällaista puuta kutsutaan *nelikkotopologiaksi (engl. quartet topology)* [CV05] tai *nelikoksi (engl. quartet)* [CVDW04]. On olemassa kolme vaihtoehtoa miten tämä puu rakentuu: a) $uv|wx$, b) $uw|vx$, c) $ux|vw$, joissa pystyviiva erottelee lapsisolmuparit alipuihin. Kuvassa 1 näkyy nelikoiden vaihtoehdot.

Jokaista määrättyä puuta T ja jokaista lapsisolmumerkin ryhmää $\{u, v, w, x\}$ kohden sanomme, että T on konsistentti $uv|wx$:n kanssa, jos ja vain jos polku u:sta v:hen ei risteä polun w:stä x:än kanssa.

Voimme kuvitella isomman puun rakentuvan monesta pienemmästä nelikosta. Yleisesti nelikkomenetelmän tarkoitus on löytää tai approksimoida mahdollisimman lähelle puu, joka sisältää maksimaalisen määrän nelikoita määrätystä

joukosta Q. Puun T kustannus lasketaan seuraavasti:

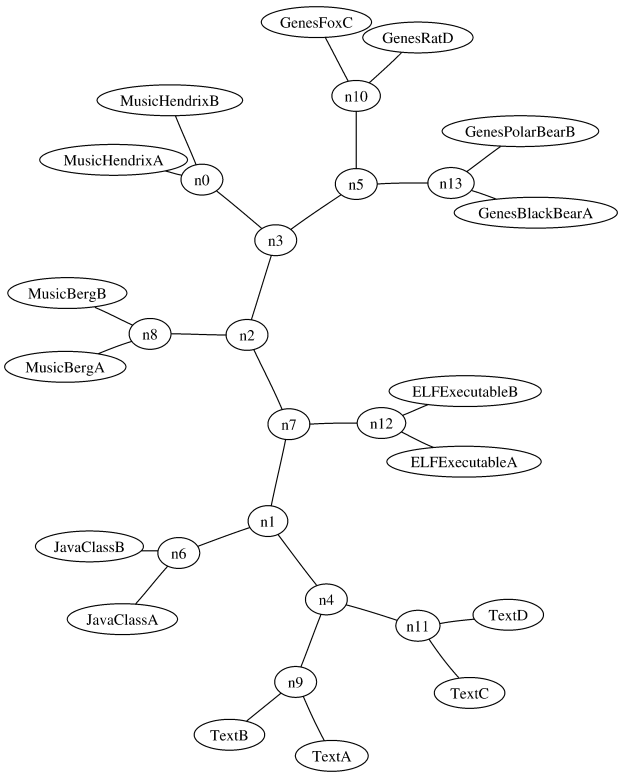
$$C_T = \sum_{\{u,v,w,x\} \subseteq N} \{C_{uv|wx} : T \text{ on konsistentti } uv|wx \text{ kanssa}\}.$$

Jotta voimme vertailla puiden pisteitä C_T, niin nämä pitää normalisoida; nyt 0 vastaa huonointa ja 1 parasta pistearvoa. Tätä termiä kutsutaan *normalisoiduksi puun hyötytulokseksi (engl. normalized tree benefit score)* $S(T)$ [CV05]. Pyrkimys on löytää puu jonka $S(T)$:n arvo on maksimaalinen, eli pienin kokonaiskustannus. Tällaisen puun löytäminen tosin on NP-vaikea, eli se on erittäin epätodennäköistä, joskin välillä löytyy vastaus ja aina voidaan approksimoida se. Tämän laskeminen on erittäin intensiivistä ja vaatii paljon resursseja, ja siihen on sen takia kehitetty yksinkertainen heuristinen metodi, joka pohjautuu satunnaisuuteen ja vuorikiipeilijä-algoritmiin *(engl. hill-climbing)*. Ensiksi luodaan satunnainen puu, johon luodaan $2n-2$ solmua. Näistä n on lehtisolmuja, nimettynä objektien mukaan, ja $n-2$ sisäsolmua jotka merkitään kirjaimella 'n' ja uniikilla numerolla. Jokaisella sisäsolmulla on tasan 3 kaarta. Tälle puulle lasketaan $S(T)$ ja se asetetaan nykyiseksi parhaaksi puuksi. Puulle on määritelty operaatioita mitä voidaan suorittaa [CV05]. Puulle suoritetaan mutaatio (yksi tai useampi operaatio) T:stä T':hen ja sitten lasketaan $S(T')$, jos $S(T') > S(T)$, niin T':stä tulee uusi paras puu. Mutaatioita suoritetaan kunnes $S(T')$ on 1 tai kunnes parempia puita ei löydy järkevässä ajassa. Suorituksen lopussa tulostetaan paras puu.

Klusteroinnista on usea esimerkki olemassa ja käsittelemme tässä muutaman näistä. Ensimmäiseksi tarkastelemme musiikkigenrejen tunnistamista. Objekteiksi valittiin 12 klassista kappaletta, 12 jazz kappaletta ja 12 rock kappaletta. Tulokseksi saadun puun $S(T)$ on 0.858. Kappaleet ryhmittyivät suurimmaksi osaksi oikeisiin kategorioihin, mutta poikkeuksiakin oli. Tämä nähdään jo alhaisesta $S(T)$:n arvosta.

Seuraavaksi on klusteroitu sama 12 klassisen kappaleen joukko. Puun kustannukseksi saatiin $S(T) = 0.958$ ja kappaleet ryhmittyvät säveltäjän mukaan.

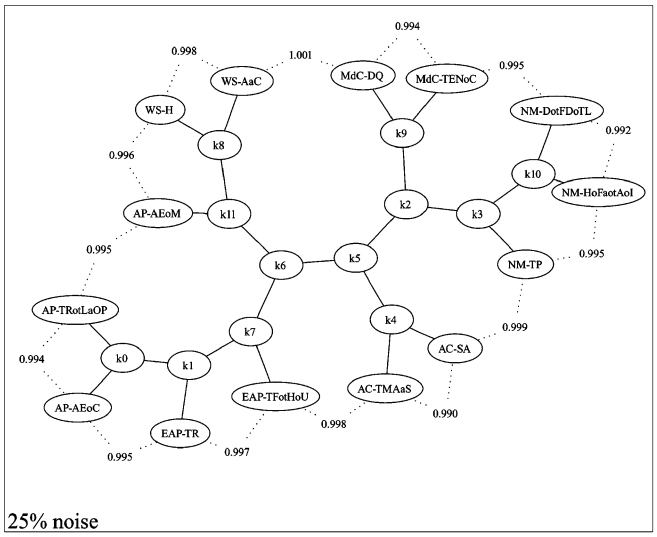
Empiirisesti on havaittu, että hierarkkinen klusterointi toimii parhaiten pienille joukoille (alle 25 objektia), ja sen toimivuus laskee huomattavasti isoille



Kuva 2: *Eri tiedostotyyppien ryhmittely. Puu vastaa hyvin NCD:n etäisyysmatriisia:* $S(T) = 0.984$. [CV05]

joukoille (yli 40 objektia). Yleinen ratkaisu isojen joukkojen hierarkkiseen klusterointiin on, että ensiksi klusteroidaan epähierarkkisesti, ja sitten klusteroimalla hierarkkisesti tästä saatavat klusterit.

Esimerkkinä NCD:n kyvystä erotella eri aihealueet tehokkaasti on kuva 2. Tässä on klusteroitu seuraavia tiedostotyyppejä: i) neljä mitokondrioiden geenisekvenssiä (jääkarhu, kettu, rotta, mustakarhu); ii) neljä kappaletta kirjasta 'The Zeppelinś Passenger'; iii) neljä MIDI tiedostoa, joita ei ole käsitelty (Jimi Hendrix, Debussy); iv) kaksi Linux x86 ELF ohjelmatiedostoa ja v) kaksi käännettyä Java luokkaa. Pakkaajana käytettiin bzip2 -ohjelmaa. Puun kustannus on $S(T) = 0.984$. Tämä koe näyttää miten tehokas ja universaali NCD:n käyttö klusteroinnissa on [CV05].

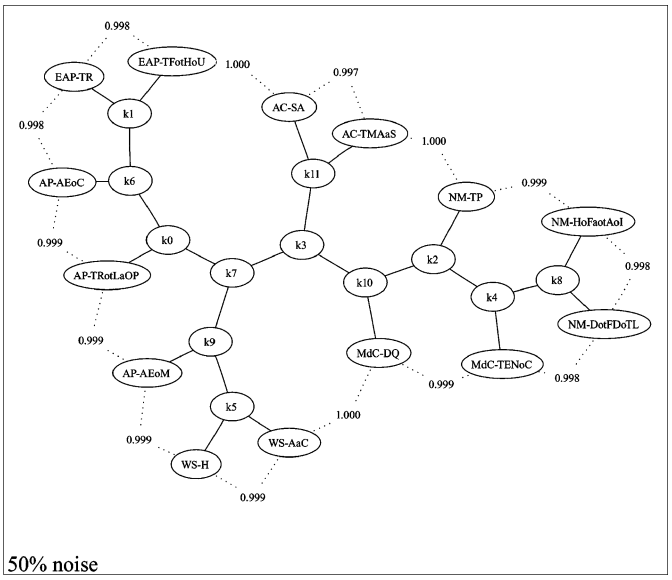


Kuva 3: *Normalisoitujen pakkausetäisyyksien pohjalta klusteroituja kirjoja eri kirjoittajilta kun tiedostoihin on lisätty 25% kohinaa. Merkinnät ovat kirjailijan nimikirjaimet: AC = Agatha Christie, AP = Alexander Pope, EAP = Edgar Allan Poe, WS = William Shakespeare ja NM = Niccolo Machiavelli.* [CAO07]

4 Algoritmin ongelmat ja ominaisuudet

4.1 Kohinansietokyky

Kun NCD:tä käytetään kahteen eri tiedostoon toista näistä voi pitää kohinallisena versiona ensimmäisestä. Kohinan lisääminen progressiivisesti tiedostoon voi tuottaa tietoa mittarista *(engl. measure)* itsestään. Kuvissa 3, 4, 5 on ternääripuut kirjojen klusteroinnista, kun objekteihin on lisätty kohinaa. Kuvissa on merkitty kohinan osuus. Tämän vastaavuuden perusteella voimme tehdä teoreettisen päätelmän odotetusta kohinan lisäämisen vaikutuksesta algoritmiin, mikä selittää miksi NCD voi saada suurempia arvoja kuin 1 joissain tapauksissa. [CAO07]



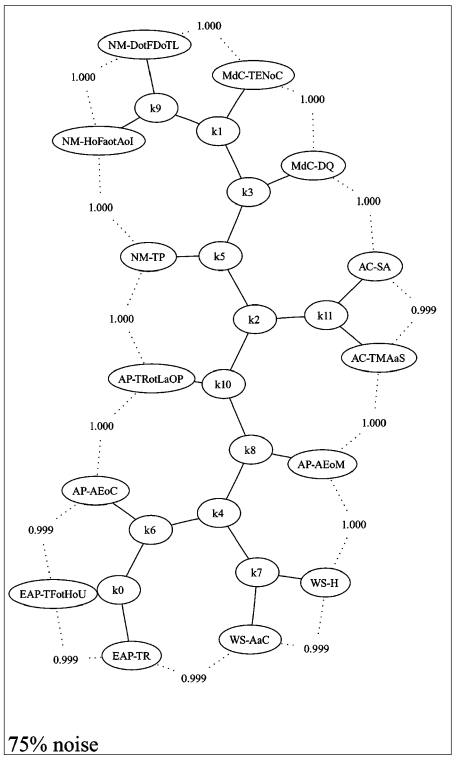
Kuva 4: *Kuvan 3 tiedostoihin on lisätty 50% kohinaa.* [CAO07]

4.2 Pakkaajan valinta

Artikkelissa [CAO05] on osoitettu, että pakkaaja, jota käytetään NCD:n laskemiseen, ei ole idempotentti kaikissa tapauksissa, koska objektien koko ja pakkaajan ikkunakoko vääristävät etäisyyksiä ja täten aiheuttavat poikkeuksia samankaltaisuuksissa. Yhteyttä etäisyyden tarkkuudesta ja objektien koosta on analysoitu monella tunnetulla pakkaajalla ja erityisesti seuraavilla kolmella: *bzip2*, *gzip* ja *PPMZ*.

Vertailussa käytettiin Cilibrasin toteuttamaa CompLearn -työkalua [Cil], josta löytyy bzip2 ja gzip pakkaajat. Aineistona käytettiin tunnettua Calgary Corpus -kokoelmaa [WBC], joka on 18 tiedoston kokoelma, jolla mitataan pakkausalgoritmien suorituskykyä. Kokoelmassa on 9 eri tyyppistä tiedostoa, jotta voidaan saada laaja näkemys pakkausalgoritmin toiminnasta. Mukana on muun muassa kuva, kirjoja, artikkeleita, lähdekoodia ja tietokoneohjelmia.

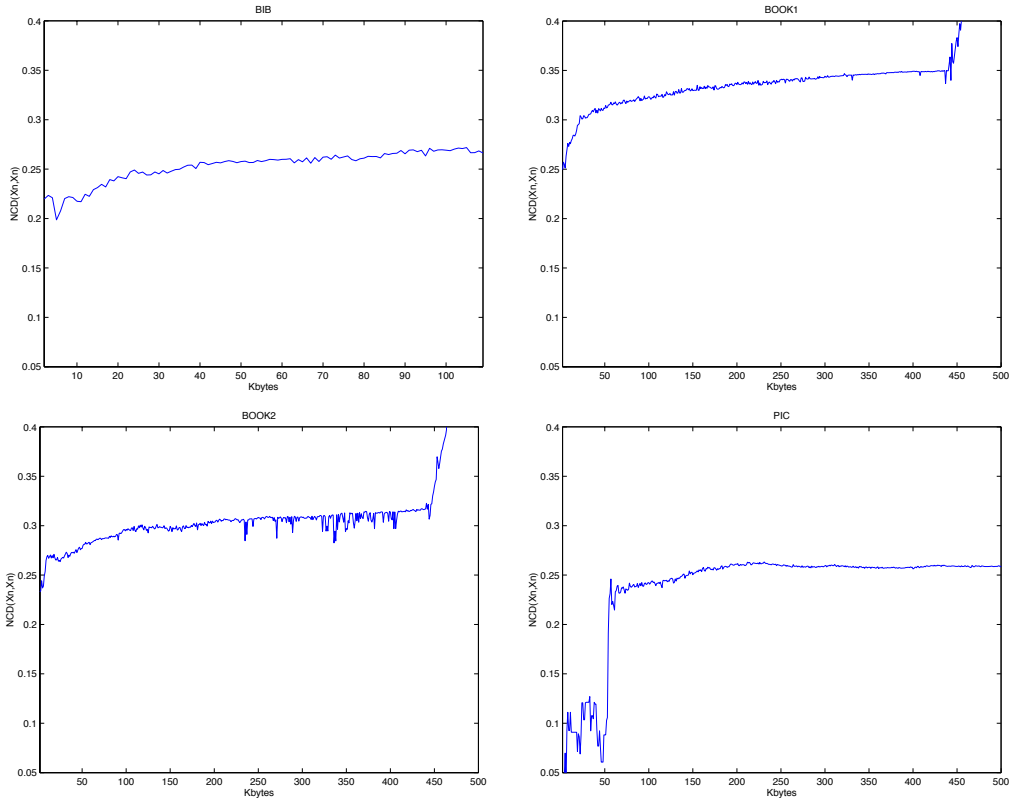
Kaikkia vertailun objekteja käsitellään merkkijonoina, jotka koostuvat tavuista. Jotta voidaan todeta pakkaajan idempotenssin pätevyys, kaikki objektit vertaillaan itsensä kanssa.



Kuva 5: *Kuvan 3 tiedostoihin on lisätty 75% kohinaa.* [CAO07]

Kuvassa 6 huomataan, että $NCD(x, x)$ on väliltä 0.2 ja 0.3, kun tiedostojen koko on sopiva bzip2 pakkaajalle. Kun taas arvo on väliltä 0.25 ja 0.9, tiedostokoko ei ole enää sopiva. Rajoittava tekijä bzip2 pakkaajassa on tälle määriteltävä lohkokoko *(engl. block size)*, joka on oletusarvoisesti 900 kilotavua, eli jos pakattavan tiedoston koko on tätä suurempi niin se pilkotaan alle 900 kilotavun kokoisiksi palasiksi ennen pakkausta. Tällaisesta pilkkomisesta aiheutuu se, ettei eri osien välillä pysty enää pakkaamaan samankaltaisuuksia ja täten pakattu koko ei ole paras mahdollinen.

Kuva 7 osoittaa, että $NCD(x, x)$ on väliltä 0.0 ja 0.1, kun tiedostokoko on sopiva gzip pakkaajalle, ja kun tiedostokoko on liian suuri, niin arvot nousevat jopa yli yhden. Vastaavanlaisesti kuin bzip2 pakkaajassa on rajoittavana tekijänä lohkokoko, on gzip pakkaajassa liukuva ikkuna *(engl. sliding window)* ja eteenpäinkatseluikkuna *(engl. lookahead window)*. Liukuva ikkuna on puskuri, jossa säilötään n viimeisintä merkkiä, missä n on liukuvan ikkunan koko.

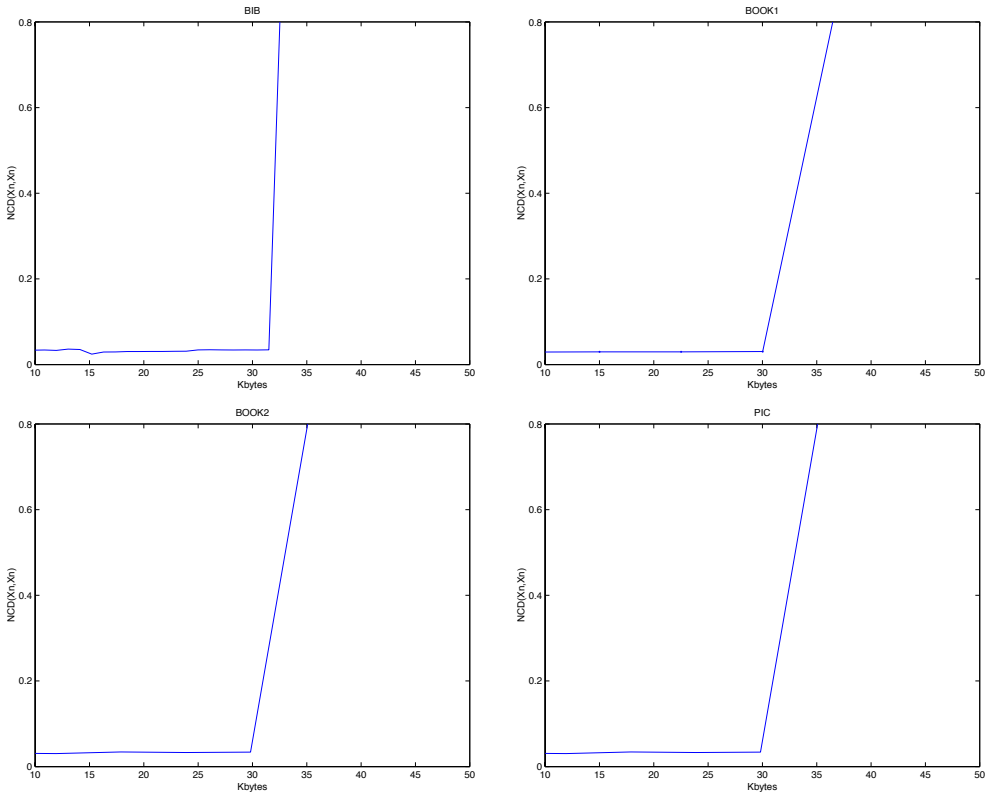


Kuva 6: *Normalisoidut pakkausetäisyydet ensimmäisestä n tavusta, neljälle tiedostolle Calgary Corpus -kokoelmasta, käyttäen* ***bzip2*** *pakkaajaa asetuksella '–best'. Tiedostot vertailtiin itsensä kanssa.* [CAO05]

Eteenpäinkatseluikkuna taas on puskuri, jossa on m tulevaa merkkiä, missä m on eteenpäinkatseluikkunan koko.

Nämä ikkunakoot aiheuttavat sen, että parhaimmallakin asetuksella gzip kykenee pakkaamaan tehokkaasti vain enintään 64 kilotavua olevia tiedostoja.

Mikä tahansa samankaltaisuusetäisyys olisi pitänyt arvoltaan olla 0 tai hyvin lähellä sitä, kun vertaillaan identtisiä objekteja. Pakkaajien vertailussa kerätyt empiiriset havainnot osoittavat, että objektien koko vääristää NCD:n tulosta riippumatta datan tyypistä (kirja, artikkeli, kuva, jne.). Siksi suositellaan PPMZ pakkaajan käyttöä kun halutaan klusteroida objekteja NCD:lla [CAO05]. Muiden pakkaajien käyttö on suositeltavaa ainoastaan, jos pyritään nopeuteen tai jos tiedostojen kokojen summa on pienempi kuin pakkaajan

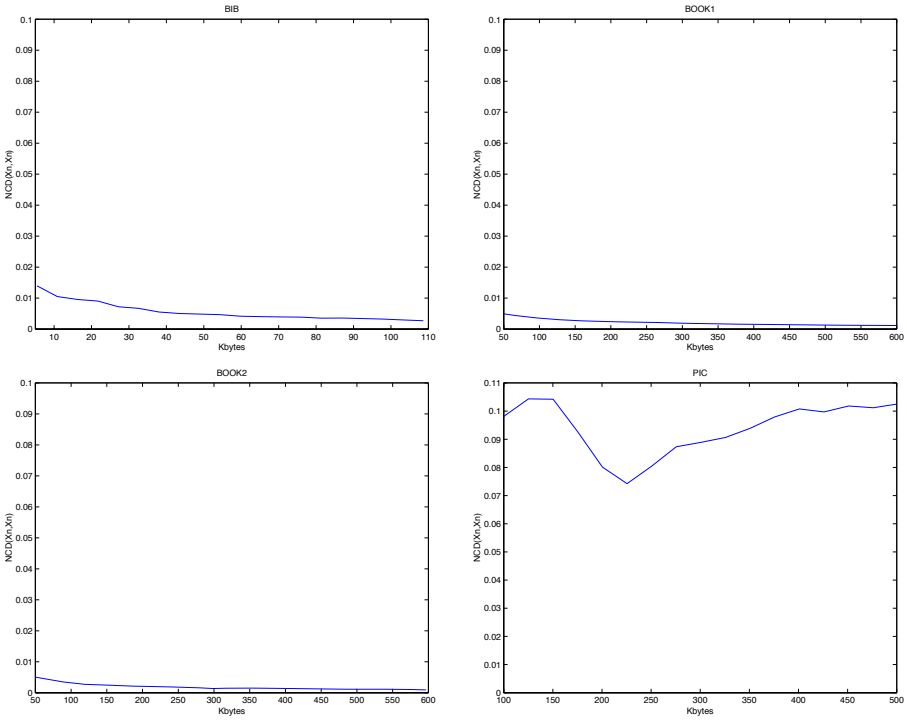


Kuva 7: *Kuva 6, käyttäen **gzip** pakkaajaa asetuksella '–best'. Tiedostot vertailtiin itsensä kanssa.* [CAO05]

käyttämä rajoite (lohkokoko, ikkunakoko).

5 Google-samankaltaisuusetäisyys

Internetin kasvu on houkutellut miljoonia käyttäjiä luomaan miljardeja internetsivuja, jotka ovat sisällöltään ja piirteiltään hyvin vaihtelevia. Suunnaton tiedon määrä miltei mistä tahansa aiheesta tekee mahdolliseksi, että ääripäät kumoutuvat, ja täten suurin osa on merkitsevää heikkolaatuisena approksimaationa. Täten on kehitetty yleinen metodi hyödyntämään tätä matalalaatuista tietoa, jota saa ilmaiseksi Internetistä. Tämä tietovarasto on kaikille avoin käyttäen mitä tahansa hakukonetta, joka pystyy palauttamaan yhteenlasketun sivulukumääräarvion, kuten Google.



Kuva 8: *Kuva 6, käyttäen* ***PPMZ*** *pakkaajaa. Tiedostot vertailtiin itsensä kanssa.* [CAO05]

Google-samankaltaisuusetäisyys (*GSD*) on hyvin vahvasti verrattavissa Normalisoituun pakkausetäisyyteen, sillä kummatkin algoritmit perustuvat samoihin tekniikoihin, kuten Kolmogorov-kompleksisuuteen ja Normalisoituun informaatioetäisyyteen. Eroavaisuuksia esiintyy siinä, mitä nämä kaksi algoritmia käyttävät samankaltaisuuden arvioimiseksi. Siinä missä NCD pakkaa ja vertaa sisältöä toisiinsa, niin GSD vertaa asioiden nimiä esiintymistiheyteen. [CV07]

5.1 Käytännön esimerkki

GSD muodostetaan siten, että haetaan ensiksi yhdellä hakutermillä, sitten toisella ja sen jälkeen kummallakin yhdessä. Hakujen lukumäärät vertaillaan keskenään ja normalisoidaan, tästä saadaan samankaltaisuusetäisyys.

Artikkelissa [CV07] käytettiin hakusanoja ‘horse’ ja ‘rider’ esimerkkinä ja

vuonna 2007 tuloksena oli $NGD(horse, rider) \approx 0.443$. Toistimme kyseisen haun 13.11.2013 ja tuloksena oli $NGD(horse, rider) \approx 0.233$. Arvojen suurta eroa on vaikea selvittää ilman laajempaa tutkimusta, mutta vaikuttavia tekijöitä voi olla epäselvyys tarkasta lukumäärästä Googlen indeksoimista sivuista.

5.2 Hakukoneet samankaltaisuuden mittaamisessa

Googlen indeksoitujen internetsivujen määrä on kasvanut suunnattomaksi ja mikä tahansa yleinen hakutermi esiintyy miljoonilla internetsivulla. Internetin sisällöntuottajiakin on suunnaton määrä, ja voidaan olettaa näiden olevan erittäin kuvaava otanta isosta osasta ihmiskuntaa. Tämän perusteella voidaan argumentoida, että Google-haun frekvenssi on kuvaava approksimaatio todellisista relatiivisista frekvensseistä yhteiskunnassa. Frekvenssi on haun tuottamien osumien lukumäärän suhde kaikkien Googlen indeksoimien sivujen lukumäärään. [CV07]

5.3 Google-jakauma

Olkoon yksittäisten Google-hakujen joukko S. Jatkossa käytämme yksittäisten ja pareittaisten hakujen joukkoja $\{\{x, y\} : x, y \in S\}$. Olkoon joukko Googlen indeksoimia internetsivuja Ω. Joukon Ω koko merkitään $M = |\Omega|$. Oletamme, että jokaisella internetsivulla on yhtä todennäköistä palautua Google-hausta, eli todennäköisyys on $\frac{1}{\Omega}$. Ω:n osajoukkoa kutsutaan *tapahtumaksi*. Jokainen hakuehto x määrittelee yksittäisen haun tapahtuman $\mathbf{x} \subseteq \Omega$, tämä on joukko internetsivuja, joista löytyy x kun suoritetaan Google-haku x:llä. Olkoon $L : \Omega \to [0, 1]$ tasajakauman todennäköisyysfunktio. Tapahtuman $\mathbf{x}$ todennäköisyys on $L(\mathbf{x}) = \frac{|\mathbf{x}|}{M}$. Samoin toimii pareittain haun tapahtuma $\mathbf{x} \cap \mathbf{y} \subseteq \Omega$, jossa haetaan pareittain hakuehdoilla x ja y.

Koska tapahtumat menevät päällekkäin, niin kaikkien tapahtumien todennäköisyys yhteensä on suurempi kuin 1, joten tarvitaan normalisointia. $|S|$ on

yksittäisten ja $\binom{|S|}{2}$ parittaisten hakujen tulosten lukumäärä. Määrittelemme

$$N = \sum_{\{x,y\} \subseteq S} |x \cap y|, \tag{5}$$

yksittäisten ja parittaisten hakujen tulosten lukumäärä yhteensä. $N \geq M$, koska jokainen Googlen indeksoima internetsivu sisältää vähintään yhden hakusanan esiintymän. Toisaalta internetsivuilla ei keskivertona ole enempää kuin α hakutermiä, täten $N \leq \alpha M$. Määrittelemme

$$g(x) = g(x, x),\ g(x, y) = L(\mathbf{x} \cap \mathbf{y}) \frac{M}{N} = \frac{|\mathbf{x} \cap \mathbf{y}|}{N}. \tag{6}$$

Ja siten looginen seuraus on $\sum_{\{x,y\} \subseteq S} g(x, y) = 1$, eli g määrittelee todennäköisyysjakauman.

5.4 Google-samankaltaisuusetäisyys

Normalisoitu Google-etäisyys (NGD) (engl. Normalized Google Distance) voidaan määritellä seuraavasti,

$$NGD(x, y) = \frac{max\{\log f(x), \log f(y)\} - \log f(x, y)}{\log N - min\{\log f(x), \log f(y)\}}, \tag{7}$$

jossa $f(x)$ merkitsee internetsivujen lukumäärää, jotka sisältävät hakutermin x, ja $f(x, y)$ merkitsee samaa, mutta sivuille jotka sisältävät termit x ja y. Tämä normalisoitu Google-etäisyys on approksimaatio normalisoidusta informaatioetäisyydestä. Käytännössä Googlen palauttamia sivujen lukumääriä käytetään frekvensseinä ja valitaan N. On huomattu, että mikä tahansa järkevä arvo voidaan käyttää normalisoinnin arvona N. Olettaen, että valitaan parametri $N \geq M$, NGD:llä on seuraavat ominaisuudet:

1. NGD:n arvo on väliltä 0 ja ∞ (mahdollisesti negatiivinen, jos Googlen lukumäärät ovat epäluotettavia, ja täten $f(x, y) > max\{f(x), f(y)\}$).

(a) Jos $x = y$ tai $x \neq y$, mutta frekvenssi on

$$f(x) = f(y) = f(x, y) > 0,$$

niin $NGD(x, y) = 0$. Eli x ja y ovat Googlen kannalta samat.

(b) Jos frekvenssi $f(x) = 0$, jokaiselle hakutermille y pätee $f(x, y) = 0$. Tämän lisäksi pätee

$$NGD(x, y) = \frac{\infty}{\infty},$$

jonka määrittelemme tarkoittavan 1.

2. NGD on aina epänegatiivinen ja $NGD(x, x) = 0$ kaikille x. Kaikille x, y pätee $NGD(x, y) = NGD(y, x)$, toisin sanoen se on symmetrinen. NGD ei kuitenkaan ole metriikka, koska se ei täytä ehtoa $NGD(x, y) > 0$ kaikille $x \neq y$, eikä se myöskään toteuta metriikan kolmioepäyhtälöominaisuutta $NGD(x, z) \leq NGD(x, y) + NGD(y, z)$. [CV07]

3. NGD on *invariantti skaalautuvuudessa* seuraavassa mielessä: oletamme, että kun Googlen indeksoimien sivujen lukumäärä N kasvaa, niin sivujen, jotka sisältävät annetun hakusanan, lukumäärä lähestyy määrättyä murto-osaa N:stä, ja niin lähestyy myös sivujen lukumäärä, jotka sisältävät hakusanojen parin.

5.5 Hierarkkinen klusterointi

Klusterointi normalisoidun Google-etäisyyden avulla onnistuu samalla tavalla kuin normalisoidulla pakkausetäisyydelläkin. Ensiksi koostetaan etäisyysmatriisi hakuehtojen pareittaisista NGD:n arvoista ja sitten nelikkomenetelmää käyttäen rakennetaan *juureton ternääripuu (engl. unrooted ternary tree).*

Esimerkkinä klusteroitiin 15 1600-luvun hollantilaisten maalareiden teosta. Kuvassa 9 on Rembrandtin, Steenin ja Bolin maalauksia ternääripuurakenteessa. Hakusanoina käytettiin ainoastaan maalausten nimiä. Klusteroinnissa käytettiin seuraavia teoksista:

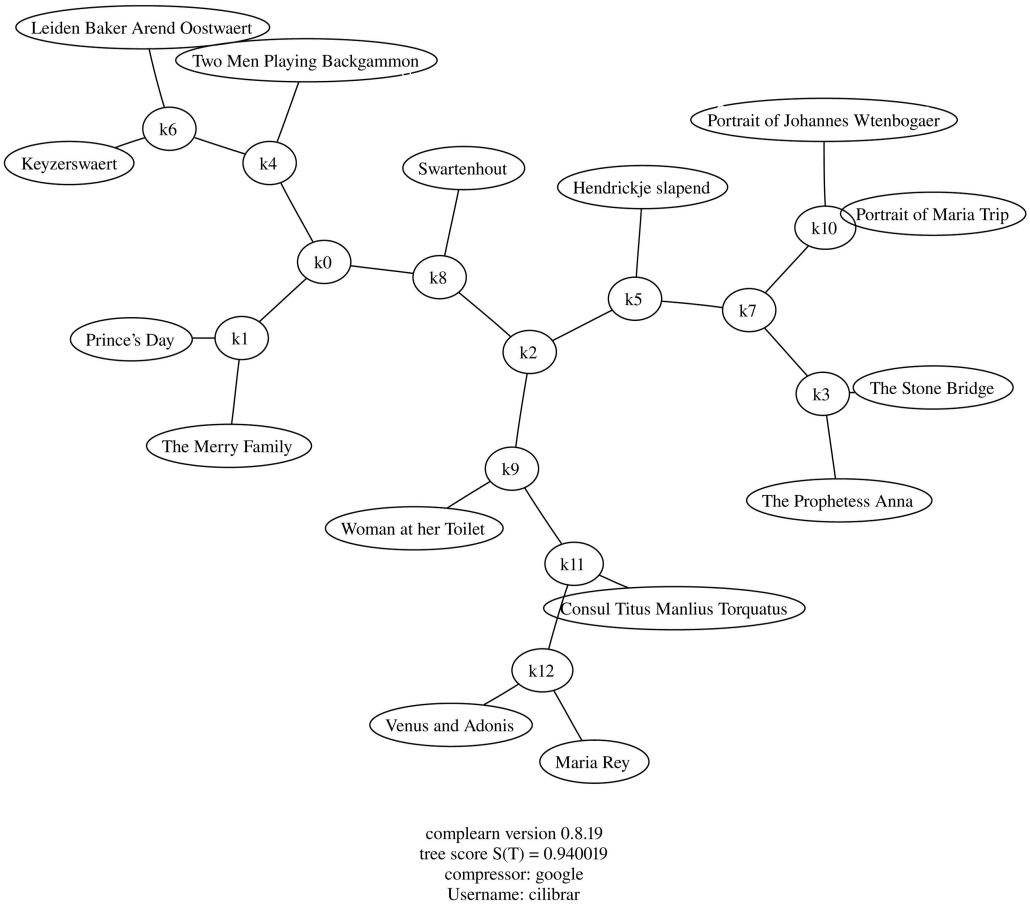
- **Rembrandt van Rijn:** *Hendrickje slapend, Portrait of Maria Trip, Portrait of Johannes Wtenbogaert, The Stone Bridge, and The Prophetess Anna.*
- **Jan Steen:** *Leiden Baker Arend Oostwaert, Keyzerswaert, Two Men Playing Backgammon, Woman at her Toilet, Prince's Day, and The Merry Family.*
- **Ferdinand Bol:** *Maria Rey, Consul Titus Manlius Torquatus, Swartenhout, and Venus and Adonis.*

Kuvassa 9 näemme, että suurin osa maalauksista on ryhmittynyt puun haaroihin taiteilijan mukaan. Rembrandtin maalaukset ovat oikealla, Steenin maalaukset vasemmalla ja Bolin maalaukset alhaalla. Muutama maalaus on hieman erillään taiteilijan ryhmittymästä ja nämä ovat Rembrandtin 'Hendrickje slapend', Steenin 'Woman at her Toilet' ja Bolin 'Swartenhout', joista viimeiset kaksi on luokiteltu väärän taiteilijan haaraan.

GSD on selkeästi [CV07] tehokas tapa klusteroida objekteja, joskin syntyvät klusterit perustuvat mielivaltaisesti internetin sisältöön eivätkä objektien ominaisuuksiin.

6 Yhteenveto

Luvussa 3 näytettiin, että NCD toimii hyvin klusterointiin ja sitä on helppo soveltaa mihin tahansa alaan kuten: musiikkiin, kirjallisuuteen ja genomeihin. Tämä johtuu siitä, että algoritmin käyttö on täysin riippumaton datasta ja sen ominaisuuksista. Algoritmi kykenee aina löytämään merkittävimmät ominaisuudet datasta ja vertailemaan näin samankaltaisuutta datan eri objektien välillä. NCD:n tehokkuutta parantaa huomattavasti, että algoritmi vaikuttaa olevan hyvin kohinansietokykyinen, eli se kykenee edelleen yhdistämään samankaltaiset objektit vaikka toisessa olisikin kohinaa. Vaikka NCD:n kehittäjät luulivat osoittaneensa, että algoritmi on riippumaton pakkaajan valinnasta [CV05], niin vertailujen tuloksena on pystytty todistamaan, että



Kuva 9: *Hierarkkinen klusterointi hollantilaisista maalauksista* [CV07]

ainoastaan tiedostonkokoa rajoittamaton pakkaaja toimii aidosti NCD:n kanssa kuten luvussa 4.2 todettiin. Kannattavin valinta pakkaajaksi on *PPMZ*, mikäli käyttää NCD:n tuloksia klusterointiin.

NCD ei tietenkään ole ainut menetelmä, jota valvomattomassa oppimisessa voi käyttää datan klusteroimiseen. Hierarkkinen klusterointi on tapa, jota NCD:n kanssa klusteroidessa käytetään, eli toisiaan lähemmät objektit liittyvät enemmän toisiinsa kuin kauempana olevat. Vaihtoehtoisesti voidaan käyttää K-keskiö klusterointia *(engl. K-means clustering)*, jossa pyritään klusteroimaan objektit k eri klusteriin klusterin keskipisteen etäisyyden perusteella, tai jakaumaan pohjautuvaa klusterointia, jossa klusteroidaan objektit,

jotka todennäköisesti kuuluvat samaan jakaumaan.

Normalisoidun pakkausetäisyyden aihealueesta on tehty paljon tutkimusta eri suuntautumishaaroihin ja seuraavaksi esitellään muutama alue, joita tämän tutkielman pohjalta voi lukea. Geeniekspressioanalyysi on tärkeä työkalu syövän tutkimisessa ja on tutkittu voidaanko NCD:tä hyödyntää siinä [NYHS05]. Tarkempaa tietoa normalisoidun informaatioetäisyyden ominaisuuksista ja sen epäapproksimoitavuudesta löytyy artikkelista [TTV11]. Tutkimusta on myös tehty siitä, miten NCD toimii kuvantunnistuksessa [Tra07]. NCD:n kohinansietokykyä on tutkittu, ja se on avannut tutkimuksen mahdollisuuksia tälle aiheelle. Tämän lisäksi on myös tutkittu miten NCD:tä voidaan hyödyntää kohinan poistossa [Vit13]. Nelikkomenetelmästä löytyy tarkempaa kuvausta artikkelissa [CV11].

Lähteet

[CAO05] Cebrian, M., Alfonseca, M. ja Ortega, A.: *Common pitfalls using the normalized compression distance: What to watch out for in a compressor*. Communications in Information & Systems, 5(4):367–384, 2005.

[CAO07] Cebrian, M., Alfonseca, M. ja Ortega, A.: *The Normalized Compression Distance Is Resistant to Noise*. Information Theory, IEEE Transactions on, 53(5):1895–1900, 2007.

[Cil] Cilibrasi, R.: *CompLearn Toolkit*. URL: `http://complearn.org/`. Luettu 26. marraskuuta 2013.

[CV05] Cilibrasi, R. ja Vitányi, P.: *Clustering by Compression*. IEEE Transactions on Information Theory, 51(4):1523–1545, Huhtikuu 2005.

[CV07] Cilibrasi, R. ja Vitányi, P.: *The Google Similarity Distance*. Knowledge and Data Engineering, IEEE Transactions on, 19(3):370–383, 2007.

[CV11] Cilibrasi, R ja Vitányi, P: *A fast quartet tree heuristic for hierarchical clustering*. Pattern recognition, 44(3):662–677, 2011.

[CVDW04] Cilibrasi, R, Vitányi, P ja De Wolf, R: *Algorithmic clustering of music based on string compression*. Computer Music Journal, 28(4):49–67, 2004.

[NYHS05] Nykter, M., Yli-Harja, O. ja Shmulevich, I.: *Normalized compression distance for gene expression analysis*. Teoksessa *Workshop on Genomic Signal Processing and Statistics (GENSIPS)(May 2005)*. Citeseer, 2005.

[Tra07] Tran, N.: *The Normalized Compression Distance and Image Distinguishability*, 2007. `http://dx.doi.org/10.1117/12.704334`.

[TTV11] Terwijn, S., Torenvliet, L. ja Vitányi, P.: *Nonapproximability of the normalized information distance*. Journal of Computer and System Sciences, 77(4):738–742, 2011.

[Vit13] Vitányi, P.: *Similarity and Denoising*. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 371(1984), 2013.

[WBC] Witten, I., Bell, T. ja Cleary, J.: *Calgary Corpus*. URL: `http://corpus.canterbury.ac.nz/descriptions/#calgary`. Luettu 26. marraskuuta 2013.