Satunnaisuuden louhinta äänitiedosta

Aki Loponen

Tampereen yliopisto
Tietojenkäsittelytieteiden laitos

Tietojenkäsittelyoppi

Pro gradu -tutkielma

Marraskuu 2006

Tampereen yliopisto
Tietojenkäsittelytieteiden laitos
Aki Loponen: Satunnaisuuden louhinta äänitiedosta
Pro gradu -tutkielma, 94 sivua.
Marraskuu 2006

Aitojen satunnaislukujen luominen pelkästään tietokoneohjelmistoilla ei ole mahdollista. Deterministiset operaatiot eivät ole satunnaisia, vaan perustuvat aina johonkin kaavaan ja tila-automaattiin. Mikäli kaava (tai automaattia) ei tiedetä, on ohjelmiston näennäisesti satunnaisella tuotoksella kryptografinen vahvuus ollakseen satunnaisluku. Yleensä tila-automaatti kuitenkin on yleisesti tunnettu tai se ainakin saatetaan saada selville, jolloin kryptografisen turvallisuuden kannalta oleellisen asemaan nousee automaatin aloitustila. Aloitustilaa kutsutaan siemenluvuksi, jonka puolestaan on oltava aidosti satunnainen, jotta koko satunnaislukujärjestelmä tuottaisi satunnaisia lukuja.

Tässä tutkielmassa perehdytään siemenlukujen merkitykseen ja saatavuuteen tietokoneohjelmistojen ulkopuolelta. Tutkielmassa esitellään muutamia menetelmiä, joilla ilmiöiden tuottamaa satunnaisuutta voidaan havaita ja louhia. Analogisten ilmiöiden digitalisoinnista ja ilmiöiden luonteista johtuen ilmiöstä taltioiduissa näytteissä usein on heikommin satunnaisia häiriöitä, joiden poistamiseksi on kehitetty louhintamenetelmiä.

Lisäksi kuvataan kokeita, joissa eri lähteistä äänitetyistä ääninäytteistä louhitaan satunnaisuutta muutamalla menetelmällä. Kokeiden tarkoituksena on selvittää lähteiden ja louhijoiden vahvuutta. Satunnaisuuden mittaaminen on ongelmallista, joten käytössä on useita eri analyysimenetelmiä. Hieman paranneltu bittilouhinta radiokohinalle tuntuu tuottavan laadullisesti ja määrällisesti parhaiten satunnaisuutta.

Avainsanat ja -sanonnat: satunnaisuus, satunnaislukugeneraattorit, siemenluku, äänitieto.
CR-luokat: G.3, E.0

Sisällys

1. Johdanto

Satunnaisuus on abstrakti käsite, jonka projektiosta reaalimaailmaan on oletettavasti mahdoton löytää todisteita. Satunnaiselta vaikuttavasta ilmiöstä ei voida satunnaisuutta täydellä varmuudella ja tarkkuudella mitata. On tehtävä oletus riittävän satunnaisesta ilmiöstä, joka saa toimia täydellisen satunnaisuuden projisiona reaalimaailmaan. Tällainen riittävän satunnainen ilmiö näyttää läpäisevän äärellisen määrän testejä ja näyttäisi täten olevan reaalimaailman kannalta paras mahdollinen. Ehdottomasti täydellinen satunnainen ilmiö läpäisee äärettömän määrän testejä. On tärkeää pitää mielessä, että vaikka satunnaisuus voidaan käsitteenä määrittää hyvin tarkkaan ja vaikka sitä voidaan mitata, ei *satunnaislukugeneraattorin* (random number generator, RNG) antamaa tulostetta voida todistaa satunnaiseksi [Chaitin, 1975].

Käytännössä satunnaislukugeneraattorit eivät luo satunnaisuutta tyhjästä, vaan ne perustuvat *siemenluvun* (seed) muuttamiseen näennäissatunnaiseksi (pseudo-random) esitykseksi tiettyjen funktioiden ja loogisten operaatioiden avulla. Tällöin siemenluvun vahvuus voi tulla itse generaattoria oleellisemmaksi osaksi järjestelmää. [Eastlake *et al.*, 1994]

Koska aitoa satunnaisuutta voi olla hyvinkin hidasta louhia, eikä lähteen satunnaisuuden aitoudesta voi olla varma, käytetään *näennäissatunnaislukugeneraattoreita* (pseudo-random number generators, PRNG), jotka venyttävät saamastaan syötteestä laajemman näennäissatunnaisen sarjan, joka siis vaikuttaisi käyttäytyvän satunnaisesti. PRNG:n ominaisuuksiin keskitytään huomattavasti useammin kuin satunnaisen syötteen ongelmaan, joka kirjallisuudessa usein sivuutetaan olettamalla sen tulevan *yleisestä satunnaisuuden lähteestä* (uniform random source). [Juels *et al.*, 1998]

Kun satunnaisuutta hyödynnetään varsinkin kryptografiassa tietokoneilla, on varmistettava, että satunnaisuuden lähteistä saadut hahmot kyetään muuntamaan joko bitti- tai tavusarjaksi. Lähde on usein niin sanottu luonnollinen ilmiö, jota mita-

taan ja arvotetaan asiaankuuluvilla tavoilla. Kolikonheitto on selkeä: kruuna voidaan määritellä 0-bitiksi ja klaava 1-bitiksi. Ongelmallisempaa on digitalisoida analogisia signaaleja siten, ettei muunnos hukkaa tai muunna lähteen satunnaisia ominaisuuksia.

Käytännössä kolikollakaan ei saada tuotettua täydellisen *reilua* (fair) satunnaisuutta, sillä kolikon täydellisyys on fysikaalinen abstraktio, jota on liki mahdoton todistaa. Lisäksi heittotilanteeseen voi liittyä tulokseen vaikuttavia tekijöitä. Vaikka löytyisi lähde, joka vaikuttaisi tuottavan vähintäänkin riittävän vahvaa satunnaisuutta, voi satunnaisuuden laatua varmistaa algoritmisin menetelmin.

Tässä tutkielmassa tutkitaan sekä satunnaisuuden lähteitä että erilaisia louhintamenetelmiä, joilla lähteiden tuottamia merkkijonoja pyritään muuttamaan satunnaisemmiksi. Lähteinä käytetään kolmea eri äänilähdettä, jotka on valittu osin intuitiivisesti ja osin lähdeaineiston antamien viitteiden perusteella. Louhintamenetelmät valitaan myös lähdeaineiston perusteella siten, että ne ovat oletettavasti tehokkaita ja kohtuullisen helposti toteutettavia.

Luvussa 2 esitellään tarkemmin satunnaisuutta, satunnaislukugeneraattoreiden toimintaa ja mahdollisia satunnaisuuden lähteitä. Luvussa perehdytään lisäksi erilaisiin menetelmiin, joilla satunnaisuutta voidaan saada lähteistä esiin. Luvussa 3 perehdytään tarkemmin äänitiedon ominaisuuksiin ja soveltuvuuteen satunnaisuuden lähteenä. Luku 4 esittää tutkimuksen koeasetelmat, selvittää käytetyt menetelmät ja kuvailee keinot, joilla kokeiden tuotoksista saadaan tehtyä analyysit. Luku 5 esittelee tulokset ja luku 6 johtopäätelmät.

2. Satunnaisuus

Satunnaisuus on käsitteenä hieman epämääräinen, ja sillä voidaan tarkoittaa samassakin yhteydessä hieman eri asioita. Seuraavassa käsitellään, mitä satunnaisuudella voidaan tarkoittaa ja määritellään tarkemmin, mitä satunnaisuudella tarkoitetaan tässä tutkielmassa.

2.1. Merkkijonon satunnaisuus

Merkkijono on satunnainen, mikäli se on valittu mielivaltaisesti kaikkien mahdollisten merkkijonojen keskuudesta – eli *merkkijonoavaruudesta* – ja mikäli kaikkien merkkijonoavaruuden alkioiden valituksi tulemisen todennäköisyys on yhtä suuri. Merkkijonoavaruuteen kuuluva mielivaltainen merkkijono voi siis olla ominaisuuksiltaan minkälainen tahansa. Satunnainen merkkijono voidaan myös määritellä merkkijonoksi, joka ei sisällä mitään toistuvaa *mallia* (pattern), eli osamerkkijonoa. Tällöin vahvasti satunnaista merkkijonoa ei voida kuvata millään lyhyemmällä jonolla, ts. merkkijonoa ei voida pakata pienemmäksi [Chaitin, 1975]. Tämän kaltaista jonoa ei puolestaan voida sanoa mielivaltaiseksi suhteessa koko merkkijonoavaruuteen. Satunnaisuus voidaan siis määritellä monesta eri lähtökohdasta tarpeen, taustan ja merkityksen mukaan.

Tarkasteltava satunnainen ilmiö – tai ilmiön ilmentymä – voi toki olla muukin kuin merkkijono. Merkkijonolla tarkoitetaan teoreettisessa tarkastelussa enemmänkin yleisempää sarjan ideaalia, *hahmoa*, joka voi konkreettisuudessaan olla mikä tahansa ilmiön ilmentymä. Käsitteenä merkkijono on konkreettisempi kuin *hahmo*, ja tämän kouriintuntuvuuden takia merkkijonoa tässä tutkielmassa käytetään. Merkkijonoavaruudella tarkoitetaan kaikkien mahdollisten merkkijonojen joukkoa, jossa jokainen alkio on uniikki, eli duplikaatteja ei ole. *Merkkijonojoukko* on merkkijonoavaruuden osajoukko. *Merkkisarja* – tai lyhyemmin tässä yhteydessä *sarja* – on sama kuin merkkijono. Louhintamenetelmiä ja äänilähteitä myöhemmin tutkailtaessa merkkijonolla tarkoitetaan konkreettista tavuarvoista muodostettujen merkkien jonoa.

Klassisen todennäköisyystieteen näkökulmasta mikä tahansa merkkijonojoukon alkio on satunnainen, mikäli jokainen joukon merkkijono ilmenee yhtä suurella todennäköisyydellä [Ritter, 1991]. Tällöin bittisarjat 0000000000 ja 0100100100 ovat yhtä satunnaisia, jos kaikki kymmenen bitin yhdistelmät esiintyvät samalla todennäköisyydellä. Tämänkaltainen satunnaisuus on helposti varmennettavissa merkkijonojen (bittisarjojen) esiintymien tarkastamisella. Mikäli on tarve saada mielivaltainen merkkijono johonkin valintatilanteeseen, klassisen satunnaisuuden tuloksissaan toteuttava menetelmä on ehdottomasti vahvin. Merkkijonon uudelleen selvittämiseksi on joko arvattava yhtä suurella todennäköisyydellä sama jono tai käytävä kaikki mahdolliset jonot läpi.

Pelkkä todennäköisyyksien yhtenevyys ei ole riittävä tae turvalliselle satunnaisuudelle, varsinkaan kryptografisten sovellusten kannalta. Vaikka käytetty jono olisikin täysin mielivaltaisesti valittu äärettömän laajasta merkkijonoavaruudesta, se voi olla kryptografisesti sovellettuna heikko ja selvitettävissä kryptojärjestelmän muiden osien ja tuotosten kautta. Tasaisesti todennäköisyyden mukaan jakautuneesta merkkijonoavaruudesta mielivaltaisesti valitun jonon rakenteen tulisi myös täyttää satunnaisuuden kriteerit. Mielivaltainen valinta koko merkkijonoavaruudesta sinällään on heikko menetelmä silloin, kun on oleellista löytää mielivaltainen ennustuksellisuutta sisältämätön jono *järjestyksellisen* (orderly) sijaan [Chaitin, 1975]. Lisäksi moista menetelmää on käytännössä vaikea toteuttaa, sillä jo mielivaltaisen merkkijonon valinta vaatii aitoa satunnaisuutta.

Michael Sipser [1983] esittää kompleksisuusteorian olevan luonnollinen ympäristö satunnaisuuden tutkimiseen. Kompleksisuusteorian näkökulmasta satunnainen merkkijono voidaan määritellä jonoksi, joka ei sisällä mitään toistuvaa mallia (pattern) ja on täten minimaalinen [Chaitin, 1975]. Merkkijonon S ei siis voida katsoa olevan satunnainen, mikäli on olemassa S:n lyhyemmin kuvaava merkkijono R. Jos merkkijono S on satunnainen, myös S:n osamerkkijonot S_1,…, S_k,…, S_n ovat satunnaisia. Jos näin ei olisi, täytyisi osamerkkijono S_k voida esittää lyhyemmin jollakin muulla jonolla. Tällöin mikäli S_k kuuluu merkkijonoon S, voidaan S esittää lyhyemmässä muodossa, jolloin se ei ole satunnainen. Satunnaisen jonon kompleksisuus on liki yhtäsuuri kuin jonon koko, mutta rajanveto tilanteen vaati-

man riittävän kompleksisuuden tarkaksi määrittämiseksi ei ole ongelmatonta [Chaitin, 1975].

Kolmogorovin kompleksisuus äärelliselle bittijonolle, jota kuvaa myös käsite *entropia* (entropy), on lyhin ohjelma, joka kykenee kyseisen bittijonon kuvaamaan. Entropia kuvaa bittijonon sisältämän informaation määrää [Sipser, 1983; Schneier, 1996, p. 233]. Numerosarjan kompleksisuus on se määrä bittejä, joka laskukoneeseen täytyy syöttää, jotta se tulostaisi syötetyn numerosarjan [Chaitin, 1975]. Entropia on *suurta* tai *vahvaa*, jos sarja ei ole esitettävissä millään yksinkertaisemmalla sarjalla. Jos sarja sisältää toistuvia malleja, se on pakattavissa ja sen entropian katsotaan olevan *heikko*. Sarja on satunnainen, jos sitä ei voida tiivistää ilman informaation menettämistä [Sipser, 1983]. Entropia ei kuitenkaan itsessään muodosta mitään binääristä luokittelua sarjojen todennäköisyyksien välille klassisen todennäköisyyden tavoin, vaan entropia kuuluu vahvasti kompleksisuusteoreettiseen näkökulmaan. Entropia kuvaa informaatioteoreettisesti sarjan sisältämää informaatiota [Schneier, 1994, p. 233]. Satunnaisuustulkinta syntyy tavasta tulkita informaatioteoreettista analysointia.

Numerosarja katsotaan satunnaiseksi, jos pienimmässä sen tietokoneelle kuvaavassa algoritmissa on (liki) sama määrä *tietobittejä* (bits of information) kuin numerosarjassa itsessään. *Occamin partaveitsi* (Occam's razor) siis pätee tässä yhteydessä: "Annetuista yhtä pätevistä teorioista tulee suosia yksinkertaisinta." [Chaitin, 1975].

2.2. Satunnaislukugeneraattori

Satunnaislukugeneraattori on matemaattinen algoritmi tai fyysinen laite, joka on suunniteltu luomaan lukusarjoja, jotka täyttävät satunnaisuuden vaatimukset. Satunnaislukugeneraattorilla voidaan tarkoittaa koko suljettua satunnaislukuja tuottavaa järjestelmää tai kyseisen järjestelmän algoritmista osaa. Algoritminen osa saa syötteekseen siemenen, jonka se laajentaa numeerisilla tai loogisilla operaatioilla satunnaisluvuksi.

Yleisesti satunnaislukugeneraattorin oletetaan tuottavan samalla siemenellä saman tulossarjan, jotta sovellusten kannalta hyödyllinen toistettavuusominaisuus olisi hyödynnettävissä. Tällöin olisi oikeastaan turvallisempaa puhua näennäissatunnaislukugeneraattorista, mutta käytäntö jälleen hämärtää asioita. Aidon ja kryptologisesti absoluuttisen vahvan satunnaisuuslukugeneraattorin toistettavuusominaisuutta (tai sen puutetta) ei voida todistaa, sillä tällöin siemenluku on tuntematon (eikä täten varmuudella toistettavissa).

Satunnaislukugeneraattori ei itsessään ota kantaa siihen, mistä siemenluku tulee, mutta olettaa siemenen olevan satunnaisuudeltaan riittävän vahva. Deterministinen tietokone ei kykene mielivaltaiseen valintaan, joten siemen on otettava jostakin riittävän satunnaiseksi havaitusta ja riittävän pitävästi epädeterministiseksi oletetusta lähteestä, mieluiten tietokoneen ulkopuolelta. Satunnaislukugeneraattoria kehitettäessä ja analysoitaessa lähteen oletetaan tuottavan täydellisen satunnaisia arvoja, joten sitäkin kautta vaatimukset satunnaisuuden lähteen ominaisuuksille on asetettu melkoisen tiukoiksi. Siemenlukua voisi verrata malmiin, josta satunnaislukugeneraattori rikastaa satunnaisuuden esiin.

Satunnaislukugeneraattorin tuotosten katsotaan hyvin usein olevan satunnaisia, jos tuotettujen lukusarjojen entropia on vahva. Tämä ei valitettavasti riitä, mikäli lukujen halutaan olevan myös kryptologisesti vahvoja. Ennustettavuus pilaa miten hyvän kompleksisen satunnaisuuden tahansa. Satunnaislukugeneraattorin täytyy kyetä luomaan sellaisia kompleksisesti satunnaisia lukuja, ettei jo luotujen lukujen perusteella pystytä mitenkään päättelemään seuraavaa luotavaa lukua.

Satunnaislukugeneraattorilla voi olla tietokoneella toteutettuna vain äärellinen määrä tiloja. Tulossarja on vain deterministinen tulos syötteestä ja generaattorin lähtötilasta, joten jokainen tietokoneelle (tai ylipäätään rajallisen tilamäärän omaavalle koneelle) toteutettu satunnaislukugeneraattori on *jaksollinen* (periodic), ja jaksollisuus johtaa käytännössä ennustettavuuteen. Rajallisen tilojen määrän vuoksi vahva satunnaislukugeneraattori tarvitsee siemenlukunsa tietokoneen ulkopuolelta lähteestä, jolla voidaan tilamäärien olettaa ja todeta olevan mahdollisimman lähellä rajoittamatonta. Riittävän vahva näennäissatunnaislukugeneraatto-

ri toki riittää silloin, kun tulossarjojen yhteinen pituus ei ylitä satunnaislukugeneraattorin jaksoa. [Schneier, 1994, p. 45].

Schneier [1996, pp. 45-46] kuvaa tietokoneelle toteutetuille satunnaislukugeneraattoreille kolme tasoa, joiden mukaisesti generaattorin käyttökohteeseen nähden riittävä turvallisuus ja vahvuus voidaan varmentaa. Kaikki kolme tasoa tietylle sovellukselle saavuttava satunnaislukugeneraattori on lähes täydellinen tarjotakseen sovelluksen ympäristön puitteissa aitoa satunnaisuutta. Käytännölliseltä kannalta kahden ensimmäisen tason saavuttaminen luetaan jo kyllin hyväksi tulokseksi satunnaislukugeneraattorille, sillä kolmannen tason vaatimuksien toteuttaminen ei ole enää käytännöllistä.

Satunnaislukugeneraattori on näennäisesti satunnainen, jos se näyttää satunnaiselta eli läpäisee kaikki olemassa olevat tilastolliset testit. Tämä ei kuitenkaan riitä turvallisiin kryptografisiin sovelluksiin, joita varten täytyy siirtyä seuraavalle tasolle. Koska tilastollinen satunnaisuus on vain osa kryptografista satunnaisuutta, täytyy kryptografisesti turvallisen näennäissatunnaisuuden olla ennalta arvaamatonta, eli ennustamatonta. Vaikka edelliset luodut bitit, algoritmi ja siemenluku (tai siemenluvut) olisivat tiedossa, täytyy olla laskennallisesti mahdotonta päätellä seuraavaa bittiä. Satunnaislukugeneraattori on aidosti satunnainen (ja täten kolmannella tasolla), mikäli sen tuotoksia ei kyetä luotettavasti tuottamaan uudelleen. Jos aitoa satunnaislukugeneraattoria ajetaan kahdesti samoilla syötteillä, tuotoksena tulee olla kaksi toisistaan riippumatonta sarjaa. Tästä luonnollisesti seuraa, että aito satunnaislukugeneraattori ei ole väistämättä ihanteellisin ratkaisu jokaiseen sovellukseen, vaan pitää tyytyä vahvaan toisen asteen generaattoriin, joka tuottaa saman tuloksen samalla syötteellä.

Ideaalin kolikon tapaan aidolla satunnaislukugeneraattorilla ei ole muistia [Walker, 1998]. Vaikka kolikko olisi tuottanut kymmenellä perättäisellä heitolla saman tuloksen, on seuraavan heiton tuloksen todennäköisyys aivan sama kuin ensimmäisenkin.

2.3. Satunnaisuuden lähteet

Jotkin luonnonilmiöt saattavat tuottaa tilastollisesti muuttumattomia tulostesarjoja, eli sarjoja, jotka eivät riipu edeltävistä sarjoista. Vaikka lähde olisi (tilastollisesti) muuttumaton tulostesarjojen luonteen suhteen, tulostesarjat ovat silti yleensä painottuneita. Voidaan ajatella, että satunnaisuus on mineraalia lähteen tuottamien painottuneiden tulostesarjojen muodostaman kallion sisällä. Tästä kalliosta satunnaisuusmalmi on louhittava ja malmi rikastettava puhtaaksi satunnaisuudeksi. [Juels *et al.*, 2000].

Luonnonilmiöt ovat kahtiajakoinen satunnaisuuden lähde. Koska luonnonilmiöön vaikuttavien muiden ilmiöiden (luonnonilmiöitä oletettavasti nekin) luonnetta ei myöskään tunneta, voidaan luonnonilmiötä tavallaan pitää mielivaltaisena. Toisaalta ilmiön sykliperiodia ei tunneta, joten ennustamattomuudesta ei voida olla riittävän varmoja. Epävarmuuden täyttämästä luonnonilmiöiden avaruudesta on vain löydettävä ne ilmiöt, jotka vaikuttavat sisältävän riittävää satunnaisuutta ja joista satunnaisuus on kustannuksiin nähden kyllin tehokkaasti irrotettavissa.

Liki kaikki louhintamenetelmät perustuvat säännöllisesti ja satunnaisesti toistuvien reaalimaailman ilmiöiden esiintymien etäisyyksiin tai eroihin. Voidaan esimerkiksi mitata kulunut aika tapahtumasta seuraavaan tapahtumaan, merkitä tulos ylös ja verrata sitä seuraavien kahden tapahtuman välissä kuluneeseen aikaan. Näin saadaan tuotettua yksi satunnainen bitti, jos kyseinen reaalimaailman ilmiö vain voidaan todeta riittävän varmasti tarpeeksi satunnaiseksi. [Schneier, 1996, p. 423].

Toisaalta louhinta voi myös perustua mahdolliseen tietoon tai hyvään arvioon lähteen suoraan tuottamasta satunnaisuudesta. Ihanteellisimmillaan mitään varsinaista louhintaa ei tarvita, vaan tulokset ovat käyttökelpoisia sellaisenaan. Käytännössä tulossarjaa joudutaan erinäisin keinoin louhimaan arvojakauman tasoittamiseksi. Ainoa ehto liittyen lähteen tuottaman satunnaisuuden muotoon on, että soveltava ohjelma tuntee muodon: rajapinta ja oletus satunnaisuuden vahvuudesta ovat tärkeimpiä seikkoja sovellukselle. Satunnaisuuden lähteen olemus ei ole sel-

keän yksijakoinen, sillä lähteestä voidaan erottaa satunnaislukugeneraattorin piirteitä. Luonnonilmiö saattaa itse asiassa laajentaa jostain muusta ilmiöstä saamaansa siementä omaksi tulosteekseen ja siksi näyttää satunnaislukugeneraattorilta itsessään. Täydellistä satunnaisuutta voi olla mahdoton mitenkään todistaa, joten on tyytyminen riittävän satunnaiselta näyttävien ilmiöiden tulkintaan. Mikäli aletaan pohtia, mistä pohjimmiltaan johtuu heitetyn kolikon putoaminen satunnaisesti, astutaan epävarmuuden ohella metafyysiseen suohon. Täytyisi tietää tarkalleen, miten maailmankaikkeus toimii ja miten kaikki maailmankaikkeudessa vaikuttaa toisiinsa, jotta voitaisiin olla täysin varmoja ilmiöiden todellisista luonteista. Louhijan tehtävä on tuoda näkyville ilmiön sisältämä satunnaisuus; louhijan ei itsessään tule pyrkiä luomaan satunnaisuutta.

Satunnaisuuden lähteen tulossarjojen painottuneisuuden lisäksi päänvaivaa aiheuttaa luonnonilmiöiden näytteistäminen digitaalisiksi signaaleiksi tietokoneelle. Mikäli menetelmä perustuu lähdeilmiön tapahtumien arvojen pisteyttämiseen, tuottaa digitaalisuus ongelmia luokittelurajan määrittelyssä: missä kohtaa tarkalleen tulee vielä nollabitti, ja missä on jo ykkösbitti. Tarkkaa rajaa ei näytteistämällä voida koskaan saavuttaa, vaan rajaa voidaan vain lähestyä äärettömän lähelle. Pyöristäminen on aina askel absoluuttisuudesta arvioon. Lähdeilmiön peräkkäisten (tai ainakin eri) tapahtumien arvojen vertaaminen on (ainakin näennäisesti) astetta helpompaa, mutta pyöristyksiä tulee myös siinä esiintymään. Vertaaminen suoritetaan yleensä digitaalisella laitteistolla, jolloin on tyydyttävä heikompaan tarkkuuteen.

Yksinkertainen ja kätevä keino havainnollistaa satunnaisen bittijonon luomista reaalimaailmasta on heittää kolikkoa ja kirjata tulokset ylös. Oletusarvoisesti puolet heitoista putoaa kruuniksi ja puolet klaavoiksi. Ideaalinen painottamaton kolikko ei kuitenkaan välttämättä mallinna todellista maailmaa täydellisesti, sillä todellinen maailma on harvoin reilu, ja todellinen kolikko ei ole täydellisen symmetrinen kruuna- ja klaavapuoliltaan - puhumattakaan kolikon heittämisen fysiikasta. Jotta voitaisiin saada vahvempaa satunnaisuutta kolikoilla, täytyisi heittotuloksiin soveltaa sopivaa louhintamenetelmää. Kolikon heittäminen ei kuitenkaan ole kovin halpaa ja nopeaa, jos tarvitaan suuri määrä satunnaisuutta. Kolikonheit-

toa voidaan tietenkin simuloida edellä mainituilla periaatteella, mutta silloin kyse on jostain muusta satunnaisuuden lähteestä.

Vaikka tietokone loogisena rakenteena ei kykene satunnaisuutta tuottamaan, voi sitä olla louhittavissa tietokoneen yhteydessä olevasta laitteistosta. Tietokoneen kello on ilmeinen ja paljon tutkittu. Kellonajan käyttäminen suoraan ei ole suositeltavaa, sillä silloin käytetyn siemenen selvittämiseksi ei välttämättä tarvitse kovinkaan suurta sarjajoukkoa käydä läpi, varsinkin silloin, kun siemenen ottamisen suurinpiirteinenkin aikaväli on tiedossa. Tietokoneen kello sisältää muitakin (lähinnä resonointiin liittyviä) ongelmia, puhumattakaan eri tietokoneiden kellojen erilaisista käyttäytymisominaisuuksista. [Schneier, 1996, p. 424].

Muita mahdollisia lähteitä tietokonelaitteiston piirissä ovat kiintolevyt sekä äänen ja kuvan sisääntuontiportit. Kiintolevyn hakuaikojen vaihteluun perustuvat menetelmät hyödyntävät suljetun tilan sisäisten ilmapyörteiden aiheuttaman kitkan vaikutusta levykiekon pyörintään ja sen kautta hakuaikaan. Vapaita signaaliportteja eli laiteportteja, joihin ei ole mitään laitetta kytkettynä, käyttävät menetelmät lukevat *lämpökohinaa* (thermal noise), joka on havaittavissa, kunhan portin päästövoimakkuus on kyllin korkea. Lämpökohinan oletetaan lisäksi olevan voimakkaan satunnaista. Levyjen hakuaikojen louhinnan heikkoutena on epävarmuus hakuaikojen mittaamisen riittävästä tarkkuudesta. Kyseiseen tarkkuuteen vaikuttaa olennaisesti tietokoneen kellon rajoitukset sekä käyttöjärjestelmän ja kiintolevyn hakuoptimointimenetelmät. Laitteistoon on luonnollisesti kiinnitettävä tarkkaa huomiota varsinkin satunnaislukuja louhittaessa, mutta välttämättä käyttäjän kannalta kätevä soveltaminen ei tätä aina suosi. Signaaliportit puolestaan ovat kovin herkkiä poimimaan muitakin signaaleja kuin vain lämpökohinaa, ja nämä ulkopuoliset tekijät voivat tuoda kohinaan säännönmukaisuuksia. [Eastlake *et al.*, 1994].

Tietokone ei laitteistona ilman huolellisia parannuksia ole kovin luotettava satunnaisuuden lähde, sillä tietokoneen satunnaisuutta tuottavat osat on harvemmin suojattu ulkopuolisilta häiriötekijöiltä, joita myös tietokoneen muut osat ja sähkömekaniikat aiheuttavat. Yksittäinen kone voidaan tietenkin rakentaa tarpeeksi suojatuksi, mutta se ei palvele joustavaa soveltamista eri ympäristöihin. Tietoko-

neen ulkopuolinen satunnaisuutta tuottava laite on huomattavasti helpompi suojata häiriöiltä ja myös siirtää vaivatta koneesta toiseen, mikäli kunnolliset rajapinnat ovat olemassa.

Säteilyyn ja sähköön liittyviä ilmiöitä on tutkittu satunnaisuuden lähteinä. Satunnaisuus louhitaan näistä ilmiöistä vertaamalla ilmiön eri ilmentymien eroja, tavallisimmin joko voimakkuuden tai keston suhteen. Digitaalisten kuvien eroihin perustuvaa menetelmää on hyödynnetty yhdessä muun muassa laavalamppujen yhteydessä. Radiokohina eli radiovastaanottimen poimima signaali sellaiselta taajuudelta, jolla ei varsinaista lähetystä ole, on oletettu myös hyväksi lähteeksi. [Schneier, 1996, pp. 423-424].

Myös ihmisen oletetaan käyvän satunnaisuuden lähteestä. Tietokoneen käyttäjän käyttäytymisestä voidaan louhia satunnaisuutta esimerkiksi laskemalla hiiren liikkeitä (tai kursorin sijaintien etäisyyksiä tiettyjen aikavälien välillä) tai näppäimistön painallusten sekvenssejä. Yleisimpiä ongelmia näppäimistön käytön louhinnassa ovat käyttöjärjestelmä ja laitteistokohtaiset rajoitteet menetelmälle, muun muassa näppäinpainallusten puskurointi, sekä suuren satunnaisuuden määrän tuottamisen hitaus. Puheesta voidaan myös louhia satunnaisuutta syvällisemminkin kuin pelkän tiedostomuodon tai ääniaaltojen etäisyyksien kautta (kuten kohinan tapauksessa on tapana tehdä). Puhe on ääntä josta on erotettavissa *hahmoja* sekä *piirteitä*. Hahmojen lausunnan piirteiden pohjalta on mahdollista tunnistaa puhuja sekä louhia satunnaisuutta, sillä piirteitä tulkitaan tarpeeksi tiukasta tunnistaminen mahdollistumisen kannalta, mutta myös riittävän löyhästi, jotta satunnaiset piirteet tulisivat esiin. [Schneier, 1996, pp. 424-425; Monrose *et al.*, 2001].

Käytettävän satunnaisuuden lähteen sekä louhintatavan valinta määräytyvät resurssien lisäksi tarpeen mukaan. Mikäli satunnaisuutta tarvitaan vain vähän kerrallaan ja suhteellisen harvoin, on satunnaisuuden louhiminen käyttäjän (tai muun ihmisen) toimista melko ongelmatonta. Jos puolestaan satunnaisuutta tarvitaan paljon, sitä on kätevintä louhia määrällisesti tuottoisammasta lähteestä.

2.4. Louhinta

Louhinta on lähdetiedon suodattamista, arvottamista ja analysointia. Louhinnassa pyritään poistamaan lähdetiedosta jakauman kallistumat ja toistuvuudet. Louhinnan määrittely ei ole oletusarvoisesti yksikäsitteistä, mutta tässä tutkielmassa louhinnalla tarkoitetaan jostakin tietystä lähteestä saatujen tulossarjojen käsittelyä satunnaisemman sarjan luomiseksi. Louhinnaksi voitaisiin kuitenkin nähdä myös esimerkiksi kolikon heittäminen, joka louhii satunnaisuutta kolikosta tai itsestään kolikonheitosta (tai ympäröivästä maailmankaikkeudesta, kuten aiemmin on todettu). Tulossarjoja tuottavien luonnonilmiöiden voidaan siis katsoa louhivan satunnaisuutta reaalimaailmasta, joka voidaan olettaa riittävän satunnaiseksi. Louhija on siis (näennäis)satunnaislukugeneraattori, joka ottaa siemensyötteekseen louhittavan syötesarjan (jonka synnyttänyt (luonnon)ilmiö on myös satunnaislukugeneraattori, jonka siemenen lähdettä ei kuitenkaan helposti voida jäljittää).

Tässä tutkielmassa louhijalla tarkoitetaan menetelmää, joka käsittelee syötteenään saamaa sarjaa ja tuottaa tulossarjan, joka on oletusarvoisesti syötesarjaa vahvemmin satunnainen. Louhijan ei vaadita tuottavan kiinteän kokoisia tulossarjoja, sillä louhijan tarkoitus on vain vahvistaa lähteen satunnaisuutta, jotta satunnaislukugeneraattoriin saataisiin vahvemmin satunnainen siemen. Lähde tuottaa satunnaisuuden ja louhija varmentaa tai vahvistaa sitä. Vaikka louhittua satunnaisuutta hyödyntävä menetelmä vaatisi kiinteämittaisen siemenluvun, ei ongelmia riittävän vahvan menetelmän kanssa synny, sillä se synnyttää tulossarjoja, joiden osasarjat ovat myös satunnaisia. Näin ollen yhdestä louhinnan tuloksesta saattaa riittää useammaksikin siemeneksi.

Yksi osa sarjan satunnaisuusoletusta on, että sarjaa ei voi esittää lyhyemmin (eli tiivistää), joten sarjan louhinnassa voisi käyttää hukkaamatonta pakkausta. Koska tietoa hävittämättä pakatusta sarjasta voidaan palauttaa alkuperäinen sarja, sisältää pakattu sarja yhtä paljon informaatiota kuin alkuperäinenkin. Näin ollen on oletettavissa, että pakattujen sarjojen joukko on tasaisemmin jakautunut kuin pakkaamattomien ja siksi pakatut sarjat ovat painottumattomampia verrattuna alkuperäi-

siin sarjoihin. Tästä seuraa, että pakkaaminen voisi toimia louhijana. [Eastlake *et al.*, 1994].

Mikäli pakkaaminen hukkaa tietoa, eli alkuperäistä sarjaa ei voida pakatusta enää palauttaa, ei sinänsä voida katsoa pakkauksen vahventaneen sarjan satunnaisuutta. Ylipäätään pakkaaminen ei ole oikotie onneen, sillä vaikka pakattu sarja olisi kompleksisuusnäkökulmasta satunnainen, voi sarjan rakenne olla todennäköisyysjakaumien perusteella heikosti satunnainen. Pakkaaminen vaikuttaa suoranaisesti vain sarjan kompleksisuuteen.

Pakkaamiseen tarkoitetut menetelmät ja sovellukset myös lisäävät tulossarjaan metatietoa, joka ohjailee muun muassa pakkauksen purkamista. Näin ollen täytyy satunnaisuutta louhittaessa käyttää varta vasten louhintaan tarkoitettuja menetelmiä, muokata pakkaajien toimintaa (esimerkiksi poistamalla tulossarjoista metatiedot) tai käyttämällä yksisuuntaisia pakkausmenetelmiä. [Eastlake *et al.*, 1994].

2.4.1. Syötteen parillisuus

Satunnaisuuden digitalisointi tehdään ennen louhintaa, jolloin louhijan rakentamisessa voidaan keskittyä käsittelemään bittejä. Yksinkertainen menetelmä louhijaksi on kuvata riittävän kokoinen syötteestä otettu osajoukko joko 0- tai 1-bitillä. Tämä menetelmä ei välttämättä tuota täysin satunnaista tulossarjaa, mutta sillä voidaan päästä hyvinkin lähelle riippuen osajoukon koosta - ja tietenkin koko syötteen luonteesta. Osajoukon *pariteettibitin* (parity) irrotus on yksinkertainen ja helposti toteutettava menetelmän sovellutus [Eastlake *et al.*, 1994]. Tulossarjan bitti määräytyy syötesarjan tavun vähiten merkitsevän bitin perusteella eli käytännössä tavun parillisuuden (bitin arvo 0) ja parittomuuden (bitin arvo 1) mukaan.

Vaikka pariteettibitin irrotuksen voidaan jo itsessään katsoa olevan louhintaa, se on hyödyllinen pohja tavuarvoisen syötteen louhimisessa muilla sellaisilla menetelmillä, jotka hyödyntävät ennemmin syötteen yksittäisiä arvoja kuin sen mahdollisesti sisällään pitämiä hahmoja. Tämä pätee varsinkin sellaisten lähteiden kohdalla, jotka eivät tuota tasaista jakaumaa koko arvoalueelleen. Esimerkiksi kuvat-

taessa aaltomuotoista signaalia tavuesityksenä voi arvojakauma keskittyä enemmän aallon nolla-arvoa kuvaavan tavun ympäristöön kuin ääriarvoille, riippuen signaalin luonteesta ja voimakkuudesta. Tällöin signaalitiedoston bittiesitys voi sisältää hyvin paljon toistoa, jolloin louhintamenetelmät voivat joutua kahlaamaan läpi huomattavan määrän hyödyttömiä bittejä. On siis kätevämpää esilouhia aineistoa varsinaiselle louhijalle optimaalisemmaksi, mihin pariteetin irrotus on omiaan.

Pariteettibitin irrotuksesta esilouhintana ei luonnollisestikaan ole hyötyä, mikäli varsinainen louhinta suuntautuu syötesarjan sisäisiin hahmoihin tai merkityksiin. Tällöin pariteetin irrotus rikkoo alkuperäistä satunnaisuutta: esimerkiksi syötteen sisältämien hahmojen etäisyyksien vertaaminen tulee mahdottomaksi, joten pariteetin irrotus ei ole kaikkialle sopiva ja sillä tavoin automaattisesti sovellettavissa.

Pariteetin irrotus louhii lohko kerrallaan, ja sen tehokkuus riippuu lähteen ja sen tuottaman tulossarjan asettamista vaatimuksista lohkon koolle. Mikäli syötesarjan bittijakauma on täysin tasainen, lohkon kooksi riittää yksi bitti. Taulukossa 2.1 on annettu bittisarjan pituus (N), joka tarvitaan, jotta tulossarjassa 0- ja 1-bitin todennäköisyys olisi välillä [0.49999,0.50001], kun jakauma vaihtelee toisen arvon suhteen funktion Prob(n) mukaan. Taulukon mukaan syötteeksi tarvitaan seitsemän bitin mittainen sarja, mikäli bitin todennäköisyys (taulukossa 1-bitin todennäköisyys, Prob(1)) on 0.7. Tässä tapauksessa tavun kokoisesta lohkosta päästään siedettävän lähelle tasajakaumaa, jos syötesarjan jakauma on toisen bittiarvon hyväksi korkeintaan hivenen päälle 0.7. [Eastlake *et al.*, 1994].

Pariteetti	
Prob(1)	N
0.5	1
0.6	4
0.7	7
0.8	13
0.9	28
0.95	59
0.99	308

Taulukko 2.1 – Pariteetin irroituksen vaatima bittijonon pituus painottuneelle jakaumalle.

2.4.2. Kolikonheiton louhinta

Yksinkertaisuus ja suora analogia bittiesitykseen ovat syyt, joiden takia juuri kolikonheitto on usein valittu satunnaisuuden lähteen symboliksi. Analogisen lähteen antamat sarjat on joka tapauksessa muutettava biteiksi, jotta niitä voidaan tietokoneella käsitellä. Satunnaisuuden lähteen luonteen ja sovellutuksen perusteella voi toisinaan olla käytännöllisempää ja tehokkaampaa tarkastella kolikon sijaan n-tahoista noppaa tai arpaa - kolikkohan on kaksitahoinen noppa. Riippuen satunnaisuuden lähteestä ja tavasta, jolla lähteestä saatu syötesarja digitaalisesti kuvataan, voi pariteettibitin irrotus olla paikallaan.

On huomattava, että mikäli kolikko olisi todistettu täydellisen reiluksi, eli painottumattomaksi, olisivat sillä heitetyt tulokset sinällään käytettävissä satunnaisen sarjan muodostamiseen, kuten edellä on mainittu. Tällaisesta ideaalisesta kolikosta ei siis tarvitsisi satunnaisuutta louhia, vaan tuloksia voitaisiin käyttää suoraan. Painottuneella kolikolla luotu sarja puolestaan vaatii louhintaa, jotta siitä saataisiin mahdollinen satunnaisuus esiin. On myös huomattava, että mielekkäässä sovellutuksessa tietokonemaailmaan kolikonheiton tulos tulisi kuvata bittinä. Kummaksi bitiksi esimerkiksi kruuna määritellään, ei ole merkitystä, kunhan valinnan suhteen ollaan jatkossa johdonmukaisia.

Seuraavaksi esitellään kolme menetelmää, jotka tuottavat tuntemattoman todennäköisyysjakauman omaavien kolikoiden heitoista tasajakaumallisia kolikonheittoja. Yleisemmin nämä menetelmät tuottavat siis satunnaisuuden vaatimukset

täyttäviä tulossarjoja syötteenä saaduista hahmoista. On toki huomattava, että mitä vahvempaa syötehahmo satunnaisuudeltaan on, sitä tehokkaammin kuvaillut menetelmät toimivat.

Menetelmät esitellään tarkemmin, sillä niiden käytännön soveltamista käsitellään vielä myöhemmin tässä tutkielmassa. Kolikonheittomenetelmiä voidaan soveltaa käytännössä suoraan bittisarjoihin, mutta lähteen muodosta johtuen voi olla hyödyllistä louhia lähteen antamia sarjoja ensin jollakin muulla menetelmällä ennen kolikonheittolouhintaa. Pariteettibitin irrotus on suositeltavaa sikäli, että se vähentää satunnaisuuden kannalta turhien ja painottuneiden bittien läpikäyntiä. Vastaavasti toistuvien ilmiöiden vertaamiseen perustuvaa menetelmää voidaan tehostaa käsittelemällä jo satunnaista tulossarjaa kolikonheittolouhinnoilla, joskaan tehostaminen ei aina ole hyödyllistä. Oleellista siis on, että lähteen satunnaisuus louhitaan esiin menetelmällä, joka ei hävitä alkuperäistä satunnaisuutta.

2.4.3. Von Neumannin menetelmä

Painottuneen kolikon muuttamisessa reiluksi voidaan lähteä von Neumannin menetelmän nimellä kulkevasta vanhasta ja yksinkertaisesta menetelmästä. Menetelmässä louhinnan kohteena olevaa sarjaa tarkastellaan kaksi aiemmin käsittelemätöntä tulosta kerraallaan. Mikäli kahden peräkkäisen heiton tulokset ovat samat, unohdetaan kyseiset heitot ja siirrytään seuraavaan heittopariin. Mikäli taas parin tulokset ovat erilaiset, valitaan lopulliseksi tulokseksi bitti sen mukaan, missä järjestyksessä kruuna ja klaava ovat (jälleen seikka, joka luonnollisesti päätetään ennen tilannetta). [Eastlake *et al.*, 1994].

Von Neumannin menetelmä kykenee luomaan täysin painottamattoman tulossarjan huolimatta kolikon painotuksista. Menetelmä kykeneekin täydellisesti eliminoimaan painotukset, vaikka heittojen tulosten todennäköisyysjakauma olisi ennalta tuntematon. [Mitzenmacher, 2001].

Menetelmän selkeä haittapuoli on sen mahdollinen tehottomuus louhitun satunnaisuuden määrän osalta. Jos heittojen todennäköisyysjakauma on hyvin tasainen,

menetelmä tuottaa satunnaista tulossarjaa korkeintaan puolet louhittavan sarjan bittimäärästä ja tämänkin vain hyvin teoreettisessa tilanteessa. Riippumatta kolikonheittojen tulosten jakaumasta, von Neumannin menetelmä hylkää ainakin puolet tulospareista. Jos kolikko on voimakkaammin painottunut, kolikon antaman tulossarjan entropia pienenee ja tällöin menetelmä vaatii suurempaa sarjaa louhittavaksi tuottaakseen satunnaisen sarjan. Mikäli kolikon antamien tulosten jakaumaa ei ennalta tiedetä, ei voida tietää tarvittavaa louhittavan aineiston määrää, jos tarkoituskena on louhia juuri tietyn kokoinen määrä satunnaisuutta. [Eastlake *et al.*, 1994; Juels *et al.*, 2000; Mitzenmacher, 2001].

Von Neumannin menetelmä toimii pohjana monille tehokkaammiksi ja mielekkäämmiksi suunnitelluille menetelmille. Von Neumannin menetelmän kyky poistaa painottuneisuutta on pyritty säilyttämään, mutta syötejoukkoa on pyritty hyödyntämään tehokkaammin.

2.4.4. Mitzenmacherin parannus von Neumannin menetelmään

Von Neumannin menetelmän taipumusta tuottaa lähdeaineistoon verraten vähän louhittua tulossarjaa parannetaan Mizenmacherin [2001] kehittämässä menetelmässä. Menetelmä hyödyntää mahdollisimman tehokkaasti niitäkin heittotuloksia, jotka von Neumannin menetelmä hylkää.

Seuraavassa kruuna kuvataan merkillä ”T” (tails) ja klaava merkillä ”H” (heads). Jos tutkittavan parin arvot ovat samat, von Neumannin menetelmä hylkää kyseisen parin ja siirtyy seuraavaan. Tässä voidaan olettaa, että tutkittavan parin arvojen ollessa HT tulokseksi tulee 0-bitti ja vastaavasti arvopari TH antaa tuloksesti 1-bitin (ks. taulukko 2.2).

Von Neumann															
Heittosarja	H	H	T	H	T	T	T	H	T	H	H	T	H	H	
Louhitut bitit		-		1		-		1		1		0		-	
Luotu sarja	1	1	1	0											

Taulukko 2.2 – Esimerkki von Neumannin menetelmästä.

Mitzenmacher rakentaa menetelmänsä yhdistämällä kaksi tapaa hyödyntää ”hylättyjä” pareja. Menetelmä on rekursiivinen ja vaatii muistitilaa toisin kuin von Neumannin menetelmä.

Menetelmän ensimmäisessä osassa luodaan alkuperäisen heittosarjan pohjalta uusi sarja heittoja (jotka ovat siis keinotekoisia eikä erikseen heitettyjä) ja pyritään hyödyntämään symmetrisyyttä entisestään. Pari HT tuottaa edelleen suoraan 0-bitin ja TH 1-bitin, mutta pari HH merkitään uuteen heittosarjaan heittotulokseksi H ja TT tulokseksi T. Kun alkuperäinen heittosarja on käyty läpi, tutkitaan seuraavaksi rekursiivisesti luotu uusi heittosarja. Tätä voidaan jatkaa kunnes enää yhtään satunnaisuutta ei ole tällä tavoin irrotettavissa. Käytännössä tämä menetelmä suhteutuu alkuperäiseen heittosarjaan siten, että osasarja HH**TT tuottaa 0-bitin ja TT**HH 1-bitin, kun ** merkitsee n kappaletta pareja HT ja/tai TH. Osasarja HH**HH tai TT**TT ei tuota suoraan bittiä vaan uuden heittotuloksen keinotekoisten heittojen sarjaan. Tätä menetelmän osaa on havainnollistettu taulukossa 2.3.

Mitzenmacher																	
Osa 1/2																	
Heittosarja	H	T	H	H	T	H	H	H	T	T	H	T	T	H	T	T	
Louhitut bitit		0		-		1		-		-		0		1		-	
Luotu sarja 1				H				H		T						T	
Louhitut bitit								-								-	
Luoto sarja 2								H								T	
Louhitut bitit																0	
Louhittu sarja	0	1	0	1	0												

Taulukko 2.3 – Esimerkki Mitzenmacher-menetelmän ensimmäisestä osasta.

Ensimmäinen osa menetelmästä tuottaa jo tehokkaammin satunnaisuutta verrattuna puhtaaseen von Neumannin menetelmään (samasta määrästä varsinaisia heittoja). Silti se on vielä suhteellisen kaukana ihannetilanteesta, jossa yksi tulossarjan bitti vastaa yhtä kolikonheittoa.

Menetelmän toinen osa hyödyntää ensimmäisen osan huomiotta jättämää symmetriaa. Osasarjat HHHT ja HTHH tuottavat menetelmän ensimmäisen osan perusteella saman tuloksen: yksi bitti satunnaisuutta ja yksi heitto uuteen heittosarjaan. Toinen osa menetelmää muodostaa jokaisesta heittoparista heiton uuteen sarjaan sekä muodostaa bittituloksen von Neumannin menetelmän ehtojen mukaan. Mikäli heittoparin alkiot ovat samat, tulee uudeksi heittotulokseksi T ja mikäli alkiot eivät ole samat, niin uuteen sarjaan tulee H. Tämän lisäksi eri alkioiset parit käsitellään von Neumannin menetelmän mukaisesti, jotta bittimuotoinen tulossarja saadaan muodostettua. Tämä toinenkin osa on siis rekursiivinen, ja jokainen rekursiivisesti käsiteltävä heittosarja on pituudeltaan puolet edellisen tason heittosarjasta (poikkeuksena ensimmäinen rekursiotaso, joka voi olla kooltaan (n-1)/2 silloin, kun alkuperäisen heittosarjan alkioiden lukumäärä n on pariton). (Taulukko 2.4.)

Mitzenmacher																
Osa 2/2																
Heittosarja	H	T	H	H	T	H	T	T	T	T	H	T	T	H	H	H
Louhitut bitit		0		-		1		-		-		0		1		-
Luotu sarja		H		T		H		T		T		H		H		T
Louhitut bitit				0				0				1				0
Louhittu sarja	0	1	0	1	0	0	1	0								

Taulukko 2.4 – Esimerkki Mitzenmacher-menetelmän toisesta vaiheesta.

Menetelmän osat yhdistettyinä louhivat alkuperäisestä heittosarjasta Mitzenmacherin mukaan riippumattomia ja reiluja bittejä. Yhdistäminen johtaa rekursiiviseen menetelmään, jonka jokaisella rekursion tasolla on mahdollista joko luoda kaksi uutta haaraa rekursiopuuhun, luoda yksi uusi haara tai lopettaa rekursio. Bittien muodostumista ei välttämättä tapahdu jokaisella tasolla, riippuen luonnollisesti alkuperäisestä heittosarjasta. Rekursio tapahtuu menetelmässä aina viimeisenä, jolloin jokaisella rekursiotasolla luodut bittisarjat tulostetaan ennen seuraavalle tasolle menoa (Taulukko 2.5).

Mitzenmacher																	
Heittosarja	H	T	H	H	T	H	T	T	T	T	H	T	T	H	H	H	
Louhitut bitit		0		-		1		-		-		0		1		-	
Luotu sarja A				H				T		T						H	
Louhitut bitit								0								1	
Luotu sarja B		H		T		H		T		T		H		H		T	
Louhitut bitit				0				0				1				0	
Louhittu sarja	0	1	0	1	0	1	0	0	1	0							

Taulukko 2.5 – Kierros yhdistettyä Mitzenmacher-menetelmää.

Mitzenmacheri-menetelmä kykenee teoriassa luomaan yhden satunnaisen bitin jokaista kolikonheittoa kohden, kunhan syötteenä oleva heittosarja on riittävän pitkä. Ongelmana voi olla hitaus rekursiopuun kasvaessa tasaisesti alkuperäisen heittosarjan kasvaessa.

Von Neumannin ja Mitzenmacherin menetelmät eivät periaatteessa ota kantaa syötteidensä lähteisiin. Kummallekin menetelmälle riittää, että lähdesarja näyttää sarjalta kolikonheittoja. Mikäli lähde on satunnaisuuden kannalta vahva, kumpikin menetelmä kykenee louhimaan satunnaisuutta tehokkaammin. Mikäli lähde puolestaan on heikko, von Neumannin menetelmän tulostesarjan koko on huomattavan pieni, eikä Mitzenmacherin menetelmältäkään voida automaattisesti olettaa suurta määrää louhittua satunnaisuutta.

2.4.5. Painotetun nopan muuttaminen reiluksi kolikoksi

Juelsin ja muiden [2000] kehittelemä menetelmä painottuneen S-tahoisen (tässä yhteydessä nopan tahoja merkitään symbolilla *S*, sillä *n* on varattu toiseen käyttöön) nopan muuttamiseksi reiluiksi kolikoiksi syventää ja laajentaa kolikonheiton mahdollisuuksia. Menetelmä ottaa kiinteän määrän painottuneita syötteitä ja tuottaa vaihtelevan kokoisen ja painottumattoman (eli reilun) tulostesarjan. Von Neumannin menetelmän ydin oli, että jos H ja T tulevat samaan heittopariin, on niiden järjestyksien todennäköisyydet yhtäsuuret, jolloin juuri tuollaisten heittoparien myötä satunnaisuus louhiintuu. Sama periaate voidaan laajentaa koskemaan S-tahoista noppaa.

Menetelmä koostuu kahdesta algoritmista, joista ensimmäinen muuttaa painottuneen nopan reiluksi nopaksi ja toinen muuntaa reilun nopan reiluiksi kolikonheitoiksi, eli reiluksi (ja ominaisuuksiltaan täten satunnaiseksi) bittijonoksi. Menetelmän ytimenä on alkuperäisen nopanheittosarjan muuntaminen sarjaksi, jossa alkiot kuvaavat nopenheittojen *asteita* (rank). Kun nopalla D on heitetty n-kertaa ja saatu täten tulossarja X = <X[1], X[2], … , X[n]>, voidaan X muuntaa *astesarjaksi* (rank sequence) p(X) = <rank(X[1]), rank(X[2]), … , rank(X[n])>, missä X[i]:n aste rank(X[i]) on arvoltaan pienempien tai yhtäsuurten alkioiden lukumäärä.

Ensimmäisessä algoritmissa oletetaan tiedetyksi vain, että n-kappaletta nopan D heittoja (lyhyemmin D^n) on tuottanut tulossarjan X ja täten *astemonijoukon* (rank multiset) {p(X)}, joka on kaikkien muotoa p(X) olevien astesarjojen järjestämätön joukko ja jossa on S osasarjaa. Koska jokainen D^n:n astesarjoista on yhtä todennäköinen, voidaan kasvavaan järjestykseen järjestetystä astemonijoukosta {p(X)} katsoa, monesko sarja p(X) on, ja merkitä tätä ”sijaa” symbolilla R. R kuvaa siis yhtä S-tahoisen nopan (kuvataan tässä symbolilla U) heiton tulosta.

Toinen algoritmi muuntaa nopalla U heitetyn tuloksen R painottumattomaksi bittisarjaksi, eli kolikonheitoiksi. Nopan U tahot jaetaan kasvavassa järjestyksessä erisuuriin joukkoihin, joiden koot ovat 2:n potensseja. Joukkojen lukumäärää merkitään kirjaimella c. Joukkojen (jotka ovat siis kokoa 2^c) sisällä alkiot viittaavaat yksi yhteen -suhteella joukon $\{0,1\}^c$ merkkijonoihin. Algoritmin tulostesarja saadaan, kun haetaan nopan U tulosta R vastaava bittijono.

Menetelmän ottaman syötteen lähde ei ole aivan yhtä yksinkertainen kuin aiemmin esitellyillä menetelmillä. Von Neumannin ja Mitzenmacherin menetelmäthän ottivat syöttekseen kolikonheittoja, joihin löytyi suora yhteys bittiesityksestä. Nopanheitto ei sinänsä ole aivan yhtä yksinkertainen. Mikäli lähde antaa menetelmälle syötettä yksi bitti kerrallaan, täytyy bittejä kerryttää ja muodostaa kerrytetystä sarjasta nopan tulokseksi sopivampi arvo. Kaksitahoinen noppa ei käytännössä sovellu tälle menetelmälle, sillä silloin bitti kopioidaan kalliisti tulossarjaksi. Mikäli puolestaan syötteeksi otetaan bittiä suurempi esitys vaikkapa kokonais-

lukuna, tulee ongelmaksi määrittää lähteen antamasta signaalista reilut ja tasaiset rajat. Jos lähde tuottaa liukulukuarvoja, niin rajojen vetäminen ei ole aivan ongelmatonta. Lähteestä saatujen bittien kerryttäminen suuremmiksi sarjoiksi toki lisää nopan tahoja ja säilyttää mahdollisuuden lähteen tulosten tulkitsemiseen bitteinä, mutta voi joissain tapauksissa vaatia huomattavan suuria määriä lähdemateriaalia pienen satunnaisuuden määrän tuottamiseksi.

2.5. Satunnaisuuden merkitys kryptologiassa

Usein salausjärjestelmien ja yleensäkin kryptograafisten menetelmien yhteydessä oletetaan käytettävissä olevan satunnaisuuden lähteen - jolla viitataan käytännössä lopullisen louhintamenetelmän tuottamaan tulossarjaan eli siemeneen - tuottavan aitoa satunnaisuutta. Vaarana oletuksessa on, että kun menetelmää aletaan käytännössä soveltaa, ei satunnaisuuden lähteen voimakkuuteen kiinnitetä riittävää huomiota, tai kehitetty menetelmä vaatii nimenomaan aidosti satunnaisen sarjan myös käytännössä. Oletus on luontevaa tehdä, sillä ensinnäkin on havaittu, että myös vahvan kryptografisen menetelmän turvallisuutta voidaan horjuttaa, jos päästään vaikuttamaan avainten luontiin [Agnew, 1998]. Toisekseen on luontevaa rakentaa ja todistaa vahvaa menetelmää silloin, kun menetelmän voidaan olettaa saavan aidosti satunnaisia sarjoja käyttöönsä.

Kryptojärjestelmissä satunnaisuuden vahvuus on loppujen lopuksi pieni osa koko turvallisuutta. Tiedon siirron, laitteistojen ja käyttäjien keskuudesta löytyvät suurimmat vaaratekijät ja haavoittuvuudet. Vaikka muu järjestelmä olisi täydellinen, voi heikko satunnaislukujen generointi esimerkiksi avaintenluontia varten vaarantaa koko järjestelmän turvallisuuden [Morris and Thompson, 1979]. Täten käytettyjen satunnaislukujen arvaamattomuutta ei auta laiminlyödä, jos järjestelmän haluaa pysyvän turvallisena [Eagini and Buce, 1999].

Fyysisesti louhijan olisi hyvä olla varsinaisen kryptojärjestelmän ulkopuolella, sekä turvallisuuden että ylläpidettävyyden vuoksi. Satunnaisuutta louhiva teknologia ei useinkaan ole suoraan yhteensopivaa kryptojärjestelmän taustalla olevan teknologian kanssa. Täten rajapinta louhijan ja kryptojärjestelmän välillä on kriit-

tinen. Louhijan ollessa kryptojärjestelmästä erillinen mahdollistuu kummankin osan laajempi uudelleenkäyttäminen. Mikäli louhittava satunnaisuuden lähde osoittautuu vialliseksi tai huonoksi, ei kryptojärjestelmälle koidu ongelmia lähteen vaihtamisesta, sillä järjestelmä näkee vain rajapinnan ja on tyytyväinen, kunhan rajapinta toimii oikein. Ulkopuolinen louhija on usein myös helpompi suojata tarkkailua ja vaikuttamista vastaan. Myös satunnaisuuden lähteeseen vaikuttavien ympäristöstä aiheutuvien ja tahallisten häiriötekijöiden (lämpö, sähkömagneettiset kentät, johtimien modulaatio, säteily ja lähteen pakotettu palauttaminen tunnettuun alkutilaan) säätely on yksinkertaisempaa ja varmempaa. Haittana ulkopuolisesta louhijasta on, että yhteys louhijasta kryptojärjestelmään voi olla hyökkäyksille altis. Louhijan olisi myös suotavaa hyödyntää luotettavaa satunnaisuuden lähdettä, jolloin tulosten tarkkailuun perustuvat hyökkäysmenetelmät (ns. passiiviset hyökkäysmenetelmät) menettävät tehonsa: louhijan toimintaan ja tulokseen vaikuttavien hyökkäysten (ns. aktiiviset hyökkäysmenetelmät) varalta louhijan fyysinen suojaaminen on välttämätöntä. [Agnew, 1998].

Satunnaisuuden käsitteiden ero aiheuttaa ongelman nimenomaan silloin, kun satunnaista sarjaa aletaan konkreettisesti luoda kryptologiseen järjestelmään, ja varsinkin, kun sarjaa yritetään ohjelmallisesti löytää. Täysin mielivaltaista sarjaa ei voi, ilman muuta tietoutta, löytää muuten kuin joko käymällä kaikki mahdolliset sarjat järjestelmällisesti läpi tai silkalla tuurilla. *Sarja-avaruuden* mahdollisimman suuri koko luonnollisesti hidastaa kummankin tavan onnistumista. Käytännössä valitun sarjan riittää olla juuri niin vahva, että murtajalla ei ole mahdollisuuksia saada sitä selville.

Raa'an voiman menetelmää voidaan optimoida poistamalla murrettaessa läpikäytävien sarjojen joukosta heikon entropian sisältävät sarjat, koska vahvoilla menetelmillä luodun sarja-avaruuden voidaan olettaa sisältävän vain vahvoja sarjoja. Mitä kompleksisesti vahvemmalla menetelmällä satunnaiset sarjat on luotu, sen nopeampaa on *sanakirjahaulla* (dictionary attack) löytää satunnaisuudeltaan heikot sarjat – olettaen, että vahvemman ja heikomman menetelmän ero on niiden hyväksymän satunnaisuuden laatu samassa sarja-avaruudessa. Sanakirjahakua voidaan tietenkin vaikeuttaa kasvattamalla alkuperäistä sarja-avaruutta, jolloin

myös kompleksisesti vahvojen sarjojen joukko oletettavasti kasvaa. Käytettäessä riittävän suuria sarja-avaruuksia nopeampi sanakirjahakukin menettää tehokkuuttaan.

Oleellista satunnaisia lukuja sovellettaessa onkin käyttää sellaisia menetelmiä, jotka tekevät mielivaltaisesti valitun sarjan löytymisen jälkeenpäin lähes mahdottomaksi. Tämä voidaan saavuttaa käyttämällä sarjaa siemenenä vahvassa yksisuuntaisessa menetelmässä. Sarjan lähteellä ei ole loppujen lopuksi suurta merkitystä, kunhan lähde kykenee tuottamaan sarjoista varmasti ennalta arvaamattomia, eikä se tuota muita sarjoja todennäköisemmin samanlaista tulosta uudestaan.

Aidosti satunnaisen sarjan erottaminen näennäisesti satunnaisesta ei ole millään muotoa yksinkertaista. Itse asiassa ei ole olemassa keinoa pitävästi todistaa mitään sarjaa aidosti satunnaiseksi. Erber ja Putterman [1985] näyttävät, että tällä on kaksi vakavaa käytännön vaikutusta viestien salaamiseen. Ensinnäkin satunnainen sarja voi vaikuttaa ei-satunnaiselta. Kun apinat lasketaan kirjoituskoneiden pariin, voi tuloksena syntyä Shakespearen Hamlet. Jos sarjaa käytetään esimerkiksi salausavaimena, tämä on ongelmallista sikäli, että ei-satunnaiselta näyttävä sarja voikin todellisuudessa olla alttiina sanakirjahyökkäykselle. Ei siis riitä, että ollaan vakuuttuneita satunnaisgeneraattorin kyvystä luoda satunnaisia sarjoja. Sarjat täytyisi voida tarkistaa etukäteen, jotta voitaisiin luoda uusi satunnainen ja ”satunnaiselta näyttävä” sarja. Käytännössä tämä tarkistus täytyy tehdä satunnaislukugeneraattorissa ennen sarjan tulostamista. Näin siksi, että tällöin satunnaislukugeneraattorin (vihamielinen) analysointi ulkopuolelta vaikeutuu. Myös satunnaislukugeneraattorijärjestelmän eheys ja ylläpito on tällöin varmempaa. Toisaalta tarkistus tekee satunnaislukugeneraattorista raskaamman. Käytännössähän tarkistus tarkoittaa sanakirjahakua, jota toki helpottaa huomattavasti se seikka, että tarkistettava sarja tiedetään. Kyse on vain sanakirjan täydellisyydestä. Satunnaislukugeneraattorin sanakirjan tulisi olla vähintään yhtä hyvä kuin paras murtamista varten kehitetty tai kehitettävissä oleva sanakirja.

Parhaan mahdollisen sanakirjan avulla tapahtuva sarjojen karsiminen ei välttämättä rajoita käytettävissä olevaa sarja-avaruutta aivan mahdottomasti. Chaitin [1975]

nimittäin huomauttaa kompleksisuuden näkökulmasta satunnaisuutta käsitellessään, että missä tahansa pitkässä ja satunnaisessa bittijonossa nollien ja ykkösten suhteellinen frekvenssi on lähellä puolta - ja vastaavasti 1/10:aa desimaalilukusarjassa. Ei-satunnaisen frekvenssijakauman omaavat numerot puolestaan ovat poikkeuksellisia. Vain noin yksi tuhannesta sarjasta voidaan tiivistää algoritmiksi, joka on enemmän kuin 10 numeroa pienempi kuin sarja itse. Chaitin kuitenkin varoittaa, että pitkien sarjojen soveltaminen sisältää muitakin ongelmia eikä täten ole aivan yksinkertaista.

Toinen Erberin ja Puttermanin [1985] esittämä aidosti ja näennäisesti satunnaisen erottamisesta seuraava ongelma on, että deterministinen prosessi voi imitoida satunnaista tapahtumaa ja täten luoda satunnaiselta näyttävän sarjan. Vaikka yksittäistä sarjaa ei voisi omilta ominaisuuksiltaan erottaa satunnaisesta, se on generoitavissa uudelleen, mikäli taustalla oleva deterministinen prosessi tunnetaan. Täten sarja on tunnistettavissa ja sen kryptologinen vahvuus on heikko. Ongelma on sikäli kiperä, että ei tosiaankaan ole mitään täysin varmaa menetelmää erottaa satunnaista sarjaa ei-satunnaisesta, eikä ole keinoa todistaa satunnaisuutta luovan menetelmän toimivan ei-deterministisesti. Sarja on mahdollista todistaa ei-satunnaiseksi, mutta satunnaistakaan sarjaa ei voi todistaa aidosti satunnaiseksi [Ritter, 1991]. On vain löydettävä näennäissatunnaisuuden lähteitä, joiden determinististä prosessia on lähes mahdotonta tunnistaa.

Koska minkään ilmiön satunnaisuudesta ei voida täydellä varmuudella vakuuttua, eikä täten voida olettaa minkään mahdollisen funktion, prosessin tai algoritmin kykenevän luomaan aidosti satunnaista sarjaa, on vain luotettava tietämättömyyteen. On luotettava tarpeeksi vahvaksi todettuun prosessiin, joka louhii satunnaisuutta ilmiöstä, jonka luonnetta ei tunneta. Tällöin louhittu satunnainen sarja vaikuttaa aidosti satunnaiselta verrattuna tunnettujen prosessien louhimaan sarja-avaruuteen. Näennäissatunnaisuus on vahvaa, mikäli sen tuottaman ilmiön kaavan ratkomiseen tarvitaan saavuttamattomissa olevaa laskentatehoa ja suunnattoman laajoja (ja lukuisia) näytteitä.

Mielivaltainen valinta on siis käytännössä mahdoton todistaa aidosti mielivaltaiseksi. Jonkin ilmiön antamat tulokset voivat vaikuttaa mielivaltaisilta, mutta ilmiön aitoa mielivaltaisuutta ei siltikään voi todistaa. Koska tietokoneet toimivat täysin deterministisesti, tiedetään tietokoneen olevan kykenemätön mielivaltaisiin valintoihin.

2.6. Avainten luonnin vaatima satunnaisuus

Kryptologiassa ilmeisin käyttökohde satunnaisuudelle on salasanojen ja salausavainten luonti. Salausavainten luonnissa oleellinen seikka on käytettävän (näennäis)satunnaislukugeneraattorin toistettavuusominaisuus. Samalla siemenluvulla satunnaislukugeneraattorin tulisi tuottaa sama tulsosarja. Tämä ominaisuus on tarpeellinen muun muassa siksi, että satunnaiseen avaimeen perustuvaa salausta luotaessa on oleellista kyetä salaus myöhemmin myös purkamaan.

Monrosen ja muiden [2001, 2002] tutkima puhujantunnistuksen ja salasanan luonnin menetelmä perustuu ajatukseen avaimenluonnista, jossa avain ei ole ennustettavissa, mutta on toistettavissa. Avaimen ennustamattomuus pyritään takaamaan ehdolla, että sitä ei saa voida päätellä kaikesta muusta siihen liittyvästä tiedosta eikä sitä luovasta menetelmästä. Menetelmän turvallisuutta lisää tehty muokkaus puheentunnistuksesta käyttäjäntunnistukseen, jolloin puhuttu salasana menettää varsinaisen kielellissisällöllisen merkityksen ja puhujan salasanan sanomisen piirteet nousevat esiin. Tällöin itse salasanasta ei tarvitse tallentaa mitään tietoa, vaan ainoastaan käyttäjän tapaa sanoa salasana kuvaava vektori tallennetaan. Vektori sinällään ei paljasta avaimesta mitään tietoa. Avain on toistettavissa, jos sama käyttäjä lausuu saman salasanan. Menetelmä hakee yhteneväisyyksiä käyttäjän antaman puhenäytteen pohjalta suoritettavan puheentunnistuksen epäyhteneväisyyksistä. Menetelmän haittapuolia on hidas ja joustamaton alkuvalmistelu. Käyttäjän tulee lausua valittu salasana useita kertoja järjestelmälle, jotta piirrevektori muodostuisi. Tämä ongelma kuuluu puheentunnistukseen, mutta kun biometrisen tunnistamisen ohella halutaan luoda vaihtelevia salausavaimia, kertautuu käyttäjältä vaadittu vaiva. Helpompaa olisi luoda avain nopeasti käytettävällä menetel-

mällä, mikäli käyttäjän tunnistaminen tai siemenen varma biometrinen uudelleenluonti eivät ole oleellisia kriteereitä.

Käytännön tasolle vaikkapa avainten luontiin sovellettuna on kiinnitettävä huomiota satunnaisuutta louhivan menetelmän luonteeseen sen tuottaman tulossarjan perusteella. Yleensä avaintenluonnissa turvaudutaan puhtaisiin bittisarjoihin, jolloin mahdollinen tavuesitys on korkeintaan bittisarjan tallennukseen tai havainnollistamiseen liittyvä piirre. Suoraan tavusarjoja luova menetelmä on toki myös toimiva, sillä mikäli tavu on satunnainen, on sen kuvaava bittisarjakin satunnainen. Kyseessä on vain kantaluvun häviötön muunnos.

3. Äänitieto satunnaisuuden lähteenä

Ääni kaikissa merkityksissään on laaja käsite. Tavallisesti ääni mielletään aistimukseksi, jolloin ääntä on kaikki, mikä on korvin kuultavissa. Tämä määritelmä ei kuitenkaan ole kovin mielekäs, ainakaan äänen digitalisoinnin ja analysoinnin kannalta. Fysikaalisesti ääni on määritelty mekaaniseksi aaltoliikkeeksi, joka vaatii välittäjäaineen edetäkseen [Pierce, 1989]. Ääni on näin määriteltynä väliaineessa tapahtuvien painevaihteluiden kautta havaittavissa olevaa mekaanista energiaa, joka etenee pitkittäisenä aaltoliikkeenä. Mekaanisen energian aiheuttamat painevaihtelut vaikuttavat paitsi väliaineeseen, myös väliaineen välityksellä ympäröiviin aineisiin, jolloin energia ja painevaihtelut voidaan mitata niiden edetessä väliaineesta toiseen.

Ääni subjektiivisena aistimuksena sinällään ei ole mitattavissa, vaikkakin kuulohavainnon fysiologinen vaikutus saattaisi olla mitattavissa. Äänen kuulemiseen liittyvät kuuloelinten lisäksi tajunta ja kuullun aistimuksen hermostollinen käsitteleminen. Täten satunnaisuutta louhittaessa ei ole mitenkään mielekästä puuttua varsinaisesti siihen, miltä ääni kuulijasta kuulostaa. Äänen ominaisuuksia ja sisältöä toki voidaan tarkastella, sillä niissä on ensinnäkin jotain universaalia, joka voi välittyä ideana useammalle kuulijalle, ja toisekseen ääni on olemassa myös ilman kuulijaa, joten sen ominaisuuksia voidaan tarkastella. On huomattava, että ainoastaan kuuloaistin omaavien elollisten olioiden voidaan varsinaisesti olettaa kuulevan. Koneet ja laitteet kykenevät vain mittaamaan olosuhteiden muutoksia (painevaihteluita ääntä mitattaessa). Näitä mittalaitteilla mitattuja väliaineen olosuhteiden aaltomaisia muutoksia kutsutaan tässä tutkielmassa äänisignaaleiksi. Jos puu kaatuu metsässä, eikä mikään laite ole siitä aiheutuvia painevaihteluita rekisteröimässä, ei ääntä tämän tutkielman määrittelemässä merkityksessä synny, vaikka äänen olemassaolo voitaisiin hyvin vahvasti olettaa.

Äänen sisällöllisiä merkityksiä tarkastellaan aina jostakin äänen ulkopuolisesta yhteydestä käsin. Puhe on fyysisesti vain tietynlaista äänisignaalia, jonka elollinen olento tulkitsee tietynlaiseksi mahdollisesti opittujen ja perittyjen taipumusten

takia. Mikäli äänen muodollisten piirteiden voidaan nähdä kuvaavan objektiivisesti ääntä, niin äänen sisällöllisten merkitysten kuvaamisen näkökulma on subjektiivinen.

Äänellä on varsinaisesti vain kaksi merkittävää ominaisuutta: ääniaallon *taajuus* (frequency) ja *voimakkuus* (amplitude). Kaksiulotteisena kuvaajana kuvattaessa taajuus on havainnollistettavissa ääniaallon pituutena ja voimakkuus korkeutena. Ääniaallon taajuudesta puhuttaessa käytetään käsitteitä matala ja korkea ääni, vaikka eräässä mielessä tarkempaa olisi puhua nopeasta ja hitaasta aallosta, sillä matala äänisignaali saa väliaineen värähtelemään hitaasti, kun taas korkea aalto värähtelee nopeasti. Ääniaalto etenee eteenpäin väliaineessa vakioisella nopeudella.

Taajuuden yksikkö on *hertsi* (Hz), joka ilmoittaa, kuinka monta kertaa ääni värähtelee yhden sekunnin aikana. Matalat äänet saavat aikaan hitaasti värähtelevän ja täten harvan ääniaallon, jossa aaltojen jaksot voivat olla pituudeltaan useita metrejä. Vastaavasti korkeiden äänien aiheuttamat aallot ovat nopeasti värähteleviä ja tiheitä: aallonpituus voi lähetä äärettömästi nollaa. Harvat ääniaallot menettävät energiaansa hitaasti ja voivat siksi kantaa kauas, kun puolestaan tiheät vaimenevat nopeasti edetessään äänilähteestä. Matalataajuisella signaalilla on ikään kuin enemmän massaa ja tämän massan tuomaa liike-energiaa, jolla se kykenee etenemään korkeataajuista pidemmälle. Äänen taajuus ja nopeus väliaineessa määrittävät *aallonpituuden* (wavelength). Äänen voimakkuuden, amplitudin, yksikkönä käytetään useinmiten *desibeliä* (db). Äänen voimakkuus on luonnollisesti myös suuri tekijä äänen etenemisessä ja vaimentumisessa. Mikäli taajuutta ajatellaan tykistä ammuttavan ammuksen massana, äänen voimakkuus on ammuksen lähtönopeus.

3.1. Äänisignaali

Äänisignaalin soveltamista satunnaisuuden lähteenä voidaan lähestyä kahdelta kannalta. Äänisignaalia voidaan tarkastella joko sisällöllisten piirteiden kautta muodostuvien merkitysten valossa tai signaalina ilman merkityksellisiä sisällölli-

siä piirteitä. Sisällöllisten piirteiden pohjalta muodostuva merkitys on tiedossa oleva viittaus johonkin reaalimaailmassa tapahtuneeseen ilmiöön, jonka aiheuttama ääni on tallennettu äänisignaaliin. Jos ilmiö on tiedossa ja sen voidaan olettaa aktivoituvan edes näennäissatunnaisesti, ilmiön äänisignaaliin antama jälki on käytettävissä satunnaisuuden louhintaan. Tällöin louhinta on suunnattava signaalin kuvaamien (merkityksellisten) ilmiöiden piirteisiin. Toinen mahdollisuus on louhia suoraan signaalista, jolloin signaalin kuvaaman ilmiön on oletettava olevan siinä määrin satunnainen, että signaalista saa tasan jakautuneita arvoja koko siltä arvojoukolta, joka tarvitaan signaalin kuvaamiseen digitaalisena.

Kohina (noise) on äänisignaalia, joka aiheutuu laitteiden poimimista sähkömagneettisista vaikutuksista. Äänenkäsittelyssä kohinaa pidetään yleisesti häiriönä ja haittana, mutta sen on todettu olevan potentiaalinen vahvan satunnaisuuden lähde [Haahr, 1999]. Periaatteessa ainoastaan kohinan antama äänisignaali voidaan tulkita sellaiseksi, että se ei sisällä sisällöllisiä merkityksiä sisältäviä säännöllisiä piirteitä, vaikka kohinan voidaan nähdä sisältävän ainakin sen piirteen, joka tekee kohinasta kohinaa. Mitenkään täysin varmaa kohinan satunnainen ja mielivaltainen luonne ei ole, sillä kohinan synnyttää aina jokin ilmiö, ja mitään ilmiöitä ei ole voitu – eikä välttämättä koskaan voida – osoittaa satunnaiseksi. On tyydyttävä riittävän satunnaisilta vaikuttaviin ilmiöihin. Tämä on selkeästi rajanveto, joskaan rajan sijainti ei ole mitenkään itsestään selvää. Mitä tahansa signaalia voidaan kuitenkin louhia välittämättä mahdollisista sisällöllisistä piirteistä, mutta louhinnan tehokkuus voi kärsiä huomattavastikin ei-optimaalisen menetelmän käytöstä.

Oikeastaan ei ole esteitä minkään digitaaliseen näytteen kuvaamiselle aaltomuodossa riippumatta siitä, onko se järkevää tai tarpeellista. Näin ollen digitalisoidun näytteen alkuperä korostuu merkittävästi. Aistittavissa oleva tai laitteellisesti mitattava ääni on myös keino kuvata signaalia, joten esimerkiksi kuvatiedostolla on tavallaan ilmentymä äänenä mikäli se ”soitetaan” sopivilla menetelmillä. Äänen perimmäisen lähteen ei näin ollen tarvitse kyetä itse tuottamaan ääniaaltoja, vaan jokin muu menetelmä voi muuntaa signaalin ääneksi, kuten sähkömagneettiset radioaallot muunnetaan ääniaalloiksi. Selkeä haitta tästä on äänen alkuperästä varmistumisen vaikeus (tai jopa mahdottomuus), jolloin on myös asetettava tiukat

vaatimukset ja varmistukset signaalin alkuperän tuntemiseksi. Yksi hyöty on mahdollisuus louhia myös muuta kuin äänisignaalia, mikäli tällainen tarve sattuu syntymään. Satunnaisuuden louhintaan liittyy vahvasti analogiseksi oletetun ilmiön digitalisoiminen käsittelyä ja tulossarjan muodostamista varten.

3.2. Digitalisointi

Digitalisoinnissa jatkuva analoginen signaali kuvataan numeeriseksi (eli digitaaliseksi) tietyn aikavälin mukaisesti otettujan *näytteiden* (samples) avulla. Yhden näytteen kuvaamiseen käytettyjen bittien määrää muuttamalla saadaan vaihdettua näytteen tarkkuutta. Näytteiden määrä (eli aikavälien tiheys), eli *näytteistämistaajuus*, määrittää koko muunnetun signaalin tarkkuuden alkuperäiseen analogiseen signaaliin nähden. Jotta signaalin alkuperäinen aaltomuoto saataisiin säilytettyä, näytteistämistaajuuden tulee olla vähintään kaksi kertaa niin suuri kuin signaalin suurin taajuus (ns. Nyquistin raja) [Watt, 2000, pp. 393-397]. Tämä suosii äänisignaalin merkityksellisestä sisällöstä louhintaa, mutta toisaalta suositeltua selvästi alhaisempi taajuus *näytteistämisessä* (sampling) voi rikkoa alkuperäisen aaltomuodon, jolloin signaali saattaa satunnaistua, mutta siitä ei oletusarvoisesti voi olla mitenkään varma.

Näytteistämistaajuuden ja näytearvon kuvaamiseen käytettävien bittien lukumäärän ohella on myös muita huomioon otettavia tekijöitä. Äänisignaalia mittaava laite – yleensä mikrofoni tai sen johdannainen – voi olla rajoittunut esimerkiksi äänisignaalin voimakkuuden osalta. Tällöin alkuperäinen äänisignaali voi kuvautua liian hiljaiseksi, jolloin hiljaisimmat näytteet voivat hukkua ja muutenkin erotustarkkuus saattaa olla heikkoa. Äänisignaali voi myös olla liian voimakasta mittalaitteelle, jolloin signaali kuvautuu *särkyneeksi* (distorted) ja täten myös häviää tietoa alkuperäisestä signaalista. On huomattava, että erittäin suurella todennäköisyydellä näytteistetty signaali eroaa alkuperäisestä jatkuvasta signaalista, sillä näytteistys ei tavoita näytekohtien välillä olevia signaalin arvoja. Kun näytteistetystä signaalista jälleen kootaan jatkuva signaali, ovat näytekohtien väliset arvot approksimaatioita. Mittalaitteen lisäksi digitalisointiin liittyy muitakin komponentteja, jotka osaltaan voivat vaikuttaa signaaliin. Jokainen komponentti tuo

hyvin mahdollisesti häiriöitä signaaliin – yleensä kohinan muodossa. Signaaliinhan vaikuttaa jo väliaineen painevaihteluiden muutoskin. Tavallisesti mikrofonin jälkeen ennen varsinaista digitalisointia signaali kulkee (välitysmedian, kuten johtojen tai radioaaltojen, lisäksi) erinäisten rajoittimien, vahvistimien ja suodattimien läpi. Rajoittimista, vahvistimissa ja suodattimista on se hyöty, että ne rajoittavat signaalin ylisuuria piikkejä ja voimistavat liian heikkoa signaalia. Haittaa tosin aiheutuu siitä, että edellämainitut käsittelyt voivat muuttaa alkuperäistä signaalia liikaa.

Ääniteknisestä näkökulmasta digitalisointi toisaalta parantaa äänen laatua, sillä näytteiden väliin jäävän signaalin approksimointi hävittää todellisia kohinoita ja häiriöitä. Toki näytteistämistaajuuden ollessa liian pieni häviää myös oleellista informaatiota. Liian suuri näytteistämistaajuus puolestaan poimii häiriöt signaaliin mukaan. Laadukkaaksi digitaaliseksi äänisignaaliksi voitaisiin sanoa riittävän, mutta ei liian, suurella näytteistämistaajuudella näytteistettyä ääntä. Äänen tekninen laadukkuus on eduksi silloin, kun satunnaisuuden louhinta kohdistuu sisällöllisiin piirteisiin, mutta louhinnan kohdistuessa signaalin arvoihin saattaa teknisesti laadukas ääni heikentää satunnaisuutta.

Digitalisointi voidaan myös nähdä satunnaisuuden louhinnaksi tai osaksi sitä, sen mukaan millä ehdoin louhintaa suoritetaan. Louhinta digitalisointina kohdistuu varmemmin analogiseen ilmiöön, sillä silloin louhintaa edeltää vähintään yksi vaihe vähemmän. Jokainen vaihe (esimerkiksi ilmanpaineen vaihtelun siirtyminen mikrofonin laitteistoon on yksi vaihe) periaatteessa avaa mahdollisuuden signaalin alkuperän hämärtymiselle. Käytännössä tämä on ongelma vasta, kun signaalia aletaan käsitellä digitaalisena.

Pelkällä analogisen äänisignaalin digitalisoinnilla tuotettujen tulossarjojen ei voida olettaa olevan satunnaisuuden kannalta vahvoja. Mikäli louhinta kohdistuu jo digitalisoituun signaaliin, moninaisemmat keinot ovat mahdollisia. Tällöin myös louhinnan tulos on yleensä vahvempaa. Ongelmana on, että digitaalisen signaalin analoginen alkuperä ei ole louhijalle mitenkään itsestään selvää. Louhijan kannalta on tällöin sama, louhitaanko digitalisoitua äänisignaalia tai vaikkapa digitaali-

seksi tuotettua ohjelmakoodia. Tuloksen vahvuuden kannalta tämä on kuitenkin kriittistä, ja se tulee ottaa huomioon järjestelmää toteutettaessa.

Signaalin aaltomuoto voi olla ennustettavuuden minimoinnin kannalta heikko piirre, jos louhinta perustuu suoraan signaalin digitalisoituihin arvoihin (tai mikäli digitalisointia on käytetty louhitana), joten aaltomuotoa tulee jollain tapaa murtaa. Toisaalta aaltomuoto jo sinänsäkin mahdollistaa monipuolisempia ja monimutkaisempia tapoja satunnaisuuden louhintaan. Tällöin satunnaisuus louhitaan sellaisista aaltomuodon vaihteluista ja ominaisuuksista, jotka voidaan osoittaa tai ainakin riittävän vahvasti olettaa satunnaisiksi. Näytteistämistaajuus on myös merkittävässä asemassa aaltomuotoisen signaalin digitaalisen approksimaation ennustettavuudessa.

Eagini ja Buce [1999] käsittelevät *valkoisen kohinan* (white noise) käyttöä satunnaisuuden lähteenä ja esittävät, että analoginen lähdesignaali tulisi muuntaa binääriseen muotoon ennen signaalin digitaalista näytteistämistä. Menetelmä kuulostaa nurinkuriselta ja oudolta: analogisen signaalin muuttaminen binääriseksi voi nimittäin kuulostaa digitalisoinnilta. Menetelmä kuitenkin perustuu näytteistämisessä käytettävien kaistanleveysrajoitteiden tuomien painotusten ja signaalimuunnosten poistumiseen, kun analoginen signaali on aluksi peitetty binääriseksi. Analogista signaalia ei vielä siis näytteistetä, vaan signaalin arvot tulkitaan valitun rajan mukaan nollina tai ykkösinä. Menetelmään kuuluu myös näytteistämistaajuuden pitäminen alhaisena, jolloin korrelaatio bittien välillä pysyy pienenä. Tällöin kohinan alkuperäinen aaltomuoto murtuu ja digitaalisen signaalin ennustaminen aiemmista signaalin arvoista saattaa vaikeutua huomattavasti.

Mikäli satunnaisuuden louhinta perustuu (aaltomuotoisen) signaalin aaltojen ominaisuuksiin, kuten pituuksiin tai voimakkuuksiin, ei periaatteessa aseteta rajoitusta signaalin sisällön merkitykselle. Signaali voi tällöin sisältää mitä tahansa ”ääntä” ja olla lähtöisin mistä tahansa äänilähteestä. Kun louhinta puolestaan perustetaan signaalin sisällön merkitykselle, signaalin lähdekin on luonnollisesti rajatumpi. Yleensäkin pätee, että mitä tarkemmin signaalin lähde on rajattu ja täten myös louhinta tarkemmin juuri sen tyyliseen signaaliin suunnattu, sitä varmempia voi-

daan olla louhinnan tuloksista. Tähän toki vaikuttaa itse louhija, mutta jos varmemmin tiedetään signaalin lähteen ominaisuuksia, voidaan paremmin arvottaa signaalin sopivuutta. Varmuus luonnollisesti kasvaa, jos menetelmää on tutkittu ja kehitetty.

Monrose ja muut [2001, 2002] ovat tutkineet puheääntä satunnaisuuden lähteenä salausavainten muodostamisessa. Heidän lähtökohtana on ollut kehittää menetelmä, joka kykenee sekä tunnistamaan käyttäjän että luomaan salausavaimen tämän sanomasta salasanasta. He ovat valinneet puhutun salasanan tutkimisen muun muassa siksi, että verrattuna muihin biometrisiin tunnistusmenetelmiin käytettävissä olevien salasanojen määrä on huomattavan suuri ja puheäänen on todettu olevan luotettava menetelmä käyttäjien tunnistamisessa. Satunnaisuus puhutusta salasanasta louhitaan sekä käyttäjän tavasta sanoa salasana että itse puhutun salasanan puhetunnistuksellisista ominaisuuksista.

3.3. Äänen vaikutus aaltomuotoon

Kun äänisignaalia louhitaan puhtaasti näytteiden arvoihin perustuvilla menetelmillä, tulee ongelmalliseksi löytää sopiva äänilähde, joka tuottaa tasaisesti jakautuneita näytteistysarvoja. Äänisignaalin tulisi olla keskimäärin tasaisen voimakasta, jossa piikkien hajonta ei olisi liian suuri. Puheääni sinällään tuottaa hyvin epätasaista signaalia, kuten myös yleensä lähellä ja kaukana mikrofonista olevat ympäristön ”satunnaiset” äänilähteet (lähellä mikrofonia oleva äänilähde tuottaa ilmetessään tehokkaamman ääni-impulsin kuin kauempana oleva äänilähde, jos äänilähteet tuottavat yhtä voimakasta ääntä). Tulokseksi syntyy siis signaali, joka sisältää paljon hiljaisia hetkiä suhteessa koko signaalin kestoon.

Mikrotietokoneissa olevien analogista tosimaailman ilmiötä mittaavien laitteiden sisääntulojen (muun muassa ääni ja kuva) kautta voitaisiin tiettyjen olosuhteiden vallitessa melkoisen helposti ja varmasti louhia satunnaisuutta. Jos sisääntuloihin ei ole kytketty mitään laitetta ja sisääntulon signaalia voitaisiin vahvistaa riittävästi, olisi sisääntulon antama signaali periaatteessa lämpökohinaa. Näytteistetty lämpökohina on oleellisesti satunnaista, joskin pariin asiaan on kiinnitettävä huo-

miota. Ensinnäkin kohinan näytteistys mikrotietokoneelle on *painottunutta* (skewed), joten näytettä täytyy *tasoittaa* (de-skew). Toisekseen täytyy olla varma, että laitteisto on sen verran hyvässä kunnossa, etteivät muut (häiriö)tekijät pääse vaikuttamaan signaalin sisääntuloon ja näytteistykseen. [Eastlake *et al.*, 1994]

Kun signaalia vahvistetaan, mukaan tulee todennäköisesti ei-toivottuja häiriösignaaleja. Oikeanlaiset suodattimet signaalin lähteen ja digitalisoivan laitteen välissä ovat miltei paras tapa karsia häiriöitä pois. Samalla saadaan varmistettua, että signaali on kaistanleveyksiltään ja taajuuksiltaan digitalisoijalle sopivaa. Signaali saadaan siis vakaaksi, jolloin sen tutkiminen ja digitalisointi on helpompaa, varmempaa ja turvallisempaa. [Eagini and Buce, 1999]

Lämpökohinan näytteistämisessä mikrotietokoneen mikrofoniportista tai muusta signaalisisääntulosta ei käytännössä ole kuitenkaan aivan yksinkertaista. Saatu satunnaisuus täytyy mieltää korkeintaan keskinkertaiseksi. Lämpökohina elektronisissa komponenteissa perustuu nimittäin (sähkömagneettisen) säteilyn aiheuttamaan häiriöön. Ongelmana on, että sähkömagneettista säteilyä on kaikkialla, missä sähkö virtaa. Mikrotietokoneen muiden komponenttien on oletettava tuottavan häiriötä mikrofonisisääntuloon, eikä tämä häiriö välttämättä ole mitenkään ”satunnaista”. Tietokoneesta irrallinen lämpökohinan digitalisoija puolestaan voi olla hyvinkin ongelmallinen suojauksen suhteen. Miten tehdä sellainen suojaus, joka laskee läpi tarkkailtavan kohinan, mutta sulkee pois muut häiriötä häiritsevät häiriöt; puhumattakaan suojauksesta tarkoituksellista häiriöntuottoa vastaan? Digitointivaiheessa täytyy ohjelmallisesti tarkkailla, että signaali ei ole epäilyttävää. Myös signaalin ajaminen soveltuvien suodattimien läpi poistaa ei-toivottuja ilmiöitä. Ylipäätään louhittua satunnaisuutta tulisi analysoida ja tarkkailla mahdollisten vikojen ja ei-toivottujen vaikutusten paljastamiseksi. Mitenkään triviaalia signaalin tarkkaileminen algoritmisesti ei tietenkään ole.

Eräs paljon tutkittu ja kohtuullisen hyväksi havaittu lähde satunnaiselle sähkömagneettiselle kohinalle on ”tyhjillä” radiotaajuuksilla esiintyvä taustasäteilystä johtuva kohina. Random.org-nimistä sivustoa ylläpitävän tahon käyttämä satunnaislukugeneraattori hyödyntää siemenlähteeksi nimenomaan sellaisia radiotaa-

juuksia, joilla ei havaita olevan mitään taustahälyä kummempia lähetyksiä. Kohinan kanssa samassa käsiteyhteydessä voidaan puhua myös yleisemmin taustahälystä. Taustahälyn (mahdollinen) voima satunnaisuuden lähteenä perustuu satunnaisesti toistuviin ääni-ilmiöihin, jotka eivät ole riippuvaisia edellisestä ilmentymästään tai muista ääni-ilmentymistä. Laboratorion, toimiston, ostoskeskuksen, rautatieaseman tai vilkkaasti liikennöidyn kadun ”sisältämien” äänten tallennus yhdeksi äänisignaaliksi voi periaatteessa toimia sangen hyvänä satunnaisuuden lähteenä. Hedelmällisintä tällaisessa tapauksessa lienee tutkia äänisignaalia sekä rakenteellisesti että sisällöllisesti. Pelkkään rakenteelliseen kokonaisuuteen pohjaava soveltamistapa on herkkä sekoittamaan päällekkäisiä ja yhtä voimakkaita äänilähteitä. Tiukan sisällöllinen tarkastelutapa voi puolestaan olla tehoton tai hidas. [Haahr, 1999; Schneier, 1996].

Sisällöllinen tarkastelutapa taustahälylle on sangen hyvä, kunhan äänisignaalista kyetään erottelemaan mielekäs ilmiö kyllin tarkasti eikä tiukkoja aikarajoja ole. Vilkkaan kadun tapauksessa voisi esimerkiksi olla mielekästä mitata aikavälejä ohiajavien, tietynmerkkisten yksityisautojen välillä. Ongelmia on lähinnä kaksi. Ensinnäkin ilmiön erottelu äänisignaalista voi olla melkoisen hankalaa johtuen muun taustahälyn aiheuttamista häiriöstä. Valittu ilmiö voi lisäksi olla äänisignaalisesti hyvinkin identtistä muiden ilmiöiden kanssa. Toisekseen toistuva ilmiö pitäisi osata valita täysin oikein. Ilmiön tulee olla ennalta-arvaamattomasti toistuva, mutta ehdottomasti toistuva. Valitun yksityisauton merkin tulisi olla mahdollisimman yleinen, eikä leimautunut tietyn ryhmän käyttöön. Näin voitaisiin varmistaa, ettei kyseisten autojen liikkeillä ole muista autoista poikkeavaa aikataulua. Esimerkiksi linja-autot tapaavat ajaa suhteellisen aikataululleen, jolloin vuorokaudenajan perusteella voidaan melko luotettavasti haarukoida mahdolliset intervallit peräkkäisten autojen välillä.

3.4. Ääni ja sen louhinta tässä tutkielmassa

Tätä tutkielmaa varten tehdyissä kokeissa tuotettiin kertaluonteisesti satunnaislukugeneraattoreissa käytettävää satunnaisuutta. Uudelleentuotettavuutta ei millään tavalla huomiotu satunnaisuuden lähteitä ja käsittelymenetelmiä valittaessa, toisin

kuin esimerkiksi Monrosen ja kumppaneiden [2001] tutkimuksessa on tehty. Lähtökohtana oli käyttää äänilähteen tuottamaa satunnaisuutta aitona siemenlukuna, johon ei oleteta voitavan sitä luodessa vaikuttaa. Tällöin louhitun satunnaisuuden jatkosoveltaminen kohdistuisi vain sellaisiin sovelluksiin, joille riittää vain suora syöte louhijalta.

Lähdekirjallisuuden perusteella radiokohinan oletettiin olevan tutkielman kokeiden kannalta ylivoimaisesti vahvin satunnaisen äänen muoto, kuten edellä on jo todettukin. Radiokohinan louhinnassa sisällöllisillä merkityksillä ei suoranaisesti ole vaikutusta, paitsi siten, että mikäli kohina sisältää joitain merkityksellisiä hahmoja, on se huonolaatuista. Kohinan todentaminen aidoksi kohinaksi on sangen vaikea, ellei jopa mahdoton ongelma.

Kohina on siis ääntä, joka ei sisällä mitään sisällöllistä merkitystä. Tämä ongelma on osin identtinen täydellisen satunnaisuuden ongelman kanssa. Määritelmän mukaan kohina ei saisi sisältää sisällöllisiä merkityksiä, mutta tämän todistaminen on yhtä mahdotonta kuin yksittäisen ja kompleksisesti vahvan sarjan todistaminen tilastollisesti satunnaiseksi. Jos kohina on sähkömagneettisista ilmiöistä lähtöisin, niin se voi periaatteessa olla kyseiselle ilmiölle merkityksellistä. Kun ilmiön oppii tuntemaan, radiokohinankin merkityksen voi ymmärtää. Kohina ei myöskään ole tilastollisesti satunnaista, jos äänilähdealue on koko äänien kirjo. Kohinan muodostamasta alajoukosta nimittäin karsiutuu välittömästi pois kaikki merkitykselliset äänet. Jälleen on tyytyminen riittävään tarkkuuteen ja varmuuteen.

Ylipäätäänkin on ongelmallista todistaa joidenkin merkityksellisten ilmiöiden olemassaoloa äänisignaalissa. Korvakuulolla tämä usein onnistuu, sillä silloin ilmiölle syntyy merkitys. Valitettavasti korvakuulon ongelmat ovat ilmiöiden toistumisen tarkan mittauksen ongelmat: tie korvasta ajastimeen on ajallisestikin liian pitkä. Kuulija voi myös erehtyä. Kätevää olisi hoitaa tällaiset tehtävät automaattisesti. Silloin ongelmana on ilmiöiden tunnistaminen ohjelmallisesti. Pitäisi kyetä rakentamaan ja ohjelmoimaan järjestelmä, joka ilmiön aiheuttaman äänen mittaamisen perusteella kykenisi vieläpä tunnistamaan oikean ilmiön. Se puolestaan

vaatii jonkinlaisen tietoisuuden tai vähintäänkin ymmärtämisen liittämistä järjestelmään.

Puheentunnistus on sovellusalue, jossa äänen sisältämää merkityksellistä sisältöä pyritään mahdollisimman luotettavasti tunnistamaan. Puheentunnistusjärjestelmät toimivat tavallaan tietyssä ikkunassa signaalin piirteiden suhteen, jolloin samaan merkitysyhteyteen voidaan liittää useita samankaltaisia signaaleja, kuten pitääkin. Tämä väljyys mahdollistaa satunnaisuuden louhinnan [Monrose *et al.*, 2001]. Väljyyttä on osaltaan aiheuttamassa myös kohina, mikä puolestaan voi antaa tilaa myös äänen ulkoisiin piirteisiin perustuvalle louhinnalle.

Radiokohina äänitettynä sellaiselta taajuudelta, jolla ei korvakuuloisesti arvioiden ollut ”keinotekoista” lähetystä, päätettiin siis kelpuuttaa yhdeksi satunnaisuuden lähteeksi tässä tutkielmassa tehdyissä kokeissa.

Kohinaa löyhemmin määritellyn taustahälyn käyttäminen avaa periaatteessa mahdollisuudet tutkia signaalia sekä ulkoisten piirteiden että sisällöllisten merkitysten valossa. Sisällölliset merkitykset syntyvät taustahälyn seassa esiintyväksi tiedetyn ilmiön laadusta ja käyttäytymisestä. Aiemmin mainitun kaltaisessa vilkkaasti liikennöidyn kadun äänimaailman tarkkailussa voitaisiin kiinnittää huomiota vaikkapa autojen äänimerkkeihin tai jarrujen kirskuntaan. Pääasia, että valittu ilmiö kyetään luotettavasti taustahälystä poimimaan. Toisaalta voidaan myös luottaa taustahälyn muodostavien ilmiöiden yhdessä luovan tallentimen näkökulmasta mielivaltaista äänimaailmaa, jolloin satunnaisuutta louhittaessa ei huomioida mitään yksittäistä ilmiötä vaan katsotaan saatua äänisignaalia kokonaisuudessaan ulkoisten piirteiden pohjalta.

Taustahälyn voimakkuus satunnaisuuden lähteenä syntyy, kun taustahälyn seasta poimittavan ilmiön voi valita helposti. Varsinkin, jos louhinta keskitetään signaalin sisällöllisiin merkityksiin. Toisaalta valinnan vapauden tuoma varmuus on hyvin näennäistä, sillä vaikka ilmiön valintaan voisi satunnaisuuden louhija vaikuttaakin, ei heikkojen oletuksien pohjalle voi edelleenkään rakentaa luotettavaa

järjestelmää. Vastuu siirtyy radiokohinaan verrattuna vieläkin enemmän lähteen valitsijalle, jolla on lisätaakkanaan hyväksyttävien tekosyiden vähäisempi määrä.

Taustahälystä poimittavien ilmiöiden esiintymisen satunnaisuudesta on oltava riittävä varmuus, ja siltikin siihen voi keinotekoisesti vaikuttaa. Ihanteellisimmillaanhan ilmiöt syntyvät täysin riippumatta louhintajärjestelmästä. Mikäli ilmiöön kyetään vaikuttamaan, ei (kryptograafisesti vahvalle) satunnaisuudelle ole takeita.

Ulkoisten piirteiden valossa taustahäly ei ole aivan ongelmatonta. Radiokohinan kaltainen kohinasignaali vaihtelee laajemmin ja tasaisemmin koko signaali-ikkunan arvoalueella, kun taustahäly puolestaan usein jakautuu epätasaisemmin. Radiokohinasignaalin voimakkuutta on helpompi säädellä siten, että signaalin leikkautuminen ei muodostu ongelmaksi. Taustahälysignaali sisältää helposti hiljaisia hetkiä sekä yllättäviä voimakkaampia signaalin jaksoja. Tietenkään yllättävyys ei ole sinänsä haitaksi satunnaisuuden kannalta, mutta mikäli liian voimakkaisiin signaaliryöppyihin ei ole riittävästi varauduttu, voi lopullinen taltioitu signaali olla särkynyt, eli äänisignaali on leikkautunut joillain osin voimakkuusrajojen yli. Taustahäly vaatii siis helposti signaalin normalisointia koko ikkunan alueelle ennen käsittelyä, jos käsittely keskittyy yksittäisten näytearvojen vaihteluun.

Luonnollisesti tiettyjen ilmiöiden tunnistaminen taustahälystä on teknisesti haastava ongelma. Mikäli ilmiö tuottaisi signaalin tallentimen näkökulmasta täysin identtisiä signaaleja, olisi siltikin tarpeen kyetä erottamaan aiottu kyseinen ilmiö muiden ilmiöiden aiheuttamasta ”häiriöstä” huolimatta. Ongelma syvenee lisää, jos ilmiö ei tuota identtisiä signaaleja. Tässä tutkielmassa satunnaisuutta ei lähdetty ensisijaisesti hakemaan taustahälyn tietyistä ilmiöistä vaan keskityttiin tarkastelemaan taustahälyä ulkoisten piirteiden valossa.

Satunnaisuus louhittuna puheesta (tarkemmin siis puheesta koostuvasta äänisignaalista) on intuitiivisesti lähtöisin signaalin sisällöllisistä merkityksistä. Juuri tähän perustuu Monrosen ja kumppaneiden [2001] työ. Puhesignaalin ulkoisten piirteiden ominaisuudet ovat myös sangen kiinnostavia. Puheentunnistuksen ja täten myös puheen (tiettyjen) elementtien kautta tapahtuvan satunnaisuuden lou-

hinnan kannalta sisällöllisten piirteiden tulisi olla mahdollisimman vakaita. Jos pitäydytään liian tarkasti puheen sisällössä, heikkoutena voi tulla esiin kielen ominaisuudet. Kun puheesta voidaan tunnistaa pienehkö määrä merkityksellisiä variaatioita verrattuna koko äänisignaaliin, on kyseisten variaatioiden tuottaminen kielen antaman frekvenssitiedon perusteella melko mutkatonta (olettaen, että variaatioiden tunnistamiseen käytetty menetelmä toimii täysin ideaalisella tavalla, eli edellä esitetty väite pätee teoriassa), ja täten menetelmä on murrettavissa.

Puheessa voisi olettaa olevan ulkomuodollista satunnaisuutta melko runsaasti, sillä sen lisäksi, että eri puhujat tuottavat ulkoisesti vivahteikkaan erilaista äänisignaalia, sama puhuja todennäköisesti tuottaa identtisen äänisignaalin sijaan enemmän tai useimminkin vähemmän erilaista ääntä. Oman variaationsa tähän tuovat äänityksen tekniikka ja tarkkuus. Taustahälyn ja radiokohinan on tässä tutkimuksessa silti oletettu sisältävän huomattavasti enemmän potentiaalia ulkomuodolliseen satunnaisuuteen. Saman puhujan toistama ja sisällöltään samanlainen puhe on laajemmilta muodoiltaan kuitenkin samanpiirteistä, jolloin lokaalimmatkin kohdat ovat taipuvaisia olemaan samankaltaisia.

4. Koejärjestelyt ja kokeet

Tutkielman empiirisessä osiossa osoitetaan äänen lähteen merkitys äänisignaalin aaltomaisuutta hyödyntävien ja hyödyntämättömien louhintamenetelmien kannalta. Joidenkin menetelmien soveltaminen jätetään tarkoituksellisesti pinnalliseksi, sillä syvällisempi soveltaminen ja analysointi muodostavat todella laajan kokonaisuuden.

Kokeisiin liittyy kaksi osaa: louhinta ja analysointi. Louhintaosiossa ajetaan syötetiedostot erilaisten louhijoiden läpi, ja näin synnytetään tulostiedostot, joihin analysointiosio pureutuu. Kokeilla kartoitetaan sekä äänen lähteen ja muodon merkitystä että louhijan vaikutusta eri lähteistä saatujen ja erimuotoisten äänten sisältämään satunnaisuuteen. Syötetiedostojen lähteet on valittu siten, että niiden voidaan olettaa tuottavan laadullisesti ja määrällisesti eri vahvuista satunnaisuutta. Louhintamenetelmiksi on tietoisesti valittu heikompia menetelmiä, jotta vahvempien menetelmien tehoa ja mielekkyyttä voitaisiin verrata. Analysoinnissa käytettäviksi menetelmiksi sen sijaan on pyritty toteuttamaan useita osaltaan vahvoiksi todettuja menetelmiä, sillä ei ole tiedossa mitään yksittäistä menetelmää, joka kykenisi satunnaisuuden aukottomasti todentamaan. Analysointimenetelmät eivät ole tässä tutkielmassa varsinaisesti tarkastelun kohteena, ja sen takia mitään heikoksi oletettua analyysia ei ole tietoisesti otettu mukaan verrokiksi.

Eri lähteistä kerättyjen syötetiedostojen ollessa käytössä on huomioitava, että mitään lähdettä ei voi aukottomasti sanoa toista heikommaksi. Satunnaisuus saattaa vain olla eri tavalla nähtävissä. Tutkielmaan valittujen louhintamenetelmien voidaan kuitenkin havaita louhivan satunnaisuutta samankaltaisilla tavoilla, jolloin tietyt lähteet kykenevät tuottamaan satunnaisempia sarjoja kuin toiset. Tämä on tietoinen rajaus, jonka tekemättä jättäminen olisi laajentanut tutkielmaa huomattavasti.

Satunnaisuuden analysoiminen liki täydellisesti vaatii useiden mittareiden antamien tulosten vertaamista. Samoin lähteen sisältämän (siis syötteestä louhittavissa

olevan) satunnaisuuden määrän selvittämiseksi lähteen tuotoksia tulisi louhia mahdollisimman monella eri tavalla. Ongelmaksi koituukin kaikkien mahdollisten satunnaisuuden ilmenemismuotojen löytäminen yhdistettynä siihen, että satunnaisuuskaan ei ole täysin yksiselitteisesti määriteltävissä. Mitään yksimielisesti yleispätevää menetelmää satunnaisuuden louhimiseen digitaalisesta datasta ei ole löydetty, vaikka useita vahvoja menetelmiä on esitetty [Eastlake *et al.*, 1994; Juels *et al.*, 2000; Mitzenmacher, 2001; Ritter, 1991; Schneier, 1996].

4.1. Louhinta

Agnew [1998] esittää kolme ehtoa kryptograafisesti vahvalle satunnaisuuden louhijalle ja lähteelle. Lähteen tulee olla yhteensopiva louhijan kanssa. Lähteen ja louhijan tulee olla immuuneja tarkkailulle tai ulkopuolisille vaikutuksille. Lähteen tuotosten täytyy olla vakaita (ts. lähteen tulee antaa satunnaisuutta ilman epämääräisiä taukoja) ja niiden täytyy täyttää satunnaisuusvaatimukset.

Lähteen yhteensopivuus louhijan kanssa on ollut helpoin ehto toteuttaa tässä tutkimuksessa, sillä louhijat ovat tietokoneella toteutettuja algoritmeja ja lähteet ovat olleet tietokoneelle tuotettua äänidataa. Käytännössä lähteen ja louhijan yhteensopivuus toteutetaan sopivalla muuntimella. Ellei näin olisi, täytyisi louhijan itsenäisesti kyetä ottamaan syötettä lähteestä. Toisin sanoen louhijan tulisi olla osa lähdettä, eli louhinnan ohessa tuottaa itselleen louhittavaa materiaalia.

Immuunius tarkkailulle on tämän tutkimuksen ulkopuolella, sillä voidaan turvallisesti olettaa, että tämän tutkimuksen tapauksessa niin sanottua vihamielistä tahoa, joka haluaa vaikuttaa tutkimuksen tuloksiin, ei ole. Tulosten analysointi olisi myös huomattavan hankalaa, mikäli minkäänlaista tarkkailua ei sallittaisi.

Lähteiden luotettavuuden ja suojaamisen toteuttamiseen ei tässä oteta kantaa. Aineisto on silti pyritty keräämään mahdollisimman luotettavasti käytettävissä olevilla resursseilla. Lähtökohtana on ollut, että mikäli lähteessä ylipäätään on satunnaisuutta, sitä saadaan riittävissä määrin esiin käytetyillä menetelmillä ja laitteis-

toilla. Sähkömagneettisen säteilyn ja muiden signaalia heikentävien ilmiöiden oletetaan vaikuttaneen aineistoihin suojauksien vähäisyyden vuoksi.

Lähteiden antamien signaalien vakaus on kyetty tuottamaan, sillä jokainen näyte koostuu ajallisestikin yhtenäisestä signaalista. Lähteiden satunnaisuusvaatimuksia ja tuotetun satunnaisuuden vakautta tässä tutkimuksessa tutkitaan, joten niiden olettaminen voimassaoleviksi on paradoksaalista.

Agnewin esittämät ehdot kryptografisesti vahvalle satunnaisuusympäristölle on hyvä ottaa huomioon myös tässä tutkimuksessa. Ehdot toimivat kuitenkin enemmänkin päämäärinä kuin lähtökohtina. Kryptografinen vahvuus ei lisäksi ole oletusarvoinen päämäärä, vaan mahdollisesti toteutuessaan ennemminkin mieluisa lisä.

4.2. Louhittava aineisto

Kun Monrosen ja kumppaneiden [2001, 2002] tutkimaa menetelmää tarkastelee tämän tutkielman näkökannalta, on satunnaisuuden käsitteessä huomattavissa selkeä ero. Tätä tutkielmaa varten tehdyissä kokeissa toistettavuuteen suhtaudutaan eri tavalla. Monrosen menetelmässä toistettavuuden tärkeys korostuu siinä, että sama käyttäjä kykenisi saman salasanan sanomalla tuottamaan saman avaimen. Oletuksena tässä tutkimuksessa on, että täsmälleen samanlainen lähdetiedosto tuottaa samanlaisen tuloksen, mutta täsmälleen samanlaisen lähdetiedoston tuottaminen oletetaan hyvin vaikeaksi ja epätodennäköiseksi.

Ääninäytteiden tuottamiseen valittiin kirjallisuuden pohjalta kolme lähdettä. Tarkoitus oli valita lähteet siten, että louhitun satunnaisuuden määrä ja laatu olisivat runsaasti vaihtelevia. Vahvimmaksi lähteeksi oletettiin radiokohinaa, jota tallennettiin *tyhjälle radiotaajuudelle* säädetystä radiosta. Tyhjällä radiotaajuudella ei ole tarkoituksellisesti lähetettyä signaalia. Taustahälyä äänitettiin yksityisasunnon huoneesta eloisana päivänä. Puhutun äänen näyte äänitettiin koehenkilön lausuessa tiettyjä sanoja lukuisia kertoja.

4.2.1. Ääninäytteiden tallentaminen

Ääninäytteiden tallentamisessa tulisi käyttää matalalla näytteistämistaajuudella näytteistettyä ääntä, sillä tällöin bittien välinen korrelaatio on keskimäärin pienempää [Eagini and Buce, 1999]. Analogista ääntä digitoidessa on kuitenkin myös huomioitava Nyquistin raja, jotta signaalin *laskostumista* (aliasing) ei pääsisi tapahtumaan [Watt, 2000, pp. 393-397]. Nyquistin rajan ylittyminen aiheuttaa vääristymiä signaalin taajuudessa. Äänitettävän signaalin päästäminen sopivan suodattimen läpi on yksi keino tasoittaa analogisen signaalin taajuuksia siten, että laskostumista ei pääse syntymään [Monrose *et al.*, 2002], mutta toisaalta analogisen signaalin rajoittaminen ennen digitalisointia todennäköisesti hukkaa dataa ja voi vahvistaa ennustettavuutta [Eagini and Buce, 1999].

Jokaisesta äänilähteestä tallennetut näytteet otettiin samoilla parametreilla. Näytteiden pituudeksi määriteltiin viisi sekuntia ja näytteistystaajuudeksi 22 050 näytettä sekunnissa. Tallennettu ääni oli yksikanavaista, eli monoääntä. Näytteiden tarkkuudeksi valittiin kahdeksan bittiä, eli yksi tavu kuvaa yhtä näytteen arvoa. Minkäänlaista pakkausta ei käytetty, vaan ääninäytteet tallennettiin WAV-tiedostomuotoon.

Suurempi tarkkuus tai näytteenottotaajuus olisi tarkemman näytteistyksen lisäksi myös sisältänyt enemmän häiriöitä. Kahdeksan bitin tarkkuus näytettä kohden on myös helpommin käsiteltävissä louhinnassa, sillä tällöin näytteen arvo nähdään suoraan luetusta tavusta eikä louhijan tarvitse enää tehdä laskutoimituksia tavujen yhdistämisessä näytteen arvoksi. Valitut parametrit pitävät myös tallennetun näytteen koon siedettävänä.

Radiokohina ja taustahäly tallennettiin ensin useiksi kolmenkymmenen minuutin mittaisiksi tiedostoiksi: radiokohinaa tallennettiin kaikkiaan kuusi tuntia ja taustahälyä viisi. Näistä tiedostoista poistettiin alussa olevat *tunnistetiedot* (header). Tunnistetiedot ovat louhinnan kannalta turhia, sillä tiedoston parametrit ovat alkujaan jo tiedossa. Edistyneempi louhintajärjestelmä voisi käyttää tunnistetietoja asettaakseen louhijan toimimaan oikeilla parametreilla. Tunnistetietojen poiston

jälkeen tiedostot pilkottiin viiden sekunnin osiin. Jako menee tasan, joten yhdellä puolen tunnin tallennusistunnolla saatiin talteen 360 näytettä.

Puhutut näytteet tuotettiin siten, että koehenkilö lausui viiden sekunnin ikkunaan oman nimensä selkeästi ääntäen. Koehenkilö pyrki lausumaan jokaisen näytteen samanlaisella rytmityksellä, äänenpainolla ja äänellä. Tallennukset tehtiin viidentoista minuutin jakoissa, jotta koehenkilöllä olisi pieni tauko aikaa virkistää ääntään ja levätä muutenkin. Kun viidentoista minuutin mittainen äänitiedosto oli muodostettu, pilkottiin se tasan viiden sekunnin mittaisten ikkunoiden mukaisesti. Ikkunoinnissa käytettiin visuaalista ajastinta, jonka avulla koehenkilö näki lausua näytteet ikkunaan.

4.2.2. Ääninäytteiden käsittely

Ääninäytteet tallennettiin ilman välivahvistimia suoraan mikrofonista tai linjakulkujen kautta äänikortille. Äänikortti ja tallennusohjelma eivät lisänneet signaaliin mitään tarkoituksellista keinotekoisuutta – luonnollisesti jokainen vaihe tuo oman häiriönsä signaaliin. Tästä seurasi myös äänisignaalien voimakkuuksien epätasaisuus.

8-bittinen WAV-tiedostomuoto kuvaa äänisignaalia esittämällä tavuarvoina näytekohdissa olevan signaalin voimakkuuden siten, että yksi tavu esittää yhtä näytekohtaa. Signaalin aallon niin sanottu nolla-akseli kuvataan arvolla 127. Suuri osa äänisignaalista kuvautuu lähelle nolla-akselia, jolloin ääriarvot voivat jäädä tyystin käyttämättä. Tämä on ongelmallista sikäli, että äänitiedostojen toivottaisiin ulkoisiin piirteisiin perustuvassa louhinnassa olevan arvoiltaan mahdollisimman tasaisesti jakautuneita.

Jakauman tasaamiseksi ääninäytteet käsiteltiin ennen lopulliseen kokoon pilkkomista äänenvoimakkuuden säätimellä. Koska radiokohina ja taustahäly tuottivat tasaista signaalia, voitiin äänenvoimakkuus säätää siten, että voimakkuutta nostettiin, kunnes signaalin voimakkain kohta osui arvoalueen ylärajalle. Tällöin signaalista ei leikkautunut mitään pois, eikä signaali keinotekoisesti muuttunut miltään

osin. Puhutuille näytteille tehtiin sama käsittely. Tulokset tarkistettiin vielä silmämääräisesti graafisen äänenkäsittelyohjelman avulla.

Äänitiedostot nimettiin lähteen ja äänitysjakson perusteella juoksevasti numeroiden. Näytetiedostot sijoitettiin tunnistetiedoista siistimisen ja oikeaan mittaan pilkkomisen jälkeen lähdekohtaisiin hakemistoihin käsittelyn helpottamiseksi ja sekaannusten välttämiseksi.

4.3. Louhintamenetelmät

Louhintamenetelmät toteutettiin yksi kerrallaan Java-kielellä. Lähtökohtaisesti jokaisen louhintamenetelmän algoritmit ohjelmoitiin ensin erillään toisistaan. Kun kaikki algoritmit toimivat virheettömästi, yhdistettiin menetelmät yhden yliohjelman alle, jotta kokeiden louhintaosio voitaisiin suorittaa nopeasti ja yksinkertaisesti. Louhinta-algoritmit saatiin asianmukaisista lähdemateriaaleista, ja niiden ohjelmallisessa toteuttamisessa käytettiin Goodrichin ja Tamassian [2001] sekä Kolehmaisten [2001] esittämiä ohjelmointitekniikoita ja tietorakenteita. Menetelmiä ei kuitenkaan optimoitu toimimaan rinnakkain, vaan menetelmät ajetaan peräkkäin.

Kokeet ajettiin ketjuttamalla menetelmät sisältävää yliohjelmaa, jolle ketjutuksen eri vaiheissa syötettiin hakemistot, joissa lähdetiedostot sijaitsivat. Yliohjelma kykeni siis louhimaan yhden hakemiston sisältämät tiedostot kerrallaan kaikilla sisältämillään menetelmillä.

Kaikki valitut louhintamenetelmät eivät ole täysin itsenäisiä, vaan osa hyödyntää pariteetin irrotusta. Von Neumann -menetelmä parantaa pariteetin irrotusta, ja Mitzenmacher-menetelmä hyödyntää tavusta vain merkitsevän bitin – eli pariteettibitin. Loput menetelmistä käyttävät kaikkia tavun bittejä louhinnassaan.

Äänitiedoston tavujen voidaan olettaa keskittyneen keskiarvojen ympärille, jolloin ainoastaan vähiten merkitsevä bitti – eli pariteettibitti – on satunnaisuudeltaa suurin. Pariteettibittien irrottaminen uusien tavujen muodostamiseksi tasoittaa ja-

kaumaa. Bittejä käsittelevien menetelmien on täten mielekkäämpää käsitellä lähdetiedoston tavuja yhtenä kuin kahdeksana bittinä. Bitteihin kohdistuvat menetelmät lukevat lähdetiedoston tavusta tavun parillisuuden, ja tavuja käsittelevät lukevat tavun arvon.

Louhintaohjelmien tuottamat tulostiedostot säilyttivät alkuperäisen näytetiedoston nimen, mutta muuttivat tiedoston tunnistetta louhintamenetelmän mukaan yksilöidyksi. Tulostiedostot tuotettiin lähdetietojen kanssa samaan hakemistoon, josta ne myöhemmin siirrettiin louhintamenetelmän ja äänilähteen mukaisesti eriteltyihin hakemistoihin analyyseja varten.

Louhintamenetelmät tuottavat Juelsin ja muiden [2000] menetelmästä, jota jatkossa kutsutaan yksinkertaisuuden vuoksi Juelsin menetelmäksi, yksinkertaistettua nopan heittoa lukuunottamatta satunnaisuutta bitteinä – nopanheitto tuottaa tavuja. Louhittu satunnaisuus tulee tallentaa tiedostoon sellaisessa muodossa, että satunnaisuus säilyy muuttumattomana ja tiedosto on helposti luettavissa ja analysoitavissa. Kätevintä on jakaa tuotetut bitit kahdeksan bitin joukkoihin ja kirjata joukot tavuina. Vaikka vain tallennetaan satunnaisia bittejä, jotka eivät siis muutu millään tavalla, on tulostiedostoista lisäksi mahdollisuus tarkastella satunnaisuutta muodostettujen tavujen valossa. Mikäli jokainen kahdeksasta bitistä on satunnainen, on niistä muodostetun tavunkin oltava satunnainen.

4.3.1. Pariteetti

Tavun pariteetin irrottaminen on käytetyistä louhintamenetelmistä yksinkertaisin. Kysessä on yksinkertainen tavun korvaus bitillä, eikä mitään tallentavaa rakennetta käytetä. Tavusta luetaan viimeinen bitti, joka kirjoitetaan tulostiedostoon.

Toteutuksessa joudutaan ylläpitämään pientä tietorakennetta, koska tällöin kirjoitus saadaan ohjemointikielen ominaisuuksien mukaan tehokkaammin suoritettua. Toteutettu algoritmi lukee lähdetiedostosta kahdeksan tavun pariteettibitit puskuriin, jonka täyttyessä puskurin kahdeksasta bitistä muodostetaan tavu, joka kirjoitetaan tulostiedostoon. Mikäli lähdetiedosto ei ole tavumäärältään kahdeksalla

jaollinen, jätetään lähdetiedoston loppuun jäävät ylimääräiset bitit huomioimatta. Ohjelmointikielellisistä syistä lähdetiedostosta luetaan kokonainen tavu bittien sijaan. Tällöin tavun pariteettibitti saadaan selvitettyä kahdella jaollisuuden ja tämän jakojäännöksen perusteella: tavun pariteetti ilmaisee samalla tavun desimaaliarvon parillisuutta.

Tulostiedoston koko on aina lähdetiedostoon nähden kahdeksasosa tai hieman alle. Lähdetiedoston tavumäärän kahdeksalla jakamisen jakojäännös määrää hyödyntämättä jäävien pariteettien määrän. Yli jääviä pariteetteja ei voida tulostavun muodostukseen käyttää, sillä vajaamääräisestä bittijoukosta muodostuu täysimääräisiin nähden helpommin ennustettavissa oleva tavu. Mikäli pariteetin irrotus on kiinteäkokoisen aineiston louhintaan valittu väline, tulisi lähteen koko kiinnittää kahdeksalla jaolliseksi, jolloin hyötysuhde olisi suurimmillaan.

Pariteetin irrotus poistaa tiedostosta täysin viitteet lähteen alkuperään ja luonteeseen. Ihanteellisin lähde toteutuksen kannalta on tavuja kohtuullisen satunnaisesti tuottava, joten sikäli lähteellä on merkitystä, vaikka sisältö irrotuksessa häivytetäänkin. Mikäli lähdetiedostossa toistuu sama arvo useaan kertaan peräkkäin, pariteetin irrotus tuottaa myös toistuvaa bittijonoa. Samoin käy, jos parillisuus peräkkäisillä arvoilla ei muutu. Hyvää tulosta tuottaakseen pariteetin irrotus vaatii lähteeltä paljon.

4.3.2. Von Neumann

Von Neumannin menetelmä jatkaa pariteettimenetelmää. Pariteettibittejä ei suoraan kopioida tulosteeseen, vaan niistä muodostetaan uusia bittejä. Pariteetin irrotus on herkkä toistuville pariteeteille, jolloin tuloste voi olla edelleen painottunutta. Tätä painottuneisuutta saadaan korjattua von Neumannin menetelmällä. Laatu ei tule ilmaiseksi, vaan satunnaisuuden määrässä voidaan joutua tinkimään.

Menetelmä kohdentuu pariteettibitteihin, sillä vaikka painottuneisuutta tehokkaasti poistetaankin, on tavuarvoja tuottavasta lähteestä järkevintä ja turvallisinta hyödyntää vähiten merkitsevää ja todennäköisimmin muuttuvaa bittiä.

Toteutuksessa käytetään kahta tietorakennetta, joista toinen toimii kirjoituspuskurina keräten muodostettuja bittejä, kunnes niitä on kaikkiaan kahdeksan kappaletta. Tällöin biteistä muodostetaan tavu, joka kirjoitetaan tulostiedostoon. Toinen tietorakenne tallentaa edellisen luetun bitin arvon, jotta vertailu voidaan suorittaa. Bittejä tarkastellaan pareittain, ja mikäli bittiparin bitit ovat samat, siirrytään seuraavaan bittipariin tulosta kirjoittamatta. Jos parin bitit eroavat toisistaan, kirjoitetaan tulospuskuriin ensimmäinen bitti. Kumpi bitti puskuriin siirretään, on käytännössä merkityksetöntä, koska lähteen tuottamien bittien jakaumaa ei tarkkaan tiedetä ja lähdettä käytetään vain kerran.

Koska louhintamenetelmät ajettiin sarjassa, olisi von Neumannin menetelmän voinut kohdentaa alkuperäisen lähdetiedoston sijaan pariteetin irrotuksen tuottamaan tulostiedostoon, jolloin suoritusajassa olisi voinut säästää. Algoritmi ei kuitenkaan ole läheskään niin raskas eivätkä lähdetiedostot niin suuria, että säästön määrä suoritusajassa tuntuisi merkittävästi. Ajot joka tapauksessa olivat kertaluontoisia, jolloin sekuntien eroilla ei merkitystä ollut. Von Neumannin menetelmä kohdennettiin siis suoraan alkuperäisiin lähdetiedostoihin. Tällöin jouduttiin jokaisesta luetusta tavusta irrottamaan pariteettibitti.

Menetelmä tuottaa oletettavasti tehokkaasti satunnaista mutta vaihtelevan mittaista tulosta. Mitä enemmän lähdetiedosto on painottunut jommankumman bitin suuntaan, sitä pienempi tulostiedostosta tulee. Suurimmillaan tulostiedosto on kuudestoistaosa alkuperäisestä tiedostosta – joka kahdeksas bitti luetaan, ja bittiparista muodostetaan yksi tulosbitti. Toteutus ei sisällä minkäänlaista lähdetiedoston luonteen tarkastusta, jolloin toteutus on altis heikolle lähdedatalle. Von Neumannin menetelmän toteutus on joka tapauksessa pariteetin irrotusta varmempi louhimaan satunnaisuutta.

4.3.3. Mitzenmacher

Von Neumannin menetelmän yhtenä heikkoutena on hukkaan heitettyjen bittien määrä. Mitzenmacher-menetelmä pyrkii parantamaan tuotettujen tulosbittien lukumäärää hyödyntämällä lähdebittejä laajemmalti. Vaikka toteutus pohjaa von

Neumannin menetelmään, ei aiemmin luotua ajon tulosdataa voida hyödyntää. Pariteetin irrotuksen tuottamaa tulostiedostoa sen sijaan hyödynnetään. Tällöin toteutuksessa ei tarvitse kiinnittää huomiota pariteetin irrottamiseen, vaan voidaan keskittyä pelkästään Mitzenmacher-menetelmään.

Pariteetin irrotus kirjoittaa tulosbittinsä kahdeksan bitin mittaisina lohkoina – eli tavuina – tiedostoon. Java-kielellä on suoraa bittilukua helpompi toteuttaa menetelmä, joka lukee tiedostoa tavu kerrallaan, muuntaa luetun tavun desimaaliesityksen binääriarvoa kuvaavaksi merkkijonoksi ja käsittelee saadun bittimerkkijonon merkki kerrallaan. Mitzenmacher-menetelmän toteutus sisältää erillisen metodin tätä muunnosta varten.

Koska Mitzenmacher-menetelmä on luonteeltaan rekursiivinen, se käyttää enemmän resursseja kuin aiemmat menetelmät. Varsinaisia tietorakenteita tarvitaan neljä: kirjoituspuskuri, aiemmalta tasolta saadut bitit ja kahdet muodostetut uudet bitit. Rekursion takia tietorakenteiden vaatima resurssien määrä voi kasvaa huomattavastikin, joten toteutuksessa rajoitutaan käsittelemään lähdetiedostoa 64 bitin lohkoissa koko tiedoston kertakäsittelyn sijaan.

Rekursiopuu muodostuu binääriseksi ja käsitellään esijärjestyksessä. Puun solmu tuottaa tulosbittejä von Neumannin periaatteen mukaan saamastaan bittisarjasta: juuri tekee tulosta alkuperäisestä syötelohkosta, alisolmut käsittelevät luotuja sarjoja. Vasemmalle lapselle luodaan uusi sarja menetelmän toisen osan perusteella, jolloin jokainen bittipari tuottaa bitin siten, että eribittinen pari tuottaa uuteen sarjaan bitin 1 ja samabittinen pari tuottaa bitin 0. Oikealle lapselle luotava sarja muodostuu menetelmän ensimmäisen osan perusteella, jolloin bitti tuotetaan vain samabittisestä parista sen mukaan, mikä bitti parissa toistuu.

Vasemman lapsen saama bittisarja on pituudeltaan aina puolet solmun saamasta. Oikealle lapselle välittyvän sarjan pituus puolestaan riippuu solmun saaman sarjan ominaisuuksista. Pisimillään sarja voi olla puolet solmun saamasta, lyhyimmillään oikealle lapselle ei välity sarjaa ollenkaan. Näin ollen rekursiopuu kasvaa voi-

makkaammin vasemman lapsen suuntaan. Parittomia bittejä saadusta sarjasta ei käsitellä.

Tulossarja muodostuu esijärjestyksen mukaisesti lohko kerrallaan. Lohkoista suoraan von Neumannin menetelmällä tulevien bittien väliin generoituu Mitzenmacher-jalostusmenetelmän synnyttämät bitit. Kun kirjoituspuskuriin on kerrytetty kahdeksan bittiä, ne kirjoitetaan tavuna tulostiedostoon.

4.3.4. Juels

Juelsin menetelmä ei käsittele lähdetiedoston tavujen pariteettibittejä (tai pariteetin irrotuksen tuottamaa tulostiedostoa), vaan tavuja kokonaisuudessaan. Toteutuksessa lähdetiedosto käsitellään kahdeksan tavun mittaisina lohkoina. Lohko mielletään painottuneen nopan tulokseksi, ja lohkosta muodostetaan tulospuskuriin bittejä. Tulostiedoston koko riippuu lähdetiedoston sisällöstä, joten ennakkoon tulostavujen määrää ei voida tietää.

Lähdetiedoston tuottavan lähteen tuottaessa aidosti satunnaisia tavuja Juelsin menetelmästä ei ole hyötyä, mutta lähteen tuottamien tavujen ollessa painottuneita menetelmä kykenee louhimaan satunnaisuutta kohtuullisen hyvin. Koska satunnaisuuden lähde on käytännössä aina painottunut, kannattaa Juelsin menetelmälle antaa vahvaksi todetuista lähteistä syötettä. Menetelmä tuottaa satunnaisuutta siedettävän tehokkaasti sangen laaja-alaisesta lähdeavaruudesta, mutta silti on suotavaa käyttää luotettaviksi todettuja lähteitä.

Juelsin menetelmä jakaantuu kahteen pääalgoritmiin – *Q1* ja *Q2* – jotka tuottavat syötesarjasta *X* tulossarjan ketjuttamalla algoritmit: *Q2(Q1(X))*. Algoritmin *Q1* tuloste sallii lisäksi kokeellisemman, lähteessä esittelemättömän, menetelmän soveltamisen varsinaisen menetelmän ohella. Ensimmäinen algoritmi tuottaa painottumattoman nopan heittotuloksen, joka voidaan muuntaa tavuesitykseksi. Kokeellinen menetelmä esitellään myöhemmin.

Louhinnan tehokkuuteen vaikuttava seikka on lohkon koko. Mikäli lohko on suuri, tuotettu satunnaisuus on vahvempaa suoritusajan ja louhinnasta saadun satunnaisuuden määrän kustannuksella. Suoritusresursseja eniten kuluttaa menetelmän ensimmäinen algoritmi, joka luo, listaa ja järjestää kaikki nopan tahot. Tahojen lukumääräksi tulee lohkon koon (eli lohkoon otettujen merkkien lukumäärä) kertoma. Mikäli lohkon koko on kaksikymmentä merkkiä, tahojen lukumääräksi tulee enemmän kuin $2.4*10^{18}$. Näin monen tahon listaaminen ja järjestäminen on kohtuuttoman suuritöistä, joten kannattaa käyttää tehokkaampaa menetelmää. Ensimmäinen algoritmi on toteutettu ehdotetulla tavalla. Nopanheitto pilkotaan pienemmiksi nopanheitoiksi ja lopullinen tulos saavutetaan yhdistämällä nämä pienemmät nopanheitot. [Juels *et al.*, 2000]

Ensimmäinen algoritmi käyttää suorituksessaan kahta alialgoritmia, joista toinen laskee alkion X[i] arvoasteen annetussa sarjassa X ja toinen laskee kertoman annetulle arvolle. Arvoasteen määrittäminen on yksinkertainen menetelmä, jossa sarjan sisältävä taulukko järjestetään kasvavaan järjestykseen ja joka iteroidaan vertaamalla alkiota kulloisenkiin sarja-alkioon. Samalla päivitetään tarpeen mukaan arvoastelaskuria, kunnes alkio ja sarja-alkio täsmäävät. Kertoman laskemiseksi joudutaan rakentamaan pienimuotoinen metodi käyttäen Kolehmaisten [2001] algoritmia, sillä Java-kielen perusosissa ei ole sisäänrakennettuna tätä laskumetodia.

Toinen algoritmi saa syötteekseen ensimmäisen algoritmin tuottaman nopanheiton tuloksen *R* sekä nopan tahojen lukumäärän *S* kokonaislukuesityksinä. Algoritmi muuntaa kokonaislukuesitykset binääriesityksiksi käyttäen alifunktiota *binary(x)*, jota vastaava valmismetodi kuuluu Java-kieleen. Mikäli nopanheiton tuloksen binääriesitys *R'* ei ole yhtä pitkä kuin tahojen lukumäärän binäärinen esitys *S'*, lisätään *R'*:n alkuun 0-bittejä tarpeellinen määrä. *R'* ei voi olla bittimäärältään suurempi kuin *S'*. Käytännössä algoritmi palauttaa *R'*:n, jos sen pituus on yli kaksi bittiä. *S'*:n jakaminen kahden potenssin mittaisiksi paloiksi on tehokkaasti toteutettavissa sangen yksinkertaisesti.

Toisen algoritmin tuottama tulossarja liitetään jatkoksi kirjoituspuskuriin. Mikäli liitoksen jälkeen kirjoituspuskurissa on riittävästi bittejä tavun tai tavujen kirjoittamiseen, muutetaan puskurin alusta lähtien kahdeksan bitin sarjat tavuiksi tulostiedostoon. Kirjoitetut bitit luonnollisesti poistetaan puskurista. Kun lohkot loppuvat eikä kirjoituspuskurista saada enää kirjoitettua täysiä tavuja, unohdetaan ylimääräiset bitit, ja suoritus on valmis.

Ensimmäisen algoritmin tuottaman heittotuloksen ja taholuvun perusteella kirjoitetaan toista tulostiedostoa. *Suoran nopan* toteutus luo 256-tahoisen nopan kertomalla heittotuloksen 256:lla ja jakamalla kyseisen tulon heittonopan tahojen lukumäärällä. Saatu tulos pyöristetään lähimpään kokonaislukuun, ja tästä pyöristetystä tuloksesta vähennetään 1. Näin saadaan nopalla heitetty tavuarvo. Menetelmän ei voi odottaa tuottavan kovinkaan vahvasti satunnaista tulosta, sillä heitetyn nopan taholukua ei voida tarkalleen ennustaa, jolloin taholuvun ollessa vähemmän kuin 256 tulostavun arvo hajaantuu liian väljästi tavuarvovälille (0-255). Kun taholuku on muu kuin 256:n monikerta, pyöristys aiheuttaa painottuneisuutta arvojakaumassa.

4.3.5. Aallonpituusmenetelmä

Edelliset menetelmät ovat käsitelleet lähdetiedoston tavuja ja bittejä sinällään hyödyntämättä lähteiden ääniominaisuuksia. Äänitiedoston kuvaaman ääniaallon ominaisuudet tarjoavat omat keinonsa satunnaisuuden louhintaan. Schneierin [1996] esimerkki toistuvan ilmiön ilmentymien suuruuksien ja/tai etäisyyksien vertaamisesta on sovellettavissa ääniaaltoihin.

Lähdetiedostoa luetaan tavu kerrallaan, mutta tavujen merkitys on olla aallon mittana. Vertailun kohteena on peräkkäisten aaltoparien pituus. Aallon pituus lasketaan siitä kohdasta lähtien, jossa signaali on uutta aaltoa lukiessa ylittänyt ensimmäistä kertaa nolla-akselin, eli kun tiedostosta luetun tavun arvo on suurempi tai yhtä suuri kuin 127 (nolla-akselia kuvaava tavuarvo 8-bittistä mono-ääntä sisältävässä tiedostossa) ja kun edellinen luettu tavu on arvoltaan pienempi kuin 127. Jos edellinen aalto loppuu nolla-arvoon 127, alkaa seuraava aalto vasta seuraavasta

tavusta. Näin yksikään tavu ei kuulu kahteen aaltoon. Kun aalto nousee ”negatiiviselta” puolelta (eli pienemmistä arvoista kuin 127) nolla-akselille, aalto katsotaan loppuneeksi. Aallon pituus on aallon alku- ja loppukohdan väliin jäävien tavujen lukumäärä.

Mikäli lähdetiedosto alkaa ”aallon keskeltä”, siirrytään ensimmäisen kokonaisen aallon alkukohtaan. Näinollen osa lähdetiedoston alun tavuista jää hyödyntämättä. Vastaavasti tiedoston lopusta jätetään vaillinaisen aallon tavut huomiotta, kuten myös edeltävän aallon pituus, jos viimeinen ja vaillinainen aalto on aaltoparin toinen aalto. Aallon pituus luetaan muistiin ja jos kyseessä on parin ensimmäinen aalto, luetaan seuraavan pituus ja verrataan näitä kahta pituutta toisiinsa. Jos ensimmäinen aalto oli pidempi, sijoitetaan kirjoituspuskuriin arvo 1. Arvo 0 sijoitetaan puskuriin, mikäli toinen aalto on pidempi. Toteutetussa menetelmässä samanmittaiset aallot ohitetaan, eli mitään arvoa kirjoituspuskuriin ei sijoiteta. Kun kirjoituspuskurissa on kahdeksan bittiä, niistä muodostetaan tavu, se kirjoitetaan tulostiedostoon ja tyhjennetään kirjoituspuskuri. Tämän jälkeen suoritusta jatketaan, kunnes lähdetiedosto päättyy.

Aallonpituusmenetelmän tuottaman tulostiedoston pituutta ei voi ennakoida, sillä alun ja lopun vajavaiset aallot sekä yhtä pitkät aallot välissä aiheuttavat bittien poisjäämistä. Aaltojen pituuksia ennalta tietämättä ei myöskään voi laskea minkäänlaista suhdetta tuotettujen tavujen määrälle. Peräkkäisten aaltojen määrä riippuu äänilähteestä, mutta käytetyllä näytteistystaajuudella äänitetyt lähteet tuottanevat kiitettävän määrän aaltoja. Peräkkäisten aaltojen kohtuullisen lyhyt pituus ja toisaalta kiivaasti vaihteleva luonne parantaa louhinnan tuloksia. Vaarana on silti lähdetiedoston tuhlaava hyödyntäminen, joka voi vaikuttaa myös sellaisiin analyyseihin, jotka vaativat mahdollisimman paljon aineistoa.

4.3.6. Menetelmien yhdistäminen

Menetelmiä ei ole ketjutettu ja täten muodostettu yhdistelmämenetelmiä. Pariteettibitin irrotusta toki hyödynnetään esikäsittelijänä von Neumannin ja Mitzenmacher-menetelmissä, mutta pariteettibitin irrotus on nähtävissä ennemminkin

yhdeksi sisäiseksi osaksi kumpaakin menetelmää. Intuitiivisesti voi päätellä, että voimakkaan louhijan tuottava tulostiedosto olisi sopivaa syötettä toiselle vahvalle menetelmälle, kunhan menetelmien toimintafilosofiat ovat kohdallaan: sisällöllisistä merkityksistä satunnaisuutta louhiva menetelmä sopii vain ketjun ensimmäiseksi menetelmäksi.

Ketjutuksen toteuttamatta jättäminen on perusteltua, koska tutkielmassa perehdytään yksittäisiin louhintamenetelmiin ja täten minkäänlaista luotettavaa *apriori* tietoa louhintamenetelmien tehokkuudesta ei ole hyödynnettävissä. Menetelmien ketjuttaminen on mahdollinen siemen jatkotutkimukselle.

4.4. Louhinnan tulosten esittäminen

Louhittu satunnaisuus täytyy kuvata sellaisella mielekkäällä tavalla, jossa satunnaisuus säilyy. Bittimuoto on sinällään luonnollinen tietokoneelle, mutta tilastollisen analysoinnin ja tuloksen luettavuuden kannalta kahdeksan bitin esittäminen yhtenä tavuna on kätevämpää. Bittejä tuottavat menetelmät muodostavat joko 0- tai 1-bitin tuloksekseen, jonka oletetaan olevan satunnainen. Mikäli tällöin jokainen tavun kahdeksasta bitistä on satunnainen, on tavunkin oltava satunnainen. Tavun bittimuotoisuutta ei sinänsä unohdeta, vaikka tarkastelussa tavusta tehdäänkin perusyksikkö.

Suoraan tavuja tuottavat menetelmät eivät yleisellä tasolla häviä tässä esitystavassa mitään. Jos tavu on tuotettu ”suoraan”, on sen sisältämien bittienkin oltava satunnaisia. Toteutettavista menetelmistä vain Juelsin menetelmän alimenetelmä tuottaa suoraan tavuja, mutta ne eivät välttämättä ole entropialtaan yhtä vahvoja kuin muiden menetelmien tuottamien bittien muodostamat tavut, vaikka satunnaisia muuten olisivatkin. Alimenetelmän tuottamat tavuthan voivat olla pienemmästä kuin 256 alkion joukosta satunnaisesti valittuja. Tällöin yhden tavun entropia voi olla huomattavastikin vähemmän kuin kahdeksan bittiä.

4.5. Analyysit

Tulosten analyysi pyrkii selvittämään louhintamenetelmien ja äänilähteiden määrälliset sekä laadulliset soveltuvuudet siemenluvun tuottamiseen. Louhintamenetelmän vahvuus on selvitettävissä vertaamalla sekä louhintamenetelmien vaikutusta eri äänilähteille että louhintamenetelmiä keskenään. Äänilähteen vahvuus puolestaan selviää vertaamalla äänilähteiden vaikutusta eri louhintamenetelmillä. Analyysien näkökulmasta louhintamenetelmät ja äänilähteet ovat hyvin lähellä toisiaan.

Satunnaisuutta tullaan analysoimaan määrällisesti ja laadullisesti. Määrällinen analysointi hoituu vertaamalla tuotetun tulostiedoston kokoa alkuperäiseen tiedostoon. Vahvalta menetelmältä odotetaan kykyä tuottaa mahdollisimman runsas määrä satunnaisuutta. Laadullinen analyysi on huomattavasti hankalampi toteuttaa, eikä sitä saa millään yhdellä arviointimenetelmällä selville.

Analysoitava tavusarja katsotaan vahvasti satunnaiseksi, mikäli se läpäisee riittävän monta laadullista ehtoa. Näitä ehtoja ovat useat analyysimenetelmät. Mitään laadullista analyysimenetelmää ei oletusarvoisesti voi arvottaa muita korkeammalle, sillä tiedetään, että yksikään vahva satunnaissarja tai satunnaisia sarjoja tuottava menetelmä ei voi suoriutua täydellisesti kaikista voimakkaista analyyseista [Hellekalek, 1998]. Oletettavaa siis on, että paras äänilähteen ja louhintamenetelmän yhdistelmä voi tuottaa heikon tuloksen yksittäisessä analyysissa, jossa muut yhdistelmät ovat menestyneet.

Käytännössä kaikkia tunnettuja ja voimakkaita analyysimenetelmiä ei tulla toteuttamaan, vaan tyydytään otantaan, joka käsittää tunnetuimmat, yleisimmät ja toteutettavissa olevat. Samaten läpäisykynnykset määritellään tulosten tarkastelun yhteydessä. Louhintamenetelmien ja äänilähteiden ei etukäteisesti oleteta olevan voimakkaita tai heikkoja, vaan vasta analyysien tulosten tulkinnassa arvioidaan onko jokin äänilähteen ja louhintamenetelmän yhdistelmä tuottanut vahvaa satunnaisuutta.

Kaikkia haluttuja analyysimenetelmiä ei ollut käytännöllisesti saatavilla avoimeen lähdekoodiin perustuvina ja yhden ohjelman alle niputettuna. Lähdekoodin vapaa tutkiminen on sikäli oleellista, että käytettyjen analyysien algoritmirakenteesta – ja täten oikeellisuudesta – voisi varmistua. Niinpä kätevin menetelmä analyysien toteuttamiseen oli ohjelmoida algoritmit itse. Analyysit kuvaavien artikkeleiden yhteydessä on myös kaavat, joiden pohjalta algoritmit ovat toteutettavissa. Suora kaavan tai pseudokoodin soveltaminen valmiiksi ohjelmaksi ei kaikkien analyysien osalta ole itsestäänselvää, mutta Kolehmaisten [2001] algoritmivinkeillä kyseiset analyysit saa toteutettua. Koska analysoitavia tiedostoja on lukuisia, on kätevintä toteuttaa analyysit mahdollisimman tiiviisti yhdeksi ohjelmaksi. Näin säästetään analysoinnin vaatimissa resursseissa, sillä useat analyysit voidaan suorittaa samanaikaisesti.

Analyysipaketin suorittamisesta syntyy lähdetiedostojen lukumäärää vastaava määrä analyysitiedostoja, joissa analyysien tulokset on esitetty pelkistetysti. Pelkistetty esitys kootaan yhteenveto-ohjelmalla, joka laskee lähde- ja louhintamenetelmäkohtaisesti keskiarvot tai ilmentymien lukumäärät analyysien tuloksille. Yhteenvedon tuloksena analyysien tulokset saadaan taulukoitua ja esitettyä kuvaajilla.

4.5.1. Merkkien lukumäärä

Merkkien lukumäärien vertaaminen on ainoa toteutettava määrällinen analyysi. Koska analysoitavan tiedoston oletetaan sisältävän pelkästään louhittua satunnaisuutta, on merkkien lukumäärän vertaaminen oikeastaan ainoa keino analysoida louhintamenetelmän määrällistä vahvuutta. Satunnaisia bittejä analysoitavassa tiedostossa on oletettavasti kahdeksan kertaa merkkien lukumäärä, mutta kuten aiemmin on todettu, louhinnan tulostiedoston tavu oletetaan yhtä satunnaiseksi kuin sen muodostavat bitit. Näin ollen on kätevämpää verrata merkkien, eli tavujen, lukumääriä.

Merkkien lukumäärä ei sinällään kerro satunnaisuudesta mitään. Määrällisessä tarkastelussa tiedoston satunnaisuuden oletetaan olevan vahvaa: vahvuuden todis-

taminen jätetään muille analyyseille. Hyvä louhintamenetelmä kykenee tuottamaan hyvästä lähteestä saadusta aineistosta paljon satunnaisuutta. Määrällinen vahvuus ei toisaalta ole tae louhijan ja lähteen hyvyydelle. Määrä täydentää mutta ei korvaa laatua.

Merkkien lukumäärän voisi helposti saada selville suoralla hakemisto-operaatiolla, jolla järjestelmältä kysyttäisiin kyseisen tiedoston tavukokoa. Koska analyysit on niputettu samaan pakettiin, tulee yhteisilmeen säilyttämisen kannalta kannattavammaksi laskea jokainen merkki erikseen samalla, kun muita analyyseja suoritetaan. Lisäksi näin saadaan varmistettua, että merkkien lukumäärä käsittää kaikki muilla keinoillakin analysoidut tavut. Merkkien lukumäärä on likiarvo, sillä se esitetään kokonaislukuna.

4.5.2. Merkkien jakautuminen arvoalueelleen

Aidosti satunnainen tiedosto sisältää todennäköisimmin tasaisesti kautta arvoalueen jakautuneen joukon tavuja. Mikä tahansa tavujen kombinaatio on yhtä todennäköinen, jolloin tavujen esiintymien summat ovat lähellä toisiaan. Toisin sanoen tavujen jakauma on tasainen, jos tavut tiedostoissa ovat yhtä todennäköisiä. Tavujen jakautumista arvoalueelle [0,255] tarkastellaan kunkin tavun *ilmentymän keskiarvolla* sekä tiedoston kaikkien tavuarvojen *kvartiileilla*.

Ilmentymien keskiarvo lasketaan pitämällä yllä laskuria käsiteltyjen tiedostojen lukumäärästä sekä kasvattamalla tarkasteltavan tavun indeksoimaa taulukkoarvoa. Kun kaikki ajon mukaiset tiedostot on käsitelty, jaetaan jokaisessa taulukon indeksissä oleva arvo tiedostojen lukumäärällä. Merkkien ilmentymien keskiarvoa on järkevintä tarkastella kokonaislukuna, jolloin ilmentymien keskiarvojen summa saattaa osoittaa eri arvoa kuin merkkien kokonaislukumäärien keskiarvo (joka sekin itse asiassa on likiarvo).

Ilmentymien keskiarvo kertoo melkoisen hyvin saavutetun satunnaisuuden laadusta. Jos jakauma on tasainen, louhintamenetelmä ja lähde kykenevät tuottamaan tämän testin läpäisevää satunnaisuutta. Valitettavasti merkkien järjestystä analyysi

ei paljasta. Analysoitavan tiedoston sisältäessä aidosti nousevassa järjestyksessä ja tasaisesti jakautuneesti luvut läpi arvoalueensa, se läpäisee tämän analyysin esimerkillisesti. Kompleksisuuden kannalta tiedosto on kuitenkin kaikkea muuta kuin satunnainen. [Walker, 1998]

Kvartiilianalyysi täydentää ilmentymien keskiarvoanalyysia. Kvartiilien sijainnit lasketaan ilmentymien keskiarvotaulukosta. Teoreettiset optimikvartaalit (arvoissa 63, 127 ja 191) osoittavat tasajakautunutta tapausta, ja tarkasteltavat keskiarvokvartaalit ovat sitä parempia, mitä lähempänä kyseisiä optimikvartaaleita ne ovat.

Luonnollisesti ilmentymien keskiarvo ja kvartiilitarkastelu arvioivat kokonaissatunnaisuutta sitä paremmin, mitä enemmän analysoitavaa aineistoa on saatavilla. Radiokohinan laajempi aineisto tuottaa täten todennäköisesti tasaisempaa jakaumaa kuin suppeampi puhutun ääninäytteen aineisto. Tämä pätee kaikkiin tehtäviin analyyseihin, joskin likiarvojen käyttämisestä johtuen on hankalaa löytää tarkkaa rajaa, jolloin aineistojen laajuuden eroavaisuus alkaa häiritä.

Raa'assa äänidatassa äänen aaltomuoto ja voimakkuus vaikuttavat voimakkaasti merkkien jakaumaan. Merkit jakautuvat nolla-akselin (eli arvon 127) ympärille sitä tiiviimmin, mitä vaimeampaa ääninäyte on. Toisaalta leikkautunut – eli *särkynyt* – ääninäytteen kohta painottaa jakaumaa kohti ääripäitään. Täten käsittelemätön äänidata muodostaa todennäköisimmin epätasaisen jakauman. Äänen voimakkuuden ja ääninäytteen hiljaisempien hetkien vaikutuksesta jakauma on voimakkaasti keskittynyt.

4.5.3. Merkkien peräkkäinen toistuvuus

Kompleksisuusehdon täyttymistä voidaan tarkastella merkkien peräkkäisen toistuvuuden avulla. Sarjojen tai yksittäisten merkkien peräkkäinen toistuvuus kasvattaa ennustettavuutta, jolloin kompleksisuus on heikkoa. Merkkien peräkkäisen toistuvuuden kannalta tiedosto on vahvasti satunnainen silloin, kun tarkastelussa seuraavan tavun arvoa ei voi arvata paremmalla todennäköisyydellä kuin 1/256.

Tavusarjan sisältämiä toistuvia merkkijonoja on kohtuullisen työlästä löytää. Kun kokeissa käytetyt tavusarjat ovat 110 250 tavun mittaisia, täytyisi järjestäen jokaista minimissään kolmen ja maksimissaan 55 125:n mittaista alisarjaa sovittaa koko tavusarjaan. Kolmen merkin minimi olisi käytännössä järkevin, sillä kahden merkin kombinaatioita voi olla enintään 256^2=65 536, jolloin joidenkin kombinaatioiden täytyy toistua 110 250 tavun mittaisessa sarjassa. Kolmen merkin mittaisia kombinaatioita puolestaan voi olla 256^3=16 777 216, jolloin käytetyn mittaisiin tavusarjoihin eivät kerralla mahdu kaikki mahdolliset kombinaatiot. Käytetyn mittaiseen tavusarjaan mahtuu kaikkiaan 110 248 kolmen peräkkäisen merkin mittaista tavusarjaa, jolloin näiden vertailujen määräksi tulee kertoma luvulle 110 248.

Kuvattu laskennan määrä ei vielä riitä tavusarjan sisältämien säännömukaisuuksien ja toistuvien mallien selvittämiseen. Toistuva tavusarja ei nimittäin välttämättä koostu peräkkäisistä merkeistä. Alisarjan {201, 177, ***3***, 45, 47, ***66***, ***24***, 152, ***3***, 3, 3, ***66***, ***24***} kursivoidut ja tummennetut alkiot muodostavat toistuvan mallin, jonka alkiot eivät ole peräkkäin. Lisäksi malli ei välttämättä ole sidoksissa tiettyihin tavuihin vaan tavujen trendiin. Malli voi muodostua esimerkiksi kahdestakymmenestä tavusta, jotka esiintyvät arvoltaan kasvavassa järjestyksessä. Tämän kaltaisten mallien tunnistaminen on mahdollista raa'alla voimalla, mikäli laskentaresursseja ja suoritusaikaa on liki rajaton määrä, mutta kehittyneitä heuristiikkojakin voi olla olemassa.

Tässä tutkimuksessa tuotettuja tiedostoja analysoidaan toistuvuuden suhteen sekä *korrelaatiokertoimen* laskemisella, että *Run*-testeillä [Bays and Durham, 1976]. Run-testit kuvaavat konkreettisesti sekä tiedoston peräkkäisten tavujen toistuvuutta – eli *juoksuja* – että tiedoston arvojen aaltomuodon tasaisuutta. Ideaalitapauksessa tavun peräkkäistä toistuvuutta ei ole, eikä tiedoston arvojen aaltomuoto ole tasainen vaan voimakkaasti vaihteleva. Korrelaatiokerroin liitty toistuvuuteen sikäli, että mikäli tavuarvot ovat toisistaan riippumattomia, tulee tavuparien perusteella lasketun korrelaatiokertoimen olla lähellä nollaa.

Ensimmäinen Run-testi – josta tulosten yhteydessä käytetään Run-testi nimitystä – analysoi samojen tavuarvojen peräkkäisten toistojen pituuksia. Jos luettu tavu on arvoltaan yhtä suuri kuin edellinen luettu tavu, kasvatetaan laskuria. Mikäli luettu tavu eroaa edellisestä, kasvatetaan tulostaulukon laskurin osoittamaa indeksiä yhdellä ja alustetaan laskuri. Kaikki yli 20 merkin mittaiset saman tavun toistot kirjataan samaan ”yli 20” lokeroon taulukossa. Analyysien tuloksista otetaan tarkasteluun kaikkien samaan äänilähteeseen ja louhintamenetelmään kohdistuneiden tiedostojen juoksujen keskiarvoinen lukumäärä. [Bays, 1976]

Ideaalin tapauksen teoreettiset raja-arvot on Run-testissä helppo laskea. Yhden mittainen juoksu, jota edeltävällä ja seuraavalla tavulla ei ole samaa arvoa, esiintyy todennäköisyydellä *255/256*, eli noin 99,6 % juoksuista olisi yhden merkin mittaisia. Tämä on laskettu siten, että tarkasteltavana oleva tavu voi olla arvoltaan mikä tahansa muu kuin edeltävä tavu. Kahden mittaisten juoksujen todennäköisyys on puolestaan *(255/256)*(1/256)*, eli kahden mittaisia juoksuja on likimäärin 0,389 %. Kolmen mittaisia vastaavasti noin 0,00152 %. Kokeissa käytetyissä louhimattomissa tiedostoissa kyseisiä juoksuja olisi likimäärin 109 809, 429 ja 2 kappaletta. Louhinta luonnollisesti pienentää tavujen lukumäärää ja siten myös juoksujen lukumääriä. [Schneier, 1996, p. 45]

Toinen Run-testi – josta käytetään nimitystä Prun – analysoi puolestaan tiedoston arvojen aaltomuotoisuutta. Prun-testissä juoksu koostuu vähintään kahdesta peräkkäisestä ja eriarvoisesta tavusta. Niin kauan kuin luetut merkit joko kasvavat tai pienenevät, kasvaa juoksun pituus. Kun tulee taitekohta muuhun suuntaan, saadaan edellisen juoksun pituus ja aloitetaan seuraavan mittaaminen. Mikäli peräkkäiset tavuarvot pysyvät samoina, ollaan tarkastelemassa *tasankoa*, jolloin kasvatetaan asianmukaista laskuria, mutta ei päivitetä juoksuihin liittyviä laskureita. Tasangot ovat Run-testin juoksuja, eli saman merkin peräkkäisiä toistoja. Samoin kuin Run-testissä, myös tässä tarkastellaan samasta äänilähteestä samalla menetelmällä louhittujen tiedostojen keskiarvoja. [Bays, 1976; Gorenstein, 1967]

Teoreettisten rajojen laskeminen tietyn mittaisille juoksuille on oletettavasti hyvin vaikeaa. Gorenstein [1967] esittää yhden kaavan, joka kuitenkin pätee vain tasan-

gottomalle datalle. Tasankojen esiitymisen mahdollisuus muuttaa ongelmaa huomattavasti. Kuitenkin, kun Prun-testin tuloksia tarkastellaan, voidaan Run-testin raja-arvoja sopeuttaen päästä jonkinlaiseen tulkintaan. Vahvasti satunnaisessa tiedostossa tavut eivät muodosta kovin pitkiä juoksuja ja tasankokohtia ei ole kovin montaa. Heikkoutena näissä päätelmissä on muun muassa hyvin pitkät tasankokohdat, jonka kuitenkin voi todeta vaihteluksi juoksujen kokonaismäärässä. Kaikenkaikkiaan käytettävissä olevaan aineistoon Prun-testi ei sovellu läheskään niin hyvin kuin Run-testi. [Gorenstein, 1967]

Pearsonin sarjakorrelaatiokertoimen (jatkossa pelkkä korrelaatiokerroin) laskeminen satunnaislukusarjasta antaa osaltaan viitettä lukujen riippumattomuudesta paljastamalla, onko tavun ja edeltävien tavujen välillä korrelaatiota. Laskenta toteutetaan Kolehmaisten [2001] esittämän algoritmin mukaisesti. Vaikka käsiteltävät tiedostot ovat sisällöltään tavallaan yksiulotteisia muotoa $\{x_1, x_2, x_3, \ldots, x_n\}$, voidaan näistä muodostaa kaksiulotteinen kuvaus siten, että parittoman indeksin alkio mielletään x-akselin arvoksi pisteelle, jonka y-akselin arvoa kuvaa seuraava parillisen indeksin alkio. Jos alkiot ovat satunnaisia, myös niiden määrittelemä kuvaajan piste on satunnainen. Tällä perusteella korrelaatiokerroin on järkevä arvo satunnaisuuden analysointiin. Mitä lähempänä kerroin on nollaa, sitä vahvemmaksi voidaan analysoidun tiedoston arvojen satunnaisuus päätellä. Ei-satunnainen tavusarja – kuten vaikkapa ohjelmakoodista koostuva tiedosto – tuottanee keskimäärin korrelaatiokertoimen 0.5, kun puolestaan vahvasti ennustettavissa oleva tavusarja – esimerkiksi pakkaamaton bittikarttatiedosto – johtaa hyvin lähellä yhtä olevaan arvoon.

Korrelaatiokerroin on toisaalta ongelmallinen analyysimenetelmä, sillä sen vahvuus on hyvin riippuvainen käytetystä toteutuksesta. Periaatteessa korrelaatiokerroin sarjalle pitäisi arvioida osasarjoille laskettujen korrelaatiokertoimien pohjalta. On kuitenkin hyvin hankala löytää optimaalisia pituuksia alisarjoille. Tässä tutkimuksessa korrelaatiokerroin lasketaan suoraan koko tiedostolle. Samassa ryhmässä käsiteltävien tiedostojen korrelaatiokertoimista lasketaan keskiarvo edustamaan kyseistä ryhmää. [Eagini and Buce, 1999 ; Kolehmainen ja Kolehmainen, 2001]

4.5.4. Muut menetelmät

Edellä esiteltyjen analyysien lisäksi on muutamia menetelmiä, jotka mittaavat koko tavusarjan – eli tiedoston – satunnaisuutta kokonaisuutena. Tässä alakohdassa esiteltävät menetelmät ovat kenties yleisimmin tunnetut ja käytetyt satunnaisuuden mitat, mutta ne eivät silti välttämättä ole sen luotettavampia kuin aiemmissa alakohdissa esitellyt menetelmät. Vaikka esimerkiksi tiedoston tavujen *entropia* on määritelmällisesti kompleksisuusnäkökulmasta tiedoston satunnaisuuden mitta, ei pelkkään entropia-arvoon voi luottaa tiedoston todellista satunnaisuutta arvioitaessa. Eri analyysimenetelmät mittaavat aina jossain määrin joko *erilaista* satunnaisuutta tai samantyyppistä satunnaisuutta hieman eri näkökulmista. Tässä alakohdassa käsiteltäviä menetelmiä on kolme. *Monte Carlo* -menetelmä *piin arvon* muodostamiseksi (Monte Carlo for Pi), entropian laskeminen ja *Khiin neliön* laskentaan perustuva menetelmä.

Piin likiarvo on määriteltävissä satunnaislukujen avulla niinsanotulla Monte Carlo -menetelmällä, mitä voidaan hyödyntää tiedoston tavujen satunnaisuutta analysoitaessa. Koska ympyrän ala on $\pi*r^2$, on yksikköympyrän ala π. Otetaan tarkasteluun yksikköympyrän se neljännes, jossa sekä x että y saa vain positiivisia arvoja (tai nollan). Kun generoidaan suuri määrä satunnaisia x- ja y-liukulukuarvoja väliltä [0,1], niin osa näiden arvojen muodostamista pisteistä osuu yksikköympyrän alalle ja pienempi osa ei. Näiden *osumien* suhde kaikkiin pisteisiin kerrottuna neljällä on yksikköympyrän ala, eli π. [Kolehmainen ja Kolehmainen, 2001]

Tiedostoja analysoitaessa osuma lasketaan kahden peräkkäisen tavun muodostamasta x- ja y-parista – parin ensimmäinen tavu otetaan *x*:ksi ja toinen *y*:ksi. Koska tietokoneella liukulukulaskenta on hitaampaa ja tarkkuudeltaan kyseenalaisempaa kuin kokonaisluvuilla toimiminen, on toteutuksessa järkevämpää käyttää yksikköympyrän sijaan ympyrää, jonka säde on 255. Kun tavupari on luettu, voidaan laskea pisteen sijainti suhteessa ”yksikköympyrään”. Mikäli *y*-tavun neliö on pienempi tai yhtä suuri kuin 255^2-x^2, on piste osunut yksikköympyrän alalle ja *osumalaskuria* päivitetään. Jos tuloksena on osuminen yksikköympyrän ulkopuoliselle alueelle, päivitetään *ohilaskuria*. Kun tiedosto on käsitelty, lasketaan π:n arvon

arvio. Lopputulokseksi lasketaan virheprosentti vertaamalla arviota Javan kovakoodattuun π:n arvoon. Samasta lähteestä samalla louhintamenetelmällä muodostetusta aineistosta laskettujen arvioiden virheprosenttien keskiarvo on tarkastelun kohteena tulosten analysoinnissa.

Entropian laskemisella selvitetään, kuinka monta bittiä tarkalleen tiedosto tarvitsee yhden tavun kuvaamiseen. Jokaisen tavun prosentuaalinen esiintyminen koko tiedostossa vaikuttaa kokonaisentropian laskemiseen, mutta tässä ei huomioida tavujen arvojen peräkkäistä toistuvuutta. Voidaankin sanoa, että tässä tutkimuksessa käytetty entropia-analyysi on vajavainen ja vaatii vähintään Run-testit taustalle varmentaakseen satunnaisuuden laatua. Tiedosto, jossa on kasvavassa järjestyksessä uniikit 256 tavua, tuottaa tällä entropiamenetelmällä tuloksen 8.0 bittiä/tavu. Tämä tarkoittaa, että tiedoston tavut on määriteltävissä vähintään kahdeksalla bitillä, mikä on entropian kannalta paras mahdollinen tulos. Olisi täysin perusteltua sanoa, että entropiamenetelmä kuuluisikin merkkien jakautumista arvoalueelleen mittaavien menetelmien joukkoon. Toisaalta filosofinen ero tarkoituksissa pitää entropian erillään tavanomaisemmista jakaumaa mittaavista menetelmistä. [Eastlake *et al.*, 1994 ; Schneier, 1996]

Entropia lasketaan tiedoston tavujen suhteellisesta jakaumasta koostetusta taulukosta, joka on muodostettu siten, että kokonaislukuja sisältävän esiintymätaulukon arvot on jaettu merkkien kokonaismäärällä. Suhteellisen jakauman esittävästä taulukosta lasketaan summa siten, että jokaiselle alkiolle p_i kaavan 4.1 mukaan lasketut tulot e_i lasketaan yhteen. Näin saadaan laskettua tiedostolle entropia. [Eastlake *et al.*, 1994]

$$e_i = p_i * log_2(1/p_i),\ i=[0,255]$$

Kaava 4.1 – Entropian osakaava suhteellisen jakauman taulukolle.

Entropia noteerataan analyysien yhteenvedossa jälleen kaikkien samasta lähteestä samalla tavalla louhittujen tiedostojen entropioiden keskiarvona. Entropian kannalta täydellisin tulos on 8.0, mutta kuten aiemmin on todettu, kyseinen tulos voi

viestiä ongelmista muiden analyysimenetelmien kohdalla. Entropioiden keskiarvon käyttäminen on sikäli ongelmallista, että erot analysoitavien materiaalien määrässä murtavat hieman pohjaa keskinäiseltä analyysiltä. Entropia yksinään ei ole tällä tavoin toteutettuna kovinkaan luotettava satunnaisuuden mitta, vaan sen tuloksia on verrattava muiden menetelmien tuottamien tulosten kanssa. Näistä menetelmistä varsinkin Run-testit ovat olennaisia.

Sopivaa viitettä tiedostojen satunnaisuudesta voidaan saada *Khiin neliö* (X^2) testillä. Testissä tarkastellaan tiedoston arvojen jakaumaa suhteessa ihannetapaukseen, ja testi suoritetaan tiedoston lukemisen jälkeen. Testi ei ota kantaa tavujen järjestykseen vaan vain niiden jakaumaan. Khiin neliön tarkastelemiseksi tavut luokitellaan siten, että jokainen tavu edustaa omaa luokkaansa, eli luokkia on 256 kappaletta. Luokille lasketaan tasajakauman mukaiset ihannefrekvenssit. Tasajakaumallinen ihannetapaus, eli *nollahypoteesi* (null hypothesis), tuottaa $X^2=0$ kaavan 4.2 perusteella. Kun tarkastellaan tavusarjoja tasajakauman ja Khiin neliön selvittämisen jälkeen, voidaan verrata tavusarjan tuottamaa tulosta ihannetulokseen. Yksinkertaisimmillaan voidaa päätellä, että mitä suurempi on kaavan tuottama tulos, sitä etäämpänä tarkasteltu sarja on ihannesarjasta ja tällöin sitä vähemmän satunnainen tarkasteltu sarja on.

$$X^2=\sum \frac{(\text{havaittu} - \text{ihanne})^2}{\text{ihanne}}$$

Kaava 4.2 – Khiin neliön kaava

Sarjan ja ihannetuloksen etäisyyden tulkinnassa hyödynnetään *vapausasteita* (degrees of freedom) sekä teoreettisia raja-arvoja. Khiin neliö -analyysissa lasketaan sekä tavusarjan absoluuttinen Khiin neliö -arvo, että prosenttiarvo, joka kertoo, kuinka usein aidosti satunnainen tavusarja ylittää saadun absoluuttisen arvon. Prosenttiarvo tulkitaan ikään kuin tarkasteltavan tavusarjan satunnaisuuden todennäköisyydeksi. Prosenttiarvoja tulkitaan siten, että mikäli tavusarja tuottaa arvon, joka on suurempi kuin 99 % tai pienempi kuin 1 %, niin voidaan päätellä että tavusarja hyvin todennäköisesti ei ole satunnainen. Jos arvo on välillä 99 % - 95 %

tai 1 % - 5 %, tavusarja ei todennäköisesti ole satunnainen, mutta epäilyksen sijaa silti on. Arvovälit 95 % - 90 % ja 5 % - 10 % kuvaavat, että sarjan satunnaisuutta sopii epäillä. Mitä lähempänä arvo on prosenttiarvoa 50, sitä todennäköisemmin sarja on satunnainen. Puolella aidosti satunnaisista sarjoista Khiin neliö -arvot ylittävät ihannetapauksen ja puolella arvot keskimäärin ihannetuloksen alittavat. [Walker, 1998; Gorenstein, 1967]

Prosenttiluokituksia käytetään hyväksi analysoitaessa tämän tutkimuksen tulostiedostoja. Jälleen luokitusten ilmentymistä lasketaan keskiarvot lähde- ja louhijakohtaisesti. Mikäli luokitusten jakauma muistuttaa normaalijakaumaa, ovat tulokset tulkittavissa hyviksi. Edelleen on kuitenkin huomioitava, että vaikka Khiin neliö on tehokas menetelmä tavujen jakautumisen selvittämiseksi, ei se yksin riitä satunnaisuuden mitaksi.

5. Kokeiden tulokset

Satunnaisuutta on mielekkäintä arvottaa satunnaisuutta tarvitsevan kohteen tarpeiden mukaan. Yleinen arviointi paljastaa lähinnä vain suunnan, josta satunnaisuutta kannattaa hakea. Tämän tutkielman kokeellisessa osiossa on käytetty louhintamenetelmiä ja äänilähteitä ilman sovelluskohteita. Kokeissa on haettu viitteitä lähteiden soveltuvuudesta mahdollisimman laajalti satunnaisuutta tarvitseviin kohteisiin.

Satunnaisuuden määrällinen arviointi on vahvasti alisteinen laadulliselle arvioinnille. Satunnaisuuden laatu paljastaa, onko määrällä merkitystä. Käyttökohteen vaatimukset asettavat laadulle rajat: joissain tilanteissa voi riittää laadullisesti heikompikin satunnaisuus, kunhan sitä on paljon. Tämän tutkimuksen tuloksia tarkasteltaessa korkeaa laatua pidetään arvokkaana ja määrä on eduksi.

Tuloksia tarkastellaan vertaamalla analyysikohtaisesti louhintamenetelmiä ja äänilähteitä samalla kertaa. Kohdassa 5.1 esitellään laadullista satunnaisuutta kuvaavien analyysien tulokset ja kohdassa 5.2 määrälliseen satunnaisuuteen keskittyvät tulokset.

5.1. Laadullinen satunnaisuus

Satunnaisen merkkijonon tärkeät ominaisuudet ovat ennustamattomuus, toistuvien osasarjojen puute ja edellä mainittujen ominaisuuksien rajoittama mielivaltaisuus. Ennustettavuuden vaikeus on helppo mieltää merkki kerrallaan saatavan jonon ominaisuudeksi: voiko merkistä ja sitä edeltävistä merkeistä ennustaa seuraavaa merkkiä. Vaikka louhinta kohdistuukin tiedostoihin reaaliaikaisen *merkkivirran* (stream) sijaan, voidaan sekä louhittaessa että analysoitaessa louhinnan tuloksia tiedosto tarkastella merkki kerrallaan virtuaalisena virtana.

Ennustamattomuus ja osasarjojen puuttuminen ovat todennäköisempiä, jos merkit ovat jakautuneet tasaisesti koko arvoalueelleen. Todellisen arvoalueen tulisi olla yhtä suuri kuin suurin mahdollinen arvoalue. Tärkeintä on arvoalueen laajuuden

suhde satunnaisuuden laadun tarpeeseen. Jos merkit ovat jakautuneet tasaisesti suurinta mahdollista arvoaluetta pienemmälle alueelle, on silti täysin mahdollista, että merkit ovat jakautuneet satunnaisesti. Luotettavampaa satunnaisuus on jakautuneena koko mahdolliselle arvoalueelle.

Laadullista satunnaisuutta tarkastellaan merkkien jakauman ja jaksollisuuden perusteella. Tarkastelussa selvitetään sekä äänilähteiden että louhintamenetelmien laadukkuutta. Ensin tarkastellaan jakaumaa kvartiilien, piin arvioinnin ja Khiin neliön avulla, ja sitten selvitetään jaksollisuutta Run-testien tuloksista. Tulostiedostojen kompleksisuuksia tarkastellaan entropia-arvoilla. Analyysien teoreettinen tausta on selvitetty kohdassa 4.5.

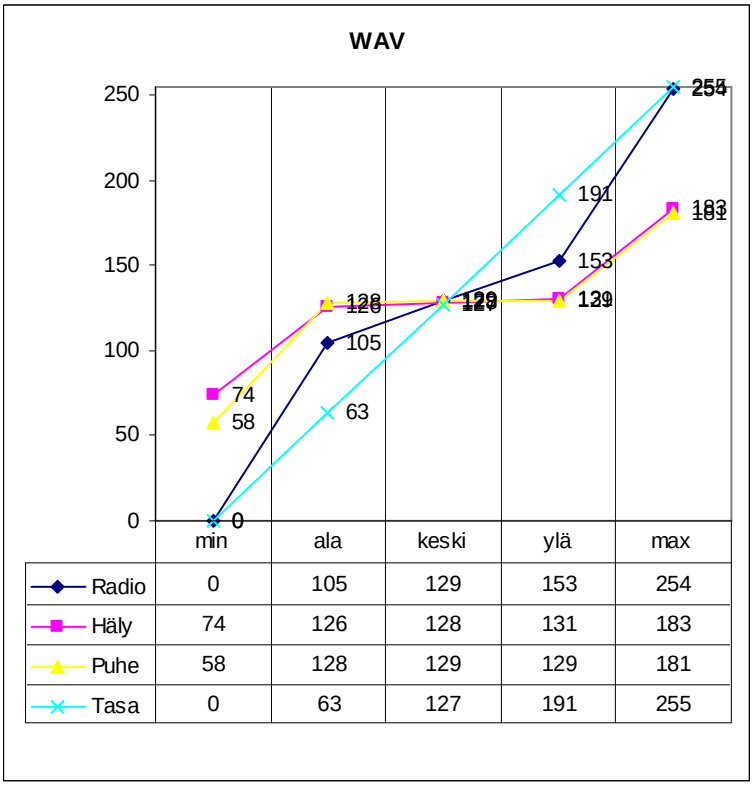
5.1.1. Merkkien jakautuminen arvoalueelleen

Kvartiilien ja merkkijakaumien avulla voidaan tarkastella merkkien jakatumista arvoalueelleen. Koska kaikkien analysoitujen tiedostojen tuloksista laskettiin keskiarvot lopullisia analyyseja varten, voidaan tarkastella louhintamenetelmiä ja lähteitä suhteellisen yleisellä tasolla. Yksittäisten tulostiedostojen ominaisuudet eivät ole läheskään niin kiinnostavia ja oleellisia kuin kaikkien vastaavien tuottama kokonaisilme. Siksi on usein mielekkäämpää tarkastella kvartiileja kuin merkkien jakaumia.

Jos kvartiilit näyttävät liputtavan tasajakauman puolesta, voivat merkit silti olla jakautuneina hieman eri arvoille. Tämän takia myös varsinaisia merkkijakaumia tarkastellaan, jotta kvartiileilta huomiotta jääneet poikkeamat voidaan havaita. Ääri- ja keskimerkeiltään suuria arvoja saava jakauma voi tuottaa ideaalit kvartiilit, joten merkkijakauman tarkastaminen on oleellista. Todennäköisempää kuitenkin on, että jakauma korostuu äärimerkeissään, jolloin kvartiilit ovat hyvin lähellä minimi- ja maksimiarvoja. Tällöin keskiarvo voi puolestaan näyttää ideaalilta. Ala- ja yläkvartiilit ovatkin keskiarvoa oleellisempia tunnuslukuja.

Käsittelemättömien äänitiedostojen merkkien oletettiin keskittyneen nolla-akselin ympärille normaalijakautuneesti. Radiokohinaa sisältäneiden näytteiden keskiar-

voinen jakauma käsittää laajimman arvoalueen. Radiokohinanäytteet toisaalta olivat hyvin tasaisia äänenvoimakkuudeltaan, joten niissä ei ollut mitään huomattavasti ympäristöä voimakkaampia piikkejä, jotka olisivat muuttaneet jakaumaa. Kohinasignaalin särkymistä – eli yli arvoalueen meneviä signaaleja – rajoitettiin. Puhenäytteet sen sijaan sisälsivät paljon hiljaisia signaalikohtia. Taustahälynäytteiden merkkijakauman kvartiilit ovat liki identtiset puhenäytekvartiilien kanssa. Tätä saattaa selittää kohinanäytteiden parempi voimakkuudensäätely.

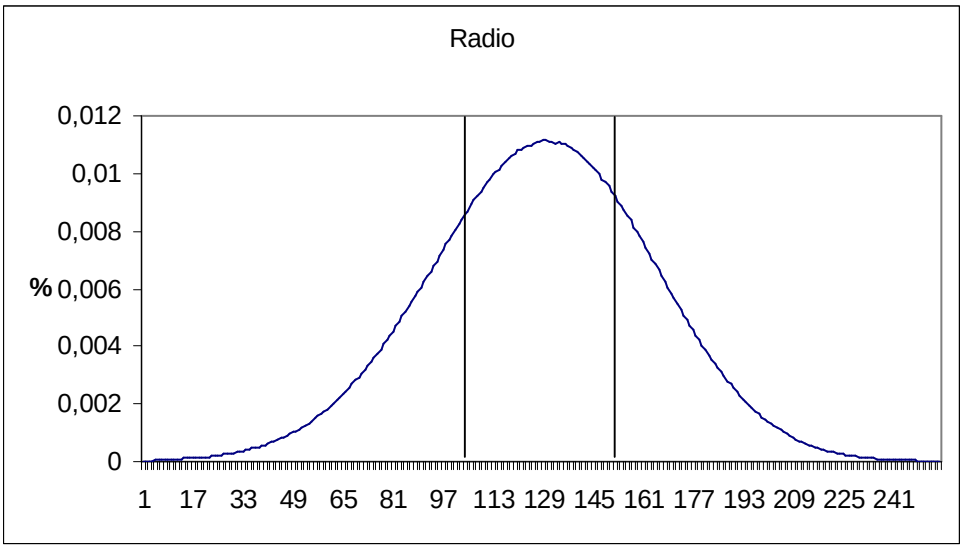


Kuvaaja 5.1 – Kvartiilit käsittelemättömillä ääninäytteillä.

Kuvaajassa 5.1 on esitetty käsittelemättömien ääninäytteiden kvartiilit suhteessa tasajakauman kvartiileihin. Häly- ja puhenäytteiden jakaumat ovat liki identtiset, mutta häly on aavistuksen puhetta laajemmalle jakautunut. Puhenäytteen signaaleista puolet on vain yhden arvon päässä signaalinäytteiden keskiarvosta. Hälynäytteissä viidenkymmenen prosentin jakauma käsittää vain viisi näytearvoa. Radiokohinan kvartiilijakauma on huomattavasti siedettävämpi, mutta silti keskittynyt voimakkaasti keskustaan ja täten kauemmas tasajakautuneesta tavoitearvosta.

Voimakkuuden säätely vaikutti osittain siihen, että puhe- ja hälynäytteiden arvoalue rajautui pienelle alueelle. Puhenäytteissä ei ollut arvoa 58 pienempiä, eikä arvoa 181 suurempia signaalinäytteitä. Hälynäytteissä vastaavat rajat olivat 74 ja 183. Jos puhe- ja hälynäytteet olisivat saatu ikkunoitua koko arvoalueelle, olisi ala- ja yläkvartiilien etäisyydet toisistaan todennäköisesti hieman suuremmat, mutta jakauma säilyisi silti hyvin keskittyneenä.

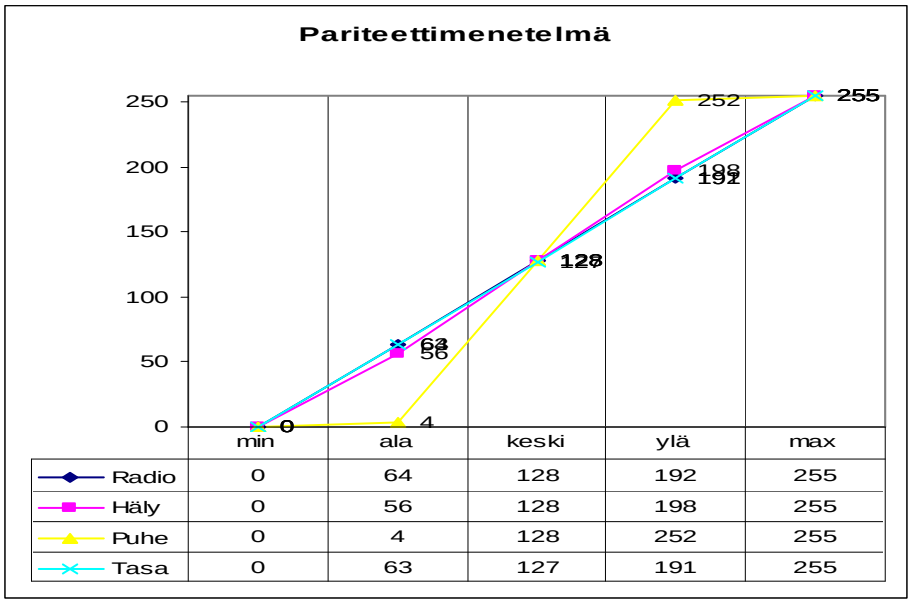
Rikkomatta signaalia radiokohinaakaan ei saada tasajakautuneesti, sillä puhdas äänisignaali on aina keskialueen ympärille keskittynyttä. Kuvaajassa 5.2 esitetään kokeissa käytettyjen radiokohinanäytteiden jakauma käyttäen suhteellisia arvoja näytteille. Äänenvoimakkuuden nostaminen aiheuttaisi leikkautumista, joka näkyisi joko piikkeinä ääriarvoissa tai häiriönä muissa arvoissa. Taulukkoon on merkitty myös alue, joka sisältää puolet arvoista.



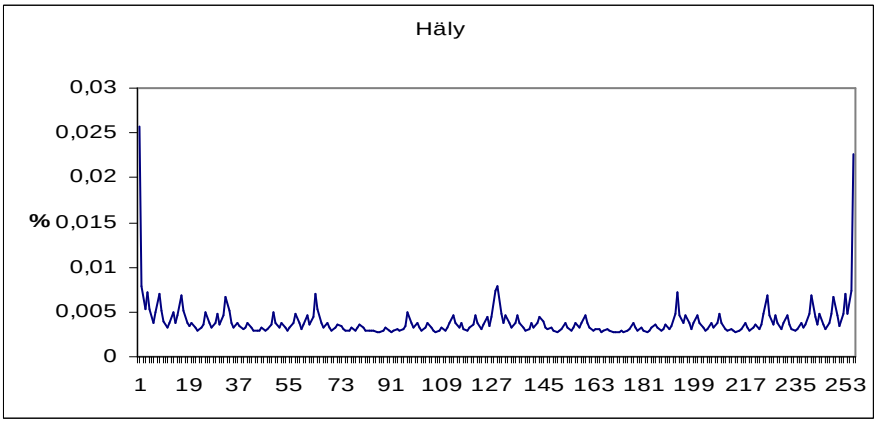
Kuvaaja 5.2 – Radiokohinan signaalinäytteiden jakauma ja kvartiilit.

Pariteetin irroitus (kuten muutkin varsinaiset louhintamenetelmät) mursi tiedoston aaltomaisuuden ja arvojen keskittymisen ”nolla-akselin” ympäristöön (kuvaaja 5.3). Radiokohinan kvartiilit olivat vain yhden suurempia kuin tasajakaumalla, mikä johtunee pyöristysvirheestä. Taustahälystä louhittu jakauma on kvartiilien osalta hyvin lähellä tasajakaumaa. Tarkempi jakauman tarkastelu kuitenkin paljas-

taa, että kovin tasaisesti arvot eivät ole jakautuneet (kuvaaja 5.4). Pienempien piikkien lisäksi ääritavut saavat korkeat arvot.



Kuvaaja 5.3 – Pariteetin irroituksen tuottama kvartiilijakauma.

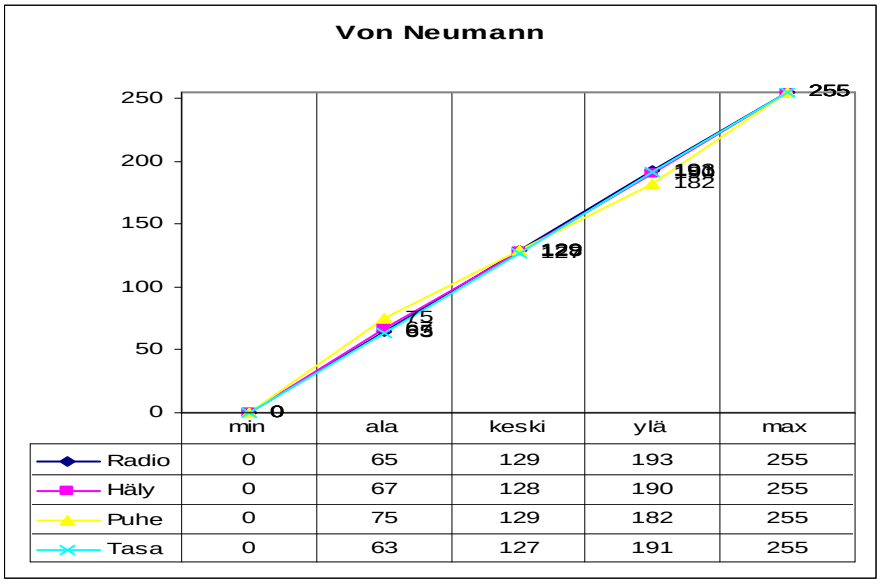


Kuvaaja 5.4 – Taustahälystä pariteetin irroituksella louhittu tulosjakauma.

Louhituista puhenäytteistä on nähtävissä taustahälyn tulosten kaltainen jakautuminen: ääritavut saavat korkeita arvoja ja loput tavut sahaavat pienemmillä arvoilla. Huomattavaa on, että puolet arvoista kohdistuu vain muutamalle ääritavulle. Pariteetin irroittaminen tuntuisi olevan kohtuullisen hyvä menetelmä, mikäli lähdetieto on tiheästi vaihtelevaa ja arvoalueeltaan riittävän laajaa. Menetelmä tuot-

taa kuitenkin kohtuullisen pitkiä saman bittiarvon toistavia merkkijonoja. Hiljaisen äänitiedon kohdan voi tällöin ajatella kallistavan bittijakaumaa jompaan kumpaan suuntaan.

Von Neumannin menetelmä hyödynsi pariteettimenetelmää parantaen toistuvien bittien ongelmaa. Menetelmä voi toki tuottaa toistuvia bittisarjoja, mutta toisto johtuu tällöin epäsuoremmin käsiteltävästä äänitiedosta. Kvartiilit jakautuvat jo hyvin lähelle ideaalisia tasajakauman rajoja. Radiokohina pärjää hitusen huonommin kuin pariteettimenetelmässä, mutta ero ei ole mitenkään merkittävä. Taustahäly pääsee hyvin lähelle tasajakaumaa, eikä puhekaan kauaksi jää. (Kuvaaja 5.5)

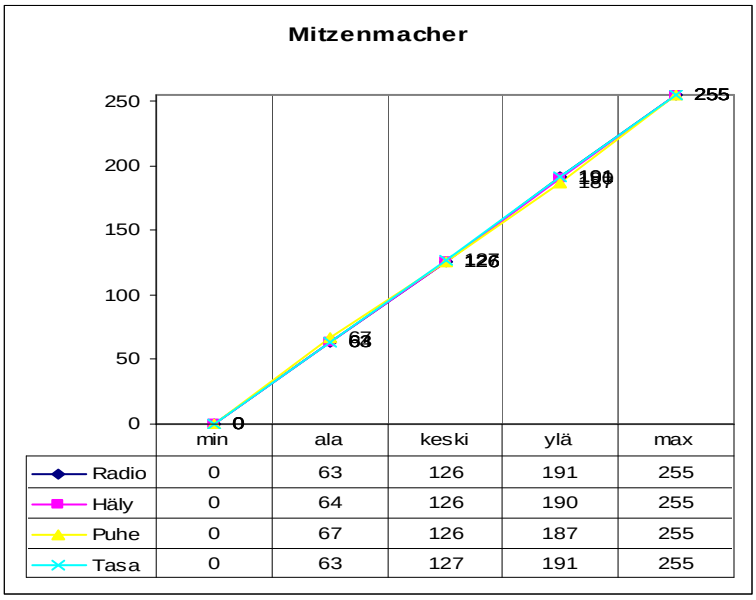


Kuvaaja 5.5 – Von Neumannin menetelmän tuottamat kvartiilit.

Arvojakauman lähempi tarkastelu paljastaa, että radiokohinan pieni heitto tasajakaumasta johtunee vain näyteaineiston koon riittämättömyydestä ja tästä johtuvasta pyöristysvirheestä. Kohinan louhos huojuu nimittäin kahden arvoasteen välillä. Taustahäly ja puhe muodostavat hieman enemmän huojuvan jakauman, joilla ei kuitenkaan ole selkeää keskittymää.

Mitzenmacher-menetelmä jalosti von Neumannin menetelmää pidemmälle ja tuloksista päätellen sangen tehokkaasti. Kuvaaja 5.6 osoittaa, että keskiarvoiltaan

kaikista lähdetyypeistä saatiin louhittua liki tasajakautuneen tilanteen mukaisesti. Ylä- ja alakvartiilitkaan eivät jääneet kauaksi ideaalista. Radiokohinasta saatiin edelleen parhaat tulokset: täysin tasajakautuneen tilanteen mukaiset. Merkkien arvojakaumaa tarkastellessa on huomattavissa, että arvot huojuvat suhteellisen pienellä välillä, radiokohina vähiten ja puhe eniten.

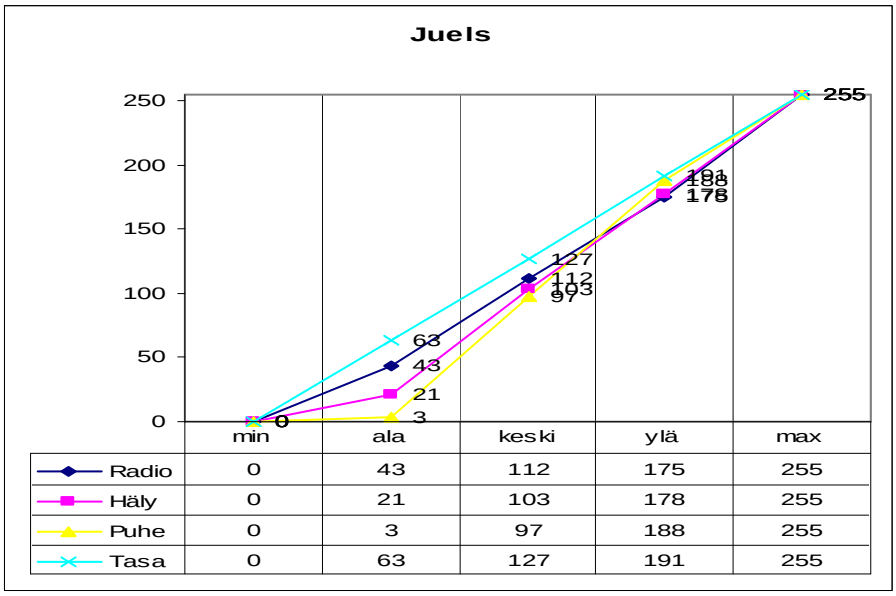


Kuvaaja 5.6 – Mitzenmacher-menetelmän tuottamat kvartiilit.

Mitzenmacher-menetelmän tuottamien tulosten paremmuus von Neumannin menetelmään nähden voi hyvinkin johtua siitä, että Mitzenmacher tuottaa keskimääräisesti enemmän satunnaisia bittejä. Mikäli von Neumannin menetelmälle syötettäisiin enemmän aineistoa, voisi senkin tulosten jakauma lähentyä tasajakaumaa. Tällöin vastaavalla syötemäärällä Mitzenmacher-menetelmä tuottaisi mahdollisesti vieläkin parempaa tulosta. Toki on selvää, että Mitzenmacher-menetelmän tehokkuus (sekä laadullinen että suoritusaikaan pohjautuva) riippuu suuresti valitusta lohkokoosta.

Juelsin tavuja käsittelevä menetelmä on erilainen kuin aiemmin esitellyt, myös tulosten valossa (kuvaaja 5.7). Kvartiilit ovat jakautuneet aiemmista poiketen epäsymmetrisesti keskittyen merkkijoukon alimpiin merkkeihin. Jokaiselle lähteelle tulee arvoja koko arvoalueelta, mutta suuri osa arvoista on keskittynyt alimmille

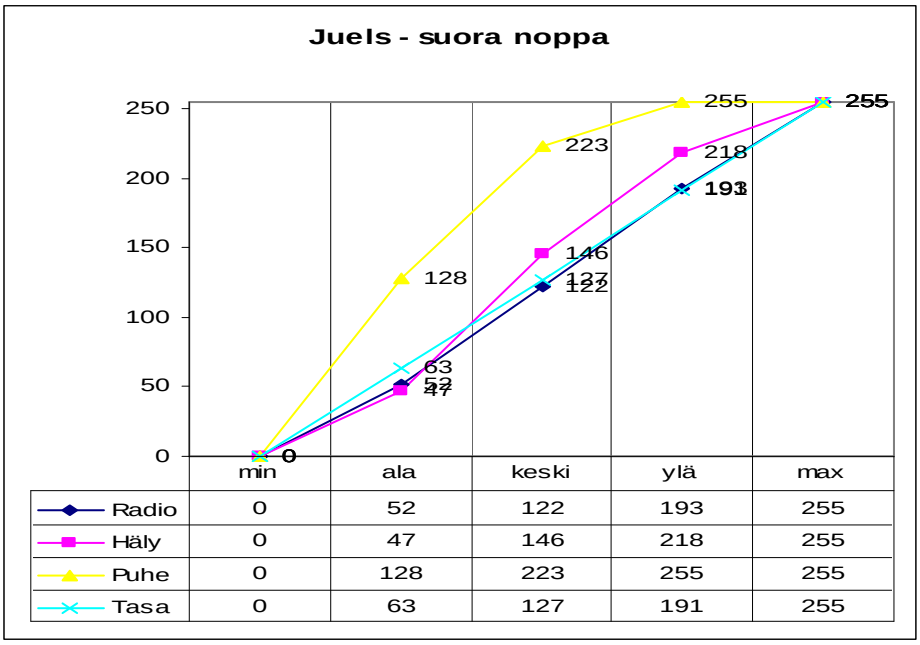
tavuille. Puheen kohdalla neljällä pienimmällä tavulla on neljäsosa arvoista. Radiokohina pääsee alakvartiililillaan jälleen lähimmäksi ideaalia jääden silti ”liian” kauas. Huomion arvoista on, että kaikkien lähteiden kvartiilit jäävät ideaalia pienemmiksi.



	min	ala	keski	ylä	max
Radio	0	43	112	175	255
Häly	0	21	103	178	255
Puhe	0	3	97	188	255
Tasa	0	63	127	191	255

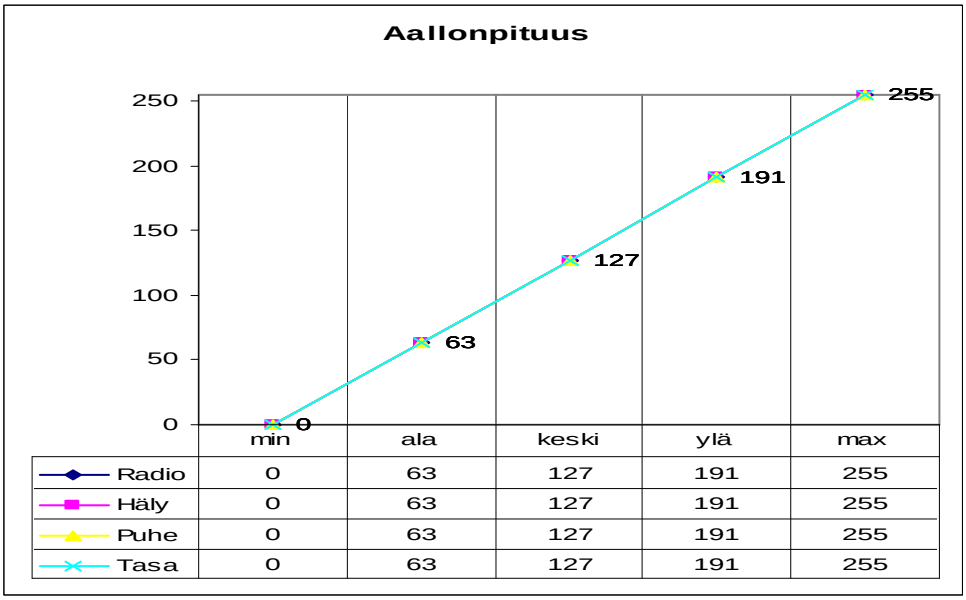
Kuvaaja 5.7 – Juelsin menetelmän tuottamat kvartiilit.

Suoran nopan toteutus Juelsin menetelmästä oli jo toteutettaessa leimattu vähintäänkin epäilyttäväksi, eikä leimaaminen osunut ollenkaan väärään. Kvartiileja analysoitaessa on nähtävissä (kuvaaja 5.8), että radiokohina tuottaisi jopa parempaa tulosta kuin varsinainen Juelsin menetelmä. Taustahälyn ja varsinkin puheen kvartiilit puolestaan kielivät jakauman vinoudesta. Jakaumaa tarkastellessa käy ilmi, että ylin ääritavu saa huomattavan suuren osan kaikkien lähteiden louhosten arvoista.



Kuvaaja 5.8 – Juelsin menetelmästä johdetun suoran nopan menetelmän kvartiilit.

Aallonpituusmenetelmän odotettiin tuottavan hyviä tuloksia, mikäli äänilähde kykeni tuottamaan pituudeltaan riittävän satunnaisesti vaihtelevia aaltoja. Pettymystä menetelmä ei kvartiilien ja merkkien arvojen jakauman suhteen tuottanut. Kvartiilit ovat juuri kuin niiden pitääkin, edes pyöristysvirhettä ei esiinny (kuvaaja 5.9). Jakauma on yhtä tasaista viivaa jokaiselle lähteelle. Näin hyvä tulos herättää välittömästi epäilyksiä. Todennäköistä onkin, että lähdeaineistoa huomattavasti muita menetelmiä enemmän tarvitsevana menetelmänä aallonpituusmenetelmälle pitäisi syöttää reilusti enemmän aineistoa, jotta sen todellinen voimakkuus olisi havaittavissa.



Kuvaaja 5.9 – Aallonpituusmenetelmän tuottamat täydelliset kvartiilit.

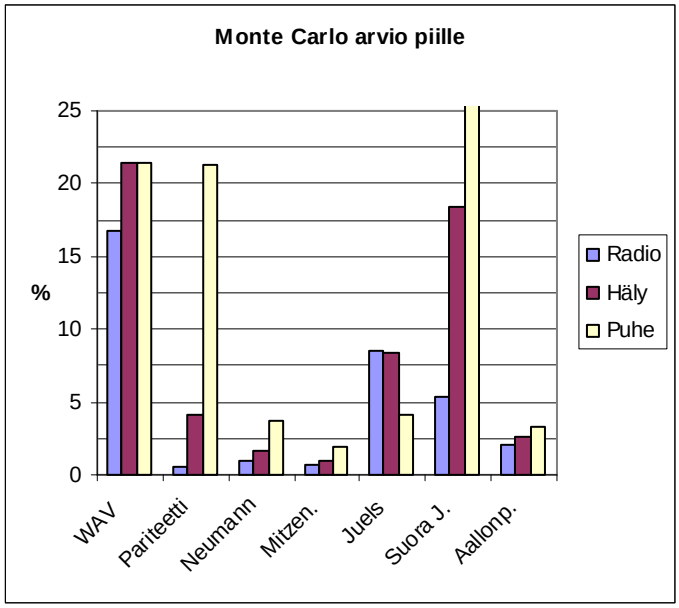
Merkkien jakautumisen perusteella radiokohina on äänilähteistä soveltuvin tuottamaan satunnaisuuden siementä. Sellaisenaan sekään ei kuitenkaan kelpaa, vaan louhinta tuo merkkien jakautumisen valossa huomattavaa kohennusta tulokseen. Louhintamenetelmistä onnistuneimpia tulosten perusteella ovat pariteettien käsittelyyn ja aallonpituuksien mittaamiseen perustuvat menetelmät. Aallonpituusmenetelmän tuottamat tulokset ovat kuitenkin hieman arveluttavia juuri täydellisyytensä vuoksi. Tarvittaisiin enemmän koeaineistoa, jotta tuloksia voisi pitää luotettavimpina. Jo pelkkä von Neumannin menetelmä tuntui tuottavan hienoa jakaumaa, jota Mitzenmacher-menetelmä kykeni vielä parantamaan. Pelkään merkkien jakaumaan ei kuitenkaan voida luottaa.

Monte Carlo -menetelmä piin arvioimiseksi kertoo tarkasteltavan tiedoston merkkien jakautumisesta. Jos jakauma ei ole tasainen, on todennäköistä, että piin arvio on virheellinen. Täysin kohdalleen osuvaa arviota ei oleteta saavutettavan ainakaan keskiarvoisessa analyysissä. Osittain syynä on menetelmän toteutuksessa käytettyjen kokonaislukujen heikko tarkkuus verrattuna liukulukuihin. Tarkkuuden puutetta voisi kompensoida huomattavan suurella näytteiden määrällä, mutta silloin oikeastaan hyödynnettäisiin keskiarvoon perustuvan analyysien niputtamisen heikkoutta.

Ennustamattomuus on oleellinen osa hyvää piin arviota tällä menetelmällä. Mikäli peräkkäiset merkit ovat aina toistensa ympäristöstä, on todennäköistä, että arviosta tulee huono. Tarkasta arviosta voi saada viitteitä ennustamattomuudesta ja myös kompleksisuuspohjaisesta satunnaisuudesta, sekä jaksollisuudesta. Menetelmässä on kuitenkin heikkouksia, joiden kompensoimiseksi tarvitaan myös muita analyysimenetelmiä. Monte Carlo -menetelmä piin arvioimiseksi ei anna todisteita, vaan viitteitä.

Menetelmä voi tuottaa hyviä tuloksia huonosti jakautuneella aineistolla. Menetelmässä lasketaan vain osumat ja kaikki (x,y) -parit. Tällöin hyvän tuloksen kaikki osumat voivat ollakin samaan (x,y) -pisteeseen tulleita. Arvion virheprosentti voi tällöin olla pieni, mutta jakauma voi silti olla hyvin vino. Virheprosentin ollessa nolla, arviossa on saavutettu piin arvo. Virheprosentilla ei ole ylärajaa, vaan virheprosentti kuvaa kerrointa, joka erottaa arvion piin arvosta.

Kuvaajaan 5.10 on koottu keskiarvoiset virheprosentit jokaisesta lähteestä kullekin louhintamenetelmälle. Louhimattomille ääninäytteille (WAV) menetelmä tuotti oletetun korkeat virheprosentit. Radiokohina tuotta hieman tarkemman arvion, mutta ei mitenkään merkittävästi. Aivan kuten kvartiilianalyysin perusteella, pariteettimenetelmä tuottaa hyviä tuloksia radiokohinasta, jossain määrin heikompia taustahälystä ja huonoja tuloksia puhenäytteistä.



Kuvaaja 5.10 – Monte Carlo -menetelmän keskimääräiset virheprosentit.

Von Neumannin menetelmä ei radiokohinasta saa aivan pariteettimenetelmän veroista arviota aikaiseksi, mutta eroa ei merkittävästi ole. Oleellisempaa on, että muista äänilähteistä saa paljon parempia arvioita. Puhenäytteidenkin virheprosenttien keskiarvo on jo alle neljä prosenttiyksikköä. Mitzenmacher-menetelmä parantaa – kuten kvartiilianalyysissakin – von Neumannin menetelmän tuottamia tuloksia. Pariteetteja käyttävistä menetelmistä Mitzenmacher-menetelmä suoriutuu tästä analyysista selkeästi muita paremmin juuri tasaisuutensa ansiosta.

Juelsin menetelmä tuotti poikkeuksellisen tuloksen. Puhenäytteiden louhinta näytti tuottavan tarkemman arvion piille kuin radiokohina ja taustahäly. Häly päihitti kohinan alle puolella prosenttiyksiköllä, mutta louhitulla puheella virheprosentti oli noin puolet edellisiä pienempi (prosenttiyksikköinä). Suoran nopan menetelmä puolestaan sai aikaan virheprosenteille muiden menetelmien kaltaisen jakauman, jossa radiokohinalla saavutettiin paras tulos ja puheella heikoin. Puhenäytelouhoksilla saatu virheprosentti oli koko aineiston suurin, yli 68 prosenttia.

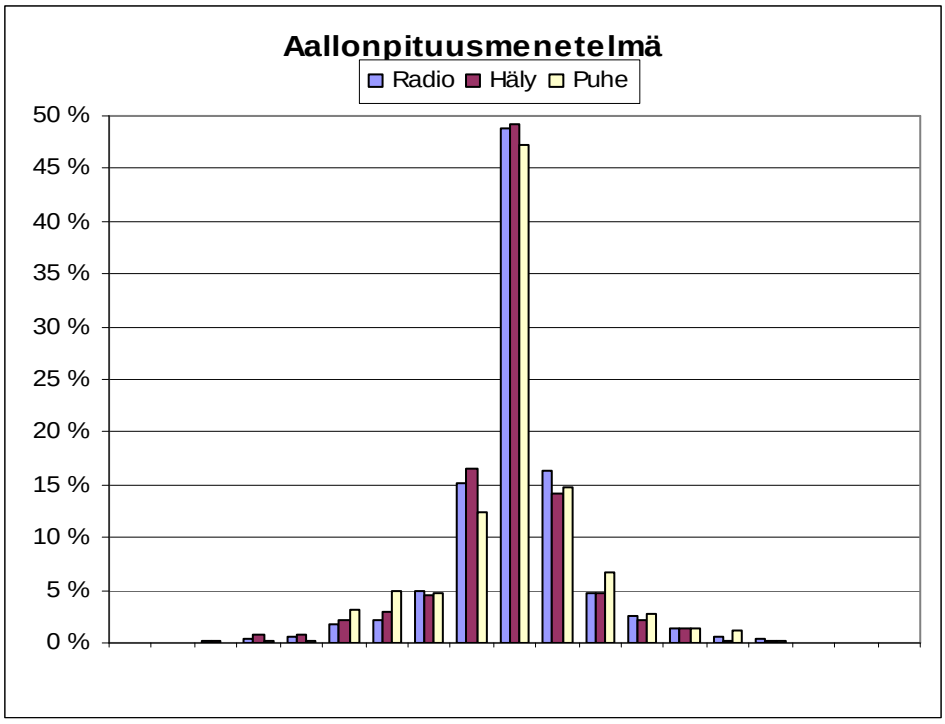
Aallonpituusmenetelmän virheprosentit ovat melko hyviä. Äänilähteiden välillä ei ole suurta eroa – vain noin 1,1 prosenttiyksikköä. Mitzenmacher-menetelmälle aallonpituus käytettävissä olevalla aineistolla häviää, joskin eroa radiokohinan ja

puhenäytteiden välillä Mitzenmacher-menetelmässä on enemmän (noin 1,3 prosenttiyksikköä). Aallonpituusmenetelmä kärsii pienestä näytemäärästä.

Merkkien jakaantumista voidaan vielä tarkastella laskemalla tiedostoille Khiin neliö. Tiedostosta lasketaan absoluuttinen Khiin neliö ja sille selvitetään todennäköisyys, jolla aidosti satunnainen tavusarja sen ylittää. Absoluuttiset prosenttiarvot on sijoitettu ja pyöristetty luokkiin, jotta analysointi ja tulosten vertaaminen olisi mielekkäämpää.

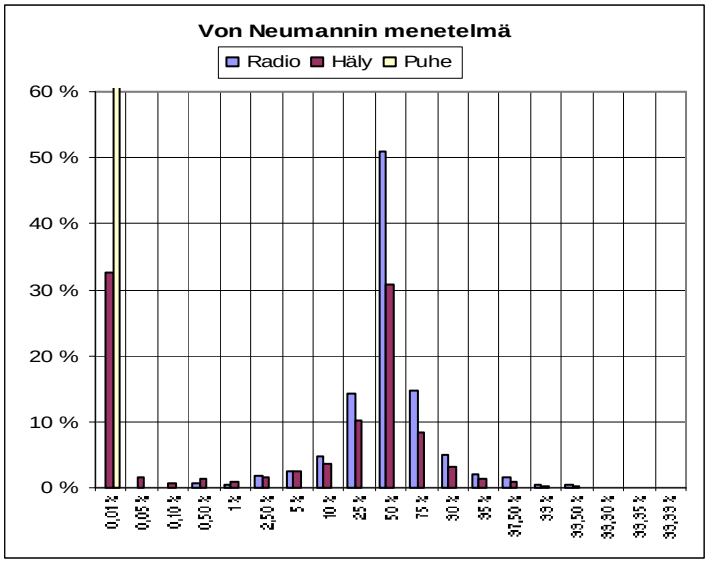
Khiin neliö tässä yhteydessä on oikeastaan toinen tapa esittää merkkien jakaantumista. Menetelmä havainnoi jakauman epätasaisuutta, mutta ei huomioi tarkemmin, missä kohdissa jakauma huojuu. Koska jokaiselle tiedostolle saatiin yksi prosenttiarvo, laskettiin jokaista äänilähde-louhintamenetelmä -paria kohden prosenttiluokituksen alkioiden lukumäärät yhteen. Näistä muodostettiin suhteelliset arvot, jotta menetelmien ja lähteiden vertaaminen toisiinsa onnistuu.

Louhimattomat äänitiedostot eivät odotetusti testissä pärjää: kaikki näytteet luokittuivat pienimpään prosenttilukuun. 0,01 prosenttia ihannetuloksista ylittää saadut arvot. Juelsin ja suoran nopan menetelmä eivät kykene tuottamaan yhtään parempaa tulosta. Kaikki näytteet luokittuvat näilläkin pienimpään prosenttiluokkaan. Aallonpituusmenetelmä pärjää todella hyvin tässä analyysissa. Luokitusten arvojen jakauma keskittyy 50 prosentin luokkaan, joskin kaikkia arvoja esiintyy (kuvaaja 5.11). Erot ääninäytteiden lukumäärässä aiheuttavat eroa lähteiden välille, mutta vaikuttaisi siltä, että Khiin neliön laskennassa suurta eroa äänilähteiden välillä ei ole.



Kuvaaja 5.11 – Aallonpituusmenetelmästä lasketut Khiin neliön prosenttiluokitukset.

Muut menetelmät onnistuvat louhimaan radiokohinasta Khiin neliön kannalta hyviä tuotoksia. Pariteetin irroituksella ja von Neumannin menetelmällä kohinan luokkajakauma on aallonpituusmenetelmän kaltainen. Puhenäytteet kuitenkin pettävät, ja kummassakin ne luokittuvat huonoimpaan luokkaan. Pariteettimenetelmällä louhitusta taustahälystä vajaa puolet luokittuu heikoimpaan luokkaan, mutta myös optimaalisia tuloksia löytyy. Jakauma paranee von Neumannin menetelmällä (kuvaaja 5.12).



Kuvaaja 5.12 – Khiin neliöprosenttien luokitukset Von Neumannin menetelmälle.

Mitzenmacher-menetelmä kykenee työstämään puhenäytteistäkin jonkin verran onnistuneempaa materiaalia. 0,01-luokassa on ”vain” noin 65 prosenttia kaikista arvoista ja optimaaliseen luokaan kuuluu 0,6 prosenttia arvoista. Radiokohina ja taustahäly tuottavat hieman samankaltaisen jakauman kuin von Neumanninkin menetelmä. Jakauma suosii enemmän alle viidenkymmenen prosentin luokkia, eli se ei ole yhtä symmetrinen kuin muilla pariteettimenetelmillä. Heikkoja arvoja on kuitenkin huomattavasit vähemmän, joten suurempi osa arvoista on keskittynyt vähintäänkin siedettäviin luokkiin.

Merkkien jakautuminen arvoalueelleen tuntuisi nostavan sekä Mitzenmacher- että aallonpituusmenetelmän toisten edelle. Mitzenmacher-menetelmän vahvuus kvartiilitarkastelussa ja piin arvon arvioinnissa perustuu tuotettujen satunnaisbittien suureen määrään. Menetelmä päihittää analyyseissa von Neumannin menetelmän suurempien tulossarjojen ansiosta. Satunnaisuuden vahvuuden voidaan arvella olevan molemmissa menetelmissä samaa luokkaa, mutta kompleksinen ja jaksollinen satunnaisuus vahvistuu, kun saadaan pitkiä ja satunnaisuudeltaan vahvoja tulossarjoja.

Khiin neliön analysointi vahvistaa Mitzenmacher-menetelmän pätevyysodotuksia. Vaikka muut menetelmät saattaisivatkin tuottaa parempia yksittäisiä tuloksia, pystyy Mitzenmacher-menetelmä louhimaan lähteestä riippumatta tasaisen hyviä tuloksia. Oikeastaan pitäisi sanoa, että Mitzenmacher-menetelmä tuottaa hyvin vähän huonoja tuloksia. Satunnaisuuden louhinnassa tärkeämpää on olla tuottamatta huonoja tuloksia kuin tuottaa toisinaan huikean hyviä tuloksia.

Aallonpituusmenetelmä on näiden kokeiden valossa kyseenalainen, sillä se vaikuttaisi tuottavan hyviä tuloksia, mutta näytemäärät ovat usein liian pieniä kunnollisen tarkkailun ja vertaamisen tekemiseen. Aallonpituusmenetelmää ei silti tule täysin tyrmätä, sillä selkeästi sillä näyttää olevan potentiaalia.

Aallonpituusmenetelmälle kvartiilianalyysi povaa menestystä satunnaisuusvertailuissa. Piin arviointi on kuitenkin herkempi sarjan pituudelle, jolloin pitkiä sarjoja tarvitseva aallonpituusmenetelmä ei toimi moitteettomasti. Juelsin menetelmällä ja sen johdannaisella tuntuisi olevan vaikeuksia pärjätä edes käsittelemättömälle äänitiedolle näissä analyyseissa. Käsittelemättömästä äänitiedosta saatavaa jakaumaa voidaan sentään säädellä äänen voimakkuutta tasaamalla, mutta Juelsin menetelmän tuottamat jakaumat sisältävät arvoja saamattomia merkkejä, jolloin varsinainen arvoalue kutistuu pieneksi.

5.1.2. Merkkien toistuminen

Merkkien toistumista tarkastellaan kahden Run-testin – eli juoksukokeen – ja korrelaatiokertoimen avulla. Ensimmäinen juoksukoe, Run, selvittää peräkkäisiä saman merkin toistumisia. Saatuja tuloksia verrataan alakohdassa 4.5.3 esitettyihin ideaaleihin raja-arvoihin. Toinen juoksukoe, Prun, puolestaan tarkastelee analysoitavan tiedoston ”aaltomuotoa”. Testi mittaa peräkkäisten arvojen kasvun tai pienemisen tuottamia juoksuja. Tämä toinen juoksukoe on ensimmäistä kyseenalaisempi, sillä teoreettisten rajojen laskeminen ei ole ongelmatonta. Korrelaatiokerroin ilmaisee koko tiedoston taipumuksen noudattaa säännöllistä trendiä.

Run-testissä radiokohinan louhinta osoittautuu kaikista tuottoisimmaksi. Louhimattomat kohinanäytteetkin sisältävät hyvin vähän saman merkin toistoa. Ihannerajoja ei aivan saavuteta. Yhden (98,4 %, ihanne 99,6 %) ja kolmen (0,03 %, ihanne 0,02 %) merkin mittaisia juoksuja on ihannetulosta vähemmän ja kahden mittaisia enemmän (1,6 %, ihanne 0,4 %). Yli kolmen merkin mittaisia juoksuja ei ole lainkaan.

Louhintamenetelmistä muut kuin Juelsin menetelmä ja sen johdannainen selviytyvät hyvin Run-kokeesta radiokohinan osalta. Merkkejä toistuu keskimäärin lähes ihannerajojen verran. Lähimmäksi ihannerajoja pääsee Mitzenmacher-menetelmä, mutta myös muut pariteettimenetelmät ja aallonpituusmenetelmä ovat hyvin lähellä. Juelsin menetelmä johdannaisineen selviytyy muita louhintamenetelmiä heikommin, mutta silti paremmin kuin louhimaton radiokohina.

Louhimaton taustahäly ei mitenkään pärjää radiokohinalle. Louhimattomasta taustahälystä yhden merkin mittaisia toistoja löytyy enää vain 68,3 prosenttia. Kahden mittaisia on 18,2 prosenttia ja kolmenkin mittaisia 6,6 prosenttia. Toistojen pituuksien kasvaessa niiden määrä luonnollisesti vähenee, mutta ilmeisen hitaasti. Toistoista 0,1 prosenttia on pituudeltaan yli kaksikymmentä merkkiä.

Taustahäly aiheuttaa louhintamenetelmien välille jo jonkinlaista eroa. Pariteetin irroituksella enää vain 98,8 prosenttia toistoista on yhden merkin mittaisia, ja pisin toistosarja on keskimäärin seitsemän merkin mittainen. Mitzenmacher-menetelmä vie niukan voiton von Neumannin menetelmästä, mutta molemmat ovat hyvin lähellä ihannerajoja. Kummankin toistot eivät keskimäärin ole kahta merkkiä pidempiä, eivätkä kummankaan ihannetoistot sitä ole. Aallonpituusmenetelmä kärsii jälleen kenties liian pienestä näytteiden määrästä, vaikka ihannerajojen lähellä onkin. Juelsin menetelmät ovat edelleen louhintamenetelmistä heikoimpia mutta voittavat käsittelemättömän taustahälyn.

Käsittelemätön puhuttu ääni ei odotettavasti uhkaa toisia äänilähteitä millään tavalla. Merkki jää toistumatta kertaakaan vain 43,8 prosentin todennäköisyydellä. Yli kahdenkymmenen merkin toistoja on jo 4,1 prosenttia. Suoran nopan mene-

telmä hajoaa täysin puheääntä käsitellessä. Yhden merkin toisto tapahtuu 83,4 prosenttisesti ja yli kahdenkymmenen merkin toistoja on 0,2 prosenttia. Juelsin menetelmä kykenee hieman parempaan. Merkki jää toistumatta kertaakaan 93,4 prosentin todennäköisyydellä ja keskimäärin pisin toisto on yksitoista merkkiä.

Aallonpituusmenetelmä pääsee jälleen hyvin lähelle ihannerajoja, mutta näytteiden yhä pienempi määrä hieman kyseenalaistaa tuloksen tarkkuutta. Pariteetin irroitus pärjää paremmin kuin suoran nopan menetelmä mutta heikommin kuin Juelsin varsinainen menetelmä. Yli kahdenkymmen mittaisia toistosarjoja on 0,2 prosenttia sarjoista, mutta merkin toistumatta jääminen on hieman todennäköisempää (89,5 %).

Von Neumannin menetelmä on hyvin lähellä ihannerajoja: 99,5 ja 0,5 prosenttia, eikä yli kahden merkin mittaisia toistoja keskimäärin ole. Näytteiden vähyys kuitenkin haittaa tulosten tarkempaa vertailua. Mitzenmacher-menetelmä kykenee tuottamaan von Neumannin menetelmää enemmän tulosarvoja, mutta se pyörii hyvin tarkalleen samoissa lukemissa.

Run-juoksukokeen perusteella radiokohina osoittautuu äänilähteistä parhaimmaksi. Louhintamenetelmistä von Neumannin ja Mitzenmacherin menetelmät erottuivat edukseen. Aallonpituusmenetelmä on vahvasti edellä mainittujen menetelmien joukossa, joskin näytteiden vähyys haittaa selvästi tarkempaa vertailua. Pariteettimenetelmä on taipuvainen tuottamaan toistuvia merkkejä, jos äänilähde itsessään tuottaa heikosti vaihtelevia tulostiedostoja. Juelsin menetelmä ja suoran nopan metodi luovat toistuvia merkkijonoja mutta sentään parantavat äänilähteiden tulosta.

Prun-juoksukoe ei tuloksia tarkastellessa osoittaudu kovinkaan hedelmälliseksi. Pariteettipohjaiset louhintamenetelmät, Juelsin varsinainen menetelmä ja aallonpituusmenetelmä tuottivat lähes identtisen jakauman erimittaisille ylös-alas -juoksuille. Ainoastaan tasakohdissa Juelsin menetelmä jäi muista hitusen jälkeen. Suoran nopan menetelmä tuotti jälleen kohtalaisia tuloksia mutta jäi tasoltaan muiden

louhintojen alle. Tasakohtia esiintyi vähiten aallonpituusmenetelmällä, mutta jälleen analysoitavien louhostiedostojen lyhyys vaikuttaa.

Korrelaatiokerroin paljastaa oletetusti louhimattomien äänitiedostojen heikkouden (kuvaaja 5.13). Radiokohinan korrelaatiokerroin 0,76 ei todellakaan ole korrelaatioanalyysin kannalta hyvä. Puhenäytteet tuottavat keskimääräisen korrelaatioarvon 0,995. Puhenäytteiden merkit ovat siis hyvin säännönmukaisesti järjestäytyneet, eikä ihme.

	Radio	Häly	Puhe
WAV	0,75873448	0,928624	0,994903
Pariteetti	-0,000126	0,036462	0,449917
Neumann	0,00023563	-0,00101	-0,00054
Mitzenm.	0,00226587	0,001328	0,000277
Juels	-0,0207557	0,021964	0,262623
Suora	-0,0815595	0,074859	0,19923
Aallonp.	-0,0013731	-0,00238	-0,00461

Kuvaaja 5.13 – Keskimääräiset korrelaatiokertoimet.

Pariteetin irroitus radiokohinasta tuottaa jo todella hyvän korrelaatiokertoimen: -0,000126. Valitettavasti taustahälystä saadaan aikaan jonkin verran heikompi tulos (0,0365) ja puhenäytteistä huomattavan heikko (0,45). Korrelaatiokertoimia vertaamalla von Neumannin menetelmä päihittää Mitzenmacher-menetelmän. Ei mitenkään suuresti, mutta päihittää silti. Juelsin ja suoran nopan menetelmät tuottavat melko siedettävät korrelaatiokertoimet. Aallonpituusmenetelmä toimii Mitzenmacher-menetelmää tehokkaammin radiokohinalle mutta häviää muissa äänilähteissä.

Merkkien toistuvuutta tarkastellessa Mitzenmacher-menetelmä erottautuu von Neumannin menetelmän kanssa muista louhintatavoista. Kyseiset menetelmät kykenevät hyvin rikkomaan aaltomuodon tiedostoista. Mitzenmacher-menetelmän heikkous merkkien toistuvuudessa on, että se tavallaan luo ylimääräisiä bittejä olemassaolevien bittien perusteella. Tällöin ei voida olla täysin varmoja, etteivätkö ylimääräiset bitit olisi hieman painottuneita. Menetelmä on joka tapauksessa vahvoilla.

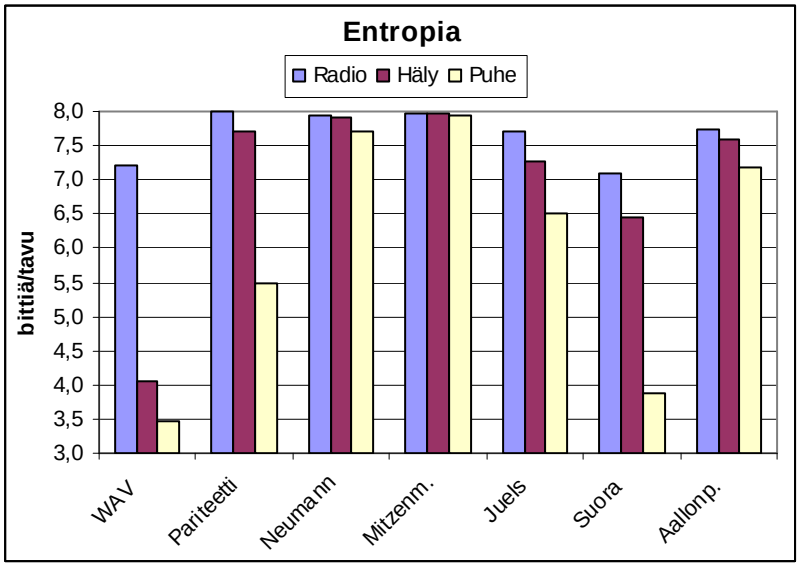
Aallonpituusmenetelmä kärsii myös merkkien toistuvuutta tarkasteltaessa kohtuullisen pienestä tulostiedoston koostaan. Materiaalia ei vain yksinkertaisesti riitä pätevän analyysin suorittamiseksi. Juelsin menetelmä pärjää paremmin tässä tarkastelussa kuin merkkien jakautumisessa. Suoran nopan menetelmä erottuu muista, mutta ei edukseen.

Jos siemenenä käytettävän merkkijohon ei haluta sisältävän peräkkäisiä saman merkin toistoja, äänitiedostoja ei voida sinällään käyttää. Vaikka jakauma olisikin liki täydellinen, tuo merkkien aaltomuotoisuus mukanaan ennalta-arvattavuuden.

5.1.3. Jaksollisuus

Tiedostojen entropian laskeminen selvittää merkkien jakatumista ja jaksollisuutta. Jos merkit ovat jakautuneet lähes tasaisesti arvoalueelleen eivätkä muodosta toistuvia alijaksoja, tarvitaan yhden tavun muodostamiseen kahdeksan bittiä. Mikäli tutkittavan tiedoston laatu on heikompi, tarvitaan tavun muodostamiseen keskimäärin vähemmän bittejä. Menetelmä ei kuitenkaan paljasta, kummalla tapauksella on suurempi vaikutus: onko merkkien jakauma vino, onko toistuvia merkkijonoja vai onko molempien vaikutus yhtä suuri. Entropiaan liittyy muitakin aiemmin mainittuja ongelmia, joiden takia täydellistä tulosta on syytä epäillä.

Äänilähteistä radiokohina odotetusti sisältää parhaimman potentiaalin suureen entropia-arvoon. Käsittelemättömän kohinan entropia on jopa suurempi kuin suoran nopan menetelmällä louhitun kohinan: käsittelemättömänä entropia on noin 7,203 bittiä tavua kohden, kun suoralla nopalla saavutetaan vain 7,106. Kymmenyksen ero on suuri tiedostojen ollessa kookkaita. Pariteetin irroitus louhii radiokohinan entrooppisimmaksi keskimääräisen arvon ollessa 7,987. Kuvaajaan 5.14 on havainnollistettu entropia-arvot lähde- ja louhintakohtaisesti.



Kuvaaja 5.14 – Entropia-arvot.

Pariteetin irroitus ei kuitenkaan toimi hyvin muilla äänilähteillä. Taustahälyn tulos on vielä jotenkuten siedettävä (7,72), mutta puhenäytteiden entropioiden keskiarvo on huono. Von Neumannin menetelmän tuottaman tuloksen entropia eri lähteiden välillä on melko tasainen (radio 7,946; taustahäly 7,9; puhe 7,697). Mitzenmacher-menetelmän tulokset ovat lähteiden välillä hyvin tasaiset. Radikohinasta louhitaan keskimääräiseksi entropia-arvoksi 7,976 ja puhenäytteistä arvo 7,935. Muista menetelmistä mikään ei yllä äänilähteiden keskinäisessä vertailussa näin hyviin arvoihin.

Juelsin ja suoran nopan menetelmä eivät pärjää muille louhintamenetelmille, eikä suoran nopan menetelmä pärjää aidolle Juelsin menetelmälle. Aallonpituusmenetelmä häviää von Neumannin ja Mitzenmacherin menetelmille, mutta tasaisuudellaan se vie voiton pariteetin irroituksesta.

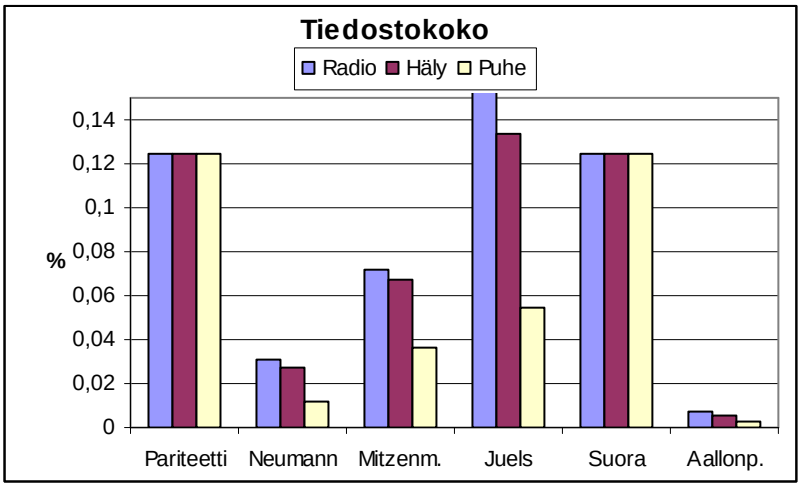
Aiempien analyysien perusteella voidaan olettaa, että Mitzenmacher-menetelmän saavuttaman hyvä entropiatulos ei johdu entropia-analyysin heikkouksista. Merkkien jakautuminen tasaisesti arvoalueelleen ja juoksukokeet tuottivat menetelmälle hyviä tuloksia, joiden avulla on helppo vakuuttua satunnaisuuden vahvuudesta. Von Neumannin menetelmä vaikuttaa vain aavistuksen heikommalta. Aallonpituusmenetelmä kärsii entropiatarkastelussakin pienestä tulosjoukon koosta.

5.2. Määrällinen satunnaisuus

On ainoastaan hyödyllistä, jos louhittavasta aineistosta saadaan mahdollisimman paljon laadukasta satunnaisuutta. Useimmissa käyttökohteissa on vaarattomampaa tuottaa liikaa satunnaisuutta kuin liian vähän. Tehokkuus asettaa kuitenkin rajansa, jolloin on tärkeää saada louhittua satunnaisuutta mahdollisimman suurella hyötyprosentilla.

Satunnaisuuden määrää voidaan tämän tutkimuksen tuotoksista tarkastella tiedostokokojen suuruuksista ja satunnaisuuden laadukkuudesta. Tässä tarkastellaan vain tiedostokokoja, ja yhteenveto jätetään lukuun 6.

Kuvaajaan 5.15 on havainnollistettu eri louhintamenetelmien aikaansaamat tiedostokokojen muutokset. Arvot vertautuvat louhimattoman näytetiedoston kokoon, joka kaikilla äänilähteillä on 110 250 tavua. Pariteetin irroitus ja suoran nopan menetelmä tekevät vakiokokoisia louhoksia. Pariteetti irroitetaan jokaisesta tavusta, jolloin tiedostokoko pienee kahdeksasosaan. Suora noppa puolestaan muodostaa kahdeksasta tavusta yhden tulostavun, joten tiedostokoko pienenee silläkin kahdeksasosaan. Muut menetelmät tuottavat vaihtelevan mittaisia tulostiedostoja.



Kuvaaja 5.15 – Louhittujen tiedostojen keskimääräisten tiedostokokojen suhteet.

Myös Juelsin menetelmä lukee kahdeksan tavua kerrallaan, mutta se saattaa saada muodostettua niistä pitkiäkin bittijonoja. Radiokohinasta Juelsin menetelmä kykenee saamaan aikaiseksi keskimäärin noin 24 500 tavun mittaisia merkkijonoja. Muidenkin äänilähteiden osalta Juelsin menetelmä on vaihtelevakokoista tulostetta tekevistä menetelmistä tehokkain.

Aallonpituusmenetelmä osoittautuu tehottomimmaksi. Menetelmähän mittaa peräkkäisten aaltojen pituuksia, jolloin se voi kuluttaa huomattavan määrän tavuja ennen yhden tulosbitin muodostamista. Von Neumannin menetelmä karsii huomattavan määrän tavuja pois pariteetin irroitukseen verrattuna. Mitzenmacher-menetelmä kykenee reilusti kaksinkertaistamaan tulostavumäärän verratuna von Neumannin menetelmään.

6. Yhteenveto

Tämän tutkimuksen tuloksia täytyy tarkastella sillä oletuksella, että esitettyjä ja käytettyjä äänilähteitä ja louhintamenetelmiä käytetään säännöllisesti. Puhuttu ääninäyte käy siemenestä satunnaislukugeneraattorille, jos sitä käytetään hyvin harvoin eikä sen generoimisen menetelmää tahattomastikaan paljasteta ulkopuolisille.

Äänilähteiden tuotoksia ei ilman louhintaa voi suositella käytettäväksi vahvaa satunnaisuutta tarvitsevissa järjestelmissä. Aaltomuoto ja arvojen keksittyminen nolla-akselin läheisyyteen tekee radiokohinastakin ennalta-arvattavaa. Äänilähteeseen täytyy silti kiinnittää huomiota. Tuloksista näkyy, että radiokohina on käytetyistä lähteistä ehdottomasti paras. Muita äänilähteitä haittasi äänen tason säätelyssä ilmenneet ongelmat. Täysin varmasti ei voida sanoa, etteivätkö muutkin lähteet olisi pärjänneet paremmin, jos äänenvoimakkuus olisi saatu jotenkin tasattua. Puhenäytteiden osalta on kuitenkin muistettava, että ellei näyte ole kurkku suorana huutamista tai muuta tasaista äännähtelyä, syntyy signaaliin hiljaisempia kohtia.

Louhintamenetelmistä Mitzenmacher-menetelmä erottui edukseen. Jo pariteetin irroitus tuotti muutamassa analyysissa kohtuullisen hyvää tulosta, ja kun sitä parannettiin von Neumannin menetelmän kautta Mitzenmacher-menetelmäksi, olivat tulokset hyvää luokkaa jo useissa analyyseissa. Mitzenmacher-menetelmä ei pärjännyt huonosti yhdessäkään laadullisessa analyysissa. Määrällisesti se olisi voinut olla tuotteliaampi. Tutkimus osoittaa, että Mitzenmacher-menetelmä todellakin on parannus von Neumannin menetelmään. Von Neumannin menetelmä pärjäsi yllättävänkin hyvin menetelmäksi, jolla ei ole sisäistä muistia.

Juelsin menetelmän heikkoja tuloksia saattaa selittää parametrien sopimattomuus louhinta-aineistoon. Lohkolouhijana Juelsin menetelmä olisi voinut pärjätä paremmin suuremmalla lohkokoolla. Menetelmän algoritmien raskaus lohkon kasvaessa kuitenkin rajoittaa menetelmän hyödyntämistä jonkin verran. Täysin hyö-

dyttömäksi menetelmää ei tule tuomita, vaan lisäkokeita tulisi tehdä erilaisella aineistolla ja lohkojen rakenteella.

Suoran nopan kokeilumenetelmä osoittautui hyödyttömäksi. Kyseessähän oli liukulukujen pakottamista suljettuun kokonaislukuarvoalueeseen. Pyöristäminen pilaa aina tarkkuuden. Menetelmästä saisi mahdollisesti paremman, mikäli sillä keskittyisi heittämään 256-tahoisen nopan sijasta binäärinoppaa.

Aallonpituusmenetelmä osoittautui yllättävän tehokkaaksi ainakin pintapuolisesti. Huikea suoriutuminen joistain analyyseista selittyy kuitenkin tiedostokokojen pienuudella ja materiaalin vähäisyydellä. Mikäli menetelmä pääsisi louhimaan kestoltaan huomattavasti pidempää ääninäytettä, voisi tuloksiinkin helpommin luottaa. Filosofialtaan aallonpituusmenetelmä on pariteettimenetelmän kanssa saman ilmiön äärellä. Onko ilmiö pidempi vai lyhyempi vertautuu kysymykseen merkityksettömimmän bitin arvosta.

Tutkimuksen tuloksen voi tiivistää lyhyesti: jos käytettävissä on esitetyt ääniläh-teet ja louhintamenetelmät, satunnaislukugeneraattorin siemen kannattaa louhia Mitzenmacher-menetelmällä radiokohinasta. Aallonpituusmenetelmää kannattaa kuitenkin harkita, varsinkin jos louhinta täytyy suorittaa merkkivirrasta.

Lähteet

[Agnew, 1986] G. B. Agnew, Random sources for cryptographic systems. In: *Advances in Cryptology: CRYPTO '85 Proceedings* (1986), 77-81.

[Bays and Durham, 1976] C. Bays, S. Durham, Improving a poor random number generator. *ACM Transactions on Mathematical Software* **2** (1976), 59-64.

[Chaitin, 1975] G. Chaitin, Randomness and mathematical proof. *Scientific American* **235**(5) (1975), 47-52.

[Eagini and Buce, 1999] V. Eagini, M. Buce, A design of reliable true random number generator for cryptographic applications. In: C. K. Koc and C. Pear (eds.), *CHES'99, LNCS* **1717** (1999), 204-218.

[Eastlake *et al.*, 1994] D. Eastlake, S. Crocker, J. Schiller, Randomness recommendations for security. Request For Comment, **RFC1750**, December 1994.

[Erber and Putterman, 1985] T. Erber, S. Putterman, Randomness in quantum mechanics – nature's ultimate cryptogram? *Nature* **318** (1985), 41-43.

[Goodrich and Tamassia, 2001] Michael T. Goodrich, Roberto Tamassia, *Data Structures and Algorithms in Java, 2nd Edition.* Wiley & Sons., 2001.

[Gorenstein, 1967] S. Gorenstein, Testing a random number generator. *Communications of the ACM* **10**(2) (February 1967), 111-118.

[Haahr, 1999] M. Haahr, Introduction to Randomness and Random Numbers (June 1999), http://random.org/essay.html [23.8.2006]

[Hellekalek, 1998] P. Hellekalek, Good random number generators are (not so) easy to find. *Mathematics and Computers in Simulation* **46** (1998), 485-505.

[Juels *et al.*, 2000] A. Juels, M. Jakobson, E. Shriver, B. K. Hillyer, How to turn loaded dice into fair coins. *IEEE Transactions on Information Theory* **46**(3) 911-921 (2000).

[Kolehmainen, 2001] S. & K. Kolehmainen, *Java – Algoritmit ja mallit*. IT Press, 2001.

[Mitzenmacher, 2001] M. Mitzenmacher, Unbiasing Random Bits (2001), *Teaching Notes from Computer Science* **124** – *Data Structures and Algorithms*, www.fas.harvard.edu/~libes124/cs124/124.html [7.4.2006]

[Monrose *et al.*, 2001] F. Monrose, M. K. Reiter, Q. Li, S. Wetzel, Cryptographic key generation from voice. In: *Proceedings of the 2001 IEEE Symposium on Security and Privacy, Oakland, CA.* (May 2001).

[Monrose *et al.*, 2002] F. Monrose, M. K. Reiter, Q. Li, D. P. Lopresti, C. Smith, Towards Speech-Generated Cryptographic Keys on Resource Constrained Devices (Extended Abstract). In: *Proceedings of the 11th USENIX Security Symposium* (August, 2002).

[Morris and Thompson, 1979] R. Morris, K. Thompson, Password security: a case history. *Communications of the ACM* **22** (November 1979), 594-597.

[Pierce, 1989] A. D. Pierce, *Acoustics: An Introduction to Its Physical Principles and Applications.* American Institute of Physics, 1989.

[Ritter, 1991] T. Ritter, The efficient generation of cryptographic confusion sequences. *Cryptologia* **15**(2) (1991), 81-139.

[Schneier, 1996] B. Schneier, *Applied Cryptography, 2nd Edition.* Wiley & Sons, 1996.

[Sipser, 1983] M. Sipser, A complexity theoretic approach to randomness. In: *Proceedings of the 15th ACM Symposium on Theory of Computing (1983)*, 330-335.

[Walker, 1998] J. Walker, RPKP Experiments: Probability and Statistics (1998), http://www.fourmilab.ch/rpkp/experiments/statistics.html [23.8.2006].

[Watt, 2000] A. Watt, *3D Computer Graphics, 3rd Edition*. Addison-Wesley, 2000.