TEKNILLINEN KORKEAKOULU
Elektroniikan, tietoliikenteen ja automaation tiedekunta

Riku Lääkkölä

LANGATTOMAN VERKON RIIPPUMATTOMAT LINKKIJOUKOT JA STOKASTINEN OPTIMOINTI

Kandidaatintyö

Espoo 10.12.2009

Työn ohjaaja:

TkT Pasi Lassila

TEKNILLINEN KORKEAKOULU

KANDIDAATINTYÖN TIIVISTELMÄ

Tekijä: Riku Lääkkölä

Työn nimi: Langattoman verkon riippumattomat linkkijoukot ja stokastinen optimointi

Päivämäärä: 10.12.2009 Kieli: Suomi Sivumäärä: 5+24

Tutkinto-ohjelma: Tietoliikennetekniikka

Vastuuopettaja: TkT Jan Eriksson

Ohjaaja: TkT Pasi Lassila

Tässä työssä perehdytään langattomiin monihyppyverkkoihin. Langattomien monihyppyverkkojen kapasiteetin ylärajaa voidaan arvioida selvittämällä verkon suurimman riippumattoman linkkijoukon painojen summa. Tämä vastaa yksittäisen aikavälin suurinta mahdollista kapasiteettia.
Tässä työssä esitellään yleisimpiä verkkojen sovelluksia sekä protokollia, joita tarvitaan langattomien monihyppyverkkojen toteuttamiseen. Verkon kapasiteetin selvittämistä lähestytään simuloitua jäähdytystä muistuttavalla menetelmällä, joka riittävän pitkällä simulointiajalla ratkaisee suurimman riippumattoman joukon. Työn tarkoituksena oli selvittää, kuinka hyvin tämä menetelmä suoriutuu verrattuna aiempiin tutkimuksiin. Tulosten tuottamiseksi verkkoja simuloitiin erilaisin parametrein noin kahden viikon ajan.
Tämän työn menetelmillä tuotettuja arvoja verrataan tutkimukseen, jossa käytetty menetelmä ratkaisi ainoastaan numeerisen arvon painojen summalle. Saatujen tulosten perusteella tämän työn menetelmä ei ole ainakaan merkittävästi tehokkaampi numeerisen tuloksen laskentaan kuin aiemmassa tutkimuksessa käytetyt menetelmät. Sen sijaan käytännön käyttökohteita menetelmälle on mahdollista keksiä esimerkiksi protokollien suunnittelussa.

Avainsanat: langaton monihyppyverkko, ad hoc, simuloitu jäähdytys, suurin riippumaton joukko

Esipuhe

Tämä työ on tehty kesällä 2009 TKK:n Tietoliikenne- ja tietoverkkotekniikan laitoksella tekemäni tutkimuksen pohjalta. Haluan kiittää kesällä ja tällä kandidaattiseminaarikurssilla ohjaajanani toiminutta Pasi Lassilaa todella aktiivisesta ja kannustavasta ohjauksesta.

Otaniemi, 10.12.2009

Riku Lääkkölä

Sisältö

Symbolit ja lyhenteet

Symbolit

R	linkin lähetys- ja interferenssisäde
ν	naapuruston koko
λ	solmutiheys
a	alueen sivun pituus
$\mathcal{G}$	graafi

Käsitteet

solmu	noodi, verkon osa, joka välittää dataa
naapurusto	yksittäisen solmun lähetyssäteen sisällä olevat muut solmut
vapaa linkki	linkki, joka on verkon tilan ja interferenssiehtojen puitteissa mahdollinen
lukittu linkki	linkki, joka on verkon tilan ja interferenssiehtojen puitteissa mahdoton
aktiivinen linkki	linkki, joka kuljettaa tietoa, ja jonka interferenssivaikutus on otettava huomioon

Lyhenteet

AODV	Ad hoc On-Demand Distance Vector
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
MAC	Medium Access Control
MANET	Mobile Ad hoc Network
MWA	Moving Window Algorithm
OLSR	Optimized Link State Routing Protocol
VANET	Vehicular Ad hoc Network
WLAN	Wireless Local Area Network, langaton lähiverkko

1 Johdanto

Langattomien laitteiden vuosien mittaan lisääntynyt saatavuus on tehnyt langattomista monihyppyverkoista suositun tutkimuskohteen. Kiinnostavia sovelluskohteita langattomalle verkolle, jossa liikkuvatkin päätelaitteet voidaan sijoitella satunnaisesti, löytyy erityisesti sotilasympäristöstä. Tämän lisäksi tietoliikenneinfrastruktuuri, joka ei vaadi johtoja tai keskitettyä hallinnointia, on nopeasti asennettavissa. Näin ollen se on omiaan esimerkiksi kohteisiin, joihin tarvitaan väliaikainen tietoliikenneyhteys.

Ad hoc -verkolla viitataan verkkoon, jolla ei ole määrättyä infrastruktuuria tai keskitettyä hallinnointia, jolloin kaiken reitityksen on tapahduttava päätelaitteissa. Tässä työssä perehdytään erityisesti verkkomalliin, jossa paikallaan olevat solmut on satunnaisesti sijoiteltu. Muita kiinnostavia malleja ovat esimerkiksi liikkuvat ad hoc -verkot (MANET), joissa päätelaitteet liikkuvat, ja sensoriverkot, joissa päätelaitteet havainnoivat ympäristöään ja välittävät havainnot eteenpäin toistensa välityksellä.

Monihyppyverkkojen kapasiteettia tutkittaessa päästään graafiteoreettisten ongelmien äärelle. Tässä työssä perehdytään simuloimalla tuotetun suuren satunnaisen verkon suurimman riippumattoman linkkijoukon etsimiseen, joka on graafiteoriassa NP-täydelliseksi todistettu [1] ongelma. Äärellisellä laskentakapasiteetilla on siis käytettävä optimointimenetelmää, joka ei varsinaisesti ratkaise ongelmaa, mutta päätyy silti tulokseen, joka on lähellä oikeaa.

Suurimmalla riippumattomalla linkkijoukolla tarkoitetaan siis sitä mahdollisten linkkien lukumäärää tai painojen summaa, joka on mahdollisimman suuri siten, ettei yksikään linkki synnytä häiriötä toiselle linkille. Tässä työssä ongelmaa on yksinkertaistettu siten, että interferenssisäde on sama kuin lähetyssädekin ja interferenssin todennäköisyys on säteen sisällä 1 ja ulkopuolella 0.

Tässä työssä tutkitaan simuloiduksi jäähdytykseksi kutsutun menetelmän soveltuvuutta suurimman riippumattoman linkkijoukon etsimiseen kolmella eri painomäärittelyllä: linkin paino on aina 1 (painottamaton), linkin paino on sen pituus ja linkin paino on sen tietyn suuntaisen projektion pituus (kuva 4).

Käytännössä suurimmasta riippumasta linkkijoukosta saadaan yläraja verkon tiedonsiirtokapasiteetille yksittäisessä aikavälissä. Todellinen jatkuva ja koko verkon kattava kapasiteetti on aidosti pienempi, myös ottamatta kantaa mallin yksinkertasuuteen, sillä suurin riippumaton linkkijoukko peittää vain osan verkosta. Tällöin sitä ei voi hyödyntää kuin yksittäisissä aikaikkunoissa, sillä liikenteen virtausta ei voida saada aikaan toistamalla suurinta riippumatonta joukkoa.

Simuloitu jäähdytys, jonka esitteli ensimmäistä kertaa Kirkpatrick vuonna 1983 [2], on stokastisen optimoinnin menetelmä, joka juontaa juurensa Metropolisin vuonna 1953 esittelemään Monte Carlo -menetelmään [3]. Simuloidussa jäähdytyksessä optimoitavaa konfiguraatiota kehitetään askel kerrallaan joko suuntaan, jossa painosumma kasvaa, tai tietyllä todennäköisyydellä suuntaan, jossa se pienenee. Tämä todennäköisyys kehittyy (”jäähtyy”) simulaation edetessä algoritmille annettujen pa-

rametrien määräämällä tavalla. Tavoitteena on, että kasvattavat valinnat painottuvat loppua kohden siten, että lopulta päästään gobaaliin maksimiin tai minimiin.

Esimerkiksi Granville et al. [4], Nahar et al. [5] ja Szu ja Hartley [6] käsittelevät simuloitua jäähdytystä tapauksissa, joissa etsitään optimointiongelman globaalia minimiä. Vastaavat simulointiparametrit eivät siten ole suoraan hyödynnettävissä tässä työssä. Simuloitua jäähdytystä käsitellään kyseisissä julkaisuissa hyvin yleisellä tasolla, kun taas tässä työssä menetelmää pyritään hyödyntämään hyvin tarkkaan rajatun ongelman ratkaisemiseen.

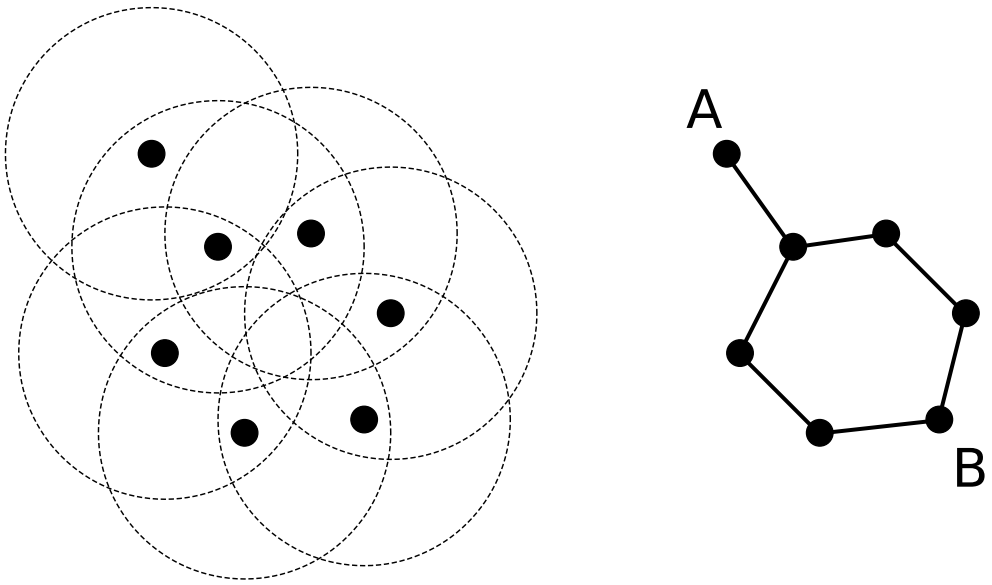
Tämän tutkimuksen tavoitteena onkin selvittää, päästäänkö simuloidulla jäähdytyksellä lähelle Nousiaisen et al. [7] suurimman riippumattoman linkkijoukon ongelmaan tuottamia viitearvoja, ja tutkia menetelmän tehokkuutta muihin verrattuna. Lisäksi pyritään löytämään laskenta-ajan kannalta optimaaliset parametrit simulointialgoritmille.

Tämän tutkimuksen rakenne on seuraava. Luvussa 2 kerrotaan langattomista monihyppyverkoista ja niiden kapasiteetin tutkimuksesta yleisesti, ja luvussa 3 määritellään käytetty verkkomalli sekä perustellaan ja esitellään simulointialgoritmi. Tutkimuksen tulokset esitellään luvussa 4, ja yhteenveto tutkimuksesta annetaan luvussa 5. Liitteeseen A on koottu tutkimuksen kannalta oleellisimpien algoritmien ohjelmakooditoteutukset.

2 Langattomat monihyppyverkot

2.1 Erilaiset verkot ja niiden ominaisuudet

Langattomalla monihyppyverkolla tarkoitetaan verkkoa, jossa viesti välittyy lähettäjältä langattomasti vastaanottajasolmulle kulkemalla välissä olevien solmujen kautta. Kuvassa 1 on esitelty yleinen langattoman monihyppyverkon malli. Kuvassa vasemmalla solmujen radion kantamat on kuvattu ympyröillä. Oikeanpuoleisessa graafissa solmut, jotka kuulevat toisensa, on yhdistetty kaarilla. Mikäli kuvan solmu A haluaa lähettää solmulle B, on reitin kuljettava usean solmun kautta.

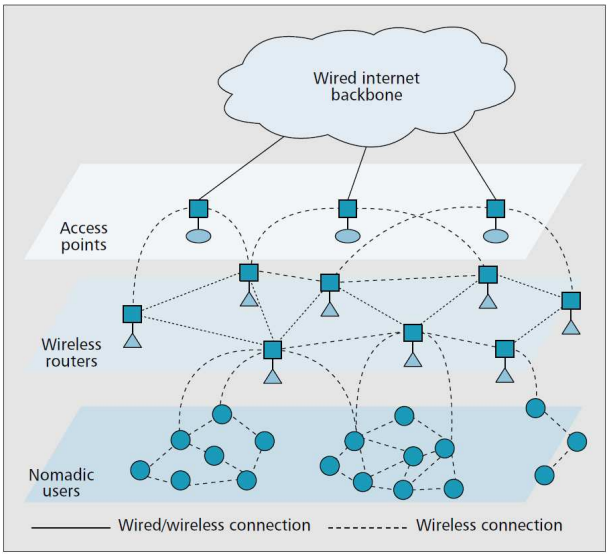


Kuva 1: Langaton monihyppyverkko. Vasemmalla esitetyillä solmujen sijoittelulla ja lähetyssäteillä muodostuu oikealla kuvattu verkko.

Langattomista monihyppyverkoista suosittuja ja kiinnostavia tutkimuskohteita ovat ainakin MANET-verkot (Mobile Ad hoc Networks), mesh-verkot sekä sensoriverkot. MANET-verkot ovat langattomia monihyppyverkkoja, joissa solmut liikkuvat. Niiden kehittämistä ja tutkimusta tapahtuu IETF:n MANET-työryhmässä. Mielenkiintoisena erikoistapauksena MANETista voidaan mainita VANET (Vehicular Ad hoc Networks), jossa verkon solmuina toimivat lähetin-vastaanottimilla varustetut autot.

Langattomat mesh-verkot ovat infrastruktuuriverkkoja, joissa verkon solmut ovat tyypillisesti paikallaan olevia tukiasemia (keskimmäinen kerros kuvassa 2) [8]. Tällaisessa verkossa ei tarvitse huolehtia verkon topologian muuttumisesta aiheutuvista ongelmista. Erona tavanomaiseen WLAN-tukiasemiin pohjautuviin verkkoihin on se, että tukiasemat liittyvät verkkoon langattomasti toistensa välityksellä.

Sensoriverkkoja sen sijaan hyödynnetään yleensä ympäristössä, jota ei päästä helposti tutkimaan muulla tavalla, kuin esimerkiksi pudottamalla tutkimuslaitteita len-



Kuva 2: Langattoman mesh-verkon arkkitehtuuri kerroksittain. Kuvan lähde: [8]

tokoneesta alueelle. Muodostettuaan sensoriverkon, jossa paketit välittyvät muiden monihyppyverkkojen tavoin itse solmujen välityksellä, solmuina toimivat tutkimuslaitteet voivat välittää keräämänsä datan kohti alueen reunaa.

2.2 MAC- ja reititysprotokollat

Edellä esiteltiin joitakin esimerkkitapauksia langattomista monihyppyverkoista. Seuraavaksi kuvaillaan joitakin yleisiä MAC- ja reititysprotokollia. Näistä osan kehittäminen on tapahtunut MANET-verkkojen tutkimuksen yhteydessä, mutta niitä ei kuitenkaan ole varsinaisesti sidottu mihinkään erityiseen käyttötapaukseen.

MAC-protokolla (Medium Access Control) määrittelee käytännöt, joilla päätelaite on yhteydessä verkkoon. MAC-protokollat pyritään suunnittelemaan siten, että verkko tulee mahdollisimman tehokkaasti hyödynnetyksi. Tilanteen mukaan verkon resurssit tulee myös jakaa riittävän reilusti eri käyttäjille.

Varhaisessa ALOHA-järjestelmässä [9] lähetykset perustuvat satunnaisliityntään, jossa päätelaitteet lähettävät satunnaisen odotusajan jälkeen riippumatta verkon tilasta ja saavat näin aikaan häiriöalueella käynnissä olevien lähetysten epäonnistumisia. Epäonnistuneen lähetyksen jälkeen päätelaite pyrkii lähettämään paketin uudelleen muutetun odotusajan kuluttua. Nykyisin yleisen WLAN:n eli IEEE 802.11 -standardin [10] mukaisen järjestelmän päätelaitteet havainnoivat verkkoa (carrier sensing), eivätkä lähetä havaitessaan häiriöalueella muita lähetyksiä. Lähetysyritysten välinen odotusaika valitaan niin sanotun backoff-algoritmin perusteella yleensä satunnaisesti tasajakaumasta, jonka maksimia kasvatetaan aina epäonnistuneen lähetysyrityksen jälkeen.

Reititysprotokollan tehtävänä on määrittää, minkä solmujen kautta paketin on kuljettava, jotta se pääsisi lähettäjältä vastaanottajalle mahdollisimman nopeasti. Lan-

gattomissa monihyppyverkoissa reitityspro tokollat jakautuvat proaktiivisiin ja reaktiivisiin protokolliin. Proaktiiviset protokollat kuten OLSR (Optimized Link State Routing) [11] toimivat siten, että verkon solmut välittävät toisilleen aika-ajoin tietoa verkon tilasta, jolloin jokaisella solmulla on jatkuvasti tieto parhaasta reitistä kuhunkin solmuun. Jos verkko on erittäin liikkuva, reittitietoja on päivitettävä hyvin usein, jolloin tällaisen protokollan käyttö on tehotonta. Reaktiivisista protokollista esimerkiksi AODV (Ad hoc On-Demand Distance Vector) [12] taas hankkii reittitietoja solmuille tarpeen mukaan, eli kun tuntematonta reittiä solmulta toiselle tarvitaan, lähettävä solmu lähettää reittipyyntöviestin ja vastauksen saatuaan lähettää paketin reittiä pitkin. Tällainen protokolla on erittäin dynaamisissa verkoissa tehokkaampi kuin proaktiivinen protokolla.

2.3 Monihyppyverkkojen kapasiteetti

Perustavanlaatuisia tutkimustuloksia monihyppyverkkojen kapasiteetista ovat tuottaneet Gupta ja Kumar vuonna 2000 [13] sekä Thiran et al vuonna 2007 [14]. Tiedetään, että suuren langattoman monihyppyverkon kapasiteetti solmua kohden skaalautuu verkon koon kasvaessa kaavan $O(1/\sqrt{N})$ mukaan [13]. Tässä N viittaa verkon solmujen lukumäärään. Tulos ei kuitenkaan O-notaation määritelmän mukaan ota kantaa siihen, miten suurta kasvu absoluuttisesti on.

Jain et al. [15] esittelivät formulaation, jolla langattoman monihyppyverkon kapasiteetti voidaan selvittää lineaarisen ohjelmoinnin tehtävänä. Lisäksi esiteltiin heuristiikkoja, joilla tämän tehtävän ratkaisulle saadaan ala- ja ylärajoja. Alaraja voidaan laskea esimerkiksi etsimällä interferenssigraafista riippumattomia joukkoja, jolloin voidaan laskea jokaisen linkin optimaalisen päälläoloajan osuus. Ylärajan selvittämiseksi voidaan käyttää interferenssigraafin klikkejä (clique).

Muun muassa edellä mainittuja heuristiikkoja ja formulaatiota hyväksi käyttäen numeerisia tuloksia suurten langattomien monihyppyverkkojen reunasta reunaan -välityskapasiteetille on tuotettu tuoreessa tutkimuksessa [16]. Tässä työssä tutkitaan yksittäisen aikavälin suurinta mahdollista kapasiteettia, joka vastaa suurimman riippumattoman joukon ongelmaa ja on edellä mainitun tutkimuksen osaongelma. Sen ratkaisemiseen Nousiaisen et al. tutkimuksessa on käytetty niin sanottua liukuvan ikkunan algoritmia (MWA) [7]. MWA:ta on sovellettu erilaisten ongelmien ratkaisemiseen [17], ja tuloksia on verrattu jatkuvan virtauksen maksimikapasiteetille tuotettuihin tuloksiin [16].

Itse suurimman riippumattoman joukon ongelmaan liittyen tässä työssä on viitattu lähinnä tutkimukseen [7], jossa ongelman muotoilu on vastaava. Nousiaisen tutkimuksessa selvitettyjä arvoja käytetäänkin vertailukohtana tämän työn menetelmillä saaduille arvoille. Nousiaisen tutkimus on osa suurempaa tutkimusta, jossa pyritään löytämään rajat suuren langattoman ad hoc -verkon jatkuvalle tiedonsiirtokapasiteetille. Käsittelyssä on siis osaongelma yksittäisen aikaikkunan suurimmasta mahdollisesta kapasiteetista, joka antaa karkeimman ylärajan jatkuvalle kapasiteetille.

Simuloidun jäähdytyksen kaltaisen algoritmin käyttöä hajautetussa MAC-protokol-

lassa, jossa solmut tekevät päälletuloyrityksiä Poisson-prosessin mukaisesti, ovat tutkineet Durvy ja Thiran [18]. He vertasivat asykronisen MAC-protokollan ja synkronisen aikaväleihin jaetun MAC-protokollan kapasiteetteja tutkimalla lähetysten lukumäärää pinta-alayksikköä kohden (spatial reuse).

Asynkronisessa järjestelmässä solmu tekee päälletuloyrityksen kaikille solmuille yhteisestä odotusaikajakaumasta arvotun odotusajan jälkeen, ja päälletulo onnistuu ainoastaan, jos häiritseviä yhteyksiä ei ole olemassa. Tällaisen järjestelmän kapasiteetti osoittautui suuremmaksi kuin vastaavan hajautetun aikaväleihin jaetun synkronisen järjestelmän. Tässä työssä sovelletaan ja kehitetään Thiranin et al. ideaan perustuvaa stokastisen optimoinnin menetelmää suurimman riippumattoman linkkijoukon painon selvittämiseksi.

3 Monihyppyverkon suurin riippumaton joukko

3.1 Malli ja oletukset

Tämän tutkimuksen verkkomalli on täysin Nousiaisen et al. [7, 16, 17] mallia vastaava, ja seuraava kuvaus on kirjoitettu kyseisestä tutkimuksesta lainaten. Verkon solmut ovat liikkumattomia ja jakautuneet satunnaisesti tasolle kaksiulotteisen Poisson-prosessin mukaisesti. Poisson-prosessin intensiteettiparametri λ määrittää verkon solmutiheyden.

Solmut kommunikoivat langattomasti samalla taajuudella, ja jokaisella solmulla on sama lähetyssäde R. Linkki solmujen välillä on mahdollinen, mikäli solmujen etäisyys on pienempi kuin tämä lähetyssäde. Lähetys linkillä onnistuu ainoastaan, jos sen vastaanottava solmu ei kuule muita lähetyksiä.

Mallia voidaan kuvata suunnattuna verkkona $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, jossa solmujen u ja $v \in \mathcal{V}$ välillä on olemassa linkki eli kaari $e = (u, v) \in \mathcal{E}$, jos $d(u, v) \leq R$, missä $d(u, v)$ on solmujen välinen Euklidinen etäisyys tasossa. Linkkien häiriövaikutusta voidaan kuvata kaarien joukolla $\mathcal{I}(e)$, joka koostuu niistä linkeistä, joihin linkillä $e \in \mathcal{E}$ on häiriövaikutus. Tämän mallin häiriövaikutus (Boolean interference model) voidaan määritellä

$$\mathcal{I}_B(e) = \{a \in \mathcal{E} \mid d(t(a), r(e)) \leq R \vee d(r(a), t(e)) \leq R\}, \tag{1}$$

jossa $t(e)$ on linkin $e \in \mathcal{E}$ lähettävä solmu ja $r(e)$ vastaanottava solmu.

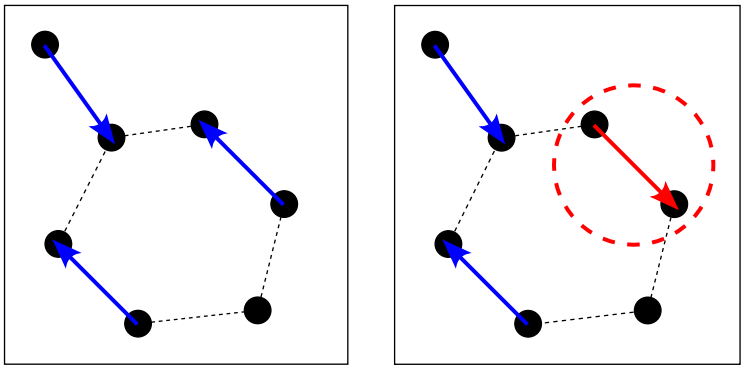
Riippumattomaksi linkkijoukoksi kutsutaan sellaista joukkoa, jossa yksikään linkki ei häiritse toista. Ylläolevia merkintöjä käyttäen voidaan määritellä, että linkkijoukko $\mathcal{L} \subset \mathcal{E}$ on riippumaton, jos

$$\forall a \neq e: \quad a \notin \mathcal{I}_B(e),\ a, e \in \mathcal{L}. \tag{2}$$

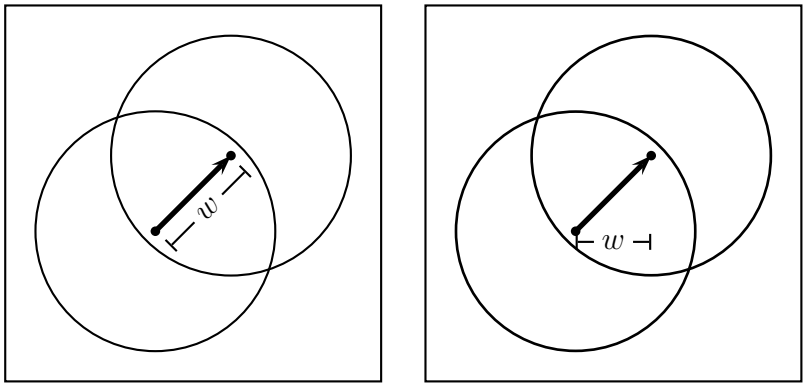
Kuvassa 3 on havainnollistettu tämän mallin interferenssiehtoja kahdella kuvan 1 esimerkkiverkon linkkijoukolla. Kuvassa aktiiviset suunnatut lähetykset on merkitty nuolilla ja epäaktiiviset katkoviivoilla. Vasemmanpuoleinen kuva määrittää tälle verkolle erään riippumattoman linkkijoukon, jossa siis jokainen vastaanottava solmu kuulee vain yhden lähetyksen. Oikeanpuoleisessa kuvassa ympyröity linkki häiritsee toisen linkin vastaanottoa, eli linkkijoukko ei ole riippumaton.

Tässä työssä on tutkittu samoja painotapauksia kuin Nousiaisen et al. työssä [7]:

1. Painottamaton, eli jokaisen linkin paino on 1.
2. Paino on linkin pituus. (Kuva 4 a)
3. Paino on linkin pituuden projektio valittuun suuntaan, tässä x-suuntaan oikealle. (Kuva 4 b)



Kuva 3: Esimerkki sekä riippumattomasta (vasemmanpuoleinen kuva) että ei-riippumattomasta (oikeanpuoleinen kuva, häiritsevä linkki ympyröity) linkkijoukosta.



a) b)

Kuva 4: Työssä käytetyt painomäärittelyt: a) w = linkin pituus, b) w = positiiviseen x-suuntaan projisoidun linkin pituus

Kaikkien mahdollisten linkkien lukumäärään verkossa vaikuttavat linkkien tiheys λ (1/pinta-alayksikkö) sekä linkkien lähetyssäde R. Näiden avulla voidaan määritellä verkolle vain yksi dimensioton ominaisuus, keskimääräinen naapuruston koko, seuraavasti:

$$\nu(\lambda, R) = \pi\lambda R^2. \tag{3}$$

Tapauksessa 1 (painottamaton) pyritään löytämään suurin mahdollinen yhtäaikaisesti päällä olevien linkkien lukumäärä pinta-alayksikköä kohden $U(\lambda, R)$. Tällöin dimensioanalyysin perusteella U voidaan esittää muodossa

$$U(\lambda, R) = \lambda u(\nu(\lambda, R)), \tag{4}$$

jossa u on dimensioton funktio. Tässä tapauksessa u kertoo itse asiassa keskimääräisen linkkien lukumäärän per solmu, ja tässä tavoitteena on löytää suurin u:n arvo eri ν:n arvoilla.

Tapauksissa 2 (merkitään alaindeksillä l) ja 3 (merkitään alaindeksillä x) pyritään löytämään suurin mahdollinen linkkien etenemä pinta-alayksikköä kohden $U_{l,x}(\lambda, R)$,

tapauksessa 2 mihin tahansa suuntaan ja tapauksessa 3 positiiviseen x-projisoituun suuntaan. Näissä tapauksissa $U_{l,x}$ voidaan siis jälleen dimensioanalyysin perusteella ilmaista dimensiottoman funktion avulla muodossa

$$U_{l,x}(\lambda, R) = \sqrt{\lambda}\, u_{l,x}(\nu(\lambda, R)), \tag{5}$$

ja vastaavasti painotetuissa tapauksissa pyritään löytämään suurin $u_{l,x}$:n arvo eri ν:n arvoilla. Tutkittavat arvot on määritelty kuten Nousiaisen et al. [7] tutkimuksessa, jotta referenssiarvoihin vertaaminen olisi mahdollisimman helppoa.

3.2 Thiranin alkuperäinen algoritmi

Kuten mainittiin osassa 2, Thiran et al. [18] tutkivat protokollaa, jossa linkki pyrkii päälle keskimäärin $1/\overline{cw}$ kertaa aikayksikköä kohden ja lähetyksen keskimääräinen kesto on T aikayksikköä. Tällaisen verkon tiloja, eli eri kombinaatioita aktiivisista linkeistä, voidaan kuvata Markovin ketjuna. Painottamattomalle tapaukselle saadaan tasapainojakauma muotoa

$$\pi(i) \sim a^i, \tag{6}$$

jossa i on aktiivisten linkkien lukumäärä ja $a = T/\overline{cw}$. Kaavasta (6) nähdään, että mitä suurempi on a:n arvo, sitä suurempi on tilan i todennäköisyys.

Simuloitu jäähdytys perustuu Monte Carlo -algoritmiin, jolla voidaan generoida liikkuvista hiukkasista koostuvan aineen tiloja eri ajanhetkillä [3]. Kyseisen algoritmin Markovin ketjun hiukkassysteemin energiaan perustuva tasapainojakauma on muotoa

$$\pi(E) \sim e^{-\frac{E}{kT}}, \tag{7}$$

jossa E on systeemin energia, k Boltzmannin vakio ja T systeemin lämpötila. Tasapainojakaumien (6) ja (7) perusteella, kun $i \sim E$, saadaan vastaavuus

$$\ln a \sim -\frac{1}{T}. \tag{8}$$

Edellä esitettyihin huomioihin perustuu idea käyttää simuloitua jäähdytystä suurimman mahdollisen riippumattoman linkkijoukon etsimiseen. Thiranin esittelemän jatkuva-aikaisen Markovin ketjun simuloiminen suoraan olisi kuitenkin huomattavan tehotonta, sillä suurilla a:n arvoilla päälletuloyrityksistä suurin osa epäonnistuu interferenssin vuoksi.

3.3 Hyppyketjualgoritmi

Tämän työn simulaatioissa on käytetty yleisen simuloidun jäähdytyksen [2] periaatteisiin pohjautuvaa hyppyketjualgoritmia. Turhien vaiheiden välttämiseksi simuloidaan suoraan diskreettiä Markovin ketjua, joka on upotettu hetkiin, jolloin jokin linkki tekee sallitun päälletulo- tai sammumisyrityksen. Lähdetään tilanteesta, jossa yksikään linkki ei ole aktiivinen. Simulaatio etenee sykleittäin alla kuvatulla tavalla.

Koko simulaation ajan on pidettävä kirjaa aktiivisista linkeistä (merkitään tätä listaa $\mathcal{N}$:lla) ja lukituista linkeistä (merkitään tätä $\mathcal{D}$:llä). Merkitään listojen $\mathcal{N}$ ja $\mathcal{D}$ alkioita vastaavasti symboleilla n_i ja d_i. Lista linkeistä, joista kunkin voi vallitsevien interferenssiolosuhteiden puitteissa aktivoida, eli vapaiden linkkien lista $\mathcal{M}$ (alkio m_i) saadaan muodostettua poistamalla kaikkien linkkien listasta $\mathcal{G}$ listojen $\mathcal{N}$:n ja $\mathcal{D}$:n alkiot.

Määritellään summat S_n ja S_m kaavoilla

$$S_n = \sum_{n_i \in \mathcal{N}} w(n_i) \quad \text{ja} \tag{9}$$

$$S_m = \sum_{m_i \in \mathcal{M}} w(m_i), \tag{10}$$

joissa $w(e)$ merkitsee linkin e painoa. Summa S_n merkitsee siis aktiivisten linkkien painojen summaa ja S_m vapaiden linkkien painojen summaa.

Itse algoritmin eteneminen painotetussa tapauksessa on kuvattu alla.

1. Syklien lukumäärä $t = 0$.
2. Todennäköisyydellä
$$P_{act}(a) = \frac{aS_m}{S_n + aS_m} \tag{11}$$
aktivoidaan vapaa linkki, tai vastaavasti todennäköisyydellä
$$P_{deact}(a) = \frac{S_n}{S_n + aS_m} \tag{12}$$
suljetaan aktiivinen linkki.
3. Aktivoitava tai suljettava linkki valitaan painojakauman mukaan satunnaisesti.
4. Päivitetään listat $\mathcal{N}$, $\mathcal{D}$ ja $\mathcal{M}$.
5. $t = t + 1$.
6. Lasketaan jäähdytysparametrin a uusi arvo $a(t)$.
7. Jos syklien määrä t ei ole saavuttanut rajaa t_{max}, palataan kohtaan 2.

Tässä työssä jäähdytysparametrin kasvatusfunktiot $a_i(t)$ on toteutettu siten, että ne riippuvat ainoastaan kuluneiden syklien lukumäärästä riippumatta parametrin edellisestä arvosta.

Painottamattomalle tapaukselle voidaan käyttää täsmälleen samaa algoritmin kuvausta, kun huomataan, että painottamaton verkko on itse asiassa painotettu verkko, jossa jokaisen linkin paino on 1, eli

$$\forall e \in \mathcal{G} : \quad w(e) = 1. \tag{13}$$

Tehokkuuden vuoksi algoritmien implementaatiot painottamattomalle tapaukselle on kuitenkin aiheellista toteuttaa erikseen.

3.4 Käytännön menetelmät

Tämän työn simulaatiot on kaikki toteutettu *Mathematica*^{TM} $\mathcal{7}$ -ohjelmistolla. Tuotettu koodi on listattu liitteessä A, ja siellä listattuihin funktioihin viitataan tässä luvussa kursivoiduilla funktionnimillä. Seuraavaksi selvitetään simulaation pääpiirteet. Simulaation vaiheet ovat seuraavat:

1. Verkkorealisaation generointi
2. Linkki- ja interferenssigraafien laskeminen
3. Suurimman riippumattoman linkkijoukon etsiminen simuloidun jäähdytyksen avulla

Tässä tutkimuksessa pyritään mallintamaan mahdollisimman hyvin äärettömän suuria verkkoja, joten realisaatiot generoidaan neliön muotoiselle tasolle, jonka vastakkaiset reunat on samastettu, eli taso on kääritty toruksen muotoiseksi reunaefektien minimoimiseksi (*sortedNeighbours*). Itse solmut taas generoidaan kaksiulotteisen Poisson-prosessin mukaisesti. Staattisen realisaation tapauksessa täytyy kullekin solmulle siis vain generoida kaksi satunnaislukua koordinaateiksi (*pUniform*).

Tällä tavoin generoidulla verkolla on itse asiassa kolme ominaisuutta: alueen koko, joka määräytyy sivun pituuden s (pituusyksikköä) mukaan, solmujen lukumäärä, joka määräytyy tiheyden λ (solmua/pinta-alayksikkö) mukaan, ja solmun lähetys- ja interferenssisäde R, joka on tämän työn simulaatioissa kiinnitetty arvoon 1. Näistä kaksi viimeistä ominaisuutta yhdistämällä voidaan laskea solmun keskimääräinen naapuruston koko $\nu(\lambda, R) = \pi\lambda R^2$. Tämä arvo kertoo miten monta solmua keskimäärin on yksittäisen solmun lähetyssäteen sisäpuolella. Siten ν on määrätyn kokoiselle verkolle ainoa tekijä, joka määrittää mahdollisten linkkien lukumäärän, jos interferenssiehtoja ei oteta huomioon. Alueen koon kasvattaminen ainoastaan tuo tilannetta lähemmäs ääretöntä tasoa.

Koska yksittäinen verkkorealisaatio säilyy muuttumattomana koko simulaation ajan, on tehokkuuden kannalta järkevää koostaa graafi mahdollisista linkeistä (painottamattomalle *allPossibleLinksList*, painotetuille *allPossibleLinksListWeight*) ja niistä linkeistä, jotka häiritsevät toisiaan (painottamattomalle *interferenceList*, painotetuille olennaisesti sama). Tässä työssä linkki on mahdollinen, jos solmu on lähettävän solmun lähetyssäteen sisäpuolella. Linkki häiritsee toista, jos jälkimmäisen vastaanottava solmu on ensimmäisen lähettävän solmun lähetyssäteen sisäpuolella.

Käytännön tasolla interferenssigraafin generointi on tässä työssä toteutettu siten, että ensin on listattu jokaisen solmun naapurit, eli solmut, jotka ovat sen lähetyssäteen sisäpuolella. Tästä listasta on sitten koostettu kaikkien mahdollisten linkkien lista. Näitä kahta listaa ristiin vertaamalla on muodostettu lista kunkin interferenssiseistä, joka itse asiassa koostuu linkkilistaan viittaavista indekseistä. Nämä indeksit kertovat mitkä linkit kyseisen linkin päälle kytkeminen lukitsee.

Naapurilistan laskemisen tehostamiseksi alue on ensin ruudutettu $r \times r$-kokoisiin ruutuihin (*sortPoints*). Tällöin yksittäisen solmun naapureita täytyy etsiä ainoas-

taan naapuriruuduista. Naapureiden listaamisen yhteydessä painotetuissa tapauksessa kirjataan linkkien mahdolliset painot.

Lopulta itse simuloitu jäähdytys (*annealAbs* ja *annealWeightAbs*) suoritetaan erilaisilla parametreilla. Luvussa 3.3 kuvatun algoritmin eri vaiheet on toteutettu liitteen A mukaisella ohjelmakoodilla taulukossa 1 kuvatulla tavalla, rivinumerot viittaavat liitteen A funktion *annealWeightAbs* riveihin, ellei toisin mainita.

Taulukko 1: Luvun 3.3 algoritmin kohdat liitteen A koodilistauksissa. Rivinumerot viittaavat oletusarvoisesti funktioon *annealWeightAbs*.

Vaihe	**Rivi**
2	13 – 21
3	15, 19, lisäksi funktiot *linkWeight* ja *unlinkWeight*
4	22, lisäksi funktiot *linkWeight* ja *unlinkWeight*
6	34
7	11

Tulosten luotettavuuden kasvattamiseksi simulaatiot on syytä toistaa. Tässä työssä jokaista ν:n arvoa kohden generoitiin useita verkkoja ja näille lasketuista u:n, u_l:n ja u_x:n arvoista laskettiin 95 % luottamusvälit. Vertailukelpoisuuden säilyttämiseksi verkkorealisaatioita ei kuitenkaan generoitu uudelleen jokaiselle eri jäähdytysparametrin kasvatusstrategialle.

Taulukko 2: Simulaatioissa käytetyt parametrit

Par.	**Arvo**
N	1000
ν	x-etenevälle painolle $[0.5, 12]$, muille $[0.5, 7]$
s	1000, 10000 ja 100000
k	10

Tämän työn simulaatioajoissa käytettiin taulukossa 2 esiteltyjä arvoja verkon koolle (N), naapuruston koolle (ν), syklien lukumäärälle (s) ja toistojen lukumäärälle (k). Jäähdytysparametri j:n kasvattaminen on tehty kuluneiden syklien lukumäärän funktiona. Tutkittiin alla listattuja kasvatusstrategioita (merkitään kuluneiden syklien lukumäärää t:llä).

$$a_1(t) = \log(t), \quad a_2(t) = t, \quad a_3(t) = t^2, \quad a_4(t) = 2^{999} \tag{14}$$

Tulokset on esitetään osassa 4.

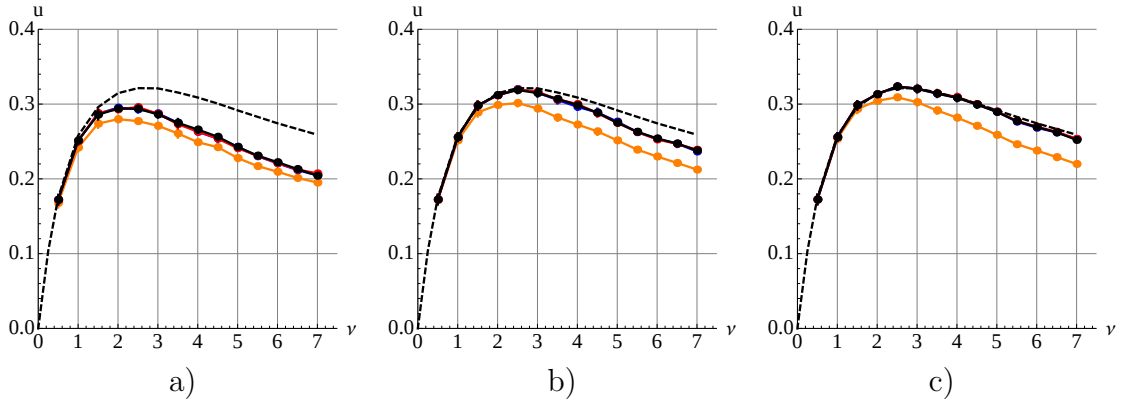
4 Tulokset

4.1 Menetelmän tehokkuus

Tämän työn simulaatioita ajettiin kahdella erilaisella tietokoneella yhtäaikaisesti yli kahden viikon ajan. Verkkorealisaatioiden ja niiden interferenssigraafien generoinnin jälkeen itse hyppyketjualgoritmi ajettiin kolmella absoluuttisella syklilukumäärällä. Vaikka syklien lukumäärä pidettiin vakiona, simulaatioiden kestot kasvoivat ν:n kasvaessa, sillä naapuruston koon kasvaessa myös joka syklillä tutkittava interferenssigraafi kasvaa.

Viitearvojen [7] tuottamiseen käytetyn menetelmän ajoajat eivät ole suoraan vertailukelpoisia, sillä niitä ei ole tuotettu *Mathematica*^{TM}-ohjelmistolla. Seuraavaksi esiteltävät tulokset eivät kuitenkaan viittaa ainakaan siihen, että tässä työssä käytetyllä menetelmällä tuloksia saataisiin erityisen paljon tehokkaammin.

4.2 Simulaatioiden tulokset

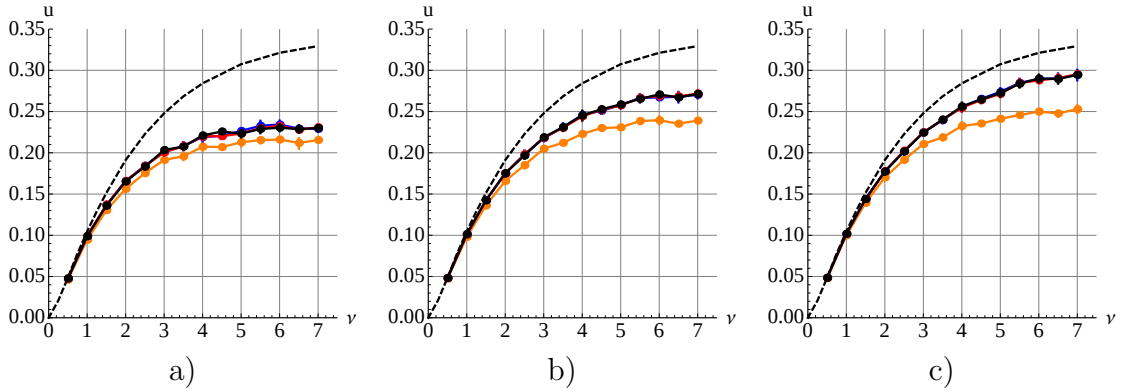


Kuva 5: Simuloidun jäähdytyksen tulokset (dimensioton parametri u naapuruston koon ν funktiona) painottamattomalle mallille syklien lukumäärillä a) 1000, b) 10000 ja c) 100000. Eri väreillä on kuvattu (14) eri kasvatusstrategioita kaavan seuraavasti: oranssi $= a_1$, sininen $= a_2$, punainen $= a_3$ ja musta $= a_4$. Vertailukäyrät aiemmasta tutkimuksesta [7] katkoviivalla. 95 % luottamusväleistä saadut virherajat on esitetty pystyviivoilla.

Kuvassa 5 on esitetty simulaatioiden tulokset painottamattomalle tapaukselle. Kuvaan on lisätty myös vertailukäyrät Nousiaisen [7] tuloksista. Kuvaajista nähdään, että pienillä ν:n arvoilla tulokset vastaavat referenssiarvoja jo lyhyilläkin simulaatioilla. Kun $\nu > 5$, tulokset alkavat jäädä vajaiksi vertailukäyrään nähden vielä suurimmallakin testatulla syklimäärällä. Suurimmalla syklimäärällä virhe ei kuitenkaan enää ole merkittävä.

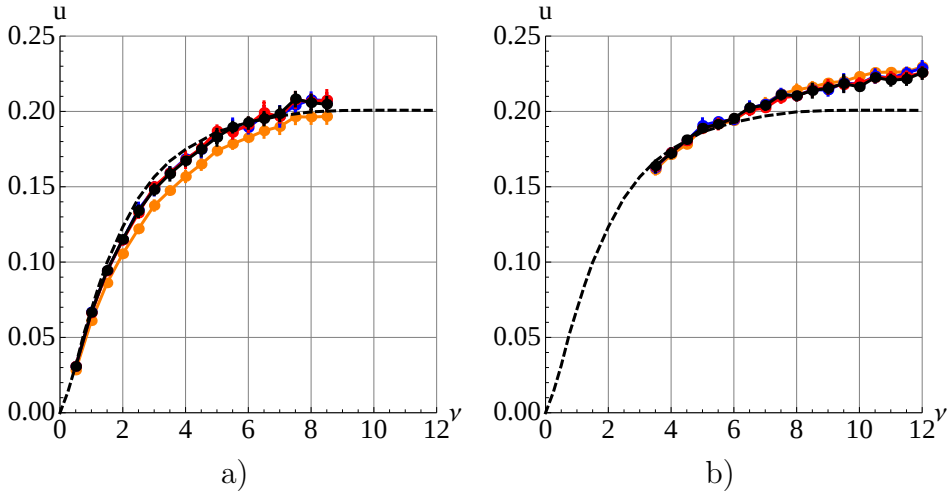
Kasvatusstrategian valinnan vaikutus on tulosten perusteella todella pieni. Riippumatta simulaation kestosta, ainoa jäähdytysparametrin kasvatusstrategia, joka kuvan 5 perusteella suoriutuu heikommin kuin muut, on logaritminen kasvattaminen.

On yllättävää, että lineaarinen kasvattaminen ja jäähdytysparametrin määrääminen suoraan hyvin suureksi vakioksi suoriutuvat lähes täysin samalla tavalla.



Kuva 6: Simuloidun jäähdytyksen tulokset kuvan 4 a painomäärittelyllä. Yksityiskohdat ovat vastaavat kuin kuvassa 5.

Kuvasta 6 voidaan havaita, että tapaus, jossa painona on linkin pituus, on tämän työn tapauksista laskennallisesti raskain tehtävä. Tuloskäyrät jäävät vielä pisimmälläkin simulaatioajalla huomattavasti vertailukäyrän alapuolelle. Kuvasta voidaan karkeasti arvioida, että syklien lukumäärä olisi pitänyt vielä vähintään kymmenkertaistaa, jotta vertailukäyrät olisi saavutettu. Kasvatusstrategian vaikutus on yhtä pieni kuin painottamattomassakin tapauksessa.



Kuva 7: Simuloidun jäähdytyksen tulokset kuvan 4 b painomäärittelyllä.. Yksityiskohdat ovat vastaavat kuin kuvassa 5, lukuunottamatta syklien lukumääriä a) 1000 ja b) 100000.

Linkin oikealle suuntautuvan x-projektion pituudella painotettu tapaus on laskennallisesti kevyempi ratkaista, sillä on huomioitava ainoastaan linkit, jotka suuntauvat oikealle. Kuvasta 7 havaitaan, että logaritminenkin kasvatusstrategia toimi tässä tapauksessa lähes yhtä hyvin kuin muutkin. Pitkällä simulaatioajalla ja suurilla ν:n

arvoilla tuloskäyrät kohoavat viitekäyrän yläpuolelle, mikä osaltaan hieman kyseenalaistaa menetelmän luotettavuuden. Ilmiö voi kuitenkin olla selitettävissä sillä, että hyvin tiheä (suuri ν) 1000 solmun verkko on varsin pieni lähetyssäteen moninkertoina mitattuna, sillä

$$N = a^2\lambda = a^2\frac{\nu}{\pi R^2} \quad \Leftrightarrow \quad \frac{a}{R} = \sqrt{\frac{N\pi}{\nu}}. \tag{15}$$

Tällöin verkko ei simuloi riittävän hyvin äärettömän suurta verkkoa, ja reunailmiöt saattavat vaikuttaa lopputulokseen.

5 Johtopäätökset

Työn perimmäinen tavoite oli tutkia, kuinka hyvin simuloitua jäähdytystä muistuttava menetelmä soveltuu annetun langattoman verkkomallin suurimman riippumattoman joukon selvittämiseen. Lisäksi oli tarkoitus perehtyä langattomiin monihyppyverkkoihin ja niiden kapasiteetin tutkimiseen syvemmin. Tässä luvussa käsitellään tuotettujen simulaatiotulosten luotettavuutta ja merkitystä.

Edellä esiteltiin simulaatioajojen tulokset kuvaajina, joihin on liitetty vertailukäyrät viitearvoista. Käytetty menetelmä suoriutui pisimmällä käytetyllä simulointiajalla hyvin painottamattomasta tapauksesta, mutta ensimmäisen painotapauksen simulaatioiden olisi pitänyt olla vähintään noin kymmenkertaisia pituuksiltaan. Toisessa painotapauksessa taas saatiin pitkällä simulaatioajalla tiheille verkoille liian suuria tuloksia viitearvoihin nähden. Tämä saattaa selittyä sillä, että verkon tihentyessä ja solmumäärän pysyessä samana alueen koko pienenee liikaa simuloidakseen hyvin ääretöntä tasoa.

Huomionarvoista saaduissa tuloksissa on myös se, että jäähdytysstrategian valinnalla ei ollut juurikaan merkitystä. Aggressiivisimmat strategiat toimivat parhaiten, ja tämä tarkoittaa sitä, että algoritmi toimi parhaiten, kun joka syklillä aktivoitiin linkki, jos se vain oli mahdollista. Simuloidun jäähdytyksen kasvatustrategian ideana on ehkäistä jumiutumista globaaliin maksimiin, mutta näiden tulosten perusteella sellaista vaaraa ei tässä ongelmassa käytännössä ole.

Painottamattoman ja ensimmäisen painotapauksen perusteella menetelmän luotettavuudesta ei synny epäilyksiä. Toinen painotapaus sen sijaan kyseenalaistaa menetelmän toimivuuden tiheämmillä verkoilla. Mikäli menetelmää halutaan vielä soveltaa, olisi luotettavuuden varmistamiseksi syytä ajaa uusia simulaatioita vielä huomattavasti suuremmilla verkoilla. Tämä taas osaltaan pidentää simulaatioaikoja.

Tulosten valossa tässä työssä esitellyn menetelmän käyttöä viitearvoissa käytetyn MWA:n sijaan ei voi perustellusti suositella tämän ongelman ratkaisemiseen. Toisaalta tämän työn menetelmä tuottaa riittävän pitkällä simulaatiolla ratkaisun suurimman riippumattoman joukon ongelmaan, kun taas MWA tuottaa ainoastaan numeerisen maksimiarvon. Ei siis voida sulkea pois mahdollisuutta, että menetelmää voitaisiin hyödyntää langattomien monihyppyverkkojen käytännön sovelluksissa.

Rajallisen ajan vuoksi pidempien simulaatioiden ajaminen tätä työtä varten ei ollut mahdollista. Olisi arvokasta tietää ainakin kuinka suuri verkko vaaditaan, jotta saadut tulokset olisivat luotettavia. Lisäksi syklimäärän kymmenkertaistamisen riittävyys ensimmäisessä painotapauksessa olisi syytä varmistaa. Olisi myös hyvä toteuttaa vastaavat algoritmit optimoidummassa ympäristössä, jotta menetelmän todellinen tehokkuus tulisi esille ja olisi paremmin verrattavissa aiemmin käytettyjen menetelmien tehokkuuteen.

Viitteet

[1] M. R. Garey & D. S. Johnson, *Computers and Intractability : A Guide to the Theory of NP-Completeness (Series of Books in the Mathematical Sciences)*, W. H. Freeman, ISBN 0716710455, Tammikuu 1979.

[2] S. Kirkpatrick, C. D. Gelatt & M. P. Vecchi, Optimization by Simulated Annealing, *Science* **220**(4598), ss. 671–680, Toukokuu 1983.

[3] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller & E. Teller, Equation of State Calculations by Fast Computing Machines, *The Journal of Chemical Physics* **21**(6), ss. 1087–1092, 1953.

[4] V. Granville, M. Krivanek & J.-P. Rasson, Simulated annealing: a proof of convergence, *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **16**(6), ss. 652–656, ISSN 0162-8828, Kesäkuu 1994.

[5] S. Nahar, S. Sahni & E. Shragowitz, Experiments with simulated annealing, sarjassa *DAC '85: Proceedings of the 22nd ACM/IEEE Design Automation Conference*, ss. 748–752, ACM, New York, NY, USA, ISBN 0-8186-0635-5, 1985.

[6] H. Szu & R. Hartley, Nonconvex optimization by fast simulated annealing, *Proceedings of the IEEE* **75**(11), ss. 1538–1540, ISSN 0018-9219, Marraskuu 1987.

[7] J. Nousiainen, J. Virtamo & P. Lassila, Maximum Weight Independent Sets in an Infinite Plane, sarjassa *21st International Teletraffic Congress (ITC)*, Pariisi, Ranska, Syyskuu 2009.

[8] R. Bruno, M. Conti & E. Gregori, Mesh networks: commodity multihop ad hoc networks, *Communications Magazine, IEEE* **43**(3), ss. 123–131, 2005.

[9] N. Abramson, THE ALOHA SYSTEM: another alternative for computer communications, sarjassa *AFIPS '70 (Fall): Proceedings of the November 17-19, 1970, fall joint computer conference*, ss. 281–285, ACM, New York, NY, USA, 1970.

[10] IEEE Standard for Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, *IEEE Std 802.11-2007 (Revision of IEEE Std 802.11-1999)* ss. C1–1184, Kesäkuu 2007.

[11] RFC 3626 - Optimized Link State Routing Protocol (OLSR), IETF Network Working Group, `http://www.ietf.org/rfc/rfc3626.txt`, Lokakuu 2003.

[12] RFC 3561 - Ad hoc On-Demand Distance Vector (AODV) Routing, IETF Network Working Group, `http://www.ietf.org/rfc/rfc3561.txt`, Heinäkuu 2003.

[13] P. Gupta & P. R. Kumar, The capacity of wireless networks, *IEEE Transactions on Information Theory* **46**(2), ss. 388–404, 2000.

[14] M. Franceschetti, O. Dousse, D. N. C. Tse & P. Thiran, Closing the gap in the capacity of wireless networks via percolation theory, *IEEE Trans. Information Theory* **53**, ss. 1009–1018, 2007.

[15] K. Jain, J. Padhye, V. N. Padmanabhan & L. Qiu, Impact of interference on multi-hop wireless network performance, *Wireless Networks* **11**(4), ss. 471–487, ISSN 1022-0038, 2005.

[16] J. Nousiainen & P. Lassila, Approximating Maximum Directed Flow in a Large Wireless Network, sarjassa *IEEE ICC*, ss. 1–6, Dresden, Saksa, Kesäkuu 2009.

[17] J. Nousiainen, J. Virtamo & P. Lassila, Forwarding Capacity of an Infinite Wireless Network, sarjassa *Proceedings of ACM MSWiM*, ss. 177–184, ACM, Vancouver, Kanada, Lokakuu 2008.

[18] M. Durvy & P. Thiran, A Packing Approach to Compare Slotted and Non-Slotted Medium Access Control, sarjassa *Infocom 2006*, 2006.

Liite A: Koodilistauksia

Tämän työn simulaatiot on toteutettu *Mathematica*^{TM}7 -ohjelmistolla. Tähän liitteeseen on koottu olennaisten funktioiden koodilistauksia.

Verkon generointi ja jako laskentaa nopeuttaviin ruudukoihin:

```
pUniform[a_,\[Lambda]_]:=
  Table[a*{RandomReal[],RandomReal[]},{\[Lambda] a^2}];


sortPoints[p_,a_,r_]:=
  Module[{sorted=Table[Table[{},{a/r}],{a/r}],i=1},
    While[i<=Length[p],
      AppendTo[sorted[[Ceiling[p[[i,1]]],
                       Ceiling[p[[i,2]]]]],p[[i]]];
      i++;];
  sorted];


sortedIndices[p_]:=
  Module[{ind={},le=p//Length,i,j},
    For[i=1,i<=le,i++,
      For[j=1,j<=le,j++,
        AppendTo[ind,{i,j,#}]&/@Range[p[[i,j]]//Length];
      ];
    ];
  ind];
```

Verkon jokaisen solmun naapuruston laskenta:

```
sortedNeighbours[p_, {i_, j_, n_}, r_] :=
  Module[{le = Length[p], possibleSq = {}, right,
          top, x = p[[i, j, n, 1]],
          y = p[[i, j, n, 2]], origin, t},
    origin = {x, y};
    possibleSq = {{i, j}, {i + 1, j}, {i + 1, j + 1},
                  {i, j + 1}, {i - 1, j}, {i, j - 1},
                  {i - 1, j - 1}, {i - 1, j + 1},
                  {i + 1, j - 1}};
    possibleSq = ((possibleSq /. (le + 1) -> 1) /. 0 -> le);
    t = Table[
          Position[
            p[[possibleSq[[m, 1]], possibleSq[[m, 2]]]],
                point_ /;
                  (point != origin &&
                   Norm[point - origin] < r ||
```

```
                    Norm[point + {le, 0} - origin] < r ||
                    Norm[point + {0, le} - origin] < r ||
                    Norm[point + {le, le} - origin] < r ||
                    Norm[point - {le, 0} - origin] < r ||
                    Norm[point - {0, le} - origin] < r ||
                    Norm[point - {le, le} - origin] < r)
            , 1] // Flatten,
    {m, 1, 9}];

    Flatten[
      Table[
        Table[{possibleSq[[k]], t[[k, s]]} // Flatten,
          {s, 1, Length[t[[k]]]}],
        {k, 1, 9}],
      1]]


allNeighboursList[spi_, sp_, r_] :=
  {#, sortedNeighbours[sp, #, r]} & /@ spi;


allPossibleLinksList[nl_] :=
  Flatten[Function[x, {x[[1]], #} & /@ x[[2]]] /@ nl, 1]
```

Seuraava funktio poikkeaa olennaisesti painotetuissa tapauksissa:

```
allPossibleLinksListWeight[nl_,sp_,r_,a_] :=
  Flatten[
    Module[{p1,p2},
      Function[x,
        {
        x[[1]], #,p1=sp[[x[[1]]/.List->Sequence]];
        p2=sp[[#/.List->Sequence]];
        N[Cases[
          {Norm[p1-p2],Norm[p1+{a,0}-p2],Norm[p1+{0,a}-p2],
          Norm[p1+{a,a}-p2],Norm[p1-{a,0}-p2],
          Norm[p1-{0,a}-p2],Norm[p1-{a,a}-p2]},n_/;n<r
          ][[1]],
          8
        ]
        } &
      /@ x[[2]]]/@ nl],
  1]
```

Interferenssigraafin laskenta:

```
interfering[ll_, nl_, i_] :=
  Module[{recvneighbours,senderneighbours,nlist},
    recvneighbours = Flatten[Cases[nl, {s_, r_} /;
                                s == ll[[i, 2]]], 1][[2]];
    senderneighbours=Flatten[Cases[nl, {s_, r_} /;
                                s == ll[[i, 1]]], 1][[2]];
      Union[
        DeleteCases[
          Flatten[
            Position[ll, {s_, r_} /;
              MemberQ[recvneighbours,s]||
              MemberQ[senderneighbours,r]||
              r == ll[[i, 1]] || s == ll[[i, 2]]]
          ]
         , i]
      ]
];


interferenceList[ll_, nl_] :=
  Table[interfering[ll, nl, i], {i, 1, Length[ll]}]
```

Simuloitu jäähdytys painottamattomalle tapaukselle:

```
annealAbs[list_,imax_,a0_,aif_,nlistp_]:=
  Module[{c,l=list,link,len=0,i,all,n,a,
          maxn,maxlist,m,ain,l2,n2list={}},
    i=0;
    all=l[[1]]//Length;
    n=l[[3]]//Length;
    m=Complement[Range[all],Union[l[[3]],l[[4]]]]//Length;
    maxn=0;
    a=a0;
    c = m*a/(n+m*a);
    While[len<imax,
      i++;
      len++;
      If[RandomReal[]<m*a/(n+m*a),
        link=linkRandom[l];
        l=link[[1]];
        If[link[[2]]!=0,n++];,
      (*ELSE*)
        l=unlinkRandom[l][[1]];
        n--;
      ];
      m=all-(Union[l[[3]],l[[4]]]//Length);
```

```
  (*paivitetaan uusi maksimiarvo*)
      If[n>maxn,
        maxn=n;
        maxlist=l;
        i=0
      (*endif*)];
      If[Quiet[Check[a<$MaxNumber,False]],
      (*then*)
        Quiet[a=Check[aif[len],$MaxNumber];];
        Quiet[c=Check[m*a/(n+m*a),1];];
      ,(*else*)
        If[m>0,
          c=1;,
          c=0;
        ];
      ];
    ];
  {maxn,maxlist}
  ]


linkRandom[{ll_, il_, o_, l_}] :=
  Module[{on = o, locked = l, available, new=0, newinterf},
    (*Listataan mahdolliset linkit, arvotaan niista paalle
      laitettava ja lisataan se paallaolevien listaan*)
    available = Complement[Range[Length[ll]],
                           Union[on, locked]];
    If[available != {},
      new = RandomChoice[available];
      on = Append[on, new];
      (*Listataan syntyvat interferenssit*)
      newinterf = il[[new]];
      (*Lisataan lukittujen listaan uudet interferenssit*)
      locked = Union[locked, newinterf];
    ];
    {{ll, il, on, locked},new}
   ]


unlinkRandom[{ll_, il_, o_, l_}] :=
  Module[{on = o, locked = l, ind, temp, pos},
    (*Arvotaan poistettava sammutettava linkki*)
    ind = RandomInteger[{1, Length[on]}];
    temp = on[[ind]];
    (*Poistetaan sammutettava paallaolevien listasta*)
```

```
      on = Delete[on, ind];
      (*Lasketaan interferenssitilanne uudelleen*)
      locked = Union[il[[on]] // Flatten];
      {{ll, il, on, locked},temp}
   ]
```

Simuloitu jäähdytys painotetulle tapaukselle:

```
annealWeightAbs[list_,imax_,a0_,aif_,nlistp_] :=
  Module[{l=list,link,len=0,i,all,n,a,maxn,maxnlist={},
            nlist=nlistp,maxlist,m,ain,l2,n2list={},c},
     i=0;
     all=Plus@@l[[1,All,3]];
     n=Plus@@(l[[1,#,3]]&/@l[[3]]);
     m=all-Plus@@(l[[1,#,3]]&/@Union[l[[3]],l[[4]]]);
     maxn=0;
     a=a0;
     c=m*a/(n+m*a);
     While[len<imax,
        i++;
        If[RandomReal[]<c,
           link=linkWeight[l];
           l=link[[1]];
              If[link[[2]]!=0,n=n+l[[1,link[[2]],3]]];,
           (*ELSE*)
           link=unlinkWeight[l];
           l=link[[1]];
           n=n-l[[1,link[[2]],3]];
        ];
        m=all-Plus@@(l[[1,#,3]]&/@Union[l[[3]],l[[4]]]);
        AppendTo[nlist,n];
        len=nlist//Length;
        (*paivitetaan uusi maksimiarvo*)
        If[n>maxn,
           AppendTo[maxnlist,{len,(maxn=n)}];
           maxlist=l;
           i=0
        (*endif*)
        ];
        If[Quiet[Check[a<$MaxNumber,False]],
        (*then*)
           Quiet[a=Check[aif[len],$MaxNumber];];
           Quiet[c=Check[m*a/(n+m*a),1];];
        ,(*else*)
           If[m>0,
              c=1;,
```

```
            c=0;];
        ];
      ];
  {maxn,nlist ,maxlist ,l ,n2list}
  ]

linkWeight[{ll_ , il_ , o_ , l_}] :=
  Module[{on = o, locked = l, available , new=0, newinterf ,
            weightSum , weightVector , pdfVector},
      (*Listataan mahdolliset linkit , arvotaan niista paalle
      laitettava painojen mukaan ja lisataan
      se paallaolevien listaan*)
      available = Complement[Range[Length[ll]] ,
                              Union[on, locked]];
      If[available != {},
        weightVector = ({ll[[#,3]]}&/@available)//Flatten;
        new = RandomChoice[weightVector->available];
        on = Append[on, new];
        (*Listataan syntyvat interferenssit*)
        newinterf = il[[new]];
        (*Lisataan lukittujen listaan uudet interferenssit*)
        locked = Union[locked , newinterf];
      ];
  {{ll , il , on, locked},new}
  ]

unlinkWeight[{ll_ , il_ , o_ , l_}] :=
  Module[{on = o, locked = l, ind , temp ,
            weightSum , weightVector , pdfVector},
      (*Arvotaan poistettava sammutettava linkki*)
      weightVector = {#, ll[[#,3]]}&/@on;
      weightSum = Plus@@weightVector[[All,2]];
      weightVector[[All,2]] = weightVector[[All,2]]/weightSum;
      pdfVector = Table[{weightVector[[i,1]] ,
                         Plus@@weightVector[[1;;i,2]]} ,
                         {i, 1, weightVector//Length}];
      ind = (Position[pdfVector,{link_,m_}/;
                      m>=RandomReal[],{1},1]
            //Flatten)[[1]];
      temp = on[[ind]];
      (*Poistetaan sammutettava paallaolevien listasta*)
      on = Delete[on, ind];
      (*Lasketaan interferenssitilanne uudelleen*)
      locked = Union[il[[on]] // Flatten];
  {{ll , il , on, locked},temp}]
```