Tiivistelmä Kunal Shahin Master of Science -työstä: Simulation Based Study of TCP Fairness in Multi-Hop Wireless Networks

Kari Kähkönen

21. maaliskuuta 2006

1 Johdanto

Ad hoc -verkkoihin on kohdistettu viime vuosina paljon aktiivista tutkimusta. Tällaisissa verkoissa jokainen solmu voi vastaanottaa ja lähettää dataa toisille solmuille sekä toimia samalla myös itse reitittimenä. Ad hoc -verkoille on tyypillistä, että verkon muoto muuttuu jatkuvasti solmujen liikkuessa sekä uusien solmujen tullessa verkkoon tai vanhojen poistuessa. Lisäksi ad hoc -laitteet toimivat yleensä akkujen varassa, joten niiden virrankulutuksen kannalta on pyrittävä välttämään raskaita virtaasyöviä operaatioita. Nämä erot perinteiseen langalliseen verkkoon aiheuttavat uusia vaatimuksia verkkojen toiminnassa ja vaikka reitysprotokollien kehittämiseen, TCP:n tehokkuuteen ja MAC kerroksen reiluuteen langattomissa verkoissa on tehty tutkimusta, on TCP:n reiluusominaisuuksien tutkiminen jäänyt hyvin vähälle.

Langallisissa verkoissa tutkimusta reiluuden osalta on tehty, mutta on huomattava, että langallisiin verkkoihin tarkoitetut TCP-versiot eivät välttämättä toimi parhaalla mahdollisessa tavalla langattomissa verkoissa, joissa solmut voivat liikkua ja joissa ilmenee langallisia verkkoja enemmän virheitä tiedonsiirrossa ja niiden seuraksena pakettien katoamisia, jotka eivät siis johdu verkon ruuhkautumisesta.

Työssään Shah pyrkii korjaamaan tätä tutkimuksen puutetta vertaamalla simulaatioiden avulla neljän suositun TCP-version reiluusominaisuuksia langattomassa verkossa ja pohtimalla mahdollisia syitä saamiinsa tuloksiin.

Loppuosa tästä tiivistelmästä on rakentunut seuraavasti. Toisessa kappaleessa kerrotaan taustatietoa TCP:n toiminnasta ja katsotaan lyhyesti millaisia ehdotuksia on esitetty TCP:n tehokkuuden parantamiseksi langattomissa verkoissa. Kappaleessa on myös esitetty huomioita reiluuden arvioimisesta ja kuvattu tapa, jota Shah käyttää simulaatiossa olevien solmjen kokeman reiluuden arvioimiseen. Kolmannessa kappaleessa kuvataan työssä käytetty simulaatiojärjestely ja sen tulokset esitetään neljännessä kappaleessa. Lopuksi on pohdittu työssä esiintyviä hyviä sekä huonoja puolia.

2 Taustatietoa

TCP on ollut käytössä pitkään langallisissa verkoissa ja se onkin yleisimmin käytetty protokolla luotettavaan pakettien välittämiseen. Vaikka protokolla on perusperiaatteiltaan pysynyt hyvin pitkälti samanlaisena syntymästään saakka, on sitä kuitenkin jatkuvasti paranneltu sitä mukaan kun Internet on kehittynyt ja kasvanut. TCP:n alkuperäinen versio tarjosi luotettavan pakettien välityksen kuittausten ja järjestysnumeroiden avulla sekä pystyi säätämään pakettien lähetysnopeutta vastaanottajan ylikuorimittamisen välttämiseksi. Se ei kuitenkaan osannut reagoida verkon ruuhkautumiseen, jota alkoi ilmentyä Internetin kasvun myötä ja tämä heikensi luonnollisesti pakettien läpäisyä. Niinpä TCP:hen alettiin kehittämään algoritmeja ruuhkautumisen hallitsemiseen.

Pohjimmainen idea lähes kaikissa tällaisissa algomitmeissa on, että lähetysnopeutta kasvatetaan kunnes pakettien katoamisesta päätellään verkon olevan ruuhkaantunut, jolloin lähetysnopeus pudototeaan tiettyyn murto-osaan edellisestä ja aletaan taas askel kerrallaan lisäämään lähetysnopeutta.

Nämä algoritmit ovat kuitenkin suunniteltu vartavasten langallisiin verkkoihin, joissa siirtovirheet ovat tänä päivänä harvinaisia verrattuna langattomiin vastineisiinsa. Niinpä TCP voi tulkita siirtovirheestä aiheutuvan paketin katoamisen ruuhkautumiseksi ja näin ollen väärin perustein vähentää lähetysnopeuttaan langattomissa verkoissa.

Seuraavaksi esitetään neljä TCP-versiota, jotka Shah valitsi vertailtavaksi työssään. Näiden neljän version valinnalle hän esitti perusteluna niiden suosiota työn kirjoitushetkellä.

2.1 Työssä käytetyt TCP:n versiot

TCP Tahoe sisältää ruuhkautumisen hallitsemiseen hidas käynnistys (slow start), ruuhkautumisen välttäminen (congestion avoidance) ja nopea uudelleenlähettäminen (fast retransmit) -mekanismit. Hitaassa käynnistyksessä ruuhkaikkunaa, joka kertoo kuinka paljon lähettäjä saa korkeintaan lähettää paketteja, jotta verkko ei ruuhkaantuisi, kasvatetaan yhdellä paketilla aina jokaisen saadun kuittausviestin jälkeen. Kun paketti katoaa, eli ei saada siitä kuittausta, protokolla puolittaa ruuhkaikkunansa ja tallentaa itselleen muistiin ikkunan koon, jonka jälkeen ruuhkaantumista alkoi esiintymään. Nyt kun tämä ikkunankoko jälleen saavutetaan, siirrytään ruukautumisen välttämistilaan, jolloin ruuhkaikkunaa kasvatetaan hitaammin kuin hitaassa käynnistyksessä. Nopea uudelleenlähettäminen puolestaan tapahtuu kun lähettäjä saa, kuittausviestien duplikaatteja tietyn määrän. Tämä kertoo sen, että joku paketti on kadonnut matkalla, mutta sitä seuraavat paketit ovat pääseet perille. Niinpä protokolla lähettää tämän puuttuvan paketin ilman, että se jää odottamaan ajastimen perusteella tapahtuvaa uudelleenlähettämistä. Tämän jälkeen TCP Tahoe jälleen puolittaa ruuhkaikkunansa ja palaa hitaaseen käynnistykseen, mikä ei ole välttämättä kaikkein tehokkain ratkaisu, jos tuon yhden paketin katoaminen ei ollut ruuhkasta johtuvaa.

TCP Reno pyrkii parantamaan TCP Tahoessa esiintyvää edellä kuvattua ongelmaa nopean toipumisen avulla (fast recovery). Protokolla tekee nopean uudelleenlähetyksen ja puolittaa ruuhkaikkunan kuten Tahoekin saadessaan tarpeeksi

kuittausviestien duplikaatteja. Reno ei kuitenkaan aloita hidasta käynnistystä vaan siirtyy nopeaan toipumiseen, missä se kasvattaa ruuhkaikkunaa jokaisen uuden saamansa duplikaattikuittauksen jälkeen ja saatuaan kuittauksen kadonneesta paketista, siirtyy ruuhautumisen välttämistilaan hitaan käynnistyksen asemesta. Tämä menetelmä toimii tehokkaasti silloin kun kadonneita paketteja ei ole useita samaan aikaan.

TCP New Reno parantaa Renon toimintaa juuri noiden useiden kadonneiden pakettien tapauksessa. Jos New Reno ei saa nopean uudelleenlähetyksen jälkeen kuittausta kaikista matkalla olleista paketeista, se tietää, että useampia paketteja oli kadonnut ja pysyy tällöin nopean toipumisen tilassa kunnes se saa kaikki puuttuvat paketit lähetettyä.

TCP SACK nopeuttaa pakettien menetyksestä selviämistä kertomalla lähettäjälle kuittausviestien mukana, mitä paketteja vastaanottaja on saanut. Näin ollen lähettäjä tietää puuttuvat paketit ja voi uudelleenlähettää ne nopeasti.

2.2 Reiluuskriteerit

TCP:n tapauksessa reiluus on tärkeä asia ottaa huomioon, koska esimerkiksi jos TCP:n ruuhkaikkunan koon kasvattaminen riippuu saatavista kuittauksista, niin sellainen yhteys, jonka round trip time on pienempi kuin toisen, pystyy kasvattamaan ruuhkaikkunaansa nopeammin, jolloin se saattaa saada suuremman osan käytettävissä olevasta kaistasta. Ja myös esimerkiksi aikaisemmiin muodostetut yhteydet voivat varata itselleen suuren osan kaistasta näännyttäen uudet yhteydet.

Shah kuvaa paperissaan muuttamia viime vuosikymmeninä esitettyjä mittoja reiluudelle ja käsittelee niiden ominaisuuksia lyhyesti ennekuin esittää mitan, jota hän soveltaa simulaatiossa saataviin tuloksiin. Reiluuden voidaan ajatella tarkoittavan sitä, kuinka lähelle päästään tilannetta, jossa jokaiselle osapuolelle annettu osuus kaistasta tai muusta ominaisuudesta kuten viiveestä, on jaettu osapuolten välille mahdollisimman reilusti. Hyvällä reiluusmitalla tulisi olla ominaisuutena riippumattomuus käytetyistä mittayksiköistä, reiluuden arvon rajoittuminen tietylle äärelliselle välille sekä reiluuden arvon jatkuvuus, jolloin pienetkin muutokset näkyvät reiluusmitassa. Yhtä oikeaa reiluusmittaa ei ole olemassa, vaan valinta eri vaihtoehtojen välillä on tehtävä aina tapauskohtaisesti.

Reiluuden mittaamiseen tässä työssä on käytetty hyväksi max-min periaatetta. Tämän periaatteen mukaan jaettava resurssi on reilusti jakautunut silloin, kun kaikille annetaan ensiksi yhtäsuuri osa resurssista. Tämän jälkeen katsotaan, tarvitseeko vähiten resurssia pyytävä osapuoli vähemmän kuin tuon tasaisen jaon määrän. Jos näin on, ylimäärä jaetan loppujen osapuolten kesken tasan. Sitten sama toistetaan toiseksi vähiten resurssia pyytävän kanssa kunnes päädytään tilanteeseen, että seuraavalla osapuolella ei ole enää ylimääräistä antaa jaettavaksi. Jos siis kaikki haluavat enemmän kuin tasaisen jaon osuuden, saavat kaikki saman määrän resurssista. Muulloin ne, jotka tarvitsevat vähemmän, saavat vain sen verran mitä pyytävät ja loppu jaetaan enemmän vaativien kesken.

Esitetyn periaatteen pohjalta voidaan asettaa reiluusmitaksi

$U = max|\frac{A_i - F_i}{F_i}|$

missä U on maksimaalinen epäreiluus, A_i on osapuolen i todellinen resurssin käyttö ja F_i on osapuolen i max-min periaatteen mukainen reilu osuus. F_i voidaan laskea, jos tiedetään systeemin kapasiiteeti ja jokaisen osapuolen vaatima osa jaetusta resurssista. Kapasiteetti puolestaan lasketaan summaamalla A_i:t yhteen. Jos systeemi on täysin reilu, U saa arvon 0. Täysin epäreilussa systeemissä se saa arvon $n - 1$.

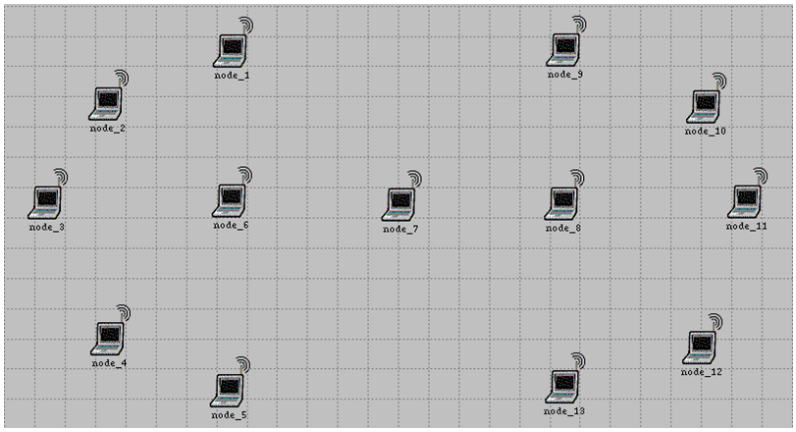
2.3 Ehdotuksia TCP:n tehokkuuden parantamiseen

TCP:n tehokkuutta on tutkittu langattomissa verkoissa ja Shah katsoi reiluusominaisuuksien ymmärtämisen kannalta hyödylliseksi myös näiden tutkimuksissa esiin tulleiden asioiden käsittelemisen. Paperissa esitellään lyhyesti joukko ratkaisuehdotuksia TCP:n kehittämiseksi langattomia verkkoja varten. Käsitellyt ehdotukset ovat Fast Retransmit TCP, Explisit Link Loss Notification TCP, Explisit Congestion Notification TCP, Mobile TCP, I-TCP, M-TCP, Spit TCP, WTCP sekä Snoop TCP.

Näitä esitettyjä menetelmiä ei paperissa jatkossa käsitellä simulaation yhteydessä, mutta kirjoittaja uskoo, että yhdistelemällä esitettyjä menetelmiä pystytään pääsemään aiempaa selvästi parempaan TCP:n toimivuuteen langattomissa verkoissa.

3 Simulaatiomalli

Aiemmin kuvattujen TCP-versioiden reiluuden tutkimiseksi työssä simuloidaan IEEE 802.11 standardin mukaista WLAN-verkkoa. Simulaation toteuttamiseen on käytetty OPNET-verkkosimulaattoria. Kuvassa 1 näkyy käytetty järjestely verkon solmujen sijainnin osalta.



Kuva 1: Simulaation järjestely

Vasemmalla olevat viisi solmua lähettävät paketteja oikeassa laidassa oleville laitteille, jotka tuhoavat paketit vastaanottamisen jälkeen. Laitteiden lähetyste-

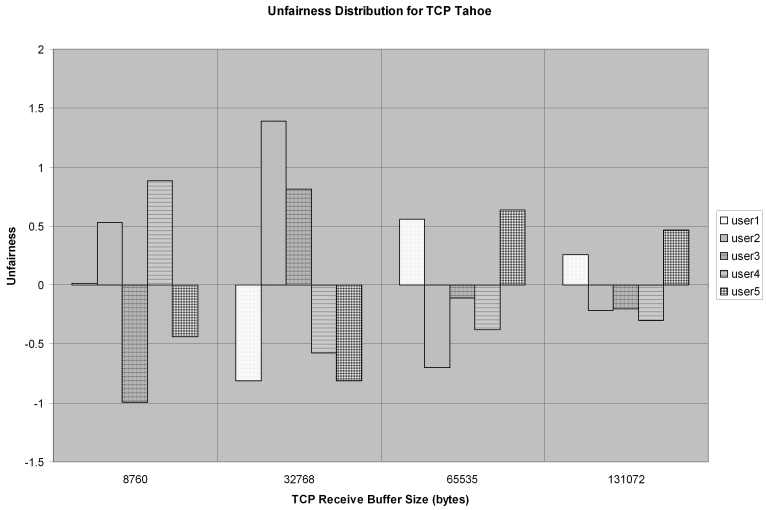
ho on asetettu niin, että kaikki paketit kulkevat kuvan keskellä olevien kolmen solmun läpi, jotka toimivat siis tässä tapauksessa reitittiminä ja muodostavat liikenteen kulkemisen kannalta pullonkaulan. Lähettävät solmut on myös määritelty siten, että node_2 haluaa lähettää kaksi kertaa niin paljon dataa sekunnissa kuin ensimmäinen solmu. Kolmas solmu puolestaan haluaa lähettää kolme kertaa enemmän verrattuna ensimmäiseen solmuun ja niin edespäin. Verkon solmujen sijainti pysyy samana koko simulaation ajan ja jokaisen simulaation kesto on 195 sekunttia.

Solmut käyttävät FTP-protokollaa datan lähtettämiseen ja jokainen viidestä lähettävästä solmusta aloittaa tiedonlähetyksen samanaikaisesti simulaation alkaessa. Pakettien reititykseen on valittu Ad Hoc On Demand Distance Vector protokolla (AODV). Aluperäisessä paperissa on selitetty tarkemmin OPNET-simulaattorin kannalta mitä tiloja paketteja lähettävälla ja vastaanottavalla logiikalla sekä TCP-, AODV- ja WLAN MAC -prosesseilla on ja miten ne toimivat. Myös laitteen akun simuloimista on käsitelty.

Jotta TCP:n eri versioiden reiluudesta saataisiin jonkinlainen kuva, työssä muutettiin muutamia simulaation parametrejä eri simulaatiokertojen välillä. Pakettien kokoa vaihdeltiin 128 tavusta 1024 tavuun. Vastaanottopuskurin kokoa vaihdeltiin 8760 tavusta 131 072 tavuun. Liikenne määrää vaihdeltiin 1.5 Mbps:stä 7.5 Mbps:ään. Lisäksi osassa simulaatioita käytettiin RTS/CTS (Request To Send/Clear To Send) -mekanismia. Muutoin kaikki simulaatiossa käytetyt parametrit olivat IEEE 802.11b standardin mukaisia.

4 Simulaation tulokset ja johtopäätökset

Kullekin neljälle TCP-versiolle suoritettiin simulaatiot aina identtisillä parametrin arvoilla. Kuvassa 2 on annettu yksi esimerkki monista alkuperäisessä paperissa esiintyvistä reiluusjakautumista, jotka on muodostettu simulaation tulosten perusteella.



Kuva 2: Esimerkki saaduista tuloksista

Tuloksista jokaiselle lähettävälle solmulle on laskettu aikaisemmin esitetyt reiluusmitan mukainen arvo ja näiden avulla voidaan sitten vertailla eri TCP-versioita keskenään.

Vaihtelemalla vastaanottopuskurin kokoa ja pitämällä paketin koko vakiokokoisena 256 tavussa, pitämällä tiedonsiirtonopeus 4.5 Mbps:ssä sekä ilman RTS/CTS -mekanismia työssa havaittiin, että reiluus on TCP Tahoella, Renolla ja New Renolla parhaimmillaan kun vastaanottopuskurin koko on suurimmillaan. TCP SACKin tapauksessa reiluus ei huonone vastaanottopuskurin koon pienemmillä arvoilla. Reiluuden parenemista puskurin koon kasvattamisen myötä perustellaan paperissa sillä, että isommilla puskurin koon arvoilla lähettäjä voi lähetttää enemmän dataa ennenkuin sen täytyy pysähtyä odottamaan kuittausviestejä. Tämä tarkoittaa, että vähemmän kaistaa tarvitseva lähettäjä voi lähettää suhteellisesti enemmän dataa muihin verrattuna, joka puolestaan parantaa työssä käytettävän kriteerin mukaista reiluutta.

Liikennemääriä vaihdellessa ilman RTS/CTS:ssää tuloksista havaittiin, että reiluus paranee suurilla liikennemäärillä. Tätä paperissa perusteltiin sillä, että pienillä liikennemäärillä tapahtuu vähemmän pakettien törmäämisiä, jolloin paljon kaistaa vaativat pystyvät lähettämään enemmän dataa. Kääntöpuolena tässä on se, että suurilla liikennemäärillä TCP:n tehokkuus kärsii suuresti törmäyksistä johtuen.

Kun liikennemääriä vaihdellaan RTS/CTS:n kanssa, tulokset ovatkin käänteiset: reiluus paranee pienillä liikennemäärillä. Lisäksi havaittiin, että reiluuden vaihtelu ei ole niin suurta eri liikennemäärien välillä kun RTS/CTS on käytössä. Tuloksia on pyritty selittämään sillä, että suurella liikennemäärällä siirtokanava on todennäköisesti jo jonkun hallussa ja RTS/CTS -mekanismia käytettäessä ei tällöin lähetettä mitään kun kanava on varattuna. Pienemmillä liikennemäärillä jokaisella on puolestaan paremmat mahdollisuudet saada lähettää paketteja reilusti.

Koska pakettien koon vaihteleminen vakiokokoisen vastaanottopuskurin kanssa ei tuottanut selkeitä tuloksia, kokeiltiin lopuksi pakettien kokojen vaihtelemista yhdessä TCP:n vastaanottopuskurin koon kanssa. Tällöin huomattiin, että jokaiselle vastaanottopuskurin koolle löytyy tietty paketin koko, jolla reiluusjakauma muodostuu paremmaksi kuin muilla koon arvoilla. Näyttäsi siis, että parhaan reiluuden saamiseksi täytyy tehdä jonkinlainen kompromissi pienten ja suurien pakettien väliltä.

Saatujen tuloksien perusteella Shah veti muutamia alustavia johtopäätöksiä. RTS/CTS:n käyttö parantaa TCP:n reiluutta verrattuna siihen, että sitä ei käytettäisi. TCP on reilumpi suurilla liikennemäärillä kun RTS/CTS:ssää ei käytetä, mutta puolestaan reilumpi pienemillä liikennemäärillä jos sitä käytetään. Isompi vastaanottopuskuri parantaa reiluutta yleisesti ottaen. Lisäksi pakettien ja vastaanottopuskurin koot ovat riippuvaisia toisistaan reiluuden kannalta. Jos vastaanottopuskurin koko on suuri, tällöin reiluus on paras suurilla paketeilla. TCP Tahoe oli tulosten mukaan reiluin protokollaversio tutkituista, mutta se kärsi huonosta tehokkuudesta muihin verrattuna.

Jatkotutkimusta varten Shah pitää tarpeellisena tutkia saatuja tuloksia vie-

lä huomattavasti syvällisemmin ja lisäksi hän pitää liikkuvuuden ja muutamien paperissa esitettyjen TCP-parannnusten tarkempaa tutkimusta tärkeinä seuraavina askelina.

5 Kritiikki

Shah esittää työssään kiittävästi taustatietoina erilaisia TCP:n reiluuteen liittyviä huomioita, mutta varsinaisessa reiluuden tutkimisessa jää myös parannettavaa.

Esitetty simulaatiomalli ja siitä saadut tulokset ja niiden tulkitseminen eivät kaikin osin pysty vakuuttamaan lukijaa siitä, että tehdyn simulaation pohjalta voisi vetää mitään lopullisia johtopäätöksiä TCP:n reiluudesta. Paperissa on alussa painotettu, että langattomissa verkoissa siirtovirheet aihettavat langallisiin verkkoihin tarkoitettujen TCP-versioiden toiminnan kannalta epäoptimaalista käyttäytymistä. Simulaation kuvauksessa ei kuitenkaan kerrottu missään vaiheessa, millainen simuloidun verkon virhealttius oli esimerkiksi langallisiin verkkoihin verrattuna. Ilmeisesti asia on käsitettävä niin, että työssä on käytetty OPNETin tarjoamia oletusarvoja langattomille verkoille. Olisi ollut mielenkiintoista tutkia miten TCP:n reiluus muuttuu erilaisilla virheen esiintymistodennäköisyyksillä vaikkapa taustahäiriön kasvaessa. Lisäksi tulosten vertailu vastaavaan tilanteeseen, jossa olisi käytetty langallista verkkoa, olisi voinut valoittaa tilannetta paremmin. Nyt lukijalle annetaan jotain tietoja, millaisessa tilanteessa TCP:n versiot toimivat reilummin muihin tilanteisiin verrattuna, mutta tämän perusteella on vaikea vetää johtopäätöstä, onko reiluus pieni vai isompi ongelma langattomissa verkoissa yleisesti ottaen. Tätäkin näkökulmaa olisi voitu käsitellä.

Verkon solmujen liikkumista ei ole käsitelty ja verkon topologia pysyy aina kaikissa simulaatiokerroissa samanlaisena. Näillä asioilla on varmasti vaikutusta TCP:n toimintaan ja sen reiluuteen, joten niihin olisi voinut kiinnittää enemmän huomiota. Nykyisestä yksinkertaisesta simutaatiosta on riskialtista yleistää mitään. Shah tosin myöntää, että asiaa olisi tutkittava huomattavasti nykyistä syvällisemmin ennenkuin kunnollisia johtopäätöksiä voisi tehdä.

Työssä on myös kuvattu turhankin tarkasti jotain simulaatioon liittyviä asioita, kuten akun mallintamista tai TCP:n sisäisiä tiloja. Nämä eivät varsinaisesti anna työn aiheeseen oleellista lisäarvoa vaan vievät enemmänkin huomiota varsinaisesta reiluuden tutkimisesta.