Ohjelman aikavaatimuksien toteutuminen empiirisen testauksen perusteella

Määrittelydokumentissa on kerrottu, että hajautustaulujen aikavaatimuksien pitäisi olla keskimäärin O(1), mutta pahimmassa tapauksessa O(n). Tässä dokumentissa esittelen empiirisesti saamiani tuloksia ja kommentoin aikavaatimuksien toteutumista.

Testiolosuhteet:

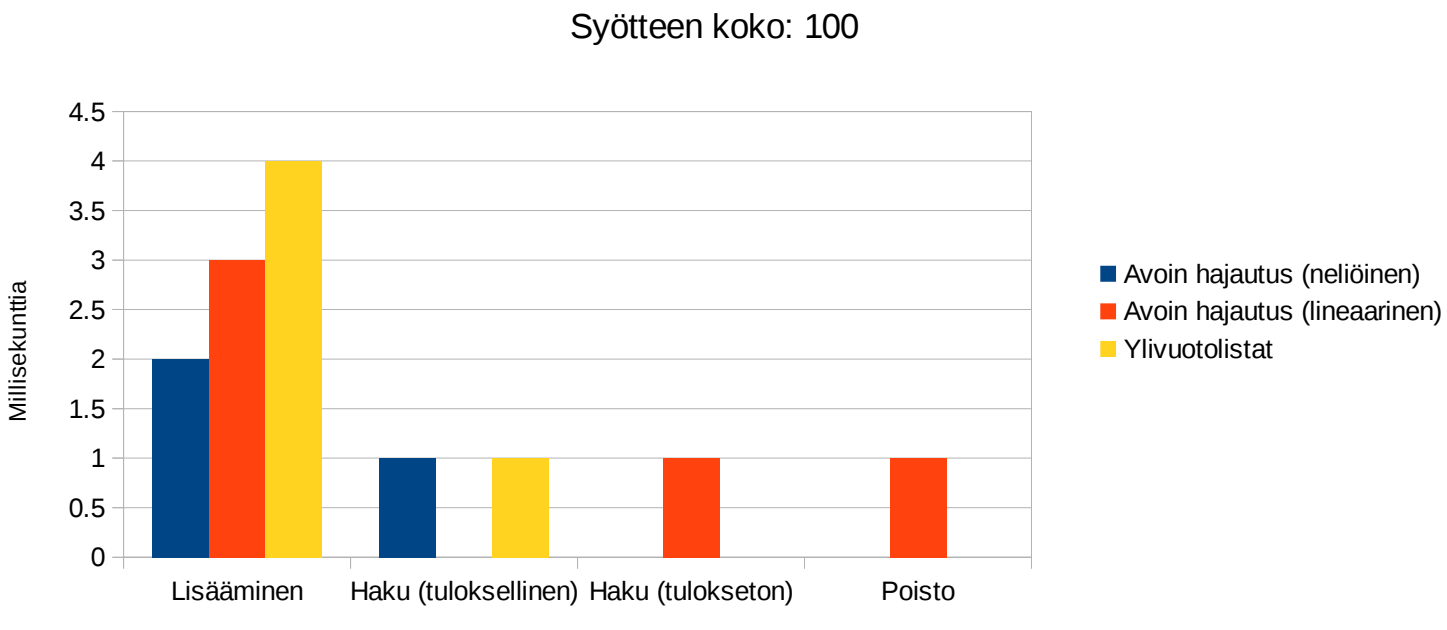
Kolme hajautustaulua, joista kaksi on toteutettu avointa hajautusta käyttäen. Näissä on molemmissa maksimitäyttösuhteeksi säädetty konservatiivinen 0.8 ja minimitäyttösuhteeksi 0.3. Kokeilin muitakin täyttösuhteita toki, mutta nämä vaikuttivat olevan lähimpänä optimaalista uudelleenhajautuksien ja kokeilujonojen pituuksien suhteen. Maksimitäyttösuhteet yli 0.9 hidastavat avoimella hajautuksella toimivia hajautustauluja liikaa kokeilujonojen pitkittyessä, kun taas alle 0.7 uudelleenhajauttaa taulun merkinnät turhan usein. 0.8 oli siis kompromissi näiden suhteen. Minimitäyttösuhde on valittu saman logiikan mukaisesti. Valitut arvot ovat siis aika pitkälti “normaaliarvot” avointa hajautusta käyttävälle hajautustaululle.

Linkitettyä listaa (ylivuotolistoja) käyttävän hajautustaulun täyttösuhteiden valitseminen ei ollut läheskään niin merkityksellistä testituloksien kannalta. Maksimitäyttösuhteeksi on laitettu 5.0 ja minimitäyttösuhteeksi 1.0. Silti testailemalla eri täyttösuhteita sain selville, että niitä voisi vaihtaa melko rajulla kädellä merkittävästi hidastamatta taulun operaatioita. Esimerkiksi maksimitäyttösuhde 25.0 ei ole merkittävästi tuota täyttösuhde 5.0 hitaampi, koska silloinkin linkitetyten listan pituus on keskimäärin vain se 25 uudelleenhajautuksen tapahtuessa. Sen pituisesta listasta on vielä erittäin nopeaa hakea avain. Valitsemani arvot 5.0 ja 1.0 täyttösuhteiksi ovat siis myös erittäin konservatiivisia.

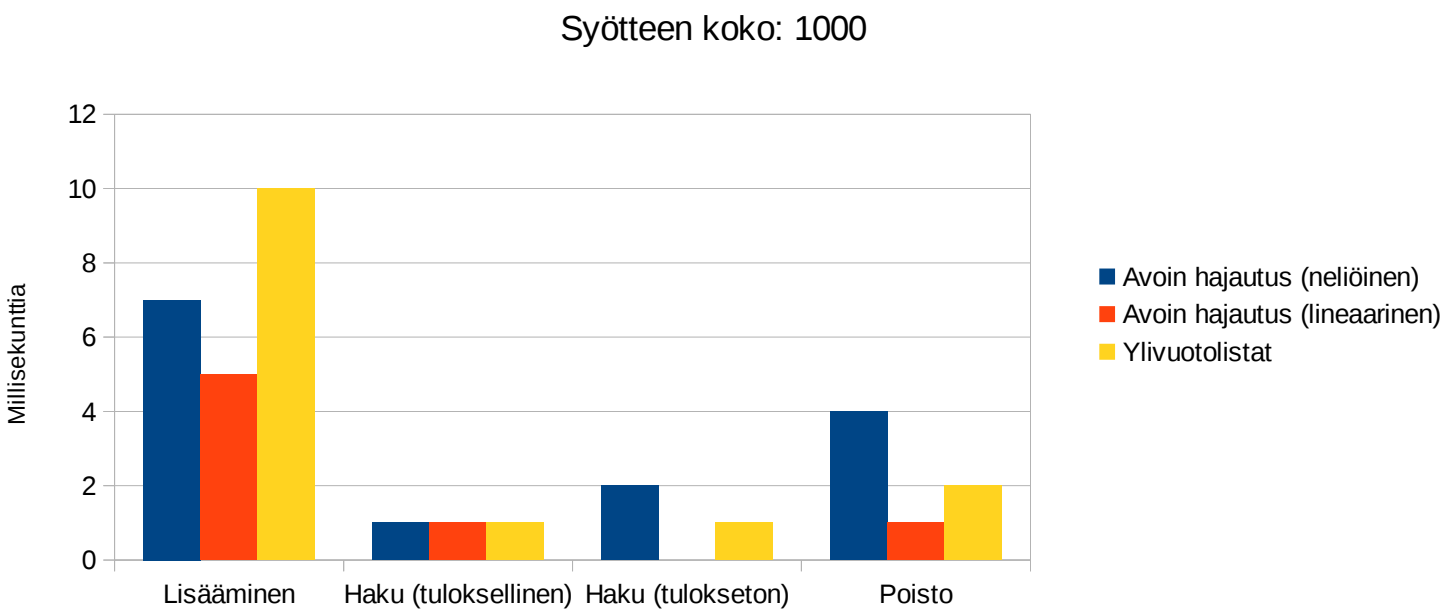
Tavoitteena oli siis valita järkevät, ja mahdollisimman standardeja olosuhteita vastaavat täyttösuhteet, jotka eivät vääristäisi testituloksia.

Hajautustaulujen alkuperäinen koko on 16.
Syötteeksi hajautustauluihin generoidaan syötteen koon määrä pseudorandomeita alfanumeerisia merkkijonoja, jotka sitten lisätään, haetaan ja poistetaan tauluista.

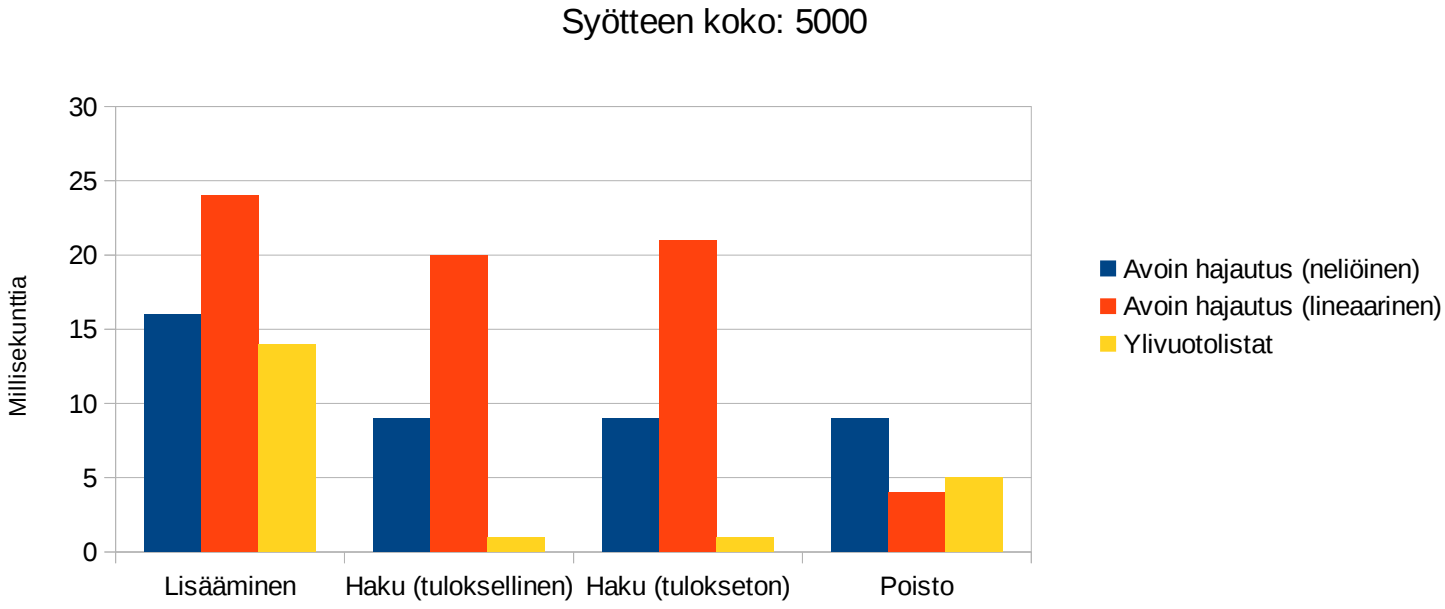
Empiiriset tulokset ja kommentit:



Kun syötteen koko on 100, emme nää juurikaan eroja hajautustauluissa. Jokaisen hajautustaulun operaatiot vievät vain muutamia millisekunteja.

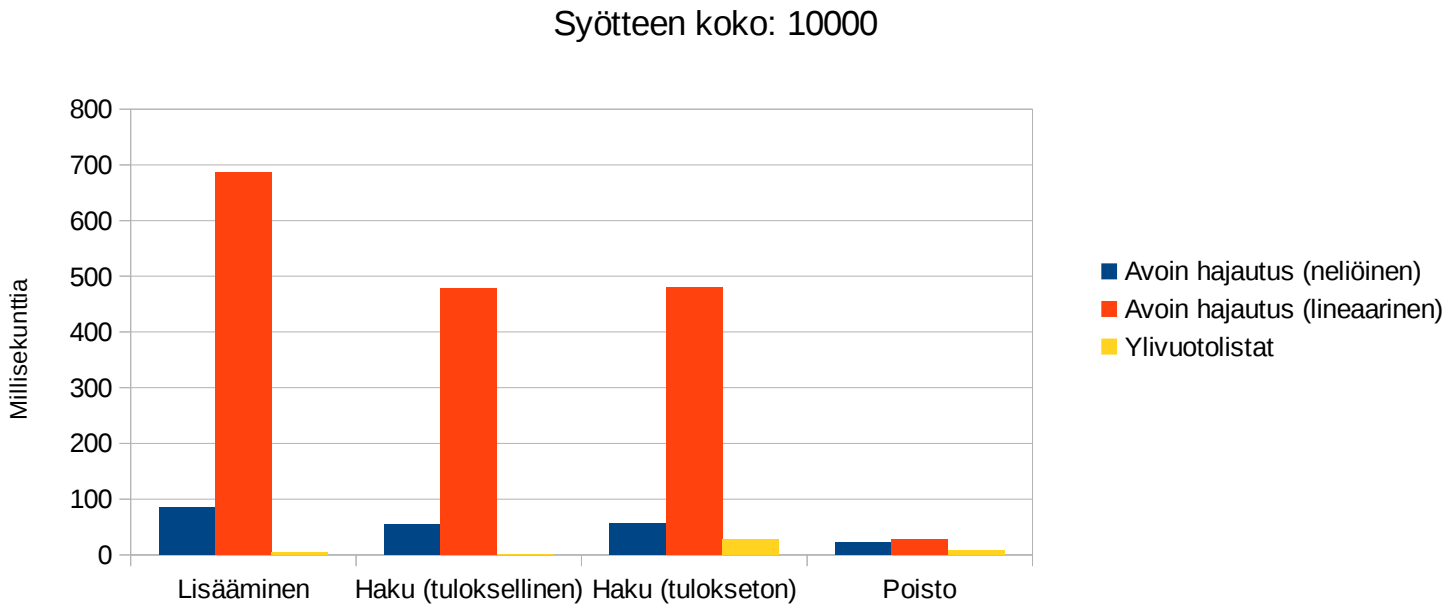


Kun syötteen koko on 1000 ovat erot yhä muutaman millisekunnin luokkaa, eikä hajautustaulujen välille synny juurikaan eroja. Merkille pantavaa on kuitenkin, että pienillä syötteillä ylivuotolistoja käyttävää hajautustauluun lisääminen on muutaman millisekunnin hitaampaa kuin avointa hajautusta käyttäviin hajautustauluihin lisääminen.



Syötteen koon ollessa 5000 alkaa tapahtua. Huomaamme, että ylivuotolistoja käyttävä hajautustaulu päihittää etenkin Haku-operaatioissa selvästi molemmat avointa hajautusta käyttävät hajautustaulut. Huomataan myös, että etenkin lineaarista kokeilujonoa käyttävä hajautustaulu alkaa olemaan jo tässä vaiheessa huomattavasti muita hitaampi, koska avaimet alkavat kasautua tiettyihin paikkoihin

hajautustaulussa, ja siten kokeilujonot niiden löytämistä varten pitkittyvät.



Kun syötteen kooksi valitaan 10000 käy erittäin selväksi, että avoimella hajautuksella toteutettu lineaarista kokeilujonoa käyttävä hajautustaulu alkaa selvästi poikkeamaan hajautustaulujen operaatioiden keskimääräisestä O(1) -aikaisuudesta. Myös neliöistä kokeilujonoa käyttävä hajautustaulu alkaa hidastua (kärsiä avainten kasaantumisesta), joskin huomattavasti vähemmän kuin lineaarinen kokeilujono. Tämä on valittu viimeiseksi testisyötteen kooksi, koska tästä eteenpäin ohjelman suorittamiseen alkoi mennä turhan kauan aikaa lineaarista kokeilujonoa käyttävän hajautustaulun hidastumisen takia.

<u>Aikavaatimuksien toteutuminen empiiristen tulosten perusteella:</u>

Empiiristen tulosten perusteella kaikkien hajautustaulujen operaatiot pysyvät pääosin vakioaikaisina alle 5000 kokoisilla syötteillä. Tulosten perusteella voidaan kuitenkin todeta, että pelkästään Ylivuotolistoja käyttävä hajautustaulu pysyy oikeasti keskimäärin O(1) aikavaatimuksessa suuremmilla syötteillä. Etenkin lineaarista kokeilujonoa käyttävä hajautustaulu hidastuu valtavasti syötteen koon kasvaessa merkintöjen kasaantumisen takia, ja yli 10000 kokoisilla syötteillä sitä voi alkaa kutsua O(n) aikaluokassa olevaksi. Neliöistä kokeilujonoa käyttävä hajaustustaulu pysyy keskimäärin vakioaikaisena huomattavasti lineaarista kokeilujonoa käyttävää hajautustaulua kauemmin ja sitä voi käyttää vielä suuremmillakin syötteillä, mutta alkaa kuitenkin jo 5000 ja 10000 merkinnän kohdalla jäämään huomattavasti ylivuotolistoja käyttävästä hajautustaulusta.

Huomaa kuitenkin, että testeissä on suoritettu hajautustauluille jokainen operaatio syötteen koon määrä kertoja, koska esimerkiksi yksittäisissä hauissa eroja oli vaikea huomata.

Nopeuden kannalta vain ylivuotolistoja käyttävä hajautustaulu vaikuttaa toteuttavan keskimääräisen O(1) aikavaatimuksen syötteen koosta riippumatta. Avointa hajautusta käyttävät toteutukset kärsivät avainten kasaantumisesta ja pakosti alhaisen (pakko olla aina alle 1.0) maksitäyttösuhteen tuomista jatkuvista uudelleenhajautuksista, jotka hidastavat niitä huomattavasti avainten määrän kasvaessa.

Lähteet:

http://www.cs.helsinki.fi/u/floreen/tira2013syksy/tira.pdf
https://github.com/TiraLabra/Loppukesa-2014/wiki/Dokumentaatio

tekemäni ohjelma