Tiralabra 2015/periodi 3: Viikkoraportti 4

Kuudennen projektiviikon luonnollinen ensiaskel oli toteuttaa viime viikolla tehdyn move to front-muunnoksen purku. Menetelmä on varsin yksinkertainen, ja se näkyykin toimivan testitiedostoille odotettavalla tavalla. Varsinaista hyötyä pakkauksessa tosin syntyy vain jos tiedostossa on paljon pitkiä saman tavun toistoja.

Seuraavaksi jatkoin pakkauksen suorituskyvyn testailua. Muokkasin viime viikolla kokeilemaani väliaikojen tulostelua hieman siistimmäksi, ja ajattelin itse asiassa jättää tämän lopulliseen toteutukseen – pitkään kestävän pakkauksen aikana on ihan hauska tietää mitä ohjelma kullakin hetkellä tekee. Suorituskykyä enemmän testaillakseni latasin parin viikon takaisessa viikkopalautteessa mainitun e-colin genomidataa sisältävän ”U00096.2.fas”-tiedoston, joka on reilusti suurempi kuin aiemmin käyttämäni testitiedostot (~5MB). Esimerkkituloste tämän pakkauksesta alla:

```
> java -jar tiralabra.jar c ../sampleFiles/U00096.2.fas aaa
Compression started.
Move to front transformation started
Phase 1/2. Initializing variables ...          Finished. Time: 0,023 sec
Phase 2/2. Replacing bytes with indices ...    Finished. Time: 7,213 sec
Move to front transform finished. Total time: 7.268859000000001 sec
Huffman encoding started
Phase 1/4. Counting bytes ...                  Finished. Time: 0,020 sec
Phase 2/4. Building Huffman tree ...           Finished. Time: 0,004 sec
Phase 3/4. Writing encoding ...                Finished. Time: 0,003 sec
Phase 4/4. Writing data ...                    Finished. Time: 2,307 sec
Huffman Encoding finished. Total time: 2.334156 sec
Compression finished. Total time: 9.609332 sec.
File size reduced from 4697740 bytes to 1350259 bytes (28,7% of original size).
```

Testit genomidatalla paljastivat mielenkiintoisia seikkoja. Pakkaustehokkuus dataan on mainittavan hyvä – tiedoston koko on pakkauksen jälkeen alle 30% alkuperäisestä (1350259 tavua). Vertailun vuoksi sanottakoon, että bzip2 pakkaa saman datan vain hieman paremmin 1328000 tavuun, ja gzip pärjää jopa ohjelmaani huonommin (tulos 1424418 tavua).

Ohjelmani oli kuitenkin varsin hidas, genomidatan pakkaus vei alunperin jopa 30 sekuntia. Seuraava tavoite olikin tämän tehostaminen, jossa koen onnistuneeni varsin hyvin: pakkaus menee nyt alle kymmenessä sekunnissa. Ikävä kyllä (opetustarkoituksen kannalta) nopeutus ei seurannut mistään suuresta oivalluksesta algoritmien tai tietorakenteiden omassa toteutuksessa, vaan eri tiedoston luku- ja kirjoitusmenetelmien käytöstä. Purkuvaihe kestää edelleen noin saman verran kuin pakkaus alunperin, esimerkkituloste alla:

```
> java -jar tiralabra.jar d aaa bbb
Decompression started.
Huffman decoding started.
Phase 1/2. Reading codewords ...               Finished. Time: 0,009 sec
Phase 2/2. Translating data ...                Finished. Time: 23,150 sec
Huffman Decoding finished. Total time: 23.184549 sec
Move to front transform started
Phase 1/2. Initializing variables ...          Finished. Time: 0,022 sec
Phase 2/2. Replacing bytes with indices ...    Finished. Time: 7,273 sec
Move to front transform finished. Total time: 7.29588 sec
Decompression finished. Total time: 30.482656000000002 sec.
```

Valtaosa ajasta menee datan kääntämiseen koodisanakirjan avulla. Tämä on luonnollisesti raskas operaatio: jokaisen luetun *bitin* jälkeen täytyy testata, vastaako kertynyt bittijono jotain koodisanakirjan sanaa. Vaikka koodisanakirja on toteutettu hajautuksella, ja yksittäinen haku saatu

siten toimimaan keskimäärin vakioajassa, näitä hakuja tulee 8*filesize kappaletta, eli genomidatan tapauksessa yli 10 miljoonaa.

Lähdin tällä viikolla toteuttamaan Burrows-Wheeler -muunnosta, josta olenkin aiemmin maininnut. Projektin loppu lähenee, enkä ollut ihan varma kannattaako tässä kohtaa ohjelmaa lähteä enää laajentamaan, mutta toisaalta BWT on olennainen muiden osien tehokkaan toiminnan kannalta, ja itseäni huvitti testailla, joten tässä sitä nyt ollaan.

Lähdin toteuttamaan pakkausta suoraan varsin optimoidusti (ei rakenneta koko matriisia vaan käytetään osoittimia, jos menetelmä on tuttu). Sainkin tuon toimimaan ihan järkevässä ajassa, ”rivien” järjestäminen pohjautuu string quicksortiin, ja se on pakkauspuolen raskain operaatio (O nlogn). Hieman tosin epäilyttää, voiko koodi olla oikein, pakkaustulokset ovat nimittäin liiankin hyviä. Aiemmin käyttämilläni testitiedostoilla (alice, e-coli, turingin kuva) pakattujen tiedostojen koko oli kaikissa < 15% alkuperäisestä, mikä ylittää huomattavasti ”oikeiden” pakkausohjelmien tuloksen. Ensi viikolla toteutan purun, ja yritän selvittää toimiiko tämä oikeasti oikein.

Ensi viikolle ja loppuprojektiin tavoitteet on nyt aika selkeät. Yritän hioa tuon BWT:n vielä kuntoon, minkä lisäksi jäljellä onkin enää testailua (muunnosten testit uupuvat vielä), ja dokumenttien kirjoittelua.

Ja viimeisenä: Laitoin mainin ajamaan myös BWT:n pakkaukselle, joten purkaminen ei tällä erää toimi oikein. Jos haluat(te) kokeilla myös purkua, voi Mainin compress-metodissa poistaa parin rivin kommentoinnit. Yritin merkata tuon selvästi.