Toteutusdokumentti

Aineopintojen harjoitustyö: Tietorakenteet ja algoritmit

Joonas Longi

Ohjelman yleisrakenne

Työssä on onnistuneesti ratkaistu tiedon tiivistämisen ognelma. Pakatun tiedoston koko on n. 35-70% alkuperäisestä tiedostosta. Pakattu tiedosto voidaan myös purkaa taikaisin alkuperäiseen muotoon.

Pääohjelma Tiralabra.java luo Huffman.javan ja käynnistää sen. Huffman.java kyselee käyttäjältä, haluaako hän pakata vai purkaa tiedoston, tiedoston sijainnin sekä uuden tiedoston nimen. Pakkauksen tai purkamisen jälkeen ohjelma kertoo siihen kuluneen ajan millisekunteina. Ohjelmassa käytetään kahta pääluokkaa: TiedostonPakkaaja ja TiedostonPurkaja. Nämä luokat käyttävät samoja tietorakenteita ja luokkia pakkaamiseen ja purkamiseen. Tietorakenteet löytyvät pakkuksesta tietorakenteet. TiedostonLukija ja TiedostonKirjoittaja löytyvät pakkauksesta tiedostonkäsittely, sekä ToistojenLaskija ja itse TiedostonPakkaaja ja TiedostonPurkaja löytyvät tiivistys pakkauksesta. Käytetyt tietorakenteet ovat Huffmanin puu, node, jono ja minimiprioriteettijono, joka toteutettiin minimikekona.

Saavutetut aika- ja tilavaativuudet

Ohjelmassa aikavaativuus riippuu kaikkien merkkien määrästä ja eri merkkien määrästä, joten merkitään kaikkien merkkien määrää n:llä ja eri merkkien määrää k:lla. koska k on aina <= 256 se voidaan nähdä vakiona. Näin ollen aikavaativuus on O(n).

Aikavaativuudet vaiheittain:

Toistojen laskeminen: Laskija lukee tiedoston merkki kerrallaan kerran läpi O(n) ja tallentaa jokaisen merkin toiston määrän taulukkoon (vakio).

Jonon muodostus: Minimikekoon laitetaan k kirjainta O (k) ja jokaiselle kutsutaan korjaa metodia (heapify). Itse korjaa metodin aikavaativuus on k kokoisessa keossa O(log k), joten jonon muodostus tapahtuu ajassa O(k log k). Keon maksimikoko on k, joten tilavaatisuu on O(k).

Puun muodostus: Binääripuuhun lisäämisen aikavaativuus h korkuiseen puuhun on O(h). Puussa on maksimissaan k lehteä (eri kirjainten määrä), jolloin solmuja on 2k-1(wikipedia).

Tiedoston kirjoittaminen: Pakattava tiedoston luetaan vielä kerran läpi O(n) ja merkit laitetaan jonoon. Mutta koska k < 256, aikaavievimmät osuudet ovat toistojen lasku ja tiedoston kirjoitus O(n).

Tiedoston purkaminen: Purku on loppujen lopuksi vain riippuvainen tiedostossa olevien merkkien määrästä, joten aikavaativuus on O(n).

Puutteet

Aluksi suunnittelin, jotakin toista pakkausmetodia toteutettavaksi Huffmanin kanssa, mutta aika ja osaaminen ei antanut tämän osalta periksi. Kahden eri menetelmän, esimerkiksi Huffmanin ja Lempel-Zivin vertailu olisi ollut mielenkiintoista toteuttaa, mutta se jäänee myöhemmäksi projektiksi. Ohjelma ei käsittele virheitä ja virhesyötteitä parhaalla mahdollisella tavalla, mutta en kokenut asiaa kovin oleelliseksi. Tietorakenteita voisi myös ehkä ajan kanssa optimoida paremmiksi.

Lähteet

Wikipedia, Huffman-coding, http://en.wikipedia.org/wiki/Huffman_coding (1.8.2013)
T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein: Introduction to Algorithms,3nd ed., MIT Press, 2009.