Toteutusdokumentti

Ohjelman keskeisimmän sisällön muodostavat luokat HuffmanKoodaus ja LempelZivKoodaus, joiden jotka sisältävät metodit, joiden avulla voidaan tiivistää syötteenä annettu tiedosto käyttäen Huffmanin tai Lempel-Zivin algoritmeja. HuffmanKoodaus-luokka on näistä rakenteeltaan monimutkaisempi johtuen itse algoritmin jakautumisesta useaan osaan. Se sisältää kokoelman metodeita, joiden avulla voidaan luoda koodauksessa tarvittavat aputietorakenteet, kuten kooditaulukko. Se sisältää myös metodit, joiden avulla voidaan ensin rakennettuja aputietorakenteita käyttäen tiivistää syöteaineisto. Lempel-Ziv-koodaus ei käytä tällä tavoin toteutettuna mitään taulukkoa monimutkaisempaa tietorakennetta. Myöskin data käydään siinä läpi ainoastaan kerran, joten algoritmin toiminta on suoraviivaisempaa.

Pääohjelma tarjoaa toiminnallisuudet, joiden avulla on mahdollista testata mainittujen luokkien pakkaus- ja purkamisalgoritmejä erilaisilla aineistoilla. Testattaessa mitataan ensisijaisesti pakkaamisen tehokkuutta laskemalla syöteaineiston pituus ja siitä tiivistetyn pakatun tiedoston pituus. Näiden tunnuslukujen perusteella voidaan laskea pakkaussuhde, joka on keskeisin tiedon tiivistyksen suorituskykyyn liittyvä tunnusluku. Pääohjelma mittaa myös tiivistämiseen ja purkamiseen kuluvan ajan, joskin algoritmin ajallinen tehokkuus on toissijainen algoritmin tehokkuuden kriteeri. Aikavaatimuksen relevanssia vähentää runsas levy-I/O:n käyttö sekä se seikka, että sekä Huffman-koodauksen että Lempel-Ziv-koodauksen aikavaatimus on lineaarinen.

Ohjelman päähakemistoon on kerätty jo joitakin testauksessa käytettäviä aineistotyyppejä ml. tekstitiedostoja, äänitiedostoja ja bitmap-grafiikkatiedostoja.

Toteutusten ajallisessa suorituskyvyssä on varmasti jonkin verran tehostamisen varaa, mutta tämä ei ole merkittävin prioriteetti, koska ajallista tehokkuutta kiinnostavampi tutkimuskohde on pakkauksen tehokkuus erilaisilla aineistoilla. Pakkaustehokkuuden osalta itse koodin analysointia hedelmällisempää on sen pohtiminen, miksi yhdenlaisella aineistolla toinen algoritmi on pakkaussuhteeltaan tehokkaampi, kun taas toisenlaisella datalla tilanne on päinvastainen.