Tiralabra-projekti syksy 2020

Pakkausalgoritmien vertailu –
ongelmakentän ja algoritmien esittely

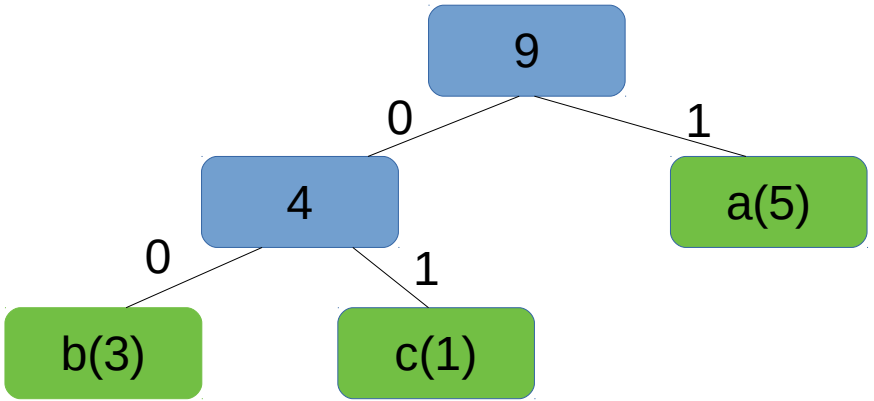
Ongelma

- Data ilmaistaan tietyllä määrällä bittejä
  - Esim. tekstimuotoinen data
    - Ascii merkit 8 bittiä
    - Utf-8 vaihteleva määrä bittejä
- Voidaanko sama data ilmaista jossain tiiviimmässä muodossa (ts. vähemmällä määrällä nollia ja ykkösiä)?
  - Tavoitteena talletustilan säästäminen tai esim. verkon yli siirrettävän datan määrän minimoiminen
  - Hintana saa olla se, että purkaminen / pakkaus vie jonkin verran koneaikaa

Huffman koodaus

- Huffman algoritmissa data ilmaistaan vaihtelevan mittaisina bitteinä
- Useimmin käytetty kirjain vähäisimmällä määrällä bittejä, harvimmin käytetty useammalla bitillä
  - Esim. ’aaaaabbbc’ → a = 1, b = 00, c = 01 → 111110000001
    - Vrt. a = 97 = 1100001, b = 98 = 1100010, c = 99 = 1100011 → 110000111000011100001110000111000011100010110001011000101100011
- Koodauksessa käytetään ns. Huffman-puuta
  - Tarvitaan purkuvaiheessa, eli pitää tallentaa tiedostoon mukaan
    - Suurentaa tiedostoa, jolloin todella pienet tiedostot voivat olla pakkauksen jälkeen saman kokokoisia tai jopa suurempia!

Huffman puu



LZW pakkaus

- Lempel–Ziv–Welch pakkauksessa idea on (hyvin karkeasti yksinkertaistaen) käyttää viittauksia aikaisemmin esiintyneisiin merkkeihin / merkkijonoihin.
- Muodostetaan viittauksista ”sanakirja” sitä mukaa kun dataa luetaan / pakataan
- Sanakirjaa ei tarvitse tallentaa pakatun tiedoston mukaan, vaan se voidaan generoida pakatun datan perusteella

LZW yksinkertaistettu esimerkki

- Esim. jos alkuperäinen merkistö käyttää 8 bitin merkkejä, a = 01100001 jne.
  - Eli: aaa = 011000010110000101100001 (yht. 8+8+8=24bittiä) jne.
- ”Laajennetaan” merkistöä siten, että käytetäänkin esim.9 bittiä
  - a = 001100001, eli huonompi kuin alkuperäinen!
    - Ehkä mahdollisuuksien mukaan voidaan toki yrittää optimoida ja käyttää eri bittimäärää alkuperäisille merkeille tms.?
  - Mutta tällöin voidaan ilmaista esim. aa = 100000000
    - Eli aaa → 001100001100000000 (yht. 9+9=18bittiä)