Määrittelydokumentti

Aineopintojen harjoitustyö: Tietorakenteet ja algoritmit (alkukesä)

Sami Korhonen
014021868
sami.korhonen@helsinki.fi

Tietojenkäsittelytieteen laitos
Helsingin yliopisto
23. kesäkuuta 2014

Työn aihe

Ongelma ja tavoite

Rahtifirma NopsaToimitus haluaa optimoida konttikuljetuksissa käytettävän tilan. Kontin tila on käytetty sitä optimaalisemmin mitä suuremman tilavuuden edestä siihen mahtuu laatikoita.

Tavoitteena on kehittää mahdollisimman nopea ja pätevä algoritmi, joka selvittää sopivan tavan pakata annetut laatikot annettuun konttiin. Ongelmaan ei ole toistaiseksi löydetty ratkaisua, joka antaisi parhaan mahdollisen ratkaisun kaikissa tilanteissa siedettävässä ajassa, eikä sitä luultavasti tule löytymään tämän harjoitustyön tuloksenakaan. Ongelman luonteesta johtuen algoritmin tarkoitus on löytää ongelmaan sopiva ratkaisu tietyllä tarkkuudella sopivassa ajassa. Tämän tarkkuuden tutkiminen on mielekkäin tapa tutkia ohjelman pätevyyttä aikavaativuuden ohella.

Määritelmiä

Yksiköt ja koordinaatisto

Työssä käytetään yksikköinä kokonaisluvuiksi pyöristettyjä senttimetrejä, sillä todellisuudessa laatikoiden mitat eivät ole esimerkiksi puristuvuuden vuoksi tämän tarkempia.

Koska kontti ja laatikot ovat suorakulmaisia särmiöitä, voidaan käyttä kolmiulotteista karteesista koordinaatistoa. Tässä työssä koordinaatiston akseleita merkitään kirjaimin x, y ja z.

Laatikot

Laatikoiden asettamiseen liittyy muutamia sääntöjä:

1. Laatikon tulee olla kokonaan kontin sisäpuolella
2. Laatikot eivät saa olla limittäin toistensa kanssa

3. Laatikon tulee olla kontin seinien suuntaisesti
4. Laatikko tulee olla tuettu koko pohjaltaan

Pakkaaminen

Toistaiseksi työssä ei kiinnitetä huomiota seuraaviin seikkoihin:

1. Laatikot tulisi latoa niin, että ne kestävät kuljetuksen
2. Rahdin tulisi olla mahdollisimman helppo pakata ja purkaa

Ratkaisu

Tietueet ja tietorakenteet

Palkki

Palkki koostuu yhdestä tai useammasta samanlaisesta laatikosta ja se on muodoltaan suorakulmainen särmiö. Palkeista kasattu lista antaa algoritmin lopputuloksen. Kaikki palkissa olevat laatikot ovat käännettynä samaan asentoon. Tämäntyyppiset yksinkertaiset palkit on esitellyt mm. Tobias Fanslau ja Andreas Bortfeldt [1, sivu 9].

Pakkaussuunnitelma

Pakkaussuunnitelma on lista asetetuista palkeista. Pakkaussuunnitelma sisältää myös tiedon palkkien kokonaistilavuudesta ja laatikoiden määrästä.

Tilapalkki

Kontissa vapaana oleva eli tyhjä tila koostuu suorakulmaisista särmiöistä, joita kutsun tässä työssä tilapalkeiksi. Näihin tilapalkkeihin asetetaan palkkeja.

Tila

Tämä pitää tietoa pakkauksen tämänhetkisestä tilasta esimerkiksi sisältäen pakkaussuunnitelman, listan vapaista laatikoista sekä pinon vapaista tilapalkeista.

Algoritmit ja funktiot

Tässä esitetyt pseudokoodit ovat hyvin pelkistettyjä versioita, joista selviää idea ohjelman takana. Esimerkiksi virheiden tarkastuksia, ja kaikkia tilanteita ei ole kirjoitettu näihin auki.

Pääalgoritmi

Kontin pakkaamisen orkesterinjohtajana on seuraava yksinkertainen algoritmi:

```
function pakkaaKontti(kontti, laatikot)
    pakkaussuunnitelma = new PakkausSuunnitelma
    tila = new Tila
    while (tila.tilapalkkipino not empty)
         tilapalkki = tila.tilapalkkiPino.pop()
        palkki = haeParasPalkkiLaatikoista(tilapalkki, tila.vapaatLaatikot)
        if (palkki not null)
         tila.paivita(palkki, tilapalkki)
    endwhile
return pakkaussuunnitelma
```

Tässä alustetaan aluksi pakkaussuunnitelma ja tilan. Ensimmäisellä while-loopin iteraatiolla tilan sisältämä tilapalkkiPino poppaa koko kontin kokoisen vapaan tilapalkin. Tähän tilapalkkiin sopivaa palkkia etsii haeParasPalkkiLaatikoista. Tämä palauttaa arvon null, mikäli sopivaa palkkia ei löytynyt. Jos sopiva palkki löytyy, päivitetään tila tämän mukaan. Palkin lisäämisen seurauksena tilapalkki jaetaan palkkeja lisätessä pienemmiksi tilapalkeiksi, joita tulee aina kolme kappaletta lisää kun asetetaan uusi palkki.

haeParasPalkki(tilapalkki, vapaatLaatikot)

```
function haeParasPalkki(tilapalkki, vapaatLaatikot)
    paras = new Palkki
    for each laatikko in vapaatLaatikot
        Palkki p = valitseOrientaatio(tilapalkki, laatikko)
        if (parempi(p, paras))
            paras = p
    end for
return paras
```

Tässä hieman yksinkertaistettu versio palkin valitsemisesta. Tämä käy läpi kaikki vapaat laatikot, ja valitsee niistä tehdyistä palkeista parhaiten sopivan. Funktio valitseOrientaatio palauttaa palkin, joka on tehty sille parametriksi annetusta laatikosta. Mikäli tämä palkki p on parempi kuin paras, asetetaan se uudeksi parhaaksi.

valitseOrientaatio(tilapalkki, laatikko)

```
function valitseOrientaatio(tilapalkki, laatikko)
    paras = new Palkki
    for i = 0 to 6
        laatikko.asetaOrientaatio(i) // valitaan orientaatio
        // luodaan nx, ny ja nz
        if (nx*x*ny*y*nz*z > suurinTilavuus)
            suurinTilavuus = nx*x*ny*y*nz*z
            paras = new Palkki(sijainti, laatikko, nx, ny, nz)
    end for
return paras
```

Tämä on myös toiminnaltaan yksinkertaistettu versio laatikon eri orientaatioiden kokeilemisesta. Kaikki 6 mahdollista ei-identtistä orientaatiota käydään läpi. Luodaan luvut nx, ny ja nz, jotka kertovat kuinka monta laatikkoa mahtuu vierekkäin x, y ja z-akseleille. Näiden luomisessa otetaan huomioon

mm. laatikoiden määrä ja tilapalkin koko. Näiden lukujen perusteella luodaan uusi palkki, mikäli sen tilavuus on suurempi kuin aiemmaksi suurimman palkin tilavuus.

tila.paivita(palkki, tilapalkki)

Palkki lisätään pakkaussuunnitelmaan, tilapalkki poistetaan, ja tilapalkki-Pinoon asetetaan kolme uutta tilapalkkia, joiden sijainti ja koko määräytyy palkin ja aiemman tilapalkin mukaan.

Syötteet ja käyttäminen

Syötteet

Ohjelma antaa käyttäjän valita täytettävän kontin muutamasta vaihtoehdosta. Sen jälkeen ohjelma lukee sopivassa muodossa olevasta tiedostosta tiedot pakattavista laatikoista.

Käyttäminen

Ohjelman käyttö tapahtuu yksinkertaisen komentorivipohjaisen käyttöliittymän kautta. Ohjelma näyttää kaikki ymmärtämänsä viisi komentoa.

Aika- ja tilavaativuudet

Aikavaativuus

Algoritmin käyttämä aika riippuu hyvin pitkälti sille annetuista laatikoista, eikä pelkästään konttiin pakattavien laatikoiden määrästä. Koska ongelmaan ei löydetä varsinaisesti oikeaa ratkaisua, on jokseenkin mielivaltaista tutkia aikavaativuutta perinteisin tavoin. Enkä näe tavotteellisten aikavaativuuksien määrittelemistä miellekkäänä. Aikavaativuuksia voisi tutkia sen mukaan, kuinka kauan on tarvittu aikaa tietyllä tarkkuudella optimaalisen pakkaussuunnitelman löytymiseen tietyllä laatikkosarjalla. Aikavaativuuteen

ja algoritmin tarkkuuteen vaikuttaa hyvin paljon laatikoiden homogenisuus.
Eli se kuinka monenlaisia laatikoita sarjassa on.

Viitteet

[1] Tobias Fanslau, Andreas Bortfeldt, 2008
A Tree Search Algorithm for Solving the Container Loading Problem
http://www.fernuni-hagen.de/wirtschaftswissenschaft/download/beitraege/db426.pdf
viitattu 22. kesäkuuta 2014