Approksimointialgoritmit (kevät 2010)
Harjoitus 4 (15. helmikuuta)
Ratkaisuja (Jouni Siren)

1. Tarkastellaan ahneita algoritmeja repunpakkausohjelmaan. Olkoot esineet $a_1, \ldots, a_n$ yksikkötuoton $\mathsf{profit}(a)/\mathsf{size}(a)$ mukaisessa laskevassa järjestyksessä ja repun koko B.

   (a) Algoritmi, joka valitsee esineet $a_1, \ldots, a_{k-1}$, voi tuottaa mielivaltaisen huonon approksimointisuhteen. Vihje esimerkin löytämiseen saadaan seuraavasta kohdasta: joskus esineen a_k valitseminen olisi parempi vaihtoehto.

   Olkoon $B = 2$ ja $0 < \alpha(n) < 1$ jokin polynomisessa ajassa laskettava funktio. Ahne algoritmi valitsee ensimmäiset $k-1$ esinettä, joille oletetaan $\mathsf{size}(a_i) = \mathsf{profit}(a_i) = \alpha(n)/(k-1)$ kaikilla $i < k$. Näiden esineiden yksikkötuotto on $\mathsf{profit}(a_i)/\mathsf{size}(a_i) = 1$. Ahneen algoritmin valitsemien esineiden yhteiskooksi ja kokonaistuotoksi tulee $\alpha(n)$.

   Asetetaan $\mathsf{size}(a_k) = 2$ ja $\mathsf{profit}(a_k) = 1$. Nyt yksikkötuotoksi tulee aikaisempia esineitä pienempi $\mathsf{profit}(a_k)/\mathsf{size}(a_k) = 1/2$, eikä esine a_k mahdu enää mukaan ahneen algoritmin valitsemien esineiden joukkoon. Kuitenkin valitsemalla pelkästään esine a_k saadaan kokonaistuotoksi $\mathsf{profit}(a_k) = 1$, joten ahneen algoritmin approksimointisuhde ei voi olla parempi kuin $\alpha(n)$. Koska $\alpha(n)$ voi olla mielivaltaisen lähellä nollaa, voi myös approksimointisuhteesta tulla mielivaltaisen huono.

   (b) Merkitään $R_1 = \{a_1, \ldots, a_{k-1}\}$ ja $R_2 = \{a_k\}$. Koska edellisen kohdan ahne algoritmi valitsee joukon R_1 mutta ei enää alkiota a_k, täytyy olla $\mathsf{size}(R_1) + \mathsf{size}(R_2) > B$. Toisaalta ainakin toisella näistä joukoista täytyy päteä $\mathsf{profit}(R_i) \geq (\mathsf{profit}(R_1) + \mathsf{profit}(R_2))/2$. Lisäksi $\mathsf{profit}(R_1) + \mathsf{profit}(R_2) \geq \mathsf{OPT}$, sillä näiden joukkojen esineet ovat yksikkötuotoltaan suurimpia ja niiden yhteiskoko on repun kokoa suurempi. Valitsemalla parempi näistä joukoista saadaan siis kokonaistuotoksi $\max(\mathsf{profit}(R_1), \mathsf{profit}(R_2)) \geq \mathsf{OPT}/2$.

2. Tehtävän yksityiskohtainen ratkaisu löytyy lähteeksi annetun artikkelin luvusta 3. Tässä lähinnä hahmotellaan ratkaisua.

   Annettuna on n positiivista kokonaislukua $a_1 < \ldots < a_n$. Käypiä vastauksia ovat osajoukkoparit $S_1, S_2 \subseteq \{1, \ldots, n\}$, joilla $A_1 = \sum_{i \in S_1} a_i \geq \sum_{i \in S_2} a_i = A_2$. Tavoitteena on minimoida suhde A_1/A_2 ja päästä kertoimen $1 + \varepsilon$ sisään optimaalisesta ratkaisusta.

   Algoritmi

   Hahmotellaan ensin pseudopolynominen algoritmi ongelmaan. Algoritmi täyttää taulukon $t[0 \ldots n, 0 \ldots \sum_{i=1}^{n} a_i]$ niin, että $t[i,j] = 1$, jos esineistä $a_1, \ldots, a_i$ voidaan valita osa niin, että alkio a_i on mukana ja yhteispainoksi tulee j. Jos jollain j pätee $t[i_1, j] = t[i_2, j] = 1$ joillain $i_1 \neq i_2$, ovat nämä osajoukot erillisiä pienimmällä tällaisella j. Tällaisessa tapauksessa sanotaan, että optimiratkaisu on eksakti. Muussa tapauksessa jokaisella j pätee $t[i,j] = 1$ korkeintaan yhdellä i, ja optimiratkaisu löytyy käymällä läpi kaikki osajoukkoparit.

   FPTAS muodostaa ja ratkaisee instanssit I_m kaikilla $m = 2, \ldots, n$ ja palauttaa parhaan löytämänsä ratkaisun. Kukin instanssi I_m koostuu m pienimmästä alkiosta $a_1, \ldots, a_m$, ja siinä käytetään skaalauskerrointa $k(m) = \varepsilon^2 \cdot a_m/(2m)$.

   Olkoon $n_0 \leq n$ suurin arvo, jolla $k(n_0) < 1$. Jos $m \leq n_0$, ovat instanssin I_m luvut polynomisen kokoisia ja se voidaan ratkaista optimaalisesti pseudopolynomisella algoritmilla. Muussa tapauksessa muodostetaan muunnettu instanssi I'_m niistä arvoista $a'_i = \lfloor a_i/k(m) \rfloor$, joilla $a'_i \geq m/\varepsilon$. Olkoon $t \geq 1$ tällaisten arvojen määrä.

   Jos $t = 1$, olkoon j pienin ei-negatiivinen kokonaisluku, jolla $\sum_{i=j+1}^{m-1} a_i < a_m$. Jos $j = 0$, valitaan $S_1 = \{m\}$ ja $S_2 = \{1, \ldots, m-1\}$, muussa tapauksessa taas $S_1 = \{j, \ldots, m-1\}$ ja $S_2 = \{m\}$. Jälkimmäisessä tapauksessa siis vasta alkion a_j lisääminen kasvatti osajoukkoa S_1 vastaavan summan suuremmaksi kuin osajoukkoa S_2 vastaava summa.

   Jos $t > 1$, ratkaistaan I'_m pseudopolynomisella algoritmilla. Jos löytynyt ratkaisu on eksakti, palautetaan se. Muussa tapauksessa käydään läpi kaikki luvuista $m - t + 1, \ldots, m$ saatavien erillisten osajoukkojen muodostamat parit (R_1, R_2), joissa toinen joukoista sisältää luvun m, ja palautetaan paras seuraavalla mekanismilla saatavista osajoukkopareista (S_1, S_2).

Oletetaan, että $B_1 = \sum_{i \in R_1} a_i \geq \sum_{i \in R_2} a_i = B_2$. Olkoon j pienin ei-negatiivinen kokonaisluku, jolla $B_2 + \sum_{i=j+1}^{m-t} a_i < B_1$. Jos $j = 0$, palautetaan $S_1 = R_1$ ja $S_2 = R_2 \cup \{1, \ldots, m-t\}$. Muuten jos $m \in R_1$, palautetaan $S_1 = R_2 \cup \{j, \ldots, m-t\}$ ja $S_2 = R_1$. Muuten palautetaan $S_1 = R_1$ ja $S_2 = R_2 \cup \{j+1, \ldots, m-t\}$.

Todistus

Osoitetaan algoritmin toimivan polynomisessa ajassa arvojen n ja $1/\varepsilon$ suhteen ja saavuttavan approksimointisuhteen $1 + \varepsilon$.

Algoritmin useimmat osat toimivat selvästi polynomisessa ajassa. Ainoa epäselvä kohta on osajoukkoparien (R_1, R_2) läpikäynti. Koska tapauksen I'_m optimiratkaisu ei ole eksakti, ovat kaikki 2^t eri osajoukkoja vastaavien alkioiden summaa erisuuria. Koska $a'_m \leq 2m/\varepsilon^2$, täytyy olla $2^t \leq \sum_{i=m-t+1}^{m} a'_i \leq 2m^2/\varepsilon^2$, koska osajoukkoja vastaavien alkioiden summat ovat kokonaislukuja. Luvun t täytyy siis tässä tapauksessa olla logaritmisen suuruinen, joten osajoukkoparit voidaan käydä läpi polynomisessa ajassa.

Olkoon m jatkossa suurin indeksi, jolla a_m on mukana optimiratkaisussa. Osoitetaan, että algoritmi löytää instanssiin I_m ratkaisun, jolla $A_1/A_2 \leq (1 + \varepsilon) \cdot \mathsf{OPT}$. Riittää tarkastella tapauksia, joissa $m > n_0$.

Jos $t = 1$ ja $j = 0$, löydetty ratkaisu on optimaalinen, koska a_m kuuluu optimiratkaisuun ja on painavampi kuin muut alkiot yhteensä. Tapauksessa $t = 1$ ja $j > 0$ taas $a'_j < m/\varepsilon$ ja siksi

$$a_j < (a'_j + 1) \cdot k(m) < \frac{2m}{\varepsilon} \cdot \frac{\varepsilon^2 \cdot a_m}{2m} = \varepsilon a_m \quad \text{eli} \quad \frac{\sum_{i \in S_1} a_i}{\sum_{i \in S_2} a_i} \leq 1 + \frac{a_j}{a_m} < 1 + \varepsilon.$$

Jos tapauksessa $t > 1$ muunnetun instanssin I'_m ratkaisu on eksakti, pätee

$$\frac{\sum_{i \in S_1} a_i}{\sum_{i \in S_2} a_i} \leq \frac{\sum_{i \in S_1} k(m) \cdot (1 + a'_i)}{\sum_{i \in S_2} k(m) \cdot a'_i} = \frac{\sum_{i \in S_1} a'_i}{\sum_{i \in S_2} a'_i} + \frac{|S_1|}{\sum_{i \in S_2} a'_i} \leq 1 + \frac{t}{m/\varepsilon} < 1 + \varepsilon.$$

Jäljellä on tapaus $t > 1$, jossa muunnetun instanssin ratkaisu ei ole eksakti. Jos parhaalla parilla (R_1, R_2) pätee $j = 0$, on sitä vastaava ratkaisu (S_1, S_2) optimaalinen parin (R_1, R_2) kanssa yhteensopivien ratkaisujen joukossa. Jos kaikilla pareilla pätee $j = 0$, löytyy näin myös optimiratkaisu.

Muuten toista osajoukoista S_1 ja S_2 vastaavien alkioiden yhteispaino on B_1 ja toisen taas välillä $[B_1 - a_j, B_1 + a_j]$. Lisäksi tiedetään, että $m \in S_2$. Näin ollen pätee

$$\frac{\sum_{i \in S_1} a_i}{\sum_{i \in S_2} a_i} \leq 1 + \frac{a_j}{\sum_{i \in S_2} a_i} \leq 1 + \frac{a_j}{a_m} < 1 + \varepsilon$$

sillä tapauksen $t = 1$ mukaisesti $a_j/a_m < \varepsilon$. □

3. Tarkastellaan rajoitettua versiota First-Fit-algoritmista laatikonpakkausongelmaan. Algoritmi käy esineet $a_1, \ldots, a_n$ läpi mielivaltaisessa järjestyksessä ja yrittää sijoittaa kunkin esineen sillä hetkellä pakattavana olevaan laatikkoon. Jos esine ei mahdu, algoritmi julistaa laatikon täydeksi ja siirtyy pakkaamaan seuraavaa laatikkoa. Osoitetaan, että tämäkin versio algoritmista saavuttaa approksimointisuhteen 2.

   Optimiratkaisulle pätee $\mathsf{OPT} \geq \sum_{i=1}^{n} a_i$. Oletetaan, että rajoitettu First-Fit käyttää yhteensä m laatikkoa, ja käydään sen pakkaamat laatikot läpi järjestyksessä. Pidetään läpikäynnin yhteydessä yllä invarianttia, että k ensimmäistä laatikkoa on pakattu keskimäärin vähintään puolilleen.

   Arvolla $k = 0$ invariantti on triviaalisti tosi. Jos laatikko $k + 1$ on vähintään puolillaan, invariantti on tosi myös arvolla $k + 1$. Jos taas laatikko $k + 1 < m$ on alle puolillaan, täytyy tämän johtua siitä ettei seuraava esine enää mahtunut laatikkoon. Näin ollen laatikoissa $k + 1$ ja $k + 2$ täytyy olla yhteensä enemmän esineitä kuin yhteen laatikkoon mahtuu, joten invariantti on tosi myös arvolla $k + 2$. Induktiolla saadaan, että ensimmäiset $m - 1$ tai m laatikkoa ovat keskimäärin vähintään puolillaan riippuen siitä, onko laatikko m pakattu alle vai vähintään puolilleen.

Jos myös viimeinen laatikko on vähintään puolillaan, pätee $m \leq 2 \cdot \sum_{i=1}^{n} a_i \leq 2 \cdot \mathsf{OPT}$. Jos taas viimeinen laatikko on alle puolillaan, pätee $m - 1 < 2 \cdot \sum_{i=1}^{n} a_i \leq 2 \cdot \mathsf{OPT}$. Koska m ja OPT ovat kokonaislukuja, seuraa myös tästä $m \leq 2 \cdot \mathsf{OPT}$. Rajoitettu First-Fit saavuttaa siis molemmissa tapauksissa approksimointisuhteen 2.

Olkoon $k > 0$ jokin kokonaisluku. Tiukka esimerkki koostuu $2k$ suuresta esineestä, joiden koko on 0,5, ja $2k$ pienestä esineestä, joiden koko on $\frac{1}{2k}$. Optimaalisessa pakkauksessa suuret esineet pakataan k laatikkoon ja pienet esineet yhteen laatikkoon, mistä seuraa $\mathsf{OPT} = k + 1$. Jos suuria ja pieniä esineitä kuitenkin tulee vuorotellen, pakkaa rajoitettu First-Fit jokaiseen laatikkoon yhden suuren ja yhden pienen esineen. Näin laatikoita tarvitaan yhteensä $2k$ kappaletta.

4. Olkoon joukkopeiteongelman perusjoukko U, osajoukkojen kokoelma $\mathcal{S}$ ja kustannusfunktio c. Ongelmaa vastaava löysennetty lineaarinen ohjelma duaaleineen on seuraava:

$$\begin{array}{lll}
\text{minimoi} & \displaystyle\sum_{S \in \mathcal{S}} c(S) x_S & \text{(primaali)} \\
\text{ehdolla} & \displaystyle\sum_{S : e \in S} x_S \geq 1 & \text{kaikilla } e \in U \\
& x_S \geq 0 & \text{kaikilla } S \in \mathcal{S}
\end{array}$$

$$\begin{array}{lll}
\text{maksimoi} & \displaystyle\sum_{e \in U} y_e & \text{(duaali)} \\
\text{ehdolla} & \displaystyle\sum_{e : e \in S} y_e \leq c(S) & \text{kaikilla } S \in \mathcal{S} \\
& y_e \geq 0 & \text{kaikilla } e \in U
\end{array}$$

Olkoot x ja y primaalin ja duaalin käypiä ratkaisuja, joilla pätee

(a) kaikilla $e \in U$: jos $y_e \neq 0$, niin $\sum_{S : e \in S} x_S = 1$;

(b) kaikilla $S \in \mathcal{S}$: jos $x_S \neq 0$, niin $\sum_{e : e \in S} y_e = c(S)$.

Tällöin näille ratkaisuille pätee

$$\sum_{S \in \mathcal{S}} c(S) x_S = \sum_{S \in \mathcal{S}} \left(\sum_{e : e \in S} y_e \right) x_S = \sum_{e \in U} \left(\sum_{S : e \in S} x_S \right) y_e = \sum_{e \in U} y_e.$$

Primaalin ja duaalin kohdefunktiot saavat siis samat arvot, kun ehdot (a) ja (b) ovat voimassa.