Approksimointialgoritmit (kevät 2010)
Harjoitus 7 (22. maaliskuuta)
Ratkaisuja (Jouni Siren)

1. Luennoilla (sivut 228–229) esitetyn mukaisesti löysennöksen ratkaisuun kuuluu kaikilla solmuilla $v \in V$ vektori $x_v = (x_v^i, x_v^j, x_v^k)$, jonka komponenttien summa on 1. Kukin verkon kaari $(u, v) \in E$ lisää optimiratkaisun painoon $c(u, v) \cdot d(u, v)$, missä $c(u, v)$ on kaaren paino ja

$$d(u, v) = \frac{1}{2}(|x_u^i - x_v^i| + |x_u^j - x_v^j| + |x_u^k - x_v^k|).$$

Luentojen perusteella (sivut 231–234) voidaan lisäksi olettaa, että vektorit x_u ja x_v joko ovat identtisiä tai eroavat toisistaan täsmälleen kahden komponentin osalta.

Huom! Jatkossa x^j viittaa vektorin x toiseen komponenttiin permutaation jälkeen ja x^2 toiseen komponenttiin ennen permutaatiota.

Tarkastellaan kaarta (u, v), jolla $x_u \neq x_v$ (muut kaaret eivät vaikuta löysennöksen optimiratkaisun eivätkä algoritmin tuottaman pyöristyksen painoon). Oletetaan, että solmut on nimetty niin, että $x_u \geq x_v$. Oletetaan lisäksi, että $x_u^1 = x_v^1$. Nyt

$$d(u, v) = |x_u^2 - x_v^2| = |x_u^3 - x_v^3|.$$

Tarkastellaan lukujen 1, 2 ja 3 permutaatioita (i, j, k) ja määritetään todennäköisyydet, joilla solmut u ja v päätyvät eri joukkoihin. Kaari (u, v) tulee tällöin osaksi leikkausta ja sen paino vaikuttaa pyöristämällä saadun ratkaisun kokonaispainoon.

(a) $(1, 2, 3)$: Jos toinen solmuista u ja v kuuluu joukkoon V_i, molemmat kuuluvat, sillä $x_u^1 = x_v^1$. Kumpikaan ei kuulu siihen todennäköisyydellä $(1 - x_u^1)$. Koska $d(u, v) = x_u^2 - x_v^2 > 0$, päätee $u \in V_j$ ja $v \in V_k$ todennäköisyydellä $(1 - x_u^1) \cdot d(u, v)/2$.

(b) $(1, 3, 2)$: Kuten edellinen tapaus.

(c) $(2, 1, 3)$: Todennäköisyydellä $d(u, v)$ pätee $u \in V_i$ ja $v \notin V_i$. Päinvastainen ei ole mahdollista, koska $x_u^2 \geq x_v^2$. Todennäköisyydellä $(1 - x_u^2)$ kumpikaan ei kuulu joukkoon V_i. Koska $d(u, v) = x_u^2 - x_v^2 > 0$ ja $x_u^1 = x_v^1$, pätee $u \in V_j$ ja $v \in V_k$ todennäköisyydellä $(1 - x_u^2) \cdot d(u, v)$. Yhteistodennäköisyys on siis $(2 - x_u^2) \cdot d(u, v)$.

(d) $(3, 1, 2)$: Kuten edellinen tapaus, mutta todennäköisyys on $(2 - x_u^3) \cdot d(u, v)$.

(e) $(2, 3, 1)$: Nyt tiedetään, että $x_u^2 > x_v^2$ ja $x_u^3 < x_v^3$. Solmut u ja v saadaan siis eri joukkoihin, jos $u \in V_i$ ja $v \notin V_i$ tai $u \in V_j$ ja $v \in V_k$. Edellisen todennäköisyys on $d(u, v)$ ja jälkimmäisen $(1 - x_u^2) \cdot d(u, v)/2$, joten yhteistodennäköisyys on $(3 - x_u^2) \cdot d(u, v)/2$.

(f) $(3, 2, 1)$: Kuten edellinen tapaus, mutta suuruuserot ovat päinvastaiset ja yhteistodennäköisyys $(3 - x_u^3) \cdot d(u, v)/2$.

Jokainen permutaatioista on yhtä todennäköinen. Todennäköisyys sille, että solmut u ja v päätyvät eri joukkoihin, on siis

$$\frac{d(u, v)}{6} \cdot \left(8 - \frac{2x_u^1 + 3x_u^2 + 3x_u^3}{2}\right) = \frac{7}{6} \cdot d(u, v) - \frac{x_u^2 + x_u^3}{12} \cdot d(u, v) \leq \frac{7}{6} \cdot d(u, v),$$

sillä $x_u^1 + x_u^2 + x_u^3 = 1$. Kaaren (u, v) vaikutus pyöristyksen kokonaispainoon on siis odotusarvoisesti korkeintaan $7/6 \cdot c(u, v) \cdot d(u, v)$, joten leikkauksen C kustannukseksi tulee $c(C) \leq 7/6 \cdot \mathsf{OPT}_f \leq 7/6 \cdot \mathsf{OPT}$.

On kuitenkin syytä huomata, ettei algoritmi välttämättä tuota sellaisenaan käypää ratkaisua. Päätesolmuista s_i päätyy kyllä joukkoon V_i ja s_k joukkoon V_k. Sen sijaan s_j päätyy joukkoon V_j vain todennäköisyydellä 1/2; muussa tapauksessa se päätyy joukkoon V_k.

Ongelma voidaan korjata vaihtamalla joukon V_j valintakriteeriksi $x_v^i/2 + x_v^j \geq \rho_2$, jolloin jokainen päätesolmu valitaan oikeaan joukkoon. Tämä vaikuttaa todennäköisyyksiin seuraavasti.

(a) $(1 - x_u^1) \cdot d(u, v)$
(b) $(1 - x_u^1) \cdot d(u, v)$
(c) $(3 - x_u^2) \cdot d(u, v)/2$
(d) $(3 - x_u^3) \cdot d(u, v)/2$
(e) $(3 - x_u^2) \cdot d(u, v)/2$
(f) $(3 - x_u^3) \cdot d(u, v)/2$

Yhteistodennäköisyydeksi tulee nyt

$$\frac{d(u,v)}{6} \cdot \left(8 - \frac{4x_u^1 + 2x_u^2 + 2x_u^3}{2}\right) = \frac{7}{6} \cdot d(u,v) - \frac{x_u^1}{6} \cdot d(u,v) \leq \frac{7}{6} \cdot d(u,v).$$

2. Luennoilla (sivu 224) esitetty algoritmi monensuuntaiseen leikkaukseen yleistyy luonnollisella tavalla monensuuntaiseen solmuleikkaukseen: etsitään jokaiselle päätesolmulle pienin eristävä solmujoukko, pudotetaan painavin näistä pois ja valitaan leikkaukseksi yhdiste muista eristävistä joukoista.

   Tarkastellaan k-sakaraista tähtiverkkoa ja valitaan sakaroiden kärjissä olevat solmut s_i päätesolmuiksi. Asetetaan keskimmäisen solmun u painoksi $c(u) = 1 + \varepsilon$ jollain $\varepsilon > 0$. Lisätään jokaista päätesolmua s_i kohti uusi solmu v_i, jonka painoksi tulee $c(v_i) = 1$. Korvataan kukin kaari (s_i, u) kaarilla (s_i, v_i) ja (v_i, u).

   Algoritmi löytää jokaiselle päätesolmulle s_i pienimmän eristävän solmujoukon $\{v_i\}$ ja valitsee leikkaukseen $k - 1$ tällaista joukkoa. Leikkauksen kokonaispainoksi tulee siis $k - 1$. Toisaalta keskisolmu u muodostaa yksin leikkauksen, jonka paino on $1 + \varepsilon$, joten algoritmin approksimointisuhde ei voi olla parempi kuin $k - 1$.

   Esimerkki on tiukka. Olkoon päätesolmua s_i vastaava eristävä joukko C_i ja sen paino $c(C_i)$. Koska jokainen monensuuntainen solmuleikkaus on myös solmun s_i eristävä joukko, pätee $c(C_i) \leq \mathsf{OPT}$. Koska leikkaukseen C valitaan $k - 1$ eristävää joukkoa, pätee sille $c(C) \leq (k - 1) \cdot \mathsf{OPT}$.

3. Tarkastellaan monensuuntaista solmuleikkausta vastaavaa kokonaislukuohjelmaa (luentojen sivu 246) ja sen löysennöksen duaalia (luentojen sivu 248) edellisen tehtävän ratkaisun mukaisessa verkossa. Asetetaan kuitenkin keskisolmun u painoksi (kapasiteetiksi) $c(u) = k$.

   Verkon maksimivuo saadaan esimerkiksi viemällä puolen yksikön verran vuota solmusta s_i solmuun s_{i+1} kaikilla $i < k$ sekä solmusta s_k solmuun s_1. Tällainen vuo on maksimivuo, sillä se kyllästää kaikki verkon ei-päätesolmut paitsi keskisolmun. Kokonaisvuoksi tulee $k/2$.

   Kaikilla $i \neq j$ verkossa on polku $s_i \to v_i \to u \to v_j \to s_j$, joten mihin tahansa monensuuntaiseen solmuleikkaukseen täytyy kuulua keskisolmu u tai $k - 1$ kappaletta solmuista v_i. Leikkauksen kokonaispainoksi tulee siis vähintään $k - 1$, mikä on $2 - 2/k$ kertaa enemmän kuin verkon maksimivuo.

4. (a) Muodostetaan approksimointisuhteen säilyttävä palautus monensuuntaisesta leikkauksesta osajoukkotakaisinkytkentäongelmaan kaarille. Palautus tapahtuu lisäämällä verkkoon uusi solmu s, josta tulee verkon ainoa kiinnostava solmu. Lisäksi verkkoon tulee kaikilla i kaari (s, s_i), jonka painoksi tulee ∞.

   Olkoot $s_i \neq s_j$ kaksi päätesolmua. Jokaista alkuperäisen verkon polkua $s_i \rightsquigarrow s_j$ vastaa uuden verkon kiinnostava sykli $s_i \rightsquigarrow s_j \to s \to s_i$, ja päinvastoin. Jokainen alkuperäisen verkon monensuuntainen leikkaus on siis käypä ratkaisu osajoukkotakaisinkytkentäongelmaan uudessa verkossa. Vastaavasti jokainen painoltaan äärellinen osajoukkotakaisinkytkentäongelman käypä ratkaisu on monensuuntainen leikkaus alkuperäisessä verkossa. Näin ollen palautus säilyttää approksimointisuhteen.

   (b) Muodostetaan approksimointisuhteen säilyttävä palautus osajoukkotakaisinkytkentäongelmasta kaarille (A) osajoukkotakaisinkytkentäongelmaan solmuille (B). Astetaan kaikkien alkuperäisen verkon kaikkien ei-kiinnostavien solmujen painoksi ∞. Korvataan alkuperäisen verkon jokainen kaari (u, v) kaarilla $(u, w_{u,v})$ ja $(w_{u,v}, v)$, missä $w_{u,w}$ on uusi solmu, johon ei liity muita kaaria. Asetetaan tämän solmun painoksi uudessa verkossa kaaren (u, v) paino alkuperäisessä verkossa.

Nyt jokainen ongelman A käypä ratkaisu alkuperäisessä verkossa voidaan muuttaa kokonaispainoltaan samaksi ongelman B ratkaisuksi uudessa verkossa. Tämä tapahtuu valitsemalla ongelman B ratkaisuun solmu $w_{u,v}$, jos kaari (u, v) kuuluu ongelman A ratkaisuun. Sama toimii myös toiseen suuntaan, jos ongelman B ratkaisu on painoltaan äärellinen. Näin ollen kysymyksessä on approksimointisuhteen säilyttävä palautus.