Approksimointialgoritmit (kevät 2010)
Harjoitus 2 (1. helmikuuta)
Ratkaisuja (Jouni Siren)

1. Olkoon verkon $G = (V, E)$ solmujen joukko $V = \{v_1, v_2, \ldots, v_n\}$. Oletetaan, että algoritmi käsittelee solmut numeroinnin mukaisessa järjestyksessä. Sanotaan, että solmu v_j omistaa kaaren (i, j), jos $i < j$.

   Algoritmin kohdassa (b) käsiteltävä solmu v lisätään joukkoon A tai B sen mukaan, kummalla tavalla suurempi osa solmun v omistamista kaarista tulee mukaan leikkaukseen. Näin ollen leikkaukseen tulee mukaan aina vähintään puolet solmun v omistamista kaarista. Koska jokainen kaari on jonkin solmun omistama, on algoritmin tuottamassa leikkauksessa aina vähintään puolet verkon kaikista kaarista. Tehtävässä esitetty algoritmi on siis 1/2-approksimointialgoritmi lukumääräiseen maksimileikkausongelmaan.

   Ylärajana maksimileikkauksen koolle käytettiin verkon kaikkien kaarten määrää. Etsityn tiukan esimerkin täytyy siis olla kaksijakoinen verkko, jonka maksimileikkauksessa ovat mukana kaikki kaaret. Esimerkiksi kelpaa verkko, jossa on kaaret (v_i, v_1) ja (v_i, v_2) kaikilla $i \geq 3$. Algoritmi valitsee aluksi solmut v_1 ja v_2 leikkauksen eri puolille, minkä jälkeen jokaisella myöhemmistä solmuista v_i vain toinen solmun omistamista kaarista tulee mukaan leikkaukseen.

   Ongelman ja algoritmin voi yleistää painotetuille verkoille korvaamalla joukkojen X ja Y välisten kaarten määrän $d(X, Y)$ niiden välisten kaarten yhteispainolla $c(X, Y)$.

2. Jos solmun v vaihto joukosta A joukkoon B ei kasvata leikkauksen kokoa, on solmulla v vähintään yhtä paljon naapureita joukossa B kuin joukossa A. Jos minkään solmun vaihto ei kasvata leikkauksen kokoa, kuuluu leikkaukseen vähintään puolet verkon kaikista kaarista. Algoritmi takaa siis approksimointisuhteen 1/2 lukumääräiseen maksimileikkausongelmaan.

   Kohta (a) voidaan suorittaa ajassa $O(|V| + |E|)$. Solmun vaihdon vaikutuksen selvittäminen vie aikaa suhteessa solmun naapureiden määrään, joten kohta (b) onnistuu myös ajassa $O(|V| + |E|)$. Kohta (c) suoritetaan tämän jälkeen ajassa $O(1)$. Kohtaan (b) palataan takaisin vain, jos leikkauksen koko $d(A, B)$ on kasvanut. Koska $0 \leq d(A, B) \leq |V| \cdot (|V| - 1)/2$ ja leikkauksen koko on kokonaisluku, suoritetaan kohta (b) $O(|V|^2)$ kertaa. Algoritmi toimii siis ajassa $O(|V|^2 \cdot (|V| + |E|))$.

3. Muodostetaan tiukka esimerkki siitä, ettei ahneelle algoritmille voi osoittaa parempaa approksimointisuhdetta lukumääräisessä joukkopeiteongelmassa kuin H_k, missä k on suurimman osajoukon koko. (Tehtävänannossa sanotaan virheellisesti, että H_k olisi suurimman osajoukon koko.)

   (a) Olkoon k kokonaisluku. Esimerkki koostuu alkioista (i, j), missä $1 \leq i \leq k!$ ja $i \leq j \leq k$. Pystysuora osajoukko V_a sisältää alkiot (a, j) kaikilla j, vaakasuora osajoukko $H_{b,c}$ taas alkiot (i, b), missä $(c-1)b < i \leq cb$. Pystysuoria osajoukkoja on yhteensä $k!$ kappaletta, vaakasuoria taas rivillä b yhteensä $k!/b$ kappaletta.

   Optimipeite koostuu kaikista pystysuorista osajoukoista, joten sen koko on $k!$. Ahne algoritmi voi kuitenkin valita ensin kaikki vaakasuorat osajoukot riviltä k, sitten riviltä $k - 1$, sitten riviltä $k - 2$ jne, jolloin se tulee valinneeksi yhteensä

   $$\sum_{i=1}^{k} \frac{k!}{i} = H_k \cdot k!$$

   osajoukkoa. Approksimointikertoimeksi tulee siis H_k. Pahin tapaus on kuitenkin epätodennäköinen, sillä ahne algoritmi voi valita missä tahansa vaiheessa myös jonkin pystysuoran osajoukon.

   (b) Olkoon $n \geq 2$ kokonaisluku. Esimerkki koostuu alkioista (i, j), missä $1 \leq i < 2^{n+1}$ ja $j \in \{0, 1\}$. Osajoukko T_0 sisältää alkiot $(i, 0)$, osajoukko T_1 alkiot $(i, 1)$ ja osajoukko S_m alkiot $\{(i, j) \mid 2^m \leq i < 2^{m+1}\}$, missä $1 \leq m \leq n$. Tapaus $n = 3$ on esitetty Wikipediassa[1] kuvana.

[1] `http://en.wikipedia.org/wiki/Set_cover_problem`

Ahne algoritmi valitsee joukot $S_n, S_{n-1}, \ldots, S_1$, sillä kullakin hetkellä suurimmassa vielä valitsemattomassa joukossa S_m on jäljellä 2^m alkiota, kun taas osajoukoissa T_0 ja T_1 on kummassakin jäljellä $2^m - 1$ alkiota. Joukkoja valitaan siis yhteensä n, vaikka osajoukkojen T_0 ja T_1 valitseminen riittäisi. Approksimointisuhteeksi tulee siis

$$\frac{n}{2} = \frac{\log_2 |S_n|}{2} = \frac{\ln 2 \cdot \ln |S_n|}{2} \approx 0{,}72 \cdot H_{|S_n|},$$

mikä on vakiokertoimen 0,72 päässä halutusta.

4. Merkitään perusjoukkoa U ja sen osajoukkojen kokoelmaa $\mathcal{S}$. Alkion x frekvenssi $f(x)$ on niiden osajoukkojen $S \in \mathcal{S}$ määrä, joilla $x \in S$. Painofunktio $w : \mathcal{S} \to \mathbb{Q}_+$ on *koon mukainen* (solmupeiteongelmassa asteen mukainen), jos $w(S) = c \cdot |S|$, missä c on vakio, kaikilla $S \in \mathcal{S}$.

   Etsitty algoritmi on seuraava (vertaa luentojen sivuun 46):

   (a) Alusta $i \leftarrow 0$, $U_0 \leftarrow U$, $\mathcal{S}_0 \leftarrow \mathcal{S}$ ja $w_0 \leftarrow w$.

   (b) Olkoon D_i niiden osajoukkojen S kokoelma, joilla $S \cap U_i = \emptyset$. Aseta $\mathcal{S}_i \leftarrow \mathcal{S}_i \setminus D_i$.

   (c) Jos $\mathcal{S}_i = \emptyset$, palauta joukkopeite $C = W_0 \cup \ldots \cup W_{i-1}$.

   (d) Laske $c_i \leftarrow \min\limits_{S \in \mathcal{S}_i} \dfrac{w_i(S)}{|S \cap U_i|}$ ja aseta $t_i(S) = c_i \cdot |S \cap U_i|$.

   (e) Olkoon W_i niiden osajoukkojen kokoelma, joilla $w_i(s) = t_i(S)$.

   (f) Aseta $U_{i+1} \leftarrow U_i \setminus \bigcup W_i$, $\mathcal{S}_{i+1} \leftarrow \mathcal{S}_i \setminus W_i$ ja $w_{i+1}(S) = w_i(S) - t_i(S)$.

   (g) Aseta $i \leftarrow i + 1$ ja palaa kohtaan (b).

   Kokoelma W_i peittää joukon $U_i \setminus U_{i+1}$ ja kokoelma S_i joukon U_i kaikilla i. Induktiolla voidaan siis osoittaa, että kokoelma C on perusjoukon U joukkopeite. Seuraava lemma yleistää luentojen sivun 49 lemman joukkopeitteelle.

   Lemma. Jos w on koon mukainen, niin $w(\mathcal{S}) \leq f_{\max} \cdot \mathsf{OPT}$, missä $f_{\max} = \max\limits_{x \in U} f(x)$.

   Todistus. Olkoon C_* optimipeite. Tällöin

   $$w(C_*) = c \cdot \sum_{S \in C_*} |S| \geq c \cdot |U| \quad \text{ja} \quad w(\mathcal{S}) = c \cdot \sum_{x \in \mathcal{S}} f(x) \leq c \cdot f_{\max} \cdot |U|.$$

   Näin ollen $w(\mathcal{S}) \leq f_{\max} \cdot \mathsf{OPT}$. □

   Jos $S \in C$, niin $S \in W_j$ jollain j ja $w(S) = \sum_{i \leq j} t_i(S)$. Muuten $S \in D_j$ jollain j ja $w(S) \geq \sum_{i<j} t_i(S)$. Jos C_* on optimipeite, niin $C_* \cap \mathcal{S}_i$ on joukon U_i joukkopeite kaikilla i. Käyttämällä funktion t_i mukaisia painoja seuraa edellä olevasta lemmasta

   $$t_i(C \cap \mathcal{S}_i) \leq t_i(\mathcal{S}_i) \leq \max_{x \in U_i} f_i(x) \cdot t_i(C_* \cap \mathcal{S}_i),$$

   missä $f_i(x)$ on alkion x frekvenssi kokoelmassa $\mathcal{S}_i$. Koska $f_i(x) \leq f_{\max}$ kaikilla i ja x, saadaan

   $$w(C) = \sum_{i=0}^{k-1} t_i(C \cap \mathcal{S}_i) \leq f_{\max} \cdot \sum_{i=0}^{k-1} t_i(C_* \cap \mathcal{S}_i) \leq f_{\max} \cdot w(C_*)$$

   jollain $k > 0$. Niinpä yllä esitetty kerrostamiseen perustuva algoritmi on $f_{\max}$-approksimointialgoritmi joukkopeiteongelmaan.

   Tiukka esimerkki saadaan yleistämällä solmupeiteongelman täydelliseen kaksijakoiseen verkkoon perustuva esimerkki. Olkoon $r \geq 2$ vakio ja $V_1, \ldots, V_r$ joukkoja, joista jokaiseen kuuluu $n > 0$ alkiota. Valitaan perusjoukoksi $V_1 \times \ldots \times V_r$ ja määritetään kaikilla i ja x osajoukot $S_{i,x}$ siten, että $(v_1, \ldots, v_r) \in S_{i,x}$, jos $v_i = x$. Asetetaan jokaisen osajoukon painoksi $|S_{i,x}| = n^{r-1}$. Algoritmi muodostaa heti ensimmäisessä vaiheessa joukkopeitteen C, johon kuuluvat kaikki osajoukot. Peitteen kokonaispainoksi tulee $w(C) = r \cdot n^r$, vaikka valitsemalla vain osajoukot $S_{1,x}$ saataisiin joukkopeite, jonka paino on n^r.