Algoritmekonstruksjon: Øving 1

Simen Keiland Fondevik Approksimasjonsalgoritmer

29. januar 2018

1 Oppgave 1

Vis at det for en konstant c ***ikke finnes noen*** $c \log |T|$***-approksimasjonsalgoritme for det rettede Steinertre-problemet, gitt at*** $P \neq NP$***.***

Bevis. Teorem 1.14 i Williamson og Shmoys sier at det finnes en konstant c slik at dersom det finnes en $c \log n$-approksimasjonsalgoritme for det uvektede mengdedekkeproblemet, må $P = NP$. Vi trenger altså bare redusere fra det uvektede mengdedekkeproblemet til det rettede Steinertre-problemet. Gitt en instans (S, E) av mengdedekkeproblemet kan dette gjøres ved å konstruere en stjernegraf med rotnoden r i senter. Ut fra rotnoden legges til nodene $s_i,\ i = 1,\ 2,\ \cdots\ |S|$ med kantvekt 1. Disse svarer til mengden $S_i \in S$. Alle $n = |E|$ elementer $e_j \in E$ kan videre legges til som barn av noden s_i med kantvekt null dersom $e_j \in s_i$. Dette er terminalene som skal dekkes. Vi har nå gjort om instansen av nodedekkeproblemet til et tilfelle av Steinertre-problemet med en polynomtid reduksjon. Altså er Steinertre-problemet minst like vanskelig som mengdedekkeproblemet, hvilket betyr at gitt en $f(|T|)$-approksimasjonsalgoritme for Steinertreproblemet må det det finnes en approksimasjonsalgoritme for mengdedekkeproblemet med ytelsesgaranti $f(n)$. Siden $f(n)$ i henhold til teorem 1.14 ikke kan bli bedre enn $c \log n$, finnes det altså ingen approksimasjonalgoritme for Steinertre-problemet bedre enn $c \log |T|$ for en konstant $c > 0$. □

2 Oppgave 2

Vis at det for en konstant c ***ikke finnes noen*** $c \log |D|$***-approksimasjonsalgoritme for det kapasitetsubegrensede fasilitetslokasjon-problemet, gitt at*** $P \neq NP$***.***

Bevis. Liknende situasjon som i oppgave 1. Gitt en instans (S, E) av mengdedekkeproblemet, lag en graf med hver mengde S_i som en node s_i ut fra rotnoden med kantvekt 1. Legg så til de $|E|$ elementene som skal dekkes som noder $e_1,\ e_2, \cdots\ e_{|E|}$. Kanten $(s_i,\ e_j)$ tilordens en vekt lik 0 hvis $e_j \in S_i$, ellers er den uendelig stor (eller minst $|S| \cdot c \log D$). Vi har nå fått en instans av det kapasitetsubegrensede fasilitetslokasjon-problemet gjennom en polynomtid reduksjon der $F = \{s_i\ :\ i = 1,\ 2,\ \cdots\ |S|\}$ og $D = \{e_j\ :\ j = 1,\ 2,\ \cdots\ |E|\}$. Helt konkret; gitt en løsning på dette problemet, finnes en minst like god løsning på mengdedekkeproplemet. Med samme argumentasjon som i oppgave 1 finnes det dermed ingen approksimasjonsalgoritme for det

kapasitetsubegrensede fasilitetslokasjon-problemet med bedre ytelsesgaranti enn $c \log |D|$ for en konstant $c > 0$. □

Finn en $O(\log |D|)$-approksimasjonsalgoritme for det kapasitetsubegrensede fasilitetslokasjon-problemet.

Løsning. La D være mengden klienter, F mendgen fasiliteter, c_{ij} kostanden å forbinde fasilitet F_i med klient j og f_i kostanden for å åpne fasilitet F_i. En grådig $O(\log |D|)$-algoritme er kjent for mengdedekkeproblemet. Ønsker derfor å transformere problemet til et mengdedekkeproblem, løse det med den grådige algoritmen, og til slutt mappe svaret tilbake. I mengdedekkepoblemet dekker hver mengde et spesifikt utvalg elementer. I fasilitetslokasjonproblemet kan imidlertid enhver fasilitet knyttes opp mot et vilkårlig utvalg klienter. Erstatt derfor hver fasilitet med nye mengder tilsvarende dets power-set. Det vil si; for en fasilitet $i \in F$ lages nye mengder $S_i^k, i \in F, k = 1, 2, \cdots 2^{|D|}$. La så kostnaden for disse mengdene være $w(S_i^k) = f_i + \sum_{j \in S_i^k} c_{ij}$. Vi har nå dessverre eksponentielt mange mengder, slik at når vi i den grådige algoritmen skal finne den mengden som er billigst å legge til relativt antall nye elementer den vil dekke, vil et lineærsøk ikke kunne gjøres i polynimisk tid. En kan imidlertid observere at når en vurderer å ta med en mengde S_i^k som, si, dekker j elementer, trenger en ikke teste alle mengder som dekker j elementer; det holder åpenbart å kun prøve dekke de j billigste. Dermed vil en kun trenge å vurdere et polynomisk antall mengder, og den grådige algoritmen kjører i polynomsik tid. Mappingen tilbake er nå triviell. □

3 Oppgave 3

Finn en 3-approksimasjonsalgoritme for k-suppliers-problemet.

Løsning. Vi kjenner en grådig algoritme for k-senterproblemt som hele tiden velger som senter det punktet som ligger lengst unna alle tidligere valgte sentre. Denne algoriten kan enkelt vises å være en 2-approksimasjonsalgoritme: La x være et senter i en opt løsning med verdi r^*. Da er er alle andre punkter (som approksimasjonsalgoritmen kan velge) i denne klyngen maksimalt en avstand $2r^*$ unna hverandre (diameteren til en sirkel rundt x). Skulle dessuten algoritmen finne på å velge to punkter fra samme klynge som to av sentrene, vil disse sentrene nødvendigvis også ligge maksimalt $2r^*$ unna hverandre. Videre betyr dette at alle andre punkter også er innenfor denne avstanden til sitt nærmeste senter, for eller ville algoritmen valgt et annet senter. Vi kan nå bygge videre på denne ideen for k-suppliersproblemet: Velg hele tiden det senteret/supplieren som ligger nærmest den kunden som ved denne iterasjonen ligger lengst unna et senter/supplier. Det vil si; finn den kunden som har det "værst", og åpne vedkommendes nærmeste supplier;

$$\underset{i \in F - S}{\operatorname{argmin}} \; d\left(i, \underset{j \in D}{\operatorname{argmax}} d(j, S)\right),$$

der $F \subseteq V$ er mengden suppliers, $D = V - F$ er mengden customers og $S \subseteq F$ er de (hittil) valgte supplierne. Gitt en optimal løsning; Velger algoritmen en supplier utenfor radien til den optimalt valgte, kan ikke denne supplieren være mer enn r^* unna periferien til sirklen rundt det optimale punktet, for ellers ville algoritmen valgt den faktisk optimale supplieren

i stedet. Siden valgt supplier er maksimalt r^* unna sirkelen rundt det optimale punktet, og to punkter inne i denne sirkelen maksimalt er $2r^*$ fra hverandre, vil maksimal avstand (i henhold til trekantulikheten) mellom en supplier og kunde med denne algoritmen være $3r^*$. Velges to suppliers innenfor samme sirkel vet vi, tilsvarende over, at ingen andre kunder heller er mer enn $3r^*$ unna en supplier. □

Vis at det ikke finnes en α-approksimasjonsalgoritme med $\alpha < 3$ for k-suppliers-problemet, gitt at $P \neq NP$.

Løsning. For k-senter kan vi ikke approksimere bedre enn $\alpha = 2$. Dette vises ved å redusere fra dominerende mengde-problemet. I dominerende mengde-problemet har vi en graf der spørsmålet er om vi kan velge ut k noder slik at alle noder er enten valgt som en node eller er nabo til en valgt node. Dette er et NP-komplett problem. Reduksjon fra dominerende sett til k-senter kan gjøres som følgende: La hver node i dominerende mengde-grafen være punkter i en instans av k-senterproblemet. Tilordn kantvekt 1 hvis to noder er naboer i grafen, kantvekt 2 (for å opprettholde trekantulikheten) hvis de ikke er naboer (legger altså til nye kanter fordi k-senter kan modelleres som en komplett graf). Hvis vi klarer approksimere dette k-senterproblemet bedre enn $\alpha = 2$, det vil si alle punkter er mindre enn en avstand 2 unna sitt nærmeste senter, betyr det at vi også klarer løse dominerende mengde-problemt! Altså; kan ikke approksimere k-senterproblemet bedre enn $\alpha = 2$, for da kan vi også løse noe som er NP-komplett i polynomtid.

Overfører så dette resonementet til k-suppliers-problemet: Plasser nodene i grafen til dominerende mengde-problemet på en vannrett rekke. Lag så en kopi av disse og tegn dem opp vannrett nedenfor. Dette er fasilitetene, og vi har nå tegnet nodene i det som skal være en komplett graf. La avstanden fra en node j til en node j' være $d(j, j') = 2,\ j, j' \in D$, $d(i, i') = 2,\ i, i' \in F$ og $d(i, j) = \{1 \text{ hvis } (i, j) \in E,\ 3 \text{ ellers}\}$. Vi ønsker i k-suppliers-problemet å velge noen av nodene i nedre rad (supplierne) slik at alle oppe (kundene) er dekket. Klarer vi å velge slik at alle kundene er maksimalt en avstand 2 unna en supplier, har vi plutselig løst dominerende mengde-problemet i polynomtid. Altså; for å kunne finne en løsning bedre enn 3, kan vi ikke bruke noen 3-er kanter. Hvis vi får lov å velge 3-ere kanter går det greit, men å unngå dem vil si vi løser dominerende mengde-problemet. Vi har altså skapt en instans av k-supplier der vi ikke kan garantere noe bedre enn 3. □

4 Oppgave 4

Formuler ryggsekk-problemet som et heltallprogram.

Løsning. La w_i og v_i være henholdsvis vekten og verdien til element i, og la W være kapasiteten til ryggsekken. La videre $x_i \in \{0, 1\}$ betegne om element i tas med eller ikke. Vi ønsker å maksimere $\sum x_i v_i$ med hensyn til $\sum x_i w_i \leq W$, altså

$$\max \sum_i x_i v_i \qquad \text{subject to} \qquad \sum_i x_i w_i \leq W, \qquad x_i \in \{0, 1\}, \quad v_i, w_i, W \in \mathbb{R}.$$

Om vi i stedet lar $x \in \mathbb{R}$ blir programmet et lineærprogram og kan enkelt løses med f.eks. Simplex-algoritmen. □

Formuler maks-flyt-problemet som et lineærprogram.

Løsning. Gitt en rettet graf $G = (E, V)$, la s være kilden, t være sluket, f_{uv} betegne flyten fra node u til node v og la c_{uv} betegne kapasiteten fra node u til node v, $(u, v) \in E$. Vi ønsker maksimere flyten ut av kilden (som er lik flyten inn til sluket). Vi krever at en node har like stor flyt inn som ut, det vil si den kan ikke akkumulere flyt, og vi krever at flyten aldri overskrider kapasiteten mellom nodene. Lineærprogrammet blir dermed som følger.

$$\max \sum_{u \in \mathrm{Adj}(s)} f_{su} \qquad \text{subject to} \qquad \begin{array}{ll} 0 \leq f_{uv} \leq c_{uv} & \forall (u, v) \in E \\ \sum_{v \in \mathrm{Adj}(u)} f_{vu} - f_{uv} = 0 & \forall u \in V \end{array}$$

□