Linköpings Universitet
Institutionen för datavetenskap
Peter Jonsson

TDDA32 Konstruktion och analys av algoritmer

Exempeltenta med facit

Uppgift 1.
Låt $G = (V, E)$ vara en godtycklig riktad graf. Konstruera en polynomisk algoritm som beräknar en mängd $E' \subseteq E$ med följande egenskaper:

1. (V, E') är acyklisk; och
2. $|E'| \geq |E|/4$.

Algoritmen får gärna vara randomiserad och då ändras krav 2 till följande:

2' den *förväntade* storleken på E' skall vara $\geq |E|/4$.

Uppgift 2.
Ett polynom $p(x_1, \ldots, x_n)$ är (som vanligt) en summa av termer och varje term är en produkt $c \cdot x_{i_1}^{j_i} \cdot \ldots \cdot \ldots x_{i_m}^{j_m}$ där (1) c är en rationell konstant; (2) $1 \leq i_m \leq n$ för alla m; och (3) j_m är ett naturligt tal för alla m. Visa att följande problem är NP-svårt.

INSTANS: En mängd polynom $\{p_1(x_1, \ldots, x_n), \ldots, p_k(x_1, \ldots, x_n)\}$.
FRÅGA: Kan $x_1, \ldots, x_n$ tilldelas reella värden sådana att $p_1(x_1, \ldots, x_n) \geq 0, \ldots, p_k(x_1, \ldots, x_n) \geq 0$?

Notera att man på goda grunder misstänker att detta problem inte tillhör NP.

Uppgift 3.
En *hamiltonväg* i en oriktad graf $G = \langle V, E \rangle$ är en väg som passerar varje nod *exakt* en gång. Att avgöra om en godtycklig graf innehåller en hamiltonväg är (inte helt överraskande) ett NP-fullständigt problem.

Vi säger att en graf G är av *grad 2* om och endast om varje nod har högst två grannar. Konstruera en polynomisk algoritm som avgör om det finns en hamiltonväg i en given graf av grad 2.

Uppgift 4.
Antag att A är en algoritm som korrekt löser följande problem i $O(1)$ tid[1]: Givet en mängd heltal S och ett heltal k så svarar algoritmen "ja" om och endast om det finns en delmängd $S' \subseteq S$ sådan att summan av talen i S' är lika med k.

Använd A för att konstruera en algoritm B som inte bara svarar "ja" eller "nej" på föregående problem, utan dessutom returnerar delmängden S' om den existerar. Algoritmen B får anropa A högst $O(n)$ gånger, där n är antalet tal i S. Vidare får den totala tidskomplexiteten för B högst vara $O(n)$.

Uppgift 5.
Låt $L = l_1, l_2, \ldots, l_n$ vara en sekvens innehållande n distinkta positiva heltal (d.v.s. alla talen i L är olika). Konstruera en algoritm som hittar den *längsta ökande delsekvensen* i L. En sådan sekvens är ett urval av element $l_{i_1}, l_{i_2}, \ldots, l_{i_k}$ ur L med följande två egenskaper:

- $i_1 < i_2 < \ldots < i_k$
- $l_{i_1} < l_{i_2} \ldots l_{i_k}$.

Betrakta exempelvis sekvensen 11, 17, 5, 8, 6, 4, 7, 12, 3. Den längsta ökande delsekvensen är då 5, 6, 7, 12. Algoritmen får högst ha tidskomplexitet $O(n^3)$.

[1]En sådan algoritm existerar naturligtvis inte. Den här typen av hypotetiska algoritmer brukar man kalla "orakel" och de spelar en av huvudrollerna i komplexitetsteori.

Facit

Uppgift 1
Antag att grafens noder är $v_1, \ldots, v_n$. Tilldela varje nod talet 0 med sannolikhet 50% och talet 1 med sannolikhet 50%. Bilda en ny graf G' där endast de bågar $v \to w$ med $v = 0$ och $w = 1$ finns kvar. Denna graf kan konstrueras i polynomisk tid. Frågan är nu hur många bågar som finns kvar i G'.

Antag att bågarna i G är $e_1, \ldots, e_m$. Låt den stokastiska variabeln X_i vara 1 om bågen e_i finns med i G'. Då är det förväntade antalet bågar A i G' lika med $\mathcal{E}[X_1 + \ldots + X_m] = /\text{linearitet}/ = \mathcal{E}[X_1] + \ldots + \mathcal{E}[X_m]$. Eftersom $\mathcal{E}[X_i] = 1/4$ följer det att $\mathcal{E}(A) = m/4$.

Uppgift 2
Polynomisk reduktion från 3SAT. Låt $F = C_1 \wedge \ldots \wedge C_k$ vara en godtycklig formel över variabelmängden $V = \{v_1, \ldots, v_n\}$. Notera först att de två olikheterna $x^2 - x \geq 0, x - x^2 \geq 0$ endast uppfylls då $x \in \{0, 1\}$. Vi börjar därför med att konstruera följande mängd av olikheter

$$\{v_i^2 - v_i \geq 0, v_i - v_i^2 \geq 0 \mid v \in V\}$$

som tvingar varje variabel att vara 0 eller 1 (och vi tolkar 0 som "falsk" och 1 som "sann"). Betrakta nu en godtycklig klausul, t.ex. $(v_1 \vee \neg v_2 \vee \neg v_3)$. Denna motsvaras av olikheten

$$v_1 + (1 - v_2) + (1 - v_3) \geq 1 \Leftrightarrow v_1 - v_2 - v_3 \geq -1 \Leftrightarrow v_1 - v_2 - v_3 + 1 \geq 0$$

Sålunda kan vi omforma varje klausul i polynomisk tid till en ekvivalent olikhet.

Uppgift 3
Notera först att om G inte är sammanhängande kan G inte innehålla en Hamiltonväg. Antag att grafen är sammanhängande. Eftersom grafens grad ≤ 2 är G antingen en enkel väg mellan två noder eller en enkel cykel och i båda dessa fall innehåller G en Hamiltonväg. Detta medför att grafen innehåller en hamiltonväg om och endast om den är sammanhängande. Att kontrollera om en graf är sammanhängande är ett polynomiskt problem (se t.ex. kursboken s. 500).

Uppgift 4.

Algoritm:

$R := \emptyset$
Repetera till $S = \emptyset$ eller $k = 0$
  Välj ett godtyckligt element $s \in S$
  Om $A(S - \{s\}, k) =$ "nej" så låt $S := S - \{s\}$, $R := R \cup \{s\}$, $k := k - s$
    annars så låt $S := S - \{s\}$
Om $k = 0$ så svara "ja" och returera R, annars svara "nej"

Algoritmens loop kan inte upprepas fler än $|S|$ gånger så oraklet A anropas maximalt $|S|$ gånger.

Uppgift 5
Låt $L = l_1, l_2, \ldots, l_n$ vara talsekvensen. Bilda en riktad graf $G = (V, A)$ där $G = \{v_1, \ldots, v_n\}$ och $v_i \to v_j \in A$ om och endast om $l_i < l_j$ och $i < j$. Denna graf är acyklisk eftersom inget tal a uppfyller $a < a$. Vidare motsvarar längsta vägen i denna graf den längsta ökande delsekvensen i L. Ge varje båge i G vikten -1. Notera nu att en väg av minimal vikt är en längsta väg i G. Enligt kapitel 24.2 i kursboken kan alla minimala vägar som startar i en viss nod hittas i en riktad acyklisk graf i tid $O(|V| + |A|)$. Genom att anropa denna algoritm n gånger kan vi hitta längsta vägen totalt sett. Sammanlagd tid för detta blir $O(n(n + 2n^2)) = O(n^3)$ eftersom en graf med n noder innehåller färre än $2n^2$ bågar.