Algoritmer, datastrukturer och komplexitet

Övning 12

Anton Grensjö
grensjo@csc.kth.se

10 december 2015

Idag

- Komplexitetsklasser
- Blandade uppgifter från tentor och muntor

Repetition

- P: mängden av problem som kan lösas på polynomisk tid.
- NP: mängden av problem vars ja-instanser kan verifieras på polynomisk tid.
- CO-NP: mängden av problem vars nej-instanser kan verifieras på polynomisk tid.
  - Ett problem $A \in$ CO-NP om komplementproblemet till A tillhör NP. Exempel: “Finns det inte någon hamiltonsk cykel?”
- EXPTIME: mängden av problem som kan lösas i exponentiell tid.
- PSPACE: mängden av problem som kan lösas med tillgång till polynomiskt mycket minne (oavsett tid).

Uppgift 1: co-NP-fullständighet

- Modell av ett tekniskt diagnosproblem:
  - Vi har någon form av system, där komponenttillstånd, systemtillstånd och omgivningstillstånd representeras av boolska variabler.
  - Systemet representeras av en boolsk formel φ.
  - Om systemet fungerar så kommer variablerna ha värden så att φ är sann.
  - En komponent c representeras av en boolsk variabel x_c (sant = fungerar, falskt = trasig).
  - Vi vet att c är trasig om: när vi sätter in alla kända variabelvärden, samt x_c sann, i φ, så är den resulterande formeln inte satisfierbar.
- Visa att det är co-NP-fullständigt att avgöra ifall c är trasig.

Uppgift 1: co-NP-fullständighet

Tillhör co-NP:

- Vi ska visa att vi i polynomisk tid kan verifiera en nej-instans, dvs att c inte är trasig.
- c är inte trasig $\iff$ det finns en variabeltilldelning som satisfierar φ där x_n sann.
- Låt en sådan variabeltilldelning vara vår lösning. Trivialt att verifiera på polynomisk tid.

Uppgift 1: co-NP-fullständigt

co-NP-svårt:

- Vilket co-NP-fullständigt problem ska vi reducera?
  - Vilka co-NP-fullständiga problem känner vi till?
- Vi reducerar CO-SAT (problemet att avgöra ifall en given boolsk formel ϕ inte är satisfierbar).
- Reduktionen:
  - Låt ϕ vara en instans av CO-SAT.
  - Givet ϕ, konstruera ett system med den extra komponenten c, representerad av x_c, och som definieras av $\varphi = \phi \vee \neg x_c$.
  - Anropa nu en algoritm som löser vårt problem, med φ som indata.
- Om x_c sätts till sann blir $\varphi = \phi$, så problemet att avgöra ifall c är trasig är precis samma som att avgöra ifall ϕ inte är satisfierbar. Alltså är reduktionen korrekt.

Problemet är alltså co-NP-fullständigt.

Uppgift 2: Komplexitetsklassrelation

- PSPACE: Komplexitetsklassen som består av alla språk för vilka det existerar en deterministisk turingmaskin som känner igen språket i polynomiskt minne.
- EXPTIME: Består av alla språk för vilka det existerar en deterministisk turingmaskin som känner igen språket i exponentiell tid.

Visa att PSPACE $\subseteq$ EXPTIME.

Uppgift 2: Komplexitetsklassrelation

Visa att PSPACE $\subseteq$ EXPTIME.

- Betrakta en turingmaskin som löser ett problem med polynomiskt minne.
- Det finns ett tal k så att antalet använda rutor på bandet är $\mathcal{O}(n^k)$ (n indatastorlek).
- Om alfabetet består av tre tecken, så är antalet möjliga kombinationer $3^{\mathcal{O}(n^k)}$.
- Antalet möjliga platser för läs/skrivhuvudet är $\mathcal{O}(n^k)$.
- Totala antalet konfigurationer är alltså $\mathcal{O}(n^k) \cdot 3^{\mathcal{O}(n^k)}$, exponentiellt i n.
- Om man kommer tillbaka till samma konfiguration igen är man i en oändlig loop, så antalet konfigurationer är en övre gräns på antal steg som behövs för att lösa problemet.
- Varje problem som kan lösas med polynomiskt minne kan alltså lösas på exponentiell tid.

Uppgift 1: Sant eller falskt?

Är följande påståenden sanna eller falska? För varje deluppgift ger riktigt svar 1 poäng och ett övertygande bevisat riktigt svar 2 poäng,

a) Problemet att avgöra ifall ett tal med n siffror är ett primtal ligger i komplexitetsklassen co-NP.

- Problemet ligger i co-NP om komplementproblemet ligger i NP.
- Komplementproblemet: avgör ifall ett tal med n siffror är sammansatt (dvs kan faktoriseras i minst två faktorer större än 1).
- Detta problem ligger i NP, eftersom en lösning (en faktorisering av talet) kan verifieras på polynomisk tid. Hur?
- Svar: sant!

Uppgift 1: Sant eller falskt?

Är följande påståenden sanna eller falska? För varje deluppgift ger riktigt svar 1 poäng och ett övertygande bevisat riktigt svar 2 poäng,

b) Det finns en konstant $c > 1$ så att $n^3 \in O(c^{\log n})$.

- Kom ihåg räkneregler för exponentialfunktionen och logaritmen!

$$n^3 = 2^{\log n^3} = 2^{3 \log n} = \left(2^3\right)^{\log n} = 8^{\log n}$$

- Dvs om $c = 8$ så är $n^3 = c^{\log n} \implies n^3 \in \Theta(c^{\log n})$.
- Dvs n^3 och $c^{\log n}$ växer lika snabbt om $c = 8$. $c^{\log n}$ måste växa snabbare om $c > 8$.
- Slutsats: Om $c \geq 8$ så gäller $n^3 \in \mathcal{O}(c^{\log n})$.
- Svar: sant!

Uppgift 1: Sant eller falskt?

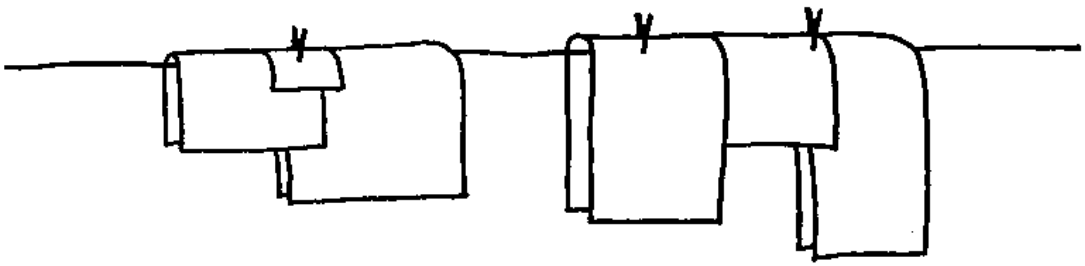
Är följande påståenden sanna eller falska? För varje deluppgift ger riktigt svar 1 poäng och ett övertygande bevisat riktigt svar 2 poäng,

c) Binära träd implementerar man vanligen genom att införa två pekare i varje post (`left` och `right`). När man implementerar ternära träd (där varje nod har tre söner) går det inte att klara sig med mindre än tre pekare i varje post.

- Falskt!
- Varje nod kan innehålla en pekare till sitt första barn, samt sitt nästa syskon.

Uppgift 2: Klädstrecket

(Algoritmkonstruktion, betyg C)



- Viggo har hängt n stycken handdukar av olika storlek på ett klädsträck.
- Varje handduk har en startkoordinat v_i och en slutkoordinat h_i.
- Viggo vill säkra handdukarna med klädnypor.
  - Det räcker med att varje handduk hålls på plats av en klädnypa.
  - Överlappande handdukar kan dela på samma klädnypa.

Indata: n, $(v_1, h_1), (v_2, h_2), \ldots, (v_n, h_n)$.
Konstruera en algoritm som så snabbt som möjligt beräknar var på sträcket man ska sätta nyporna, för att minimera antalet nypor som krävs.
Motivera korrekthet och analysera med enhetskostnad.

Uppgift 2: Klädstrecket

- Vilken typ av algoritm kan vara lämplig här?
- Vi försöker konstruera en girig algoritm.

1.
   - Sortera alla par, dels med avseende på vänsterkanten och dels med avseende på högerkanten.
   - Lagra resultaten i två listor, $Vsort$ och $Hsort$.
   - Håll under sorteringen reda på var varje post i den ena listan hamnar i den andra listan.
2. Så länge listan $Hsort$ är icke-tom:
   1. Ta det första återstående paret i $Hsort$, säg (v, h). Detta är den handduk i listan vars högerkant ligger längst till vänster.
   2. Sätt en klädnypa längst till höger på denna handduk.
   3. Ta bort alla par vars vänsterkant är till vänster om h (dvs ta bort alla handdukar som vår nya nypa sätter fast).
      - Dvs ta bort par från början av $Vsort$.
      - För varje par som tas bort, ta bort motsvarande par från $Hsort$.

Tidskomplexitet? $\mathcal{O}(n \log n)$

Uppgift 2: Klädsträcket

Korrekthet:

- Vi plockar bara bort handdukar som redan har fått en klädnypa i sig $\implies$ alla handdukar kommer sitta fast när algoritmen är klar.
- Visa att antalet klädnypor blir optimalt:
  - Betrakta handduken med lägst högerkoordinat. Den måste sitta fast, men var ska vi placera klädnypan?
  - Om man sätter fast den i punkten p så kommer alla handdukar med vänsterkant mindre än p att sitta fast.
  - För att sätta fast så många handdukar som möjligt, vill vi alltså välja p så långt till höger som möjligt.
  - Dvs det måste vara optimalt att sätta nypan i handdukens högerkant.
  - ...upprepa argumentet för de övriga handdukarna...

Uppgift 3: Gruppprepresentanter

komplexitet, betyg C

- Det finns många olika grupper av personer på KTH.
- Rektorn vill ha ett smidigt sätt att snabbt föra ut information till alla personer på KTH.
- Därför vill han skapa en grupp av informatörer, med följande krav:
  - Varje grupp ska vara representerad.
  - Informatörsgruppen ska vara så liten som möjligt.

Problemet är alltså att givet en uppsättning grupper (ej nödvändigtvis disjunkta!) hitta en så liten skara av representanter som möjligt.
Uppgift:

- Formulera problemet matematiskt som ett mängdproblem och beskriv det som ett beslutsproblem.
- Visa att beslutsproblemet är NP-fullständigt.

Uppgift 3: Gruppprepresentanter

Matematisk formulering:

- Låt k vara ett positivt heltal.
- Låt S vara mängden personer.
- Låt $C = \{C_1, \ldots, C_m\}$ vara de m grupperna.
- Problemet: Existerar det en delmängd $S' \subseteq S$ med högst k element så att $S' \cap C_i \neq \emptyset$ för $1 \leq i \leq m$?

Ligger problemet i NP?

- Låt en lösning vara mängden S'.
- Vi kan verifiera en lösning genom att:
  1. Kolla att $S' \subseteq S$.
  2. Kolla att $S' \cap C_i \neq \emptyset$ för $1 \leq i \leq m$.
- Detta kan uppenbarligen göras på polynomisk tid.
- Problemet är NP-svårt.

Uppgift 3: Gruppprepresentanter

NP-svårt?

- Vilket problem ska vi reducera?
- Hörntäckning!
  - Givet en instans $G = (V, E)$ till hörntäckning, låt $S = V$ och $C = E$.
  - En hörntäckning av storlek k motsvarar precis en delmängd $S' \subseteq S$ av storlek k som innehåller minst ett element från varje C_i.

Fler blandade uppgifter

Det finns två uppgifter till i övningsanteckningarna, som vi går igenom om vi hinner.

Sista övningen

Tack för den här kursen!

Jag tar väldigt gärna emot feedback angående övningarna, så prata gärna med mig, eller skicka iväg ett mejl.

Lycka till på mästarprovsredovisningarna och på tentan!