Viggo Kann, KTH CSC

Lösningar till teoritenta i Algoritmer (datastrukturer) och komplexitet för KTH DD1352/2D1352/DD2354/2D1354 och SU 2010-05-27

1. (8 p) Är följande påståenden sanna eller falska? För varje deluppgift ger riktigt svar 1 poäng och ett övertygande motiverat riktigt svar 2 poäng.

   a) Det existerar problem som är så svåra att inte någon algoritm kan lösa dom, oavsett hur lång tid algoritmen får på sig.

   Sant. Stopproblemet är oavgörbart, så det är ett problem som uppfyller kriteriet.

   b) En algoritm med tidskomplexiteten $O(\log(n!))$ går i polynomisk tid.

   Sant. $\log n! < \log n^n = n \log n$ som växer polynomiskt.

   c) Bloomfilter kan med fördel vara av tvåpotensstorlek, om man väljer rätt hashfunktioner.

   Sant. Bloomfilter består av en boolesk hashtabell. Med Bob Jenkins hashfunktion så går det utmärkt att använda tvåpotensstorlek på tabellen (vilket gör att man slipper kostsamma moduloberäkningar).

   d) Om problemet A kan reduceras till problemet B så är B mindre och enklare än A, komplexitetsmässigt.

   Falskt. Om A kan reduceras till B så kan A lösas med hjälp av en algoritm för B. Alltså kan inte B vara enklare än A (men däremot kan B vara svårare än A).

2. (3 p) Följande utsaga är sann: *Varje beslutsproblem vars ja-lösningar kan verifieras i polynomisk tid kan lösas av en algoritm som bara använder polynomiskt mycket minne.*

   a) Formulera utsagan med bara matematiska tecken och namn på komplexitetsklasser.

   NP $\subseteq$ PSPACE

   b) Skissa ett bevis för att utsagan är sann.

   En algoritm som går igenom alla tänkbara lösningar och för varje lösning testar om den är riktig med hjälp av verifieringsalgoritmen tar bara polynomiskt mycket minne. Anledningen är att lösningen är polynomiskt stor och att verifieringen tar polynomisk tid och därför även polynomiskt minne.

3. (3 p) Totalsökning är en av de fyra viktigaste algoritmkonstruktionsmetoderna som nämnts i kursen. Vilka är de andra tre? Ge för varje konstruktionsmetod exempel på **en** algoritm från kursen som bygger på den metoden. Skriv antingen namnet på algoritmen eller ange både problemet som algoritmen löser och ett Θ-uttryck för tidskomplexiteten.

Dekomposition	FFT
Dynamisk programmering	Maximal triangelstig $O(n^2)$
Giriga algoritmer	Dijkstras algoritm

4. (6 p) Vi söker i denna uppgift en polynomisk heuristik som ger en så bra lösning till TSP (handelsresandeproblemet) som möjligt, ju kortare tur desto bättre. Algoritmen ska också ge ett så bra mått som möjligt på hur långt bort från den optimala lösningen som den hittade lösningen som mest är, uttryckt i procent.

   Till ditt förfogande finns följande färdiga polynomiska algoritmer:

$\langle \Pi, s \rangle = \text{Christofides}(n, D)$	Approximationsalgoritm med approximationskvot 3/2.
$\langle \Pi, s \rangle = \text{NearestInsertion}(n, D)$	Deterministisk heuristik.
$\langle \Pi, s \rangle = \text{RandomInsertion}(n, D)$	Probabilistisk heuristik.
$\langle \Pi_2, s \rangle = \text{LocalSearch2Opt}(n, D, \Pi_1)$	Deterministisk heuristik som ger en förbättring av lösningen Π_1 med hjälp av lokal sökning.

   där Π beskriver en lösning som en permutation av städerna, s är lösningens (turens) längd, n är antalet städer och D är en symmetrisk $n \times n$-matris som beskriver avstånden mellan städerna. Anta att avstånden är positiva heltal och uppfyller triangelolikheten.

   Din algoritm ska ta n och D som indata och returnera trippeln $\langle \Pi, s, g \rangle$ som utdata, där g är måttet på hur långt algoritmens lösning som mest är från den optimala.

   Om s_C är det värde på turens längd som Christofides algoritm ger så vet vi att $s_C/\text{OPT} \leq 3/2$. Det betyder att om vi lyckas hitta en ännu bättre lösning med värdet s så gäller $s/\text{OPT} = (s/s_C) \cdot s_C/\text{OPT} \leq (3/2) \cdot (s/s_C)$. Avståndet till optimala värdet uttryckt i procent är $100 \cdot ((s/\text{OPT}) - 1) \leq 100 \cdot ((3/2) \cdot (s/s_C) - 1)$.

   $\langle \Pi_C, s_C \rangle = \text{Christofides}(n, D)$
   $\langle \Pi, s \rangle = \text{LocalSearch2Opt}(n, D, \Pi_C)$
   $\langle \Pi_N, s_N \rangle = \text{NearestInsertion}(n, D)$
   $\langle \Pi_t, s_t \rangle = \text{LocalSearch2Opt}(n, D, \Pi_N)$
   if $s_t < s$ **then** $\Pi \leftarrow \Pi_t; s \leftarrow s_t$
   for $i \leftarrow 1$ **to** 100 **do**
       $\langle \Pi_R, s_R \rangle = \text{RandomInsertion}(n, D)$
       $\langle \Pi_t, s_t \rangle = \text{LocalSearch2Opt}(n, D, \Pi_R)$
       if $s_t < s$ **then** $\Pi \leftarrow \Pi_t; s \leftarrow s_t$
   if $s_C < s$ **then** $\Pi \leftarrow \Pi_C; s \leftarrow s_C; g \leftarrow 50$
   else $g \leftarrow 100 \cdot ((3/2) \cdot (s/s_C) - 1)$
   return $\langle \Pi, s, g \rangle$

På kursens webbsida http://www.csc.kth.se/DD1352/adk10/ ligger en kursenkät som var och en uppmanas att svara på så snart som möjligt. Viggo och 2011 års elever på kursen tackar på förhand!

Vill du ha högre betyg på kursen? Om du har fått minst betyg C på två av dom betygsatta kursmomentet (teoritentan, mästarprov 1, mästarprov 2) och minst betyg E på det tredje så får du boka in dej på muntlig redovisning 31 maj eller 1 juni. Boka en tid på kursens webbsida redan idag och senast 30 maj klockan 9.00. Välj om du vill redovisa för betyg C, B eller A.