DD1352 Algoritmer, datastrukturer och komplexitet, hösten 2013

Mästarprov 1: Algoritmer

Mästarprovet ska lösas **individuellt** och redovisas både skriftligt och muntligt. *Inget samarbete är tillåtet, se vidare hederskodexen.* Du ska alltså inte diskutera lösningar med någon annan fram till dess att alla muntliga redovisningar är avklarade.

Skriftliga lösningar ska lämnas senast **måndag 14 oktober klockan 11.15** på föreläsningen eller senast klockan 11.00 samma dag i kursens inlämningslåda på studentexpeditionen på Osquars backe 2, plan 4. Det är viktigt att du lämnar in i tid!

Skriv ditt namn och personnummer överst på framsidan av lösningarna. Se till att spara en kopia av dina lösningar så att du kan läsa på inför den **muntliga redovisningen** som kommer att ske 18–23 oktober för någon av lärarna. Boka tid för en femton minuters muntlig redovisning på kurswebbsidan senast 14 oktober klockan 11. Bokningslistorna läggs upp senast 10 oktober. Om du inte hinner göra uppgiften så avbokar du enkelt din bokning.

Det är viktigt att du förbereder dig inför den muntliga redovisningen. För att en uppgift ska godkännas ska du kunna förklara och motivera algoritmen muntligt och reda ut eventuella oklarheter.

Läs uppgifterna mycket noga så att du inte råkar basera dina lösningar på en missuppfattning. Fråga en lärare på kursen om något är oklart.

Mästarprovet är ett obligatoriskt moment i kursen. Det består av tre uppgifter som motsvarar betygskriterierna för E, C respektive A. För godkänt (betyg E) krävs helt rätt på en av uppgifterna. Helt rätt på två av uppgifterna ger betyg C och alla rätt ger betyg A. Ett mindre fel på en uppgift sänker betyget ett steg. Läs mer om betyg på kursens webbsida.

För att se exempel på hur utförliga lösningarna bör vara kan du titta på lösningar till *tidigare mästarprov* på kurswebben.

1. **Hur kort kan texten bli?**
   Betygskriterium: utveckla algoritmer med datastrukturer för enkla problem givet en konstruktionsmetod.

   En text består av n stycken ord. Ordens längd i tecken betecknas $w_1, \ldots, w_n$ (i samma ordning som orden förekommer i texten). För enkelhets skull ingår skiljetecken i det ord dom följer, så vi slipper skilja på bokstäver och skiljetecken. När vi skriver ner texten ska det som vanligt stå ett mellanslag mellan två ord som följer på varandra på samma rad. Målet är att formatera texten (utan avstavningar) så att den ryms på så få rader av längd *len* som möjligt.

   Konstruera och beskriv pseudokoden för en effektiv girig algoritm som givet n, *len* och $w_1, \ldots, w_n$ beräknar och returnerar det minimala antalet rader av längd högst *len* som texten kan skrivas ner på. Du kan anta att inget enskilt ord i texten är längre än *len* tecken.

   Som vanligt för giriga algoritmer måste du bevisa att din algoritm returnerar det optimala värdet. Analysera också tidskomplexiteten för algoritmen (med enhetskostnad).

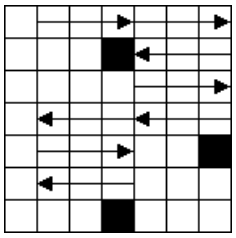
Vänd!

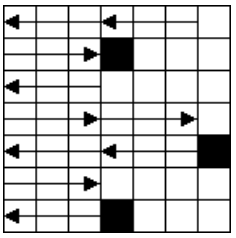
2. **Optimal kedjevikning**
Betygskriterium: utveckla algoritmer med datastrukturer för icketriviala problem.

En kedja består av länkar som är tre enheter långa. En snickare har byggt ett kvadratiskt etui för förvaring av kedjan. Etuiet kan beskrivas som en $n \times n$-matris där vissa rutor är blockerade (innehåller spikar som sätter fast etuiet i underlaget). Kedjan ska vikas fram och tillbaka rad för rad i etuiet, uppifrån och ned. Kedjan får inte gå upp igen till en tidigare rad, den får inte gå in i en blockerad ruta eller in i sig själv, och den får inte hoppa över en rad (dvs länkar måste ligga horisontellt, inte vertikalt). Kedjan behöver inte börja på översta raden. Frågan är hur lång den längsta kedjan är som kan rymmas i etuiet.

Indata består av en bitmatris $B[1..n, 1..n]$ där blockerade rutor i etuiet motsvaras av ettor i motsvarande positioner i matrisen. I exemplet nedan kan du se två möjliga kedjevikningar i samma 7×7-etui.

Din uppgift är att konstruera en effektiv algoritm som givet $B[1..n, 1..n]$ beräknar antalet länkar i den längsta kedja som kan rymmas i etuiet med ovanstående regler för vikningen. Beskriv algoritmen med pseudokod och analysera dess tidskomplexitet uttryckt i n. Den måste vara polynomisk. Motivera också att algoritmen är korrekt.


Kedja med 8 länkar.


Kedja med 10 länkar

3. **Kan ämnet framställas?**
Betygskriterium: utveckla algoritmer med datastrukturer för svårare problem med den metod som passar bäst.

Vi tänker oss att vi har ett antal kemiska ämnen (representerade av heltal mellan 1 och m) som råvaror och vill framställa något annat ämne. Med hjälp av laboratorieutrustning kan vi framställa diverse olika ämnen. Från dessa kan vi i sin tur framställa nya ämnen. Nu gäller det att ta reda på om vi givet råvaror och framställningsmetoder kan framställa ett visst ämne. Du har obegränsad mängd av varje råvara.

Indata består av en array med tillgängliga råvaror $a[1..p]$ ($p < m$) där varje element är ett ämnesnummer (mellan 1 och m), ett ämnesnummer b (den önskade produkten), en array med alla reaktioner vi kan åstadkomma $r[1..s]$, där varje element är en tupel $(j, [i_1, i_2, \ldots, i_k])$. En tupel $(j, [i_1, i_2, \ldots, i_k])$ betyder att man från ämnena med nummer $i_1, i_2, \ldots, i_k$ kan framställa ämne nummer j. Det kan finnas flera reaktioner som framställer samma ämne. Låt n vara totala antalet tal i indata.

Konstruera och analysera en effektiv algoritm som avgör ifall det är möjligt att framställa ämne b med råvarorna och reaktionerna i indata. Endast en algoritm som har optimal tidskomplexitet ger godkänt. Visa därför att övre och undre gräns sammanfaller.

Motivera att din algoritm är korrekt.