Tentamen i Datastrukturer och algoritmer, fk, DVA246

Akademin för innovation, design och teknik

2022-06-02

Skrivtid: 14.30 – 18.30
Hjälpmedel: Inga hjälpmedel
Lärare: Daniel Hedin, kan nås via mail eller på telefon: 021-107052 (15.00 – 16.00, 17.00-18.00)

Preliminära betygsgränser

Betyg 3:	50%
Betyg 4:	70%
Betyg 5:	90%
Max:	44p

Tentamen består av 8 frågor.

Krav och allmänna råd:

- Läs igenom hela tentan för att veta hur du skall disponera din tid
- Frågorna står inte i svårighetsordning
- Skriv tydligt vilken uppgift du svarar på
- Ordna svaren i rätt ordning med svaret på uppgift 1 först och svaret på uppgift 8 sist
- Om du är osäker på vad som avses och gör antaganden skriv ut vilka dessa antaganden är.
- Det går att få poäng för partiella lösningar givet att de är meningsfulla

Uppgift 1: Alternativ till sorterade arrayer (6+6p)

Binär sökning av en sorterad array tar logaritmisk tid, men insättning är linjär i arrayens storlek. Det går att skapa en alternativ datastruktur som baserar på ett antal sorterade arrayer vars storlek följer binärrepresentationen av antalet. Låt n vara antalet element, $k = \lceil \lg(n+1) \rceil$ och låt $<n=, n_{k-2}, ..., n_0>$ vara binärrepresentationen av n. Låt $A_0, A_1, ..., A_{k-1}$ vara k sorterade arrayer där i = 0, 1, ..., k-1, längden på array A_i är antingen 0 eller 2^i beroende på om $n_i = 0$ eller om $n_i = 1$. Även om varje array är sorterad finns det ingen inbördes ordningen mellan elementen i olika arrayer.

1) Beskriv hur man kan göra sökning i ovanstående datastruktur. Vilken komplexitet får sökalgoritmen?
2) Beskriv hur insättning av ett nytt element i datastrukturen går till. En korrekt insättningsalgoritm är amorterad O(lg n). Troliggör denna komplexitet genom att visa hur bokföringsmetoden skulle fungera på 10 konsekutiva insättningar.

Uppgift 2: Rekurrensformler (3p)

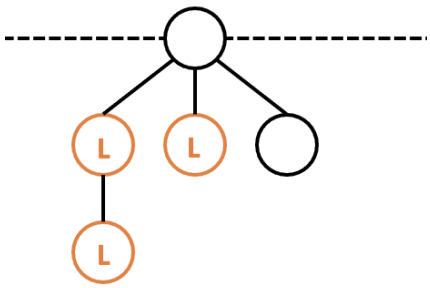
Förklara i korthet vad en rekurrensformel (recurrence) är och när de uppstår.

Uppgift 3: Heap (2p)

Var i en min-heap (binär) hittar vi den största nyckeln?

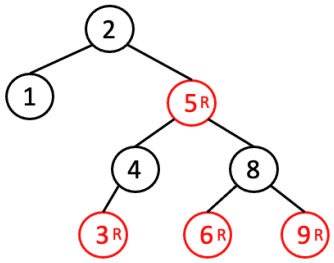
Uppgift 4: Fibonnaci-heap (6p)

Ge en möjlig sekvens av operationer som resulterar i en Fibonnaci-heap med samma form som nedan. De orangea noderna (också markerade med 'L') är markerade noder (det vi på föreläsningen kallat "loser"-noder).



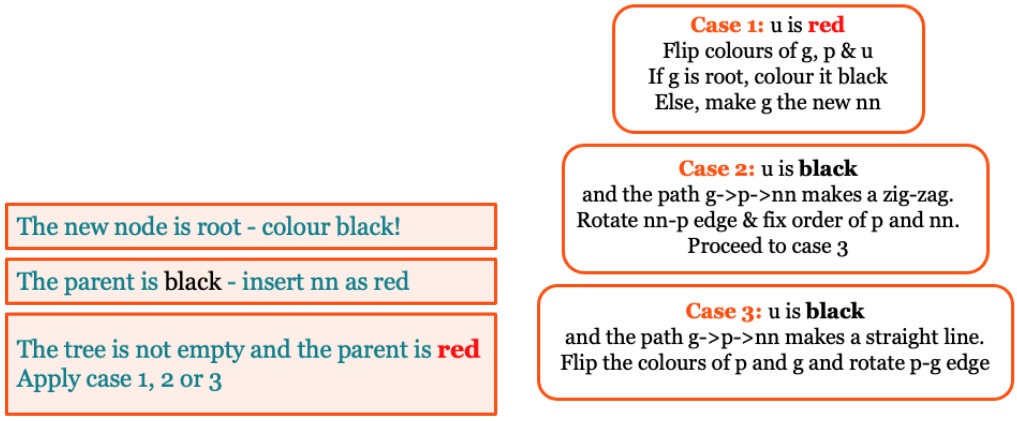
Uppgift 5: Röd-svarta träd (6p)

Antag att du har följande rödsvarta träd (de röda noderna är markerade med ett 'R', de noder som inte är markerade är svarta):



Rita hur trädet ser ut efter att 7 har sats in i det.

Till hjälp får du följande bild hämtad från föreläsningen om rödsvarta träd.



Uppgift 6: Dijkstras algoritm (4p)

Ge ett exempel på en riktad graf med negativt viktade bågar, men utan negativa cykler, där Dijkstra's algoritm misslyckas med att generera ett korrekt svar. Rita grafen och förklara varför Dijkstra's misslyckas på ditt exempel. Startnoden ska vara markerad med 'S'.

Uppgift 7: Giriga algoritmer (8p)

Aktivitetsselektionsproblemet går ut på att hitta en optimal mängd av ömsesidigt kompatibla aktiviteter från en given mängd av aktiviteter. Varje aktivitet a_i har en starttid s_i och en sluttid f_i. Två aktiviteter a_i och a_j är kompatibla om det halvöppna intervallet $[s_i, f_i)$ inte överlappar med det halvöppna intervallet $[s_j, f_j)$. På föreläsningen visade vi att hela tiden välja den aktivitet som slutar först och som är kompatibel med de redan valda aktiviteterna leder till en optimal lösning. Intuitionen var att valet av något som slutar så tidigt som möjligt ger oss maximala möjligheter att välja resterade aktiviteter. Dualen till denna algoritm är att välja den aktivitet som börjar så sent som möjligt och som är kompatibel med de redan valda aktiviteterna. Ge ett övertygande argument (bevis) för varför detta ger en optimal lösning.

Uppgift 8: Dynamisk programmering (3p)

Vilken egenskap utnyttjar dynamisk programmering för att snabba upp exekveringstiden och hur görs det?