TDT4120 Algoritmer og datastrukturer

Eksamen, 14. desember 2021, 09:00–13:00

Faglig kontakt	Magnus Lie Hetland
Hjelpemiddelkode	A

Løsningsforslag

Løsningsforslagene i rødt nedenfor er *eksempler* på svar som vil gi uttelling. Det vil ofte være helt akseptabelt med mange andre, beslektede svar, spesielt der det bes om en forklaring eller lignende. Om du svarte noe litt annet, betyr ikke det nødvendigvis at du svarte feil!

På grunn av koronapandemien er dette en hjemmeeksamen med alle hjelpemidler tillatt. Følgelig bør ikke eksamenssettet ses som representativt for ordinære eksamener.

10 % **1** Hva er stabil sortering? Forklar kort med egne ord.

> Det er en sortering der like elementer ikke bytter plass.
>
> Om man tar med eksempler som Counting-Sort, og anvendelser som Radix-Sort, så er det naturligvis helt i orden, men det kreves ikke.

10 % **2** Hva er topologisk sortering? Forklar kort med egne ord.

> Det er en rekkefølge av nodene i en rettet, asyklisk graf, som er slik at om grafen har en kant fra u til v, så havner u før v i rekkefølgen.
>
> Man kan ev. også beskrive at den kan finnes ved å sortere etter synkende *finish time* fra DFS.

10 % **3** Når vi finner minimale spenntrær, bruker vi den samme strategien i begge pensumalgoritmene. Forklar denne strategien kort med egne ord.

Merk: Her er vi ute etter noe mer spesifikt enn f.eks. grådighet.

Vi velger hele tiden en av de korteste kantene over et snitt som respekterer løsningen så langt.

Med at snittet respekterer løsningen menes altså at ingen av kantene valgt så langt krysser snittet.

10 % **4** Hva er forskjellen mellom den sentrale egenskapen til et binært søketre og den sentrale egenskapen til en binærhaug? Forklar kort med egne ord.

Den sentrale egenskapen til et binært søketre (den såkalte *binary-search-tree property*) er at noder i venstre deltre har en nøkkel som er mindre eller lik rota, mens noder i høyre deltre har en nøkkel som er større eller lik. Haug-egenskapen (*the heap property*), derimot, sier at barna (og dermed, induktivt, alle noder i begge deltrær) har nøkler som er mindre eller lik (for en maks-haug) eller større eller lik (for en min-haug).

Her vil man også få noe uttelling om man har forklart anvendelsene og virkemåtene til de to datastrukturene.

10 % **5** Følgende algoritme har kjøretid $\Theta(n \lg n)$ i beste tilfelle og $\Theta(n^2)$ i verste:

```
LOREM(A)
  n = A.length
  QUICKSORT(A, 1, n)
  for i = 1 to n
      IPSUM(A, i)
```

Hva kan du si om kjøretiden til IPSUM i beste og verste tilfelle? Forklar kort.

Her aksepteres tre ulike svar (med tilhørende forklaring).

Svar 1:

Antar implisitt at kjøretiden til IPSUM er uavhengig av i, eller at den vil ha verste kjøretid i hver iterasjon.

I beste tilfelle har QUICKSORT kjøretid $\Theta(n \lg n)$, så for å ikke gi LOREM en verre kjøretid, må IPSUM (som utføres n ganger) da ha kjøretid $O(\lg n)$.

I verste tilfelle har QUICKSORT kjøretid $\Theta(n^2)$, så for å ikke gi LOREM en verre kjøretid, må IPSUM da ha kjøretid $O(n)$.

Svar 2:

Antar implisitt at den verste kjøretiden forekommer for $1 \leqslant i \leqslant n$.

Kjøretiden til IPSUM kan være avhengig av i, så den verste kjøretiden trenger bare dukke opp én gang i **for**-løkken. Derfor har den i beste tilfelle kjøretid

$O(\lg n)$ og i verste tilfelle $O(n^2)$.

Svar 3:

Kjøretiden til IPSUM kan være avhengig av i, og trenger ikke forekomme for $1 \leqslant i \leqslant n$. Dermed kan den være vilkårlig stor, uten å påvirke total kjøretid.

Om man her bruker Θ i stedet for O, så vil det fortsatt gi noe uttelling.

10 % **6** Forenkle følgende uttrykk:

$$\frac{\Omega(n^3)}{O(n^2)} + \frac{\Theta(n^5)}{\Theta(n^3)} + \frac{O(n^7)}{\Omega(n^4)}$$

Uttrykk svaret med asymptotisk notasjon. Forklar og diskuter kort.

Du kan anta at uttrykket er veldefinert.

Svaret blir $\Omega(n^2)$.

I det første leddet er telleren asymptotisk lik n^3 eller mer, mens nevneren er asymptotisk lik n^2 eller mindre, og forholdet dem imellom blir dermed asymptotisk lik n eller mer, altså $\Omega(n)$.

I det andre leddet kan vi forkorte direkte, og får $\Theta(n^2)$.

I det tredje leddet vil nevneren være asymptotisk lik n^7 eller mindre, mens nevneren er asymptotisk lik n^4 eller mer, og forholdet imellom dem blir dermed asymptotisk lik n^3 eller mindre, altså $O(n^3)$.

Vi har da forenklet uttrykket til:

$$\Omega(n) + \Theta(n^2) + O(n^3)$$

Den minste verdien dette uttrykket kan få er asymptotisk lik n^2, pga. leddet $\Theta(n^2)$, mens det ikke finnes noen øvre grense, pga. leddet $\Omega(n)$. Dermed blir den mest presise beskrivelsen vi kan gi $\Omega(n^2)$.

Her kan forklaringen godt være atskillig kortere enn den som er gitt i løsningsforslaget.

10 % **7** Løs følgende rekurrens eksakt, med iterasjonsmetoden:

$$\begin{aligned} T(1) &= 0 \\ T(n) &= T(n-1) + \lg(n/(n-1)) \qquad (n > 1) \end{aligned}$$

Oppgi svaret uten bruk av asymptotisk notasjon. Vis fremgangsmåten din. Vis hvordan du kan verifisere løsningen ved hjelp av induksjon (altså med substitusjonsmetoden).

Hint: Husk at $\lg(a \cdot b) = \lg a + \lg b$.

Vi ender med $T(n) = \lg n$.

Utregning:

$\lg(n/(n-1)) = \lg(n) - \lg(n-1)$.

Løsning med iterasjonsmetoden (der jeg flytter T sist, for enkelhets skyld):

$$\begin{aligned} T(n) &= \lg(n) - \lg(n-1) + T(n-1) \\ &= \lg(n) - \lg(n-1) + \lg(n-1) - \lg(n-2) + T(n-2) \\ &= \lg(n) - \lg(n-2) + T(n-2) \\ &= \lg(n) - \lg(n-2) + \lg(n-2) - \lg(n-3) + T(n-3) \\ &= \lg(n) - \lg(n-3) + T(n-3) \\ &\vdots \\ &= \lg(n) - \lg(n-i) + T(n-i) \end{aligned}$$

Vi lar $i = n - 1$ og bruker at $T(n - n + 1) = 0$:

$$T(n) = \lg(n) - \lg(n - n + 1) + T(n - n + 1) = \lg(n)$$

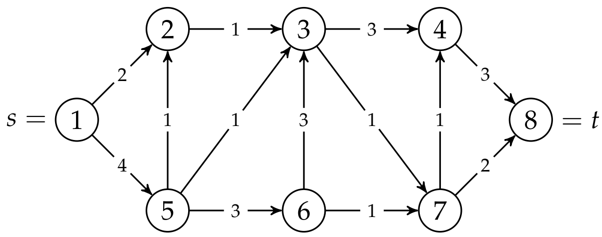
Vi kan verifisere ved å bruke induksjonshypotesen $T(n-1) = \lg(n-1)$:

$$T(n) = \lg(n-1) + \lg(n) - \lg(n-1) = \lg(n)$$

I eksamenssettet var grunntilfellet $T(0) = 0$, der det generelle tilfellet gjaldt for $n \geqslant 1$. Det ble gitt informasjon om korreksjonen kl. 10:16.

En utregning basert på opprinnelig oppgaveformulering vil i hovedsak være lik, bortsett fra at man ender med et udefinert tilfelle, med $1/0$ eller $\lg 0$. Et slikt svar vil også kunne gi full uttelling, dersom det demonstrerer bruk av iterasjonsmetoden.

10 % **8** Hvilke noder vil Ford-Fulkerson traversere i siste iterasjon, i dette flytnettet?



Oppgi svaret ved å liste opp nodene i sortert rekkefølge. Forklar kort.

Hint: Den maksimale flytverdien er 5.

Svar: 1, 2.

Siden maksimal flytverdi er 5, må alle snitt med kapasitet 5 fylles opp (og forsvinne fra restnettet/residualnettverket). Det inkluderer snittet mellom $\{1, 2\}$ og $\{3, \ldots, 8\}$. Siden maksimalt 1 flytenhet kan gå videre fra node 2, så vil kanten $(1, 2)$ aldri fylles helt, så vi når frem til noder 1 og 2 i siste iterasjon.

Her godtas andre og ev. enklere forklaringer, så lenge de demonstrerer noe innsikt i hvorfor svaret blir som det blir.

Om sensur: I oppgaven spørres det spesifikt etter hvilke noder som *traverseres* (jf. beskrivelsen nederst på s. 725, der dybde-først- og bredde-først-søk nevner som muligheter). Det er altså snakk om *alle* nodene som traverseres, og ikke bare en forøkende sti.

Om man ser bort fra fremstillingen i forelesningene, inkl. diskusjoner om å finne minimale snitt, kan man i prinsippet tolke «siste iterasjon» *nest siste* traversering, dvs. den iterasjonen som finner siste forøkende sti, heller enn den siste traverseringen, som altså ikke når frem til t. Om man tolker det slik, vil ikke svaret være entydig, enten man oppgir den siste forøkende stien eller det nest siste settet med noder som traverseres.

Hensikten med oppgaven var først og fremst å teste evnen til å utføre Ford-Fulkerson, og ev. å kunne benytte seg av minimale snitt for å finne svaret, ikke spesifikt å kunne slutte seg til at det var siste traversering og ikke siste forøkende sti det ble spurt etter. Derfor tas oppgaven ut av sensur der det er til fordel for kandidaten. Dvs., man får maksimum av (*i*) opprinnelig poengsum og (*ii*) summen for de resterende oppgavene, skalert med 10/9.

10 % **9** La u og v være to forskjellige noder i en vektet, rettet graf G. La p være den korteste stien fra u til v, og la p' være den korteste av alle de stiene fra u til v som ikke inneholder negative sykler.

Betrakt følgende påstander:

1. Stien p vil aldri inneholde en negativ sykel.
2. Hvis G inneholder en negativ sykel, finnes ikke p.
3. Hvis G inneholder en negativ sykel, kan vi likevel finne p'.

Forklar om utsagnene stemmer eller ikke, ev. med hvilke unntak, antagelser eller forbehold.

Svar relativt grundig (f.eks. ca. 100–200 ord).

Du kan anta at p og p' er unike; ingen andre stier av samme type er like korte.

Det er riktig at p aldri vil inneholde en negativ sykel; hvis den gjorde det, ville det finnes en kortere sti (som fulgte sykelen en gang til), som er en selvmotsigelse.

Hvis G inneholder en negativ sykel, er det riktig at p ikke finnes, dersom det finnes en sti fra u til v som går innom denne negative sykelen. Ellers finnes p så lenge det finnes en sti fra u til v.

Så lenge det finnes en sti fra u til v, vil p' alltid eksistere, så vi kan i prinsippet finne den, men ikke nødvendigvis i praksis. Hvis vi tillater at G inneholder negative sykler, er det generelt NP-hardt å finne p'. Men dersom ingen av stiene fra u til v går innom noen negativ sykel, så kan vi bruke f.eks. BELLMAN-FORD til å finne korteste vei fra u til v. Den vil kunne returnere FALSE, men stien vi har funnet vil likevel være p'.

10 % **10** Konstruer og beskriv kort en effektiv algoritme som tar inn to rettede asykliske grafer og finner den lengste stien som forekommer i begge.

Om det er flere, skal du finne én av de lengste stiene; det er samme hvilken. Du får altså inn grafene $G = (V, E)$ og $G' = (V', E')$ og skal finne en lengst mulig sekvens $\langle v_1, \ldots, v_k \rangle$, der $(v_i, v_i + 1) \in E \cap E'$ for $i = 1 \ldots k - 1$.

Dette kan løses med dynamisk programmering, enten direkte på de to grafene eller ved å konstruere en ny graf.

Direkte løsning:

For enhver node $v \in V \cap V'$ kan problemet løses rekursivt for alle noder u der $(u, v) \in E \cap E'$; man velger den beste av disse og legger 1 til lengden. Svaret blir maksimum over V.

Dette kan enten memoiseres, eller løses fra bunnen (*bottom-up*) ved å iterere over (f.eks.) V i topologisk sortert rekkefølge. I begge tilfeller må man lagre både lengden og hvilken delløsning (forgjenernode) den bygger på, så man kan rekonstruere stien.

Løsning vha. ny graf:

Man lager en ny graf $G'' = (V \cap V', E \cap E')$. Enhver sti som er felles for G og G' vil også finnes i G''. Man kan deretter finne den lengste stien i G'' vha. dynamisk programmering, ev. ved å bruke DAG-SHORTEST-PATHS med kantvekter satt til -1.

Om man kun beskriver hvordan man finner lengden til stien, og ikke hvordan man finner selve stien, så gis det nesten full uttelling.

Om man konstruerer en ny graf med kantmengde $E \cap E'$ og nodene som tilhører disse kantene, risikerer man å få en tom nodemengde, og dermed ingen sti, selv om $V \cap V'$ inneholder én eller flere noder (og dermed minst én sti av lengde 0, dvs. $k = 1$). Dette vil likevel gi nesten full uttelling.