TECHNISCHE UNIVERSITEIT DELFT
Faculteit Elektrotechniek, Wiskunde en Informatica

TU Delft

Tentamen IN2505-A Algoritmiek

15 april 2011, 14.00-17.00

- Het gebruik van boek, aantekeningen of rekenmachine tijdens dit tentamen is ***niet*** toegestaan.
- Dit tentamen heeft 15 meerkeuzevragen in totaal goed voor 3 punten en 4 open vragen in totaal goed voor 6 punten. Je krijgt sowieso 1 punt kado. Merk op dat je eindcijfer voor dit vak ook voor $\frac{1}{3}$ uit het onafgeronde gemiddelde van het practicum bestaat (mits beide ≥ 5.0).
- Wat betreft de meerkeuzevragen:
  - Er is voor iedere vraag telkens maar één goed antwoord mogelijk.
  - Alle vragen tellen even zwaar, maar bij de puntentoekenning wordt wel gokkanscorrectie toegepast. Als je geen enkel antwoord geeft, wordt dit altijd fout gerekend.
  - Schrijf je antwoorden eerst op kladpapier en neem ze later over op het antwoordformulier.
  - Vul je studienummer zowel met cijfers in als met streepjes/blokjes.
  - Vul het antwoordformulier bij voorkeur in met pen (geen rode pen) of potlood (goed zwart maken); gebruik geen doorhalingen.
  - Probeer in totaal niet meer dan een uur aan de meerkeuzevragen te besteden. De overige twee uur heb je nodig voor de open vragen.
- De open vragen worden als volgt gewogen:

Vraag:	1	2	3	4	Totaal:
Punten:	6	7	9	8	30

  - Geef antwoord in correct Nederlands of Engels en schrijf leesbaar (gebruik eerst kladpapier).
  - Geef geen irrelevante informatie. Dit kan leiden tot puntenaftrek. Het aangegeven maximaal aantal regels is van toepassing op (getypte) regels en wordt dus aangepast aan de grootte van je handschrift en het gebruik van witruimte.
  - Als er wordt gevraagd om een efficiënt algoritme, geef dan het meest efficiënte algoritme dat je kunt bedenken. Een langzamer algoritme kan minder punten opleveren.
  - Als er wordt gevraagd om pseudocode, dan mag je daarin algoritmen uit het boek aanroepen, tenzij anders vermeld.
  - Voordat je je antwoorden inlevert, controleer of op ieder blaadje je naam en studienummer staat en geef het aantal ingeleverde bladen voor de open vragen aan op (tenminste) de eerste pagina.
- De tentamenstof bestaat uit Chapters 1 to 7 uit Algorithm Design van Kleinberg en Tardos (behalve secties 4.9*, 5.6, 6.10*, 7.4*, en 7.13*), en bovendien de notities over de Master Method en over bewijstechnieken (Proving guide).
- Totaal aantal pagina's: 6.

Meerkeuzevragen

1. Gedurende het algoritme van Gale-Shapley
   A. komen nooit instabiele koppels voor.
   B. blijft een vrouw, eenmaal gekoppeld, altijd gekoppeld.
   C. wordt een vrouw altijd gekoppeld aan haar minst favoriete man.
   D. wordt elke opvolgende partner voor een man hoger in zijn voorkeurordening.

   Welke van deze beweringen is correct?

2. Gegeven $f(n) = n \log n$ en $g(n) = n^{1+\epsilon}$. Welke van de volgende beweringen is correct?
   A. $g(n)$ is $\Omega(f(n))$ voor $\epsilon < 0$
   B. $f(n)$ is $O(g(n))$ voor $\epsilon > 0$
   C. $f(n)$ is $\Omega(g(n))$ voor $0 < \epsilon < 1$
   D. $g(n)$ is $\Theta(f(n))$ voor $0 < \epsilon < 1$

3. Gegeven is een ongerichte graaf zonder cycles met n knopen, bestaande uit k verbonden (connected) componenten. Hoeveel kanten heeft deze graaf?
   A. $n - 1$
   B. $n - k$
   C. $n - k - 1$
   D. Dat kun je niet uit deze gegevens afleiden.

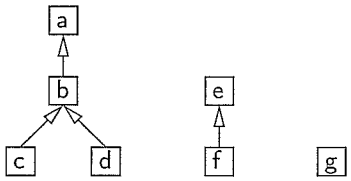
4. Gegeven is een gerichte graaf zonder cycles (DAG) met n knopen en m kanten. Wat is de krapste worst-case bovengrens op de looptijd van de meest efficiënte implementatie om een topologische volgorde (topological sort) te bepalen?
   A. $O(m + n)$
   B. $O(n \log n)$
   C. $O(nm)$
   D. $O(n^2)$

5. Gegeven een verbonden graaf G en een knoop s. De afstand tussen twee knopen is gedefinieerd als het minimaal aantal kanten van een pad tussen die twee knopen. Wat is de efficiëntste manier om de knoop te vinden die het verst verwijderd ligt vanaf s? Gebruik een variant op
   A. Depth-first search
   B. Breadth-first search
   C. Dijkstra's shortest-path algorithm
   D. Bellman-Ford's shortest-path algorithm

6. Huffman's algoritme voor het construeren van een prefix code maakt een binaire boom van de te coderen letters. Bij het construeren wordt de volgende greedy regel toegepast.
   A. De twee letters met de laagste frequentie komen naast elkaar onderin de boom.
   B. De letters worden gesorteerd op aflopende frequentie en daarna een voor een vanaf bovenaan in de boom gehangen: dus highest frequency first (hoogste frequentie eerst).
   C. De letters worden gesorteerd op oplopende frequentie en daarna een voor een vanaf bovenaan in de boom gehangen: dus lowest frequency first (laagste frequentie eerst).
   D. De te coderen letters worden zodanig in twee groepen verdeeld dat te coderen letters ongeveer even vaak in de ene groep zitten als in de andere. Het algoritme wordt dan recursief op beide groepen aangeroepen.

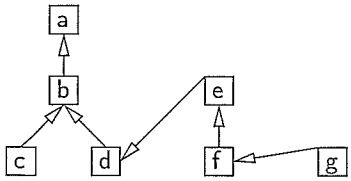
7. Gegeven is een snede (cut) (A, B) in een graaf G met op iedere kant andere kosten. Noem e de kant in de snedeverzameling (cut set) van (A, B) met de minste kosten. Wat kunnen we nu concluderen?
   A. Voor e geldt de cycle property.
   B. Alle minimale opspannende bomen (MSTs) bevatten e.
   C. Er is precies één minimale opspannende boom die e bevat.
   D. Er is een kant f in de snedeverzameling die in geen enkele minimale opspannende boom voorkomt.

8. Stel dat we de meest efficiënte pointer-based implementatie van union-find gebruiken met padcompressie en op een bepaald moment ziet de datastructuur voor de elementen a t/m g er uit als in figuur 1.

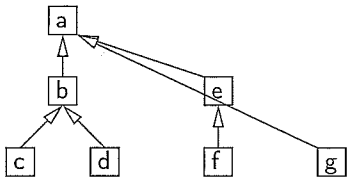


Figuur 1: In deze union-find datastructuur geven de pijlen pointers aan en de vierkanten de data.

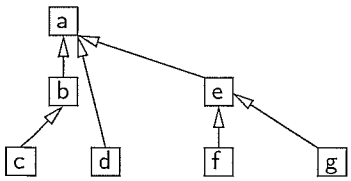
We voeren vervolgens de volgende twee operaties uit: union(find(d),find(f)) en union(find(g),find(f)). Welk van de figuren geeft dan correct het eindresultaat weer?



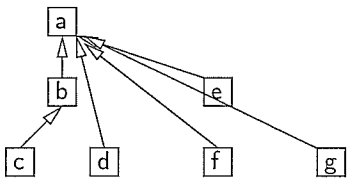
A.



B.



C.



D.

9. Gegeven zijn twee integers x en y van ieder n bits. We definiëren x_1 als de $\lfloor \frac{n}{2} \rfloor$ meest significante bits van x en x_0 als de $\lceil \frac{n}{2} \rceil$ minst significante bits van x. Analoog definiëren we y_1 en y_0. Het algoritme om in $O\left(n^{\log_2 3}\right)$ tijd twee integers te vermenigvuldigen is gebaseerd op de volgende herschrijving van een vermenigvuldiging van x en y en een recursieve aanroep van de hierin voorkomende vermenigvuldigingen.
   A. $xy = \left(x_1 \cdot 2^{\frac{n}{2}} + x_0\right)\left(y_1 \cdot 2^{\frac{n}{2}} + y_0\right)$
   B. $xy = x_1 y_1 \cdot 2^n + (x_1 y + x_0 y) \cdot 2^{\frac{n}{2}}$
   C. $xy = x_1 y_1 \cdot 2^n + (x_1 y_0 + x_0 y_1) \cdot 2^{\frac{n}{2}} + x_0 y_0$
   D. $xy = x_1 y_1 \cdot 2^n + ((x_1 + x_0)(y_1 + y_0) - x_1 y_1 - x_0 y_0) \cdot 2^{\frac{n}{2}} + x_0 y_0$

10. Welke van de volgende operaties in de functie voor het bepalen van de twee dichtstbijzijnde punten in het platte vlak (Closest-Pair-Of-Points) draagt het meest bij aan de looptijd?

    A. Het bepalen van een lijn L zodat er evenveel punten links van L als rechts van L zijn.

    B. Het bepalen van de punten A die dichterbij L zitten dan een bepaalde afstand δ.

    C. Het sorteren op y-coordinaat van de punten die dichterbij L zitten dan de afstand δ.

    D. Het controleren of een punt links van L in A dichterbij een punt rechts van L in A zit dan δ.

11. Voor het berekenen van de afstand tussen twee strings, zoeken we naar de beste alignment: gegeven strings X en Y van lengte m en n, match posities i van X met posities j van Y zodanig dat de boete voor mismatches in een alignment geminimaliseerd wordt. Laat een alignment M een verzameling van koppels (i,j) zijn waarbij $i \in \{1,\ldots,m\}$ en $j \in \{1,\ldots,n\}$, iedere i en iedere j hooguit eenmaal voorkomt en er geen kruizingen zijn: als $(i,j),(i',j') \in M$ en $i < i'$ dan $j < j'$. De boete voor mismatches in een alignment M wordt als volgt berekend:

    1. een boete $\delta \geq 0$ voor iedere positie $i \in \{1,\ldots,m\}$ of $j \in \{1,\ldots,n\}$ die in geen enkel koppel in M voorkomt, plus
    2. een boete $\alpha_{i,j} \geq 0$ voor iedere $(i,j) \in M$ waarvoor karakter i in X ongelijk is aan karakter j in Y, en waar $\alpha_{i,j} = 0$ als karakter i in X gelijk is aan karakter j in Y.

    Met welke van de volgende formules voor OPT kun je (door middel van dynamisch programmeren) efficiënt bepalen wat de boete is voor de beste alignment van X tot en met positie i met Y tot en met positie j?

    $\mathsf{OPT}(0,j) = j\delta$ en $\mathsf{OPT}(i,0) = i\delta$ en voor $i,j > 0$:

    A. $\mathsf{OPT}(i,j) = \min\{\alpha_{i,j} + \mathsf{OPT}(i-1,j-1),\ \delta + \mathsf{OPT}(i-1,j-1)\}$

    B. $\mathsf{OPT}(i,j) = \min_{i \leq t \leq j}\{\alpha_{t,t} + \mathsf{OPT}(t,j-t),\ \delta + \mathsf{OPT}(t,j),\ \delta + \mathsf{OPT}(t,j-t)\}$

    C. $\mathsf{OPT}(i,j) = \min\{\alpha_{i,j} + \mathsf{OPT}(i-1,j-1),\ \delta + \mathsf{OPT}(i-1,j),\ \delta + \mathsf{OPT}(i,j-1)\}$

    D. $\mathsf{OPT}(i,j) = \min_{1 \leq t_1 \leq i}\{\min_{1 \leq t_2 \leq j}\{\alpha_{t_1,t_2} + \mathsf{OPT}(t_1,t_2),\ \delta + \mathsf{OPT}(t_1,j),\ \delta + \mathsf{OPT}(i,t_2)\}\}$

12. Gegeven een verzameling punten $(x_1,y_1),\ldots,(x_n,y_n)$. We definiëren $e_{i,j}$ als de (minimale) fout die wordt verkregen als de punten i tot en met j worden benaderd met één lijnstuk.[1] We definiëren C als de kosten voor het introduceren van een extra lijnstuk. Welke van de volgende formules voor OPT geeft dan correct weer wat de minimale kosten zijn voor het benaderen van de punten 1 tot j met behulp van lijnstukken?

    $\mathsf{OPT}(0) = 0$, en voor $j > 0$

    A. $\mathsf{OPT}(j) = e_{1,j} + C + \mathsf{OPT}(\frac{j}{2})$

    B. $\mathsf{OPT}(j) = e_{j-1,j} + C + \mathsf{OPT}(j-1)$

    C. $\mathsf{OPT}(j) = \min_{1 \leq i \leq j}(e_{i,j} + C + \mathsf{OPT}(i-1))$

    D. $\mathsf{OPT}(j) = \min(e_{j-1,j} + C + \mathsf{OPT}(j-1),\ \mathsf{OPT}(j-1))$

[1] Hiertoe wordt de minimale som van de kwadraten van de Euclidische afstanden van de punten i tot en met j tot het lijnstuk berekend.

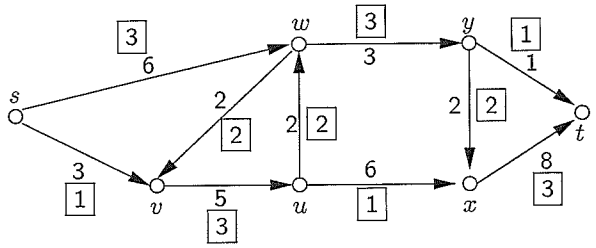
13. Bekijk het stroomnetwerk in figuur 2 en teken de restgraaf (residual graph). Met hoeveel kan de stroom hier nog verhoogd worden door het vinden van stroomverhogende paden (augmenting paths) in de restgraaf?

    A. met 2

    B. met 3

    C. met 4

    D. met 5



Figuur 2: Een stroomnetwerk. De getallen bij de kanten zijn de capaciteiten. De stroom op een kant is weergegeven in een vierkantje.

14. Welke van de volgende beweringen over een stroomnetwerk is geldig?

    A. De waarde van de stroom is gelijk aan de capaciteit van iedere willekeurige $s-t$ snede (cut).

    B. De waarde van de maximale stroom is gelijk aan de capaciteit van de grootste $s-t$ snede (cut).

    C. De waarde van de maximale stroom is gelijk aan de rest-capaciteit over iedere willekeurige $s-t$ snede (cut).

    D. De waarde van de stroom is gelijk aan de netto waarde van de stroom door iedere willekeurige $s-t$ snede (cut).

15. Gegeven een stroomnetwerk met n knopen en m kanten met geheeltallige capaciteiten kleiner dan of gelijk aan C. Van het algoritme van Ford-Fulkerson bestaat ook een variant die een stuk efficiënter kan zijn door het gebruik van een schaling (scaling) techniek met een factor Δ. Wat is de krapste bovengrens op de looptijd van het algoritme met en zonder schaling?

    A. Zonder schaling is $O(mC)$, met schaling is $O(m \log C)$.

    B. Zonder schaling is $O(mC)$, met schaling is $O(m^2 \log C)$.

    C. Zonder schaling is $O(m^2C)$, met schaling is $O(m \log C)$.

    D. Zonder schaling is $O(m^2C)$, met schaling is $O(m^2 \log C)$.

Open vragen

1. Geef een asymptotische boven- en ondergrens voor $T(n)$ in ieder van de volgende recurrente betrekkingen. Maak je grenzen zo krap (tight) mogelijk en geef aan hoe je aan je antwoord komt (in maximaal 5 regels per recurrente betrekking).
   (a) (3 punten) $T(n) = 3T(n/3) + n$
   (b) (3 punten) $T(n) = 4T(n/2) + n^2\sqrt{n}$

2. Een server heeft n gerelateerde rekentaken. Na een voorbereidingstijd van t_i minuten kan iedere taak $i \in \{1, \ldots, n\}$ verder parallel opgelost worden in p_i minuten. We laten het sequentiële deel op het bestaande mainframe doorrekenen en de parallelle berekeningen op een computer die minstens zoveel cores heeft als er deelproblemen zijn.

   Laat s_i de starttijd van taak i zijn. Het doel is om deze starttijden zo te bepalen dat de voorbereidingstijd van geen enkel paar taken overlapt en alle taken zo snel mogelijk klaar zijn. Bijvoorbeeld, stel dat we twee taken hebben met $t_1 = 6$, $p_1 = 5$ en $t_2 = 4$ en $p_2 = 8$ en stel dat we eerst taak 2 doen en dan taak 1. In dat geval zijn de starttijden $s_2 = 0$ en $s_1 = 4$, is taak 2 klaar na $4+8 = 12$ minuten en taak 1 na $4+6+5 = 15$ minuten. Na 15 minuten zijn dus alle taken klaar.
   (a) (3 punten) Geef een algoritme voor het bepalen van een rooster dat de laatste eindtijd minimaliseert (in maximaal 6 regels). Bepaal in het algoritme ook de starttijden $s_1, s_2, \ldots, s_n$.
   (b) (1 punt) Geef een krappe bovengrens op de looptijd van je algoritme.
   (c) (3 punten) Gegeven zijn de optimale starttijden o_i voor $1 \leq i \leq n$. Bewijs dat je algoritme uit de voorgaande vraag optimaal is (in maximaal 15 regels). Geef aan welke bewijstechniek je gebruikt.

3. Om efficient een DNA sequentie $b_1 b_2 b_3 \ldots b_n$ te analyseren wordt deze eerst opgedeeld in segmenten. Er is een tabel *score*$[i, j]$ die aangeeft in hoeverre de basen $b_i, b_{i+1}, \ldots, b_j$ bij elkaar horen (altijd positief, hoger is beter).[2] De score van een opdeling in segmenten is de som van de scores over alle segmenten. Segmenten zijn aaneensluitend, mogen niet overlappen en iedere base moet voorkomen in een segment.
   (a) (3 punten) Geef een formule voor een recursieve functie OPT die de maximale score (voor het opdelen in segmenten) bepaalt van een DNA sequentie $b_1 b_2 b_3 \ldots b_n$. Geef ook aan welke waarde(n) je als argument(en) meegeeft bij de eerste aanroep van je recursieve functie en beschrijf het deelprobleem dat wordt opgelost door je functie kort in woorden (in maximaal 8 regels).
   (b) (3 punten) Geef *gedetailleerde* pseudocode (dus geen gebruik van bibliotheekfunctieaanroepen) van een efficiënt iteratief algoritme voor het vinden van de score van de optimale oplossing met behulp van dynamisch programmeren op basis van je functie uit a) (in maximaal 15 regels).
   (c) (1 punt) Geef aan wat een krappe bovengrens is op de looptijd van je algoritme en hoe je hieraan komt (in maximaal 3 regels).
   (d) (2 punten) Geef pseudocode voor een algoritme dat (na het uitvoeren van het vorige algoritme) de beginindices van alle segmenten print voor de optimale opdeling in segmenten (in maximaal 10 regels).

(Sla om voor de laatste opgave.)

[2] Onder andere wordt hierin rekening gehouden met de frequentie van de verschillende aminozuren.

4. De Noorse Spoorwegmaatschappij (NS) moet beslissen of het voldoende materieel heeft om aan de eisen van de regering te voldoen. Het Noorse spoorwegnet is opgedeeld in n trajecten die ieder precies een dag duren om af te leggen. Voor ieder traject $i \in \{1, 2, \ldots, n\}$ heeft de Noorse regering een minimaal (c_i^-) en maximaal (c_i^+) aantal treinen per dag ingesteld. Een complicerende factor is dat niet iedere trein geschikt is voor ieder traject (bijvoorbeeld elektrisch of niet, vermogen mbt de helling, etc.). Voor ieder treintype $j \in \{1, 2, \ldots, k\}$ is het aantal t_j bekend dat de NS van dit type ter beschikking heeft. Uiteraard is ook voor ieder traject i bekend welke treintypes j hiervoor geschikt zijn. De vraag is of de NS voldoende materieel heeft om aan alle eisen van de regering te voldoen.

   (a) (3 punten) Modelleer dit probleem door middel van een kringloop met ondergrenzen (circulation with lower bounds) zodat er een geldige kringloop bestaat dan en slechts dan als de NS voldoende materieel heeft. Beschrijf wat de knopen zijn, wat de kanten zijn en wat de capaciteiten op de kanten zijn (in maximaal 8 regels). Maak ook een tekening van een eenvoudig voorbeeld met drie trajecten en twee treintypen.

   (b) (3 punten) Leg uit hoe je een kringloop met ondergrenzen in twee stappen kunt vertalen naar een equivalent netwerkstroomprobleem dat direct opgelost kan worden met het algoritme van Ford-Fulkerson (in maximaal 8 regels) en illustreer dit met je voorbeeld uit a).

   (c) (2 punten) Leg uit hoe je kan afleiden of de NS voldoende materieel ter beschikking heeft (in maximaal 3 regels).

Einde tentamenopgaven