TECHNISCHE UNIVERSITEIT DELFT
Faculteit Elektrotechniek, Wiskunde en Informatica

TU Delft

Tentamen TI2300 (oud)/TI2305 (nieuw) Algoritmiek[1]

29 januari 2014, 14.00-17.00

- Het gebruik van boek, aantekeningen of rekenmachine tijdens dit tentamen is ***niet*** toegestaan.
- Dit tentamen heeft 4 open vragen in totaal goed voor 6/10 van je cijfer (onderverdeeld in 30 punten) en 15 meerkeuzevragen goed voor 3/10 van je cijfer. Je krijgt sowieso 1/10 kado. Merk op dat je eindcijfer voor dit vak ook voor $\frac{1}{3}$ uit het onafgeronde gemiddelde van het practicum bestaat (mits beide ≥ 6.0).
- Voor de open vragen heb je ongeveer twee uur nodig:
  - Geef antwoord in correct Nederlands of Engels en schrijf leesbaar (gebruik eerst potlood of 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 de vakcode en het aantal ingeleverde bladen voor de open vragen aan op (tenminste) de eerste pagina.
- 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 tentamenstof bestaat uit Chapters 4–7 uit Algorithm Design van Kleinberg en Tardos (behalve secties 4.4, 4.9*, 5.6, 6.8–6.10*, 7.4*, en 7.13*), en bovendien de notities over de Master Method en over bewijstechnieken (Proving guide).
- Totaal aantal pagina's (zonder dit voorblad): 6.

[1] Zet de door jouw gewenste code op al je antwoordvellen.

Open vragen

1. Gegeven zijn een aantal verzoeken (requests) voor het gebruik van een zaal (room) vanaf een gegeven begintijd tot (exclusief) een gegeven eindtijd. We willen al deze verzoeken toewijzen (allocate), maar zo min mogelijk zalen gebruiken (minimise the number of rooms).
   (a) (4 punten) Introduceer notatie voor de begin- en eindtijd van de verzoeken en nummer de zalen. Geef een (greedy) algoritme voor het toewijzen van een zaal aan ieder verzoek, zodanig dat er geen twee verzoeken die overlappen in tijd aan dezelfde zaal worden toegewezen (no overlap), en zodanig dat er zo min mogelijk zalen worden toegewezen. Je mag ervan uit gaan dat er altijd genoeg zalen beschikbaar zijn. Laat je functie het aantal benodigde zalen teruggeven (return) (functie in maximaal 10 regels).
   (b) (1 punt) Geef een krappe asymptotische bovengrens (tight upper bound) op de looptijd (run time) van je algoritme. Leg uit (explain) hoe je aan je antwoord komt.

2. (4 punten) Laat een graaf gegeven zijn met verschillende (distinct) kosten op iedere kant (edge). Bewijs dat er één unieke minimale opspannende boom is (proof that there is a unique MST).
   - Geef aan welke bewijstechniek je gebruikt.
   - Introduceer je notatie.
   - Maak duidelijk wanneer je een aanname doet.
   - Geef een duidelijke verwijzing als je een stelling gebruikt.
   - Zorg dat iedere stap in je bewijs volgt uit voorgaande beweringen.

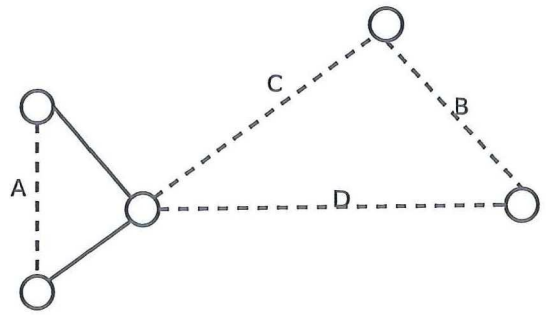
3. (3 punten) Gegeven is een array met verschillende getallen (distinct numbers). Het is bekend dat de getallen strikt oplopen (increase) naar een piek (peak) en daarna weer strikt aflopen (decrease). Geef een $O(\log(n))$ *recursief* algoritme dat de waarde van de piek bepaalt (return the peak).

4. (3 punten) Geef een krappe asymptotische boven- en ondergrens voor $T(n)$ in de volgende recurrente betrekking. Geef aan hoe je aan je antwoord komt (in maximaal 5 regels).
   $T(n) = 9T(n/3) + n^3 \log n$

5. In de nieuwe versie van een navigatiesysteem wil men rekening houden met vaak voorkomende vertragingen (delays) als gevolg van drukte in een netwerk (V, E). (De achterliggende reden is dat op sommige tijden van de dag een alternatieve route mogelijk sneller is.) Hiertoe is voor ieder kwartier van de dag $t \in [0, 95]$ de verwachte reistijd (travel time) $w(e, t) > 0$ over ieder stuk weg $e \in E$ verzameld en opgeslagen. De vraag aan jou is om gegeven deze informatie, een vertrek knoop $s \in V$, vertrektijd (departure time) t_0, en doelknoop $d \in V$ de duur (length) van de snelste (shortest time) route van s naar d te bepalen met behulp van dynamisch programmeren. Als het niet mogelijk is om dezelfde dag nog je doel te bereiken, geef dan ∞ als reistijd.
   (a) (4 punten) Geef een recursieve formule voor het bepalen van de duur van de snelste route naar d vanaf een willekeurige knoop v en vertrektijd t. Geef ook aan met welk(e) argument(en) de eerste aanroep gedaan wordt om de lengte van de snelste route vanaf s te bepalen en leg je formule kort uit.
   (b) (3 punten) Geef pseudocode voor een efficiënt iteratief algoritme dat de duur van de snelste route berekent met behulp van dynamisch programmeren en de functie van de vorige vraag. Je mag alleen gebruik maken van opdrachten (statements) die standaard in een taal als Java beschikbaar zijn.
   (c) (1 punt) Geef een krappe asymptotische bovengrens (tight upper bound) op de looptijd (run time) van je algoritme uit de vorige vraag.

Meerkeuzevragen

1. De doorgetrokken kanten (solid edges) in de onderstaande graaf zijn de gekozen kanten (selected edges) tijdens een uitvoering van het algoritme van Kruskal. De gewichten van een kant zijn evenredig met de afstand tussen de knopen op papier (the edge weights are proportional to their length on paper). Welk van de gestippelde (dashed) kanten zal dit algoritme als eerstvolgende (next) selecteren?



   A. kant A
   B. kant B
   C. kant C
   D. kant D

2. Voor een triathlon bestaande uit eerst zwemmen, dan fietsen en dan rennen, is voor iedere deelnemer i bekend wat de verwachte tijd is voor deze drie onderdelen (respectievelijk z_i, f_i en r_i). Enkel het zwemmen kan niet door de deelnemers tegelijk gedaan worden (only swimming is not in parallel); de volgende deelnemer begint met zwemmen precies als de vorige klaar is. In welke volgorde moeten de deelnemers starten met het zwemmen om de verwachte duur (expected length) van de gehele triathlon te minimaliseren?
   A. In aflopende (decreasing) volgorde van zwemtijd z_i.
   B. In oplopende (increasing) volgorde van zwemtijd z_i.
   C. In aflopende (decreasing) volgorde van fiets- en rentijd $f_i + r_i$.
   D. In oplopende (increasing) volgorde van fiets- en rentijd $f_i + r_i$.

3. Op welke manier kun je efficiënt een k-clustering met maximale afstand (max spacing) vinden van een verzameling punten in twee dimensies?
   A. Op basis van dynamisch programmeren.
   B. Op basis van het algoritme van Kruskal.
   C. Op basis van het algoritme van Bellman-Ford.
   D. Op basis van het closest-pair-of-points algoritme.

4. Welke van de volgende coderingen is een *prefix* codering van minimale lengte voor *abccdebeeece*?
   A. $a \to 000, b \to 01, c \to 10, d \to 001, e \to 11$
   B. $a \to 000, b \to 001, c \to 01, d \to 011, e \to 1$
   C. $a \to 000, b \to 010, c \to 001, d \to 011, e \to 1$
   D. $a \to 0001, b \to 001, c \to 01, d \to 0000, e \to 1$

5. Laat een recursief algoritme gegeven zijn met een looptijd beschreven door de recurrente betrekking $T(n) = aT(\frac{n}{b}) + f(n)$ voor een of andere functie f. In welk geval moet de *regularity condition* nog gecontroleerd worden voordat de Master Methode toegepast kan worden om de looptijd te bepalen?
   A. Als $f(n)$ is $\Omega(n^{\log_b a+\epsilon})$ voor een $\epsilon > 0$
   B. Als $f(n)$ is $O(n^{\log_b a-\epsilon})$ voor een $\epsilon > 0$
   C. Als $n^{\log_b a}$ is $\Omega(c \cdot f(n))$ voor een $c < 1$
   D. Als $n^{\log_b a}$ is $O(c \cdot f(n))$ voor een $c > 0$

9. 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 $j \in \{1,\ldots,n\}$ en iedere i en iedere j hooguit eenmaal voorkomt. De boete (penalty) 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?

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

   A. $\mathrm{OPT}(i,j) = \min\{\alpha_{i,j} + \mathrm{OPT}(i-1,j-1),\ \delta + \mathrm{OPT}(i-1,j-1)\}$
   B. $\mathrm{OPT}(i,j) = \min_{i \leq t \leq j}\{\alpha_{t,t} + \mathrm{OPT}(t,j-t),\ \delta + \mathrm{OPT}(t,j),\ \delta + \mathrm{OPT}(t,j-t)\}$
   C. $\mathrm{OPT}(i,j) = \min\{\alpha_{i,j} + \mathrm{OPT}(i-1,j-1),\ \delta + \mathrm{OPT}(i-1,j),\ \delta + \mathrm{OPT}(i,j-1)\}$
   D. $\mathrm{OPT}(i,j) = \min_{1 \leq t_1 \leq i}\{\min_{1 \leq t_2 \leq j}\{\alpha_{t_1,t_2} + \mathrm{OPT}(t_1,t_2),\ \delta + \mathrm{OPT}(t_1,j),\ \delta + \mathrm{OPT}(i,t_2)\}\}$

10. Om een lange tekst beter leesbaar te maken, delen we die op in paragrafen. Gegeven is een tabel met voor iedere mogelijke paragraaf van zin i tot en met zin j de score $p_{i,j}$ van die paragraaf. De score van een opdeling in paragrafen is dan de som van de scores van die paragrafen. Met welke van de volgende recursieve formules voor OPT kun je (door middel van dynamisch programmeren) efficiënt bepalen wat de hoogst mogelijke score is voor het opdelen van de tekst in paragrafen?

    A. $\mathrm{OPT}(0) = 0$ en voor $j > 0$: $\mathrm{OPT}(j) = \max_{1 \leq i \leq j}(p_{i,j} + \mathrm{OPT}(i-1))$
    B. $\mathrm{OPT}(0) = 0$ en voor $j > 0$: $\mathrm{OPT}(j) = \max(p_{j-1,j} + \mathrm{OPT}(j-2),\ \mathrm{OPT}(j-1))$
    C. $\mathrm{OPT}(0,j) = 0$ en $\mathrm{OPT}(i,0) = 0$ en voor $i,j > 0$:
       $\mathrm{OPT}(i,j) = \max_{1 \leq i \leq j}(\mathrm{OPT}(i-1,j) + p_{i,j})$
    D. $\mathrm{OPT}(0,j) = 0$ en $\mathrm{OPT}(i,0) = 0$ en voor $i,j > 0$:
       $\mathrm{OPT}(i,j) = \max(\mathrm{OPT}(i-1,j), \mathrm{OPT}(i-1,j-i) + p_{i,j})$

11. Gegeven een verzameling punten $(x_1,y_1),\ldots,(x_n,y_n)$ gesorteerd op x coördinaat. Gegeven is ook voor iedere i,j de (minimale) fout $e_{i,j}$ die wordt verkregen als de punten i tot en met j worden benaderd met één lijnstuk.[2] We definiëren C als de kosten voor het introduceren van een extra lijnstuk. De volgende formule voor OPT geeft weer wat de minimale kosten zijn voor het benaderen van de punten 1 tot j met behulp van lijnstukken.

    $\mathrm{OPT}(0) = 0$, en voor $j > 0$: $\mathrm{OPT}(j) = \min_{1 \leq i \leq j}(e_{i,j} + C + \mathrm{OPT}(i-1))$.

    Wat is een asymptotische krappe grens op de looptijd van een efficiënte dynamische programmeren implementatie van deze functie?

    A. $O(n)$
    B. $O(n \log n)$
    C. $O(n^2)$
    D. $O(n^2 \log n)$

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

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