TECHNISCHE UNIVERSITEIT DELFT
Faculteit Elektrotechniek, Wiskunde en Informatica

TU Delft

Oefententamen TI2306 Algoritmiek

januari 2016

- Het gebruik van boek, aantekeningen of rekenmachine tijdens dit tentamen is ***niet*** toegestaan.
- Dit tentamen heeft 7 open vragen (en daarover verdeeld 65 punten) in totaal goed voor 6.5/10 van je cijfer en 15 meerkeuzevragen goed voor 3.5/10 van je cijfer. 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).
- Als je voor de huiswerkopgaven van dit collegejaar gemiddeld minstens een 5.8 hebt, dan wordt je cijfer voor dit tentamen met 0.5 verhoogd (maar nooit hoger dan 10.0).
- 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 4.9*, 5.6, 6.10*, 7.4*, en 7.13*), en bovendien de notities over de Master Method en over bewijstechnieken (Proving guide), ook toegepast op eenvoudige graafalgoritmen (Chapter 3).
- Totaal aantal pagina's (zonder dit voorblad): 8.

Open vragen

1. Autofabriek Auto-op-Maat gaat vanwege de kredietcrisis enkele weken dicht. Vanwege een overschot aan personeel lopen de kosten te hoog op. Voordat de fabriek dicht gaat moeten er nog n auto's gemaakt worden. Het in elkaar zetten gebeurt grotendeels door een machine, maar daarna moet er nog heel wat maatwerk met de hand gedaan worden. Als er genoeg personeel is (en dat is er), kan het handwerk aan verschillende auto's parallel gebeuren. Voor iedere auto i zijn de benodigde tijden bekend: de machine heeft eerst m_i uur nodig en daarna is er nog h_i uur handwerk vereist. Laat s_i de starttijd zijn waarop de machine aan auto i begint. In deze opgave word je gevraagd de optimale starttijden s_i voor iedere auto $i \in \{1, \ldots, n\}$ te bepalen zodat alle n auto's zo snel mogelijk af zijn en de fabriek (tijdelijk) dicht kan.

   (a) (2 punten) Beschouw het voorbeeld van drie auto's (dus $n = 3$) in de onderstaande tabel. Het rooster waarbij de machine de auto's produceert in volgorde van hun nummer leidt tot een eindtijd van 6 voor auto 1, 17 voor auto 2 en 25 voor auto 3. Dus na 25 uur zijn alle auto's klaar. Laat zien in welke volgorde deze 3 auto's het snelste klaar zijn en geef de tijd waarop iedere auto dan klaar is (in maximaal 3 regels).

auto i :	m_i	h_i
1	2	4
2	5	10
3	11	7

   Antwoord: De beste volgorde is 2, 3, 1: de eindtijden zijn dan 15, 23 en 22.

   (b) (6 punten) Geef de pseudocode voor een efficiënt algoritme dat de starttijd s_i voor iedere auto i in een array S invult zodat de fabriek zo snel mogelijk dicht kan (in maximaal 8 regels).

   Antwoord: (Dit lijkt op 4.7 uit het boek.)

```
1 sorteer de auto's aflopend op h_i
2 S[1] := 0
3 for i := 2 to n do
4    S[i] := S[i-1]+m_{i-1}
5 end
```

   (c) (2 punten) Geef een krappe bovengrens op de looptijd van je algoritme. (Leg ook kort uit waarom.)

   Antwoord: Dit algoritme loopt in $O(n \log n)$.

2. (8 punten) We zeggen dat een binaire boom *vol* (*full*) is als iedere interne knoop twee kinderen heeft. Bewijs dat de boom van een optimale prefix codering vol is.

Antwoord: Suppose T is binary tree of optimal prefix code and is not full. (1 punt: uit ongerijmde)
Thus there is a node u (without label) with only one child v.

- Case 1: u is the root; create TÕ: delete u and use v as the root
- Case 2: u is not the root; create TÕ: let w be the parent of u delete u and make v be a child of w in place of u

In both cases the number of bits needed to encode any leaf in the subtree of v is decreased. The rest of the tree is not affected. Clearly this new tree TÕ has a smaller ABL than T. Contradiction. (1 punt)

3. In het algoritme voor het vinden van de twee punten die het dichtst bij elkaar liggen (closest pair of points) worden de punten binnen een bandbreedte van δ van een middellijn $x = L$ bekeken (gesorteerd op y-coördinaat).

   (a) (3 punten) Wat is de betekenis van deze δ en hoe wordt die bepaald?

   Antwoord: δ is het minimum van de kortste afstand tussen twee punten links van de lijn en de kortste afstand tussen twee punten rechts van de lijn. Beide worden bepaald door een recursieve aanroep (op respectievelijk het linker- en rechter deel van de punten).

   (b) (6 punten) Tot welke andere punten wordt in dit algoritme voor een punt i binnen de bandbreedte de afstand bekeken om zo tot het tweetal te komen met de kleinste onderlinge afstand? Leg uit waarom dit tot een correct antwoord leidt.

   Antwoord:

   - Ieder punt i moet vergeleken worden met de 7, 11 (of 15) eerstvolgende punten in deze gesorteerde lijst. (1 punt)
   - Voor alle punten vanaf het 12e geldt dat de afstand tot i groter dan δ: immers, er past zowel links als rechts van L in ieder vierkantje van 0.5δ slechts 1 punt, omdat zowel links als rechts de onderlinge afstanden minimaal δ zijn. Na 11 volgende punten zijn er minstens 2 rijen van deze vierkantjes overgeslagen en is de afstand tot ieder volgend punt dus strikt meer dan δ. (1 punt)
   - 1 punt voor alleen "binnen δ van L"met uitleg

4. (8 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).

   $T(n) = 6T(n/3) + n^2 \log n$

   Antwoord: $\log_b a = \log_3 6 < 2$, dus $f(n) = n^2 \log n$ is $\Omega\left(n^{\log_3 6+\epsilon}\right)$ voor een $\epsilon > 0$. Dit is geval 3 van de Master Method. Check regularity condition: voor $6\left(\frac{n}{3}\right)^2 \log(n/3) = \frac{6}{9}n^2(\log n - \log 3) \leq cn^2 \log n$ bestaat een constante c met $0 < c < 1$, namelijk $c = \frac{2}{3}$. Dus $T(n)$ is $\Theta\left(n^2 \log n\right)$. Geen regularity condition: 2 punten. Wel noemen, maar beweren dat $f(n)$ polynomiaal is: 2.5 punten.

5. Vele astronomen willen de nieuwe telescoop Integral (International Gamma-Ray Astrophysics Laboratory) gebruiken om hun observaties te doen. Hiertoe dienen ze observatieverzoeken in bij de ESA voor de periode van 0 tot T. Een observatieverzoek i bestaat uit een begintijd s_i, een eindtijd f_i (met $0 \leq s_i < f_i \leq T$) en de geschatte waarde van de observatie $v_i \geq 0$ (als niet aan s_i en f_i voldaan wordt, is de waarde 0). Integral kan helaas maar één observatie tegelijk doen. Daarnaast moet voor iedere observatie i de telescoop in de juiste richting gebracht worden. Dit gaat gepaard met enige extra tijd die afhangt van de voorgaande observatie j. We gebruiken $t(j,i)$ voor de tijd die nodig is om de telescoop voor observatie i klaar te zetten nadat j is afgelopen (met $1 \leq j \leq n-1$). De tijd voor het klaarzetten van de eerste observatie wordt gegeven door $t(0,i)$. Er zijn n observatieverzoeken bij ESA binnengekomen. Deze zijn al gesorteerd op hun eindtijd. Het doel van deze opgave is om een selectie van observaties te bepalen met maximale totale waarde.

   (a) (2 punten) Beschouw de volgende situatie met drie verzoeken tot observatie, hun begin- en eindtijden, hun geschatte waarde, en de tijd die nodig is om van de ene observatie naar de andere te gaan. Wat is hier de optimale selectie van observaties en wat is de bijbehorende totale waarde? (in maximaal 3 regels).

observatie i :	s_i	f_i	v_i
1	2	4	3
2	3	9	8
3	5	10	6

tijd $t(j,i)$ van j \ naar i	1	2	3
0	1	1	1
1	-	2	2
2	-	-	2

Antwoord: Geen van de observaties kan gecombineerd worden. De beste selectie is dus om alleen observatie 2 te doen. De waarde is dan 8.

   (b) (8 punten) Geef een recursieve formule voor de functie $\mathsf{OPT}(j)$ die de maximaal haalbare totale waarde beschrijft voor een keuze uit de eerste j observaties (in maximaal 5 regels). Gebruik zo nodig een hulpfunctie. Als je nieuwe variabelen of parameters introduceert, geef dan aan wat ze betekenen.

Antwoord: Definieer predecessor p van j als volgt: $p(j) = \max\{i \mid f_i + t(i,j) \leq s_j\}$
Beschouw dan de volgende formule:

$$\mathsf{OPT}(j) = \begin{cases} 0 & \text{als } j = 0 \\ \max\{\mathsf{OPT}(p(j)) + v_j,\ \mathsf{OPT}(j-1)\} & \text{anders} \end{cases}$$

Eventueel kan ook de volgende formule gebruikt worden in combinatie met $\mathsf{opt}(t) = \max_j \mathsf{OPT}(j) + v_j$

$$\mathsf{OPT}(j) = \begin{cases} 0 & \text{als } \{i \mid f_i + t(i,j) \leq s_j\} = \emptyset \\ \max_{i \in \{i \mid f_i + t(i,j) \leq s_j\}} \mathsf{OPT}(i) + v_i & \text{anders} \end{cases}$$

Als je geen rekening houdt met het feit dat twee observaties niet tegelijk kunnen, verlies je tenminste de helft van de punten.

6. Er is ook nog een ander plan om de fabriek Auto-op-Maat door de moeilijke tijd heen te helpen. Een aantal werknemers is al weg. Het lijkt namelijk zo dat als de resterende n werknemers voor 60% aanwezig zijn dat dan komende maand precies voldoende uren gemaakt worden om aan de afgenomen vraag van nu in totaal l auto's te voldoen. De vraag is echter of dit mogelijk is, aangezien sommige werknemers specialist zijn en voor sommige auto's meer specialistisch werk nodig is dan voor andere.

   Gegeven is een verzameling J van m verschillende specialistische taken. Laat voor iedere werknemer i zijn specialismen $J_i \subseteq J$ en zijn nieuwe aantal uren h_i gegeven zijn. Laat verder voor iedere auto k het aantal uren t_{kj} per specialistische taak j gegeven zijn. Het is al berekend dat het totale aantal uren $\sum_{1 \leq k \leq l} \sum_{1 \leq j \leq m} t_{kj} = \sum_{1 \leq i \leq n} h_i$. De vraag is echter of inderdaad voldoende uren voor iedere specialistische taak iemand aanwezig is om alle auto's te voltooien.

(a) (8 punten) Beschrijf hoe je dit probleem kunt modelleren door middel van een stroomnetwerk (flow network) of kringloop (circulation) (in maximaal 8 regels). Teken ook een voorbeeld van een dergelijk stroomnetwerk/kringloop met minimaal 2 werknemers en 3 specialismen.

Antwoord: Creëer voor iedere werknemer een knoop i. Creëer voor iedere specialistische taak een knoop j. Verbind i met j met een kant met capaciteit ∞ als $j \in J_i$. Creëer een knoop s en verbind die met iedere i met een kant met capaciteit h_i. Creëer een knoop t en verbind iedere knoop j met t met een kant met capaciteit $\sum_{1 \leq k \leq l} t_{kj}$.
In plaats hiervan is het ook mogelijk om een kringloop te maken. Het is ook mogelijk om de auto's mee te modelleren.
Als je geen capaciteiten hebt aangegeven, verlies je tenminste de helft van de punten.

(b) (2 punten) Leg uit hoe je met behulp van dit stroomnetwerk efficiënt kunt bepalen of er nog voldoende uren specialisme in huis is (in maximaal 3 regels).

Antwoord: Als de waarde van de maximale stroom f in dit netwerk gelijk is aan $\sum_{1 \leq i \leq n} h_i$ dan is dit plan haalbaar.

(c) (4 punten) Leg uit hoe je voor iedere werknemer kunt bepalen hoeveel uur die moet werken aan welke specialistische taken en beargumenteer dat een dergelijke toewijzing aan alle voorwaarden voldoet (in maximaal 8 regels).

Antwoord: De (maximale) stroom op de kanten van i naar j geven aan hoeveel uur werknemer i aan taak j moet besteden. Deze toewijzing voldoet aan de volgende voorwaarden:

- iedere werknemer werkt hooguit h_i uur omdat dit de capaciteit is op de kant s naar i en de stroom in knoop i behouden blijft,
- aan ieder specialisme j wordt precies $\sum_{1 \leq k \leq l} t_{kj}$ uur gewerkt; meer kan niet, want dit is de capaciteit op de (enige) kant van j naar t en de stroom in knoop j blijft behouden. Minder kan ook niet want de waarde van de stroom van s naar t is $\sum_{1 \leq k \leq l} \sum_{1 \leq j \leq m} t_{kj}$.

7. (6 punten) Gegeven is een stroom netwerk $G = (V, E)$, met capaciteiten $c : E \to \mathbb{N}$. Gegeven is een stroom $f : E \to \mathbb{N}$ op dit netwerk. Bewijs dat voor iedere willekeurige $s - t$ snede (A, B) geldt dat $v(f) \leq cap(A, B)$.

(Hints: 1. Geef duidelijk aan waar je de definities van $v(f)$ en $cap(A, B)$ gebruikt. 2. Je mag het flow value lemma gebruiken.)

Antwoord: Laat een willekeurige snede (A, B) gegeven zijn.

$$v(f) = \sum_{e \text{ out of } A} f(e) - \sum_{e \text{into } A} f(e) \leq \sum_{e \text{ out of } A} f(e)$$

(using flow value lemma and fact that for all e: $f(e) \geq 0$)

$$\sum_{e \text{ out of } A} f(e) \leq \sum_{e \text{ out of } A} c(e) = cap(A, B)$$

(by definition of capacity)

Omdat (A, B) willekeurig was, geldt dit voor iedere (A, B).

Meerkeuzevragen

1. Voor je volgende project moet je van n taken de looptijd meten. Voor ieder van de taken $i \in \{1, \ldots, n\}$ ben je naar verwachting t_i minuten bezig met het implementeren en daarna heeft die taak naar verwachting p_i minuten nodig om uitgevoerd te worden op een van de vele pc's op de Drebbelweg. Dit laatste kan parallel.

   In welke volgorde moet je de taken implementeren om zo snel mogelijk met het project klaar te zijn?

   A. Aflopend op p_i
   B. Oplopend op p_i
   C. Aflopend op t_i
   D. Oplopend op t_i

   Antwoord: A

2. Gegeven $f(n) = n^2$ en $g(n) = n \log n$. Welke van de volgende beweringen is waar?

   A. $f(n)$ is $O(g(n))$.
   B. $g(n)$ is $\Theta(f(n))$.
   C. $f(n)$ is $\Omega(g(n))$.
   D. $g(n)$ is $\Omega(f(n))$.

   Antwoord: C

3. Gegeven een ongerichte graaf zonder cycles met n knopen, bestaande uit c verbonden (connected) componenten. Hoeveel kanten heeft deze graaf?

   A. $n - 1$
   B. $n - c$
   C. $n - c - 1$
   D. Dat kun je niet uit deze gegevens afleiden.

   Antwoord: B

4. Gegeven een ongerichte verbonden graaf. Wat is de meest efficiënte correcte implementatie om het kortste pad van een knoop s naar een knoop t te vinden?

   A. Gebruik als datastructuur een linked list en zoek met behulp van depth-first search.
   B. Gebruik als datastructuur een linked list en zoek met behulp van breadth-first search.
   C. Gebruik als datastructuur een twee-dimensionale array en zoek met behulp van depth-first search.
   D. Gebruik als datastructuur een twee-dimensionale array en zoek met behulp van breadth-first search.

   Antwoord: B

5. Gegeven een gerichte verbonden graaf G en een knoop v zonder inkomende kanten. Welke van de volgende beweringen is waar?

   A. G is niet sterk verbonden.
   B. G is een gerichte acyclische graaf (DAG) als $G - \{v\}$ een DAG is.
   C. G heeft een topologische ordening als $G - \{v\}$ een topologische ordening heeft.
   D. Alle bovenstaande beweringen zijn waar.

Antwoord: D

6. Beschouw het volgende probleem. Gegeven een serie getallen $x_1, \ldots, x_n$ in oplopende volgorde zodat iedere twee opeenvolgende getallen maximaal d van elkaar verschillen. Geef een zo klein mogelijke deelverzameling van deze getallen waar tenminste x_1 en x_n in zitten en die ook ieder maximaal d van elkaar verschillen. Welk van de volgende algoritmen geef een correcte optimale oplossing voor dit probleem op de meest efficiënte manier?

   A. Begin bij x_1 en kies telkens het volgende getal uit de serie zo groot mogelijk, maar nog binnen de afstand d, totdat x_n is opgenomen.

   B. Bereken de verschillen tussen iedere twee opeenvolgende getallen. Sorteer deze verschillen in aflopende volgorde en kies dan getallen net zolang totdat zowel x_1 als x_n zijn opgenomen.

   C. Bekijk alle deelverzamelingen van deze getallen waar zowel x_1 als x_n zitten en waarin elke twee opeenvolgende getallen hooguit d verschillen. Kies hieruit de kleinste deelverzameling.

   D. Deel de verzameling in tweeën en los recursief het probleem voor het linkerdeel tot $x_{\lfloor \frac{n}{2}-1 \rfloor}$ en het rechterdeel vanaf $x_{\lfloor \frac{n}{2}+1 \rfloor}$ op. Als de afstand tussen het grootste getal links en het kleinste getal rechts meer dan d is, voeg ook $x_{\lfloor \frac{n}{2} \rfloor}$ toe.

Antwoord: A

7. Gegeven een snede (cut) (A, B) in een graaf G. Noem e de kant in de cut-set van (A, B) met de minste kosten. Wat kunnen we nu concluderen over e?

   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 geen enkele minimale opspannende boom die e bevat.

Antwoord: B

8. Gegeven een codering c van een verzameling karakters S. Wanneer is deze codering een optimale prefix codering?

   A. Als de prefix-boom vol is.

   B. Als de codering van ieder karakter een prefix is van ieder ander karakter.

   C. Als de twee karakters met de laagste frequentie in de onderste laag van de prefix-boom zitten.

   D. Geen van deze antwoorden is juist.

Antwoord: D

9. 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. $a \to 0001, b \to 001, c \to 01, d \to 0000, e \to 1$

Antwoord: D
Leerdoel: 2, greedy.
Niveau: Inzicht.

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.

Antwoord: C

11. Geef een zo krap (tight) mogelijke asymptotische bovengrens voor de volgende recurrente betrekking: $T(n) = 4T(n/2) + n\sqrt{n}$.
    A. $O(n \log n)$
    B. $O(n\sqrt{n})$
    C. $O(n\sqrt{n} \log n)$
    D. $O(n^2)$

Antwoord: D

12. 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 $j \in \{1, \ldots, n\}$. De boete voor mismatches in een alignment M wordt als volgt berekend:

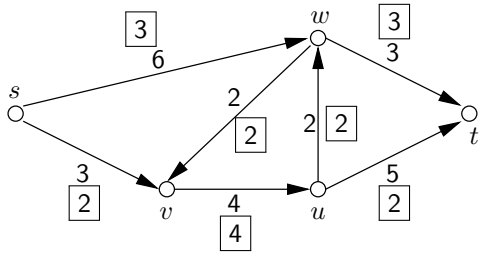
    1. een boete $\delta \geq 0$ voor iedere positie $i \in \{1, \ldots, m\}$ of $j \in \{1, \ldots, n\}$ die in geen enkel koppel in M voorkomt, plus
    2. een boete $\alpha_{i,j} \geq 0$ voor iedere $(i, j) \in M$ waarvoor karakter i in X ongelijk is aan karakter j in Y.

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

Antwoord: D

13. Bekijk het stroom netwerk in figuur 1. Met hoeveel kan de stroom nog verhoogd worden? Teken eventueel eerst de restgraaf (residual graph) op je kladpapier.
    A. met 1
    B. met 2
    C. met 3
    D. de stroom kan niet meer verhoogd worden



Figuur 1: Een stroomnetwerk. De getallen bij de kanten zijn de capaciteiten. De stroom op een kant is weergegeven in een vierkantje.

Antwoord: B

14. Gegeven een flow netwerk $G = (V, E)$ met voor iedere kant (edge) $e \in E$ een capaciteit c_e, maar ook een vraag (demand) $d_v \in \mathbb{Z}$ voor iedere knoop (vertex) $v \in V$, zodanig dat $\sum_{v \in V} d_v = 0$. Om te controleren of er een geldige flow in G bestaat zodanig dat de netto flow in iedere knoop gelijk is aan de demand van die knoop, maken we een netwerk G' als volgt. Neem netwerk G als basis voor G' en voeg daarna een super-source node s^* en een super-sink node t^* toe, en voor iedere knoop $v \in V$ met $d_v > 0$ een gerichte kant (v, t^*) met capaciteit d_v, en voor iedere knoop $v \in V$ met $d_v < 0$ een gerichte kant (s^*, v) met capaciteit $-d_v$. Wanneer bestaat er een geldige flow in G?

    A. Er bestaat een geldige flow in G dan en slechts dan als er een geldige flow in G' bestaat.

    B. Er bestaat een geldige flow in G dan en slechts dan als de maximale flow in G' gelijk is aan $\sum_{v \in \{v \in V | d_v > 0\}} d_v$.

    C. Er bestaat een geldige flow in G dan en slechts dan als de maximale flow in G' gelijk is aan de som van de vraag (demand) d_v van alle knopen $v \in V$.

    D. Er bestaat een geldige flow in G dan en slechts dan als er een geldige flow in G' bestaat die gelijk is aan de som van de vraag (demand) d_v van alle knopen $v \in V$.

Antwoord: B

15. Wat is de meest krappe (tight) worst-case bovengrens voor de looptijd van het Scaling Max-Flow algoritme op een graaf met n knopen, m kanten en een maximale stroom van c^*?

    A. $O\left(m^2 \log c^*\right)$

    B. $O\left(n^2 \log mc^*\right)$

    C. $O\left(m^2 \log nc^*\right)$

    D. $O\left(mn \log nc^*\right)$

Antwoord: A

Einde tentamenopgaven