

Antwoorden: Algorithms & Data Structures

- (a) Het sorteren van een array ter lengte 1 of 2 kost 1 vergelijking. Voor arrays met meer elementen vinden er 3 recursieve aanroepen plaats, ieder met een array ter lengte $\frac{2}{3}$ van de originele lengte. De niet-recursieve kosten in dit geval zijn 1, daar er 1 vergelijking van A-elementen plaatsvindt. Dit leidt tot de volgende recurrente betrekking voor $W(n)$:

$$\begin{aligned} W(n) &= 1 && \text{voor } n < 3 \\ W(n) &= 3 \cdot W\left(\frac{2n}{3}\right) + 1 && \text{voor } n \geq 3 \end{aligned}$$

Voor het bepalen van de worst-case tijdscomplexiteit passen we het Master theorema toe. Er volgt dat $a = 3$, $b = \frac{3}{2}$, $f(n) = 1$, en $E = \log a / \log b = \log 3 / \log \frac{3}{2}$. Aangezien $f(n) \in O(n^{E-\epsilon})$ voor bijvoorbeeld $\epsilon = \frac{1}{2}$ is de eerste case van het Master theorema van toepassing, en geldt dat $W(n) \in \Theta(n^E)$.

- (b) Merk op dat $E = \log 3 / \log \frac{3}{2} = 1.5 \log 3 > 2$ omdat $1.5^2 = 2.25$. Dus de worst-case tijdscomplexiteit van *sort* is significant slechter is dan die van de andere sorteer-algoritmen; we geven dus nooit de voorkeur aan *sort*.
- Stel x is de node die k bevat. Er zijn 2 mogelijkheden:
 - x heeft linkerkinderen. Ga dan 1 keer naar linksbeneden, en blijf dan vervolgens naar rechtsbeneden gaan zolang als dat kan. Je hebt dan de grootste waarde kleiner dan k .
 - x heeft geen linkerkinderen. Ga eerst net zolang rechtsonhoog tot dat niet meer kan, en ga dan linksomhoog. Kun je niet linksomhoog, dan was k het kleinste element en lever je *null* op; anders bereik je zo de grootste waarde kleiner dan k .

Het algoritme:

```
def bstPred(x):
    if x.left != null :
        x = x.left
        while x.right != null :
            x = x.right
        return x

    else :
        y = x.parent
        while y != null and y.left == x :
            x = y
            y = y.parent
        return y
```

- We passen dynamisch programmeren toe. Uiteindelijk moeten we het bovenste vakje vinden met het meeste aantal punten.

```
def maxpoints(p,n):

    R=[[0 for j in range(n+1)] for i in range(n+2)]

    for j in range(2,n+1):
        for i in range(1,n+1):
            R[i][j] = max(R[i-1][j-1]+p[i-1][j-1][R],
                          R[i+1][j-1]+p[i+1][j-1][R])
```

```

mx=R[1][n]
for i in range(2,n+1):
    mx=max(mx,R[i][n])
return mx

```

De complexiteit van dit algoritme is $\Theta(n^2)$.

Antwoorden: Discrete Mathematics

4. Gebruik het Euclidische Algoritme om $\gcd(1000, 444)$ te bepalen:

```

1000 = 2*444 + 112
444 = 3*112 + 108
112 = 1*108 + 4
108 = 27*4 + 0

```

Dus $\gcd(1000, 444) = 4$. We weten ook dat $\gcd(1000, 444) = \min\{1000s + 444t > 0 \mid s, t \in \mathbb{Z}\}$. Omdat $2 < 4$, kan de vergelijking $1000s + 444t = 2$ geen oplossing in \mathbb{Z} hebben.

5. (a) Het karakteristieke polynoom van de bijhorende homogene recurrente betrekking is $x^2 - 10x + 21 = (x - 3)(x - 7)$. De wortels hiervan zijn $x_1 = 3$ en $x_2 = 7$. Dus de algemene oplossing van de homogene recurrente betrekking is

$$a_n^{(h)} = c_1 3^n + c_2 7^n.$$

Als particuliere oplossing kiezen wij

$$a_n^{(p)} = An3^n,$$

omdat $A3^n$ niet lineair onafhankelijk zou zijn. Inzetten in de inhomogene recurrente betrekking levert $An3^n - 10(n-1)3^{n-1} + 21(n-2)3^{n-2} = 60 \cdot 3^n$ voor alle n , dus $An3^n(1 - \frac{10}{3} + \frac{7}{3}) + A3^n(\frac{10}{3} - \frac{14}{3}) = 60 \cdot 3^n$, dus $A = -45$, en de algemene oplossing voor de inhomogene recurrente betrekking is

$$a_n = c_1 3^n + c_2 7^n - 45n3^n.$$

Nu hebben we $a_0 = 2 = c_1 + c_2$, en $a_1 = -5 = 3c_1 + 7c_2 - 135$. Hieruit volgt dat $c_1 = -29$ en $c_2 = 31$, en het antwoord is

$$a_n = -29 \cdot 3^n + 31 \cdot 7^n - 45n3^n.$$

- (b) Laat a_n^k het aantal strings (met de genoemde eigenschappen) zijn wat eindigt op een k , dan

$$a_n = a_n^0 + a_n^1 + a_n^2.$$

Verder, $a_n^0 = a_{n-1}$, $a_n^1 = a_{n-1}^1 + a_{n-1}^2$, en $a_n^2 = a_{n-1}^1 + a_{n-1}^2$. Dat geeft

$$a_n = a_{n-1} + (a_{n-1} - a_{n-1}^0) + (a_{n-1} - a_{n-1}^0) = 3a_{n-1} - 2a_{n-2}.$$

Tenslotte, $a_1 = 3^1 = 3$, $a_2 = 3^2 - 2(\text{voor } 01 \text{ en } 02) = 7$, $a_3 = 3^3 - 3 \cdot 4(\text{voor } 01X \text{ en } X01 \text{ en } 02X \text{ en } X02) = 15(= 3 \cdot 7 - 2 \cdot 3)$.

6. Laat $D(s)$ de verzameling lijnen zijn uit $\delta(s)$ met de laagste kosten binnen de lijnen in $\delta(s)$. Dan geldt dat $D(s) \subseteq D_s$, want $d_e \geq 0$ voor alle $e \in E$. Verder, vanwege de 'cut condition' voor MSTs, moet minimaal een lijn $e \in D(s)$ bestaan zodat $e \in T$. Dus $T \cap D_s \neq \emptyset$.

7. (a) Neem $G = K_{3,3} + \{v\}$ en verbind v met een willekeurige punt van $K_{3,3}$. Dan is deze graaf niet planair, want hij bevat een $K_{3,3}$ als deelgraaf, maar $m = 10$, $n = 7$, dus $m \leq 2n - 4$. Dus deze graaf is een tegenvoorbeeld.
- (b) Bewijs: Neem een willekeurige planaire tekening van G . Merk op dat, als we in een gesloten pad langs de "grens" van een willekeurig gebied lopen, dat we minstens 4 lijnen moeten aflopen, want er bestaan *geen cycles van oneven lengte*, dus elke cycle heeft lengte minstens 4 (merk op dat K_2 hier een tegenvoorbeeld zou zijn, daarom hebben we $m > 1$ aangenomen). Laat f_i de aantal lijnen zijn die we aflopen langs de grens van gebied i . We concluderen $2m = \sum_{i=1}^r f_i \geq 4r$, met $r =$ aantal gebieden van G . Dus $r \leq \frac{1}{2}m$. Nu geldt de Euler formule voor G , dus

$$n - m + r = 2.$$

en daarom $2 = n - m + r \leq n - m + \frac{1}{2}m = n - \frac{1}{2}m$. Dus $m \leq 2n - 4$.

8. We berekenen eerst de aantal mogelijkheden waarbij een vaste persoon, zeg 1, hooguit 15 krijgt. Dat is gegeven door de coefficient van x^{50} van de genererende functie

$$f(x) = (x^{10} + x^{11} + \dots)^2 (x^{10} + x^{11} + \dots + x^{15}) = x^{30} \frac{1 - x^6}{(1 - x)^3}.$$

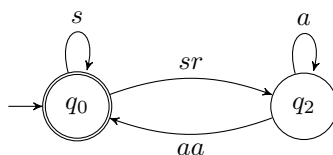
Dat is dus de coefficient van x^{20} van

$$\frac{1}{(1 - x)^3} - \frac{x^6}{(1 - x)^3}.$$

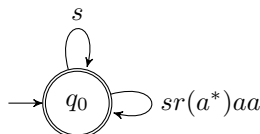
Dus dat is $\binom{-3}{20} - \binom{-3}{14} = \binom{22}{20} - \binom{16}{14} = 111$. Het antwoord is dus $3 \cdot 111 = 333$.

Antwoorden: Languages & Machines

9. (a) Elimineer eerst q_1 en q_3 (ik doe het tegelijk, het is onafhankelijk)



Nu q_2 elimineren:

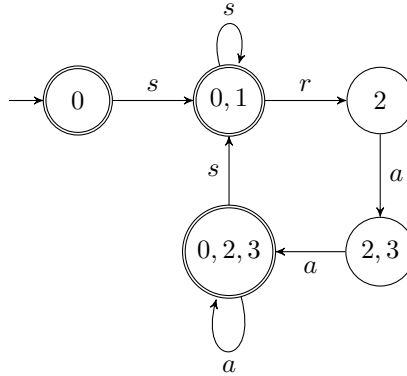


Nu kun je de reguliere expressie aflezen: $(s \cup sra^*aa)^*$.

- (b)

| | λ | a | s | r |
|---|------------|---------------|-------------|-------------|
| 0 | $\{0\}$ | \emptyset | $\{0, 1\}$ | \emptyset |
| 1 | $\{0, 1\}$ | \emptyset | $\{0, 1\}$ | $\{2\}$ |
| 2 | $\{2\}$ | $\{2, 3\}$ | \emptyset | \emptyset |
| 3 | $\{2, 3\}$ | $\{0, 2, 3\}$ | \emptyset | \emptyset |

- (c) De subset constructie levert de volgende onvolledige DFA op (we laten transitie naar \emptyset weg):



10. (a) Regulier, L_1 bevat precies alle drievouden van a 's. Dit kun je met een reguliere expressie definiëren: $L_1 = \mathcal{L}\left((aaa)^*\right)$.
- (b) Niet-regulier, bewijs met de pompstelling: Laat $k > 0$ willekeurig gegeven zijn, kies $z = a^k b a^{2k}$, dan $|z| \geq k$ en $z \in L_2$. Laat u, v, w willekeurig gegeven zijn, zodanig dat $z = uvw$, $|uv| \leq k$ en $|v| > 0$. Dan moet $u = a^m$, $v = a^n$ en $w = a^{k-m-n} b a^{2k}$ zijn voor zekere $m \geq 0$ en $n > 0$. Kies $i = 2$, dan $uv^i w = a^{k+n} b a^{2k} \notin L_2$, omdat $n \neq 0$. We hebben laten zien dat $\forall k > 0 : \exists z : |z| \geq k \wedge z \in L_2 \wedge \forall u, v, w : |uv| \leq k \wedge |v| > 0 \wedge z = uvw \rightarrow \exists i : uv^i w \notin L_2$. Dus L_2 is niet regulier ("contrapositie" van de pompstelling).
- (c) $L_3 = L_{3a} \cap \overline{L_{3b}}$, (ofwel: $L_{3a} - L_{3b}$) met

$$L_{3a} = \mathcal{L}\left((b+c)^* \cdot (a(b+c+\lambda)^4)^* \cdot (b+c)^*\right)$$

$$L_{3b} = \mathcal{L}\left((a+b+c)^* \cdot (abcba) \cdot (a+b+c)^*\right),$$

die dus beide regulier zijn. Reguliere talen zijn gesloten onder complement en doorsnede (ofwel: verschil) en dus is L_3 ook regulier.

- (d) Stel dat $L_4 \cup L_5$ regulier is. Iedere eindige taal is regulier; L_5 is eindig, dus is ook $L_5 - L_4$ eindig, en dus ook regulier. Reguliere talen zijn gesloten onder verschil (doorsnede en complement), dus dan is ook $L_4 = (L_4 \cup L_5) - (L_5 - L_4)$ regulier. Dit is een tegenspraak met het gegeven dat L_4 niet regulier is. Dus $L_4 \cup L_5$ is ook niet regulier.