1e deeltentamen Algoritmiek 2017 / 2018

Je hebt drie uur voor dit deeltentamen.

Schrijf op elk van de ingeleverde bladen je naam en studentnummer, en op het eerste blad het aantal ingeleverde bladen.

Geef duidelijke antwoorden in helder en correct Nederlands of Engels. Wanneer niet expliciet anders gevraagd, mag je resultaten uit het college gebruiken.

Als een algoritme wordt gevraagd is het een goed idee om dit algoritme zowel te geven in pseudocode, als een toelichting in woorden te geven.

Schrijf netjes, en lever overzichtelijk werk in.

Deel je tijd goed in: sommige opgaven kunnen moeilijker zijn dan andere.

1. String matching (punten)

Beschouw het algoritme voor string matching dat gebruik maakt van een eindige automaat. **Teken de eindige automaat die dit algoritme geeft in het geval dat het patroon de string ababb is.**

2. Dynamisch programmeren I (punten)

Het MAXIMUM WEIGHT COMMON SUBSEQUENCE probleem is het volgende. Gegeven zijn twee strings X en Y. We hebben ook een gewichtsfunctie: ieder karakter in X en ieder karakter in Y heeft een gewicht. Het gewicht is een positief geheel getal. Het gewicht van het i-de karakter in X is $w_X(i)$, en het gewicht van het j-de karakter in Y is $w_Y(j)$. We zoeken nu naar een *common subsequence* van X en Y zodat het totale gewicht van de gekozen karakters zo groot mogelijk is; hierbij tellen we zowel de gewichten van de karakters op zoals ze in X zitten, als zoals ze in Y zitten.

Net als in het college gebruiken we de volgende notatie: X_i is de prefix van X met lengte i, en $x(i)$ is het i-de karakter in X; Y_j is de prefix van Y met lengte j, en $y(j)$ is het j-de karakter in Y.

Om dit probleem op te lossen met dynamisch programmeren gebruiken we de volgende definitie: $W(i,j)$ = het maximale gewicht van een gemeenschappelijke subsequence van X_i en Y_j.

De volgende recurrente betrekkingen gelden voor W:

- Als $i = 0$ of $j = 0$, dan is $W(i,j) = 0$.
- Als $i > 0$ en $j > 0$ en $x(i) = y(j)$ dan $W(i,j) = \max\{W(i-1,j-1) + w_X(i) + w_Y(j), W(i-1,j), W(i,j-1)\}$.

- Als $i > 0$ en $j > 0$ en $x(i) \neq y(j)$ dan $W(i,j) = \max\{W(i-1,j), W(i,j-1)\}$.

Je mag aannemen dat deze recurrente betrekkingen correct zijn, en hoeft ze niet te bewijzen.

(a) **Geef duidelijke pseudocode van een algoritme dat uitrekent wat het maximum gewicht is van een common subsequence van een gegeven X en gegeven Y. Je algoritme moet dynamisch programmeren gebruiken, en mag geen gebruik maken van een recursief programma met memorisatie.**

(b) Stel dat X lengte n heeft en Y lengte m heeft. **Hoeveel tijd gebruikt je algoritme (in O-notatie)?**

3. Dynamisch programmeren II: Verdelen in drie groepen (punten)

Het volgende probleem kan worden opgelost met dynamisch programmeren. Gegeven zijn n positieve gehele getallen, $x_1, \ldots, x_n$. Gevraagd wordt om deze getallen in drie groepen te verdelen, zodat elke groep dezelfde som heeft.

In deze opgave kijken we naar een van de stappen van het dynamisch programmeren: het opstellen van de recurrente betrekking.

Schrijf: $S = \sum_{i=1}^{n} x_i$. Voor $0 \leq a \leq S$, $0 \leq b \leq S$, $0 \leq c \leq S$, $0 \leq i \leq n$, definieer

- $A(i,a,b,c)$ = true, als we de getallen $x_1, \ldots, x_i$ in drie groepen kunnen verdelen, zodat de som van de eerste groep a is, de som van de tweede groep b is, en de som van de derde groep c is.
- $A(i,a,b,c)$ = false, als zo'n verdeling niet bestaat.

Bijvoorbeeld: Als we de getallenrij $5, 10, 15, 6, 11, 16, 7, 12, 17$ hebben, dan: $A(4,11,15,10)$=true (groepen $\{5,6\}$, $\{15\}$, $\{10\}$), en $A(3,6,11,16)$ is false (omdat $i = 3$ kijken we naar de getallen 5, 10 en 15).

(a) **Geef aan bij welke waarde(s) van a, b, en c, $A(0,a,b,c)$ true is.**

(b) **Stel $i \geq 1$. Geef een recurrente betrekking voor $A(i,a,b,c)$.** (Je betrekking moet bruikbaar zijn voor dynamisch programmeren: druk uit met behulp van $A(j,a',b',c')$-termen met $j < i$.)

4. Piek (✋ punten)

Voor een extra zakcentje beleg je al n dagen in het aandeel Fang. Je houdt iedere dag nauwlettend de koers bij, en of de koers gestegen of gedaald is ten opzichte van de dag ervoor. Je hebt dus een array met n dagen en voor iedere dag een 'S' of een 'D' als het aandeel Fang gestegen of gedaald is. Een *piek* is een dag die zelf een 'S' is en waarvoor de navolgende dag een 'D' is.

Je mag aannemen dat dag 1 een 'S' was en dag n een 'D'. Dus bijvoorbeeld 'SSDSD'. In dit voorbeeld zijn er twee pieken: op dag 2, en op dag 4. Merk op dat er altijd tenminste één piek is.

(a) **Geef een algoritme dat een piek vindt. Je algoritme moet een looptijd hebben van $O(\log n)$.** In het eerdere voorbeeld mag je of 2 of 4 retourneren.

(b) **Leg uit waarom je algoritme correct is.**

(c) **Analyseer de looptijd van je algoritme.**

5. Adverteren (✋ punten)

Je werkt bij een grote internetwinkel. Er zijn nog n dagen tot en met de Kerst, en de grote internetwinkel wil iedere dag één advertentie plaatsen bij een bekende zoekmachine. Er zijn n advertenties ontwikkeld; sommige text-only, andere met een klein plaatje, etc. De kosten van advertentie i zijn c_i, vermenigvuldigd met de dagkosten (zie hieronder). Je moet alleen nog beslissen op welke dag je welke advertentie wilt plaatsen. Nu heeft de bekende zoekmachine ook wel door dat iedereen vlak voor Kerst graag wil adverteren. Daarom kost het plaatsen van een advertentie steeds meer naarmate Kerst dichterbij komt. Op dag j zijn de dagkosten van het plaatsen van een advertentie daarom 2^j (op dag n is het Kerst). Dus: als we op dag j advertentie i plaatsen dan kost dit $c_i \cdot 2^j$.

Bijvoorbeeld: stel dat je drie advertenties hebt, met kosten $c_1 = 5$, $c_2 = 3$, $c_3 = 10$ en je plaatst op dag 1 advertentie 2, op dag 2 advertentie 3, en op dag 3 advertentie 1, dan zijn je totale kosten $c_2 \cdot 2^1 + c_3 \cdot 2^2 + c_1 \cdot 2^3 = 3 \cdot 2 + 10 \cdot 4 + 5 \cdot 8 = 86$.

Uiteraard wil de grote internetwinkel de totale kosten minimaliseren.

(a) **Geef een efficient algoritme dat bepaalt op welke dag welke advertentie geplaatst moet worden zodat de totale kosten geminimaliseerd worden.** Je moet iedere dag een advertentie plaatsen en mag een advertentie niet twee keer plaatsen.

(b) **Bewijs dat je algoritme correct is.**

(c) **Analyseer de looptijd van je algoritme.**

6. Binaire teller (✾ punten)

Je houdt een teller bij met n bits, die initieel allemaal 0 zijn. Er is één operatie, `verhoog`, die de teller met precies één verhoogt. Als alle bits op 1 staan, dan gaat de teller terug naar de stand waar alle bits op 0 staan. (Bijvoorbeeld: als $n = 2$, dan krijgen we opeenvolgend 00, 01, 10, 11, 00, 01, 10, enz.)

De complexiteit van deze operatie meten we aan de hand van het aantal bits dat we moeten veranderen. Bijvoorbeeld, als een binaire teller met 5 bits op 2 (00010) staat en je een `verhoog` naar 3 (00011) doet, hoef je maar één bit te veranderen. Echter, als de teller op 7 (00111) staat en je een `verhoog` naar 8 (01000) doet, moet je vier bits veranderen. In het ergste geval moet je dus n bits veranderen. Maar dit ergste geval komt bijna nooit voor.

Gebruik de potentiaal-methode om aan te tonen dat de geamortiseerde kosten van `verhoog` in een serie operaties $O(1)$ is.