Proeftentamen Algoritmiek

U mag gebruik maken van resultaten die in het college bewezen zijn.

Opgave 1. (2 punt) Laat V een verzameling van n voorwerpen zijn, allemaal met een bepaalde waarde. De waarde van een voorwerp v wordt aangegeven met $w(v)$. We willen V in drie deelverzamelingen, V_1, V_2, en V_3, opdelen op een zo eerlijk mogelijke manier. Preciezer gezegd, we willen een opslitsing zodanig dat

$$max(|\sum_{v\in V_1} w(v) - \sum_{v\in V_2} w(v)|, |\sum_{v\in V_1} w(v) - \sum_{v\in V_3} w(v)|, |\sum_{v\in V_2} w(v) - \sum_{v\in V_3} w(v)|)$$

zo klein mogelijk is. Schrijf een brute-force algoritme dat de waarde van een optimale opsplitsing berekent, en analyseer de looptijd van je algoritme.

Opgave 2. ($\frac{1}{2} + \frac{1}{2} + 1$ punt) Een bergwandelaar kan maximaal G kilo bagage dragen. Er zijn n voorwerpen die hij eventueel wel zou willen meenemen. Het totale gewicht van de voorwerpen is echter groter dan M, zodat hij zal moeten kiezen. Daartoe heeft hij elk voorwerp een waarde gegeven die aangeeft hoe nuttig hij het vindt. Laat g_i het gewicht van het i-de voorwerp zijn (in kilo's) en laat w_i de waarde daarvan zijn. Een optimale deelverzameling voorwerpen is dus een deelverzameling met een maximale totale waarde onder de voorwaarde dat het totale gewicht kleiner of gelijk aan G is. Ga er vanuit dat $g_i \leqslant G$, dat g_i een geheel getal is, voor elke $1 \leqslant i \leqslant n$, en dat G een geheel getal is.

(a.) Een mogelijke aanpak om een optimale deelverzameling te vinden is met een greedy algoritme dat de voorwerpen in volgorde van waarde/gewichtverhouding bekijkt:

Algorithm *Greedy*()
1. Sorteer de voorwerpen op volgorde van dalende waarde/gewicht verhouding.
($*$ Nu geldt dus: $w_1/g_1 \geqslant w_2/g_2 \geqslant \cdots \geqslant w_n/g_n$. $*$)
2. $g \leftarrow 0$; $S \leftarrow \emptyset$
3. **for** $i \leftarrow 1$ **to** n
4. **do if** $g + g_i \leqslant G$
5. **then** $S \leftarrow S \cup \{i\}$
6. $g \leftarrow g + g_i$
7. **return** S

Laat zien dat dit greedy algoritme niet altijd een optimale oplossing vindt.

(b.) Omdat een greedy aanpak niet werkt gaan we het probleem met dynamisch programmeren oplossen. Hiertoe definieren we voor integers i en g met $1 \leqslant i \leqslant n$ en $1 \leqslant g \leqslant G$:

$c_{i,g}$ = de maximale waarde van een deelverzameling voorwerpen gekozen uit $\{i, \ldots, n\}$ onder de voorwaarde dat het totale gewicht kleiner of gelijk aan g is.

Geef een recurrente betrekking voor $c_{i,g}$.

(c.) Geef een dynamisch-programmeer algoritme dat, aan de hand van de recurrente betrekking die je in onderdeel (b.) hebt gegeven, de waarde van een optimale deelverzameling berekent. Het algoritme moet in $O(nG)$ tijd werken, maar dit hoef je niet te bewijzen.

Opgave 3. (1 punt) Stel een gerichte graaf $G = (V, A)$ is gegeven met voor elke pijl $(v, w) \in A$ een capaciteit $c(v, w)$. Elke capaciteit is een positief geheel getal. Stel een stromingsfunctie $f : V \times V \to \mathbf{Z}$ is gegeven. Beschrijf een algoritme dat in $O(n + m)$ tijd bepaalt of f een maximum stroming is. ($n = |V|$, $m = |E|$.)

Opgave 4. (1 punt) Zij $G = (V, E)$ een samenhangende ongerichte graaf met voor elke kant $(x, y) \in E$ een gewicht $w(x, y)$. Een *maximum opspannende boom* van G is een opspannende boom van G met maximum gewicht over alle opspannende bomen. Stel $w(x, y) = \max_{e \in E} w(e)$, d.w.z., (x, y) is een kant met maximum gewicht in E. Laat zien dat er een maximum opspannende boom bestaat die e bevat.

Opgave 5. (1+1 punt)

(a.) Stel gerichte graaf $G = (V, A)$ is gegeven met voor elke pijl $(v, w) \in A$ een afstand $w(v, w) \in \mathbf{Z}$. Zij ook een integer k gegeven. Beschrijf een algoritme dat in $O(k \cdot n^3)$ tijd voor elk paar knopen $x, y \in V$ bepaald wat de korste afstand is van een pad van x naar y dat ten hoogste k pijlen bevat; deze afstand wordt geacht ∞ te zijn als er geen pad van x naar y met hooguit k pijlen bestaat. Leg uit waarom Uw algoritme correct is. ($n = |V|$.)

(b.) Een programmeur heeft bij het implementeren van het Bellman-Ford algoritme een typfout gemaakt waarbij niet $n-1$ keer maar $n+1$ keer over alle pijlen van G gerelaxeerd wordt:

```
for i ← 1 to n + 1
do for each directed edge (u, z) of G
    do if D[u] + w(u, z) < D[z]
        then D[z] ← D[u] + w(u, z)
```

Opnieuw begint het algoritme met initieel $D[v] = 0$ en $D[w] = \infty$ voor alle $w \neq v$. Berekent deze code nog steeds de afstanden van v naar de andere knopen in G op een correcte manier? Toon Uw antwoord aan.

Opgave 5. (0.5+1.5 punt)

(a.) Leg uit wat het verschil is tussen Monte Carlo en Las Vegas algoritmen.

(b.) Stel ongerichte graaf $G = (V, E)$ is gegeven door middel van de adjacency matrix datastructuur. Stel G heeft $n^2/4$ kanten. Geef een Monte Carlo algoritme dat in $O(1)$ verwachte tijd een paar knopen dat door een kant verbonden is vindt. Bewijs de tijdgrens.