



## Stof

- slides college 9
- boek 5.3, 5.1

## Opgaven

1. Deze vraag gaat over houthakken.  
Pas het dynamic programming algoritme toe op de situatie uit de slides met  $n = 3$ .
2. Deze vraag gaat over houthakken.  
Definieer de *dichtheid* van een stuk hout ter lengte  $i$  als  $\frac{p_i}{l_i}$ , dat wil zeggen, de waarde per decimeter. Een greedy algoritme voor houthakken hakt eerst een stuk ter lengte  $i$  met een hoogste dichtheid eraf, en gaat dan verder voor het stuk ter lengte  $n - i$ . Geef een voorbeeld waaruit blijkt dat deze aanpak niet altijd tot een optimale oplossing leidt.
3. Deze vraag gaat over houthakken.  
We passen het probleem aan: nu is er een vaste kostenpost  $c$  per hak. De winst is nu dus de som van de waardes van de afzonderlijke stukken, min de kosten voor het hakken. Pas het dynamic programming algoritme aan aan deze nieuwe situatie.
4. Deze vraag gaat over het maximale subarray probleem.  
Pas het dynamic programming algoritme toe op het array  $[-3, 10, -8, 9]$ .
5. Deze vraag gaat over fractional knapsack.  
Geef een (klein) voorbeeld waaruit blijkt dat een greedy keuze voor een item met grootste waarde niet perse tot een optimale oplossing leidt.
6. Deze vraag gaat over fractional knapsack.  
Geef een (klein) voorbeeld waaruit blijkt dat een greedy keuze voor een item met laagste gewicht niet perse tot een optimale oplossing leidt.
7. (Dit is ongeveer opgave R-5.1.)  
Pas het algoritme voor fractional knapsack toe op totaalgewicht  $W = 18$  en de volgende verzameling items met benefit-gewicht waarden:

$$\{(12, 4), (10, 6), (8, 5), (11, 7), (14, 3), (7, 1), (9, 6)\}$$

8. (Dit is ongeveer opgave R-5.11.)

Pas met de gegevens uit de vorige opgave nu het algoritme voor knapsack-0-1 toe. (Laat ook zien hoe.)

9. (Dit is ongeveer vraag R-5.12.)

Sally organiseert een online-veiling van  $n$  dingen, en krijgt daarvoor  $m$  biedingen binnen, allemaal van de vorm ‘ik wil  $k_i$  dingen voor  $e_i$  euro’, met  $i \in \{1, \dots, m\}$ .

Geef haar optimalisatie-probleem als een knapsack-probleem. Wanneer is het een fractional knapsack-probleem en wanneer een knapsack-0-1-probleem?

10. Leg uit waarom het algoritme voor fractional knapsack in  $O(n \log n)$  is, met  $n$  het aantal elementen in de verzameling  $S$  waaruit we (delen van) items kiezen.

11. (Dit is ongeveer opgave C-5.9.)

Verander het algoritme voor knapsack-0-1 zó dat het niet alleen de best mogelijke benefit geeft, maar ook de deelverzameling van items die daartoe leidt.

Pas dit algoritme toe op de volgende verzameling  $S$ , met items  $s_i$  met benefit  $b_i$  en gewicht  $w_i$ :

	$b$	$w$
$s_1$	2	1
$s_2$	6	4
$s_3$	10	5
$s_4$	3	2

met maximaal totaalgewicht  $W = 8$ .

12. Stel we hebben een verzameling  $S$  en maximum totaalgewicht  $W$  zó dat het algoritme voor fractional knapsack een oplossing geeft met alleen *hele* items (dus geen ‘fractions’). Geeft het algoritme voor knapsack-0-1 in dat geval dezelfde oplossing?

13. (Dit is ongeveer opgave R-5.3.)

Pas het machine-scheduling algoritme toe op de volgende verzameling taken, met begin- en eind-tijd gegeven:

$$\{(1, 2), (1, 3), (1, 4), (2, 5), (3, 7), (4, 9), (5, 6), (6, 8), (7, 9)\}$$

14. (Dit is ongeveer opgave C-5.2.)

Gegeven: een verzameling taken, elk met begin- en eind-tijd. Geef een greedy (scheduling) algoritme dat een zo groot mogelijke deelverzameling taken geeft die op één machine uitgevoerd kunnen worden (dus sequentieel).

15. Laat zien dat de greedy methode bij het Traveling Salesman Probleem niet werkt.

(NB: het Traveling Salesman Probleem heeft als input een ongerichte gewogen graaf met een start-knoop. De vraag is om een pad van minimaal totaal-gewicht te vinden dat elke knoop precies één keer bezoekt.)

16. Geef een naïef algoritme voor het berekenen van de Fibonacci-getallen, gedefinieerd door:

$$\begin{aligned}F(1) &= 1 \\F(2) &= 1 \\F(n) &= F(n-1) + F(n-2)\end{aligned}$$

Wat is de tijdscomplexiteit van je algoritme?

17. Geef een dynamic programming algoritme voor het berekenen van de Fibonacci-getallen.

Wat is de tijdscomplexiteit van je algoritme?