

Tentamen Discrete Optimalisering

30-01-2014, 12:00u - 14:00u

Puntentoekening

Dit tentamen bestaat uit 4 opgaven, waaruit 40 punten te behalen zijn.

<i>Opgave</i>	<i>a</i>	<i>b</i>
Opg. 1	10	-
Opg. 2	5	5
Opg. 3	10	-
Opg. 4	10	-

$$\text{Eindcijfer} = \max\{1, \text{puntentotaal}/4\}$$

Opgave 1

De rijen van de volgende tabel representeren personen en de kolommen representeren jobs. Een getal in rij i en kolom j geeft aan wat de kosten zijn indien persoon i job j doet. Los het toewijzingsprobleem op. Laat daarbij duidelijk de stappen zien die je maakt om tot de oplossing te komen.

	1	2	3	4
1	9	5	4	7
2	4	3	3	9
3	1	3	2	5
4	5	8	1	4

Answer. Distract the lowest number in each column from all elements of the column, and then the lowest from each row from all elements of the row. This yields the matrix:

	1	2	3	4
1	6	0	1	1
2	3	0	2	5
3	0	0	1	1
4	4	5	0	0

The 0-elements of this matrix can be covered by 3 lines only, namely, for example, the lines corresponding to the first two columns and the last row (unfortunately, I cannot draw this in the matrix). Thus we have not found the solution yet.

In the next iteration, we select the minimal uncovered element: this is element 1 (which in fact occurs at various places). We distract 1 from all uncovered elements and add it to the elements that are on the intersection of covering row- and column lines. This yields the matrix:

	1	2	3	4
1	6	0	0	0
2	3	0	1	4
3	0	0	0	0
4	5	6	0	0

The 0-elements in this matrix need 4 lines to cover them, hence an optimal solution can be found just using 0-elements. For example, an optimal solution is Persoon 3 krijgt Job 1, Persoon 2 krijgt Job 2, Persoon 1 krijgt Job 3, en Persoon 4 krijgt Job 4. De optimale totale kosten zijn 12.

Opgave 2.

Opgave 2a: Beschrijf formeel het algoritme van Prim voor het oplossen van het Minimum Opspannende Boom probleem.

Answer. The formal description of Prim's algorithm can be found in the book.

Opgave 2b: Leid de rekentijd van dit algoritme af als functie van het aantal knopen van de graaf. Geef uiteindelijk de rekentijd weer in termen van de grote O notatie.

Answer. Let us denote the number of vertices of the graph by n and the number of edges by m . The start-up requires setting the two labels for each vertex except the initial vertex, which requires $2(n - 1)$ operations. In every iteration we determine the vertex with lowest label, i.e. the one that can be added in the cheapest way to the already existing tree. This takes at most $2(n - 1)$ comparisons. Then we determine for each vertex not yet in the tree if the distance to the newly added vertex is shorter than his current label. If so the label is changed and then also the neighbour-label is changed. Thus this requires at most 3 operations. Thus, the total number of operations per iteration is at most $2n + 1$. Since there are exactly $n - 1$ iterations, the total number of operations including the start-up is $(2n + 3)(n - 1)$. This is $O(n^2)$ operations.

Opgave 3.

Gegeven is de ongerichte graaf op 11 knopen met kanten en lengtes op de kanten zoals aangegeven in de volgende tabel. Bepaal de optimale Chinese Postbode route op deze graaf. Laat duidelijk zien welke stappen je maakt om tot de oplossing te komen.

kant	lengte	kant	lengte
(v_1, v_2)	5	(v_6, v_5)	1
(v_1, v_3)	5	(v_6, v_7)	1
(v_1, v_4)	5	(v_6, v_8)	2
(v_2, v_3)	2	(v_6, v_9)	3
(v_2, v_5)	3	(v_7, v_{10})	3
(v_2, v_6)	2	(v_8, v_9)	2
(v_3, v_6)	5	(v_8, v_{11})	5
(v_3, v_7)	2	(v_9, v_{10})	2
(v_4, v_7)	3	(v_9, v_{11})	4
(v_5, v_8)	5	(v_{10}, v_{11})	4

Answer. First we determine the vertices with an odd degree. These are vertices v_1, v_5, v_{10}, v_{11} . Compute the shortest distances between each pair of these points. This can be done by application of Dijkstra's algorithm, but it is also considered correct if they are just found by inspection. The distances are

$d(v_1, v_5) = 8$ $d(v_1, v_{10}) = 10$ $d(v_1, v_{11}) = 14$ $d(v_5, v_{10}) = 5$ $d(v_5, v_{11}) = 8$ $d(v_{10}, v_{11}) = 4$
--

From this we determine how we can match vertices pairwise in a cheapest possible way. This is, again just by inspection, to match vertex v_1 with v_5 and vertex v_{10} with v_{11} . This raises the degree of the four vertices by one, making them even at a total increase of the edges to be traversed of $8 + 4 = 12$. Notice that adding the shortest path between v_1 and v_5 amounts to adding edges (v_1, v_2) and (v_2, v_5) , and adding the shortest path between v_{10} and v_{11} amounts to adding edge (v_{10}, v_{11}) .

Having added the extra edges we have now a graph that is connected and the degree of all its vertices being even. Hence there is an Euler Circuit. This is determined by simply extending a cycle starting in any vertex. E.g. start in v_1 and make a cycle by continuing on a path as long as possible until getting back in v_1 and all edges incident to v_1 have been traversed. A possible such cycle is

$v_1, v_2, v_5, v_6, v_9, v_{10}, v_{11}, v_{10}, v_7, v_4, v_1, v_3, v_2, v_1$. Delete the edges on this cycle. The edges remaining are then (including the added shortest paths

$$(v_2, v_5), (v_2, v_6), (v_6, v_7), (v_6, v_8), (v_3, v_6), (v_3, v_7), (v_5, v_8), (v_8, v_9), (v_8, v_{11}), (v_9, v_{11}).$$

Then we start from any vertex that has still an edge to be traversed. E.g. v_2 and find a cycle on the remaining edges. E.g. this could be $v_2, v_5, v_8, v_9, v_{11}, v_8, v_6, v_3, v_7, v_6, v_2$.

This exhausts all remaining edges. Then we insert the last cycle at any point where v_2 appears in the first one, resulting in the Chinese Postman tour:

$$v_1, v_2, v_5, v_6, v_9, v_{10}, v_{11}, v_{10}, v_7, v_4, v_1, v_3, v_2, v_5, v_8, v_9, v_{11}, v_8, v_6, v_3, v_7, v_6, v_2, v_1.$$

Opgave 4: Dynamische Programmering

Een meubelzaak heeft na de januari uitverkoop weer genoeg winkelruimte over om haar assortiment aan te vullen. Daartoe maakt zij een selectie uit een een collectie van N meubels (genummerd van 1 tot en met N). Elk meubel neemt een bepaalde hoeveelheid vloeroppervlak in beslag en levert een bepaalde winst op bij verkoop. Precies geformuleerd: meubelstuk j neemt p_j eenheden vloeroppervlak in beslag en levert w_j winst op bij verkoop.

Het totaal aan vrijgekomen vloer oppervlak is V . Daarnaast is besloten om, onafhankelijk van de vloer capaciteit, niet meer dan $K < N$ meubels in te kopen. Dus de meubelzaak wil zijn totale winst maximaliseren, onder de voorwaarde dat de hooguit K aangeschafte meubels niet meer dan V vloeroppervlak in beslag nemen.

Formuleer dit probleem als een dynamisch programmeringsprobleem. Beschrijf daarvoor in ieder geval de toestand, de interpretatie van de waardefunctie in woorden, de recursierelatie, de start-recursie en wat je moet berekenen in termen van de waardefunctie.

Answer. The state consists here of a pair (v, k) , where v is the remaining surface to be filled and k is the remaining number of pieces of furniture to be acquired.

As a phase we choose j which tells that for piece of furniture j we need to decide if we select it or not, having selected already from the pieces $1, 2, \dots, j-1$.

As value function we choose for all $j = 1, \dots, N$, $v = 1, \dots, V$, $k = 1, \dots, K$,

$f_j(v, k)$ = the optimal total profit made on pieces $j, j+1, \dots, N$, when surface v is still free and there are still k pieces to be acquired.

Then we need to compute

$$f_1(V, K)$$

and the optimal selection.

The recursion is given by

$$f_j(v, k) = \max \{f_{j+1}(v, k), w_j + f_{j+1}(v - p_j)\}.$$

with start conditions

$$f_j(0, k) = 0, \quad \forall j = 1, \dots, N, \quad k = 1, \dots, K,$$

$$f_j(v, 0) = 0, \quad \forall j = 1, \dots, N, \quad v = 1, \dots, V,$$

and

$$f_j(v, k) = -\infty, \quad \text{if } v < 0, \quad \forall j = 1, \dots, N, \quad k = 1, \dots, K.$$