Operationsanalyse 1
Obligatorisk opgave 3

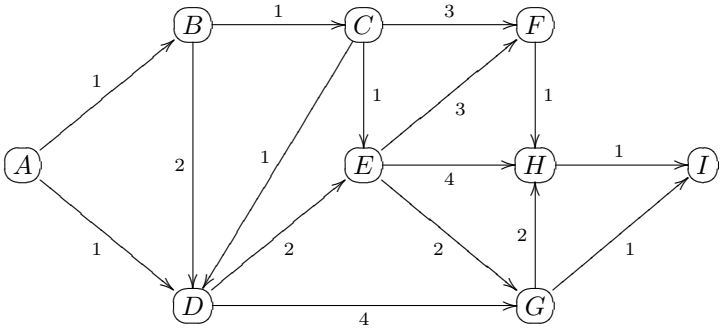
Anders "Bongo" Bjerg Pedersen
070183

14. juni 2006

Opgave 1

(a)

Vi anvender algoritmen fra bogens afsnit 9.3, idet vi for overskuelighedens skyld omdøber knuderne fra tal til bogstaver (1=A,2=B,...,9=I):



Figur 1: Directed graf til Opgave 1

n	Solved nodes directly connected to unsolved nodes	Closest connected unsolved node	Total distance	n'th nearest node	Minimum distance	Last connection
1, 2	A	B	1	B	1	$A-B$
	A	D	1	D	1	$A-D$
3	B	C	2	C	2	$B-C$
	D	E	3			
4	C	E	3	E	3	$C-E$
	D	E	3	E	3	$D-E$
5	C	F	5			
	D	G	5	G	5	$D-G$
	E	G	5	G	5	$E-G$
6	C	F	5	F	5	$C-F$
	E	F	6			
	G	I	6			
7	E	H	7			
	F	H	6	H	6	$F-H$
	G	I	6			
8	G	I	6	I	6	$G-I$
	H	I	7			

Vi kan nu finde de korteste ture fra A til de andre nodes ved at læse 'opad' i skemaet og får følgende resultater:

Node :	Shortest path:	Path length
A	A	0
B	$A-B$	1
C	$A-B-C$	2
D	$A-D$	1
E	$A-B-C-E$	3
	$A-D-E$	3
F	$A-B-C-F$	5
G	$A-B-C-E-G$	5
	$A-D-E-G$	5
	$A-D-G$	5
H	$A-B-C-F-H$	6
I	$A-D-E-G-I$	6
	$A-B-C-E-G-I$	6
	$A-D-G-I$	6

(b)

Vi opskriver ligningssystemet for problemet (nu igen med tal i stedet for bogstaver på de forskellige nodes), idet vi skal minimere de totale omkostninger ved at bevæge sig gennem (en delmængde af) alle grafens kanter. Dette ses i objektfunktionen. Vi opskriver problemet som et MCNF-problem reduceret til et shortest path-problem, dvs. ingen upper bounds, der sendes netop 1 vare gennem systemet fra supply node 1 (constraint (1)), modtages 1 fra demand node 9 (constraint (9)), og alle andre er transshipment nodes. Vores andre constraints (2)-(8) sikrer, at der kommer lige så meget ind i hver node, som der kommer ud. Vores x_{ij}-variable (kanter) er så 1, hvis de bliver anvendt og 0 ellers:

$$\begin{aligned}
&\text{Minimize } Z = x_{12} + x_{14} + x_{23} + 2x_{24} + x_{34} + x_{35} + 3x_{36} \\
&+2x_{45} + 4x_{47} + 3x_{56} + 2x_{57} + 4x_{58} + x_{68} + 2x_{78} + x_{79} + x_{89} \\
&\text{subject to:} \\
&x_{12} + x_{14} = 1 && (1)\\
&-x_{12} + x_{23} + x_{24} = 0 && (2)\\
&-x_{23} + x_{34} + x_{35} + x_{36} = 0 && (3)\\
&-x_{14} - x_{24} - x_{34} + x_{45} + x_{47} = 0 && (4)\\
&-x_{35} - x_{45} + x_{56} + x_{57} + x_{58} = 0 && (5)\\
&-x_{36} - x_{56} + x_{68} = 0 && (6)\\
&-x_{47} - x_{57} + x_{78} + x_{79} = 0 && (7)\\
&-x_{58} - x_{68} - x_{78} + x_{89} = 0 && (8)\\
&-x_{79} - x_{89} = -1 && (9)\\
&x_{ij} \in \{0,1\},\ i = 1,\ldots,8,\ j = 2,\ldots,9.
\end{aligned}$$

Opgave 2

Vi anvender Prims algoritme til at finde vores minimum weight spanning tree G^T til grafen $G = (V, E)$ i Figur 2, idet vi starter algoritmen i 1 ($V^T = \{1\}$ og $E^T = \emptyset$):

(1) Den billigste vej, der forbinder G^T med resten af grafen, er vejen fra 1 til 4, dvs. vi opdaterer vores delgraf med kanten (1,4) med vægtning $w_{1,4} = 9$, så vi får $G^T := (V^T, E^T)$ med $V^T = \{1, 4\}$ og $E^T = \{(1, 4)\}$.

(2) Den billigste vej, der forbinder G^T med resten af grafen, er vejen fra 4 til 2, dvs. vi opdaterer vores delgraf med kanten (2,4) med vægtning $w_{4,2} = 5$, så vi får $G^T := (V^T, E^T)$ med $V^T = \{1, 2, 4\}$ og $E^T = \{(1, 4), (2, 4)\}$.

(3) Den billigste vej, der forbinder G^T med resten af grafen, er vejen fra 2 til 3, dvs. vi opdaterer vores delgraf med kanten (2,3) med vægtning $w_{2,3} = 4$, så vi får $G^T := (V^T, E^T)$ med $V^T = \{1, 2, 3, 4\}$ og $E^T = \{(1, 4), (2, 3), (2, 4)\}$.

(4) Den billigste vej, der forbinder G^T med resten af grafen, er vejen fra 3 til 5, dvs. vi opdaterer vores delgraf med kanten (3,5) med vægtning $w_{3,5} = 3$, så vi får $G^T := (V^T, E^T)$ med $V^T = \{1, 2, 3, 4, 5\}$ og $E^T = \{(1, 4), (2, 3), (2, 4), (3, 5)\}$.

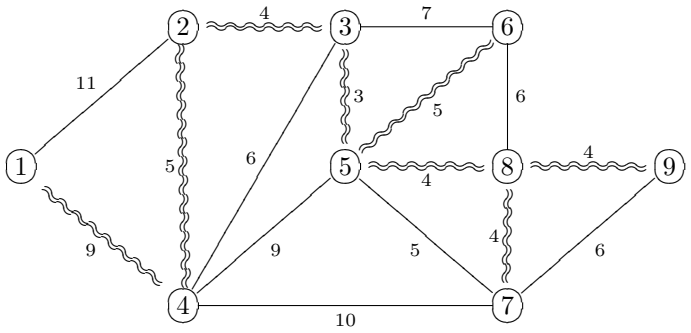
(5) Den billigste vej, der forbinder G^T med resten af grafen, er vejen fra 5 til 8, dvs. vi opdaterer vores delgraf med kanten (5,8) med vægtning $w_{5,8} = 4$, så vi får $G^T := (V^T, E^T)$ med $V^T = \{1, 2, 3, 4, 5, 8\}$ og $E^T = \{(1, 4), (2, 3), (2, 4), (3, 5), (5, 8)\}$.

(6) Der er nu to billigste veje, der forbinder G^T med resten af grafen. Vi vælger så vejen fra 8 til 7, dvs. vi opdaterer vores delgraf med kanten (7,8) med vægtning $w_{7,8} = 4$, så vi får $G^T := (V^T, E^T)$ med $V^T = \{1, 2, 3, 4, 5, 7, 8\}$ og $E^T = \{(1, 4), (2, 3), (2, 4), (3, 5), (5, 8), (7, 8)\}$.

(7) Den billigste vej, der forbinder G^T med resten af grafen, er vejen fra 8 til 9, dvs. vi opdaterer vores delgraf med kanten (8,9) med vægtning $w_{8,9} = 4$, så vi får $G^T := (V^T, E^T)$ med $V^T = \{1, 2, 3, 4, 5, 7, 8, 9\}$ og $E^T = \{(1, 4), (2, 3), (2, 4), (3, 5), (5, 8), (7, 8), (8, 9)\}$.

(8) Der er nu to billigste veje, der forbinder G^T med resten af grafen. Vi vælger så vejen fra 5 til 6 (for at undgå kredse), dvs. vi opdaterer vores delgraf med kanten (5,6) med vægtning $w_{5,6} = 4$, så vi får $G^T := (V^T, E^T)$ med $V^T = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ og
$E^T = \{(1, 4), (2, 3), (2, 4), (3, 5), (5, 6), (5, 8), (7, 8), (8, 9)\}$. Da nu $V = V^T$, er vi færdige, og G^T definerer et spanning tree for G.

Vægten af vores spanning tree er så $9 + 5 + 4 + 3 + 4 + 4 + 4 + 5 = 38$. Vores minimum weight spanning tree kan ses i Figur 2 (de dobbelt-bølgede kanter er dem fra G^T).

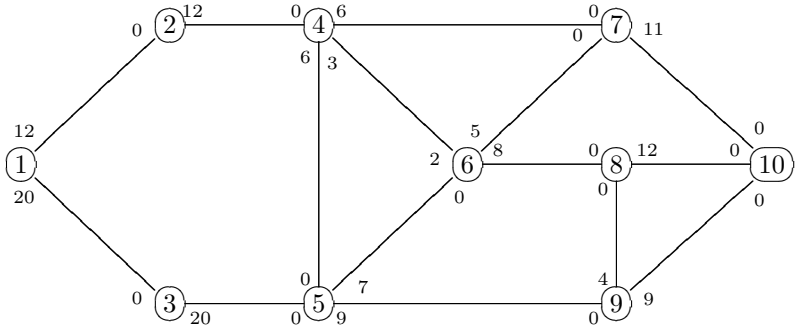


Figur 2: Minimum weight spanning tree til Opgave 2

Opgave 3

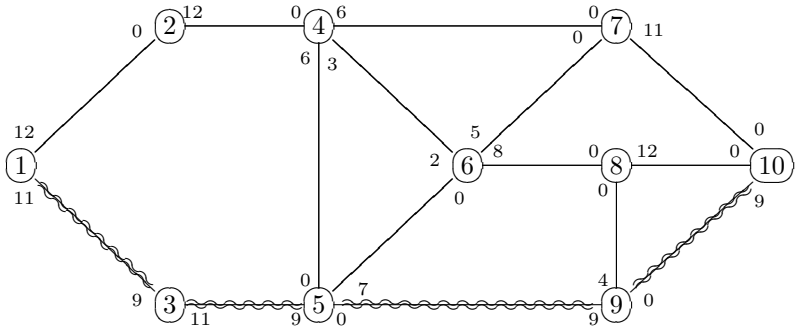
(a)

Vi tegner først grafen for maximum flow-problemet, idet vi sætter residuale kapaciteter på begge retninger på alle kanter (vi kan så fjerne pilene på kanterne):



Figur 3: Initial graf for maximum flow-problemet

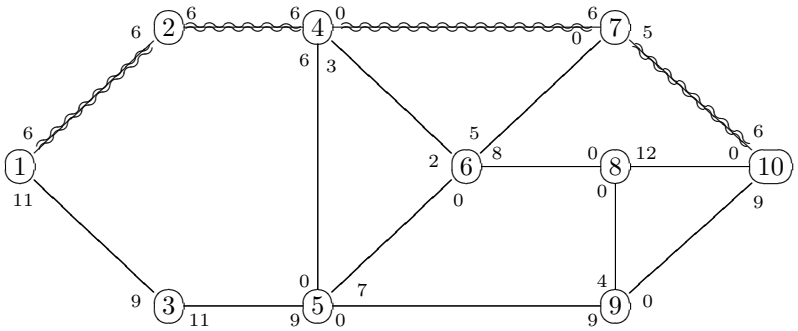
Vi anvender Augmenting Path-algoritmen fra bogen og finder nu en mulig vej fra source til sink, hvor der kan transporteres en positiv mængde gods. Vi vælger vejen 1-3-5-9-10, der samlet kan transportere $c^* = 9$, da mængden begrænses af strækningen mellem knuderne 5 og 9. Derefter nedjusterer vi de residuale kapaciteter på de anvendte strækninger i positiv retning og opjusterer dem i negativ retning, som vist på Figur 4:



Figur 4: Efter første iteration

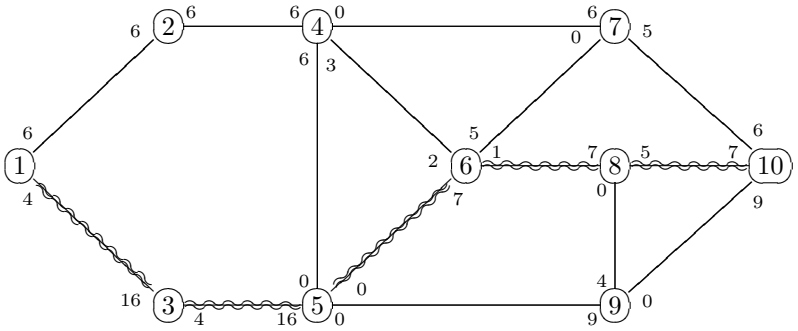
Vi finder nu igen en mulig vej fra source til sink, hvor der kan transporteres en positiv mængde gods. Vi vælger vejen 1-2-4-7-10, der samlet kan transportere

$c^* = 6$, da mængden begrænses af strækningen mellem knuderne 4 og 7. Derefter nedjusterer vi de residuale kapaciteter på de anvendte strækninger i positiv retning og opjusterer dem i negativ retning, som vist på Figur 5:



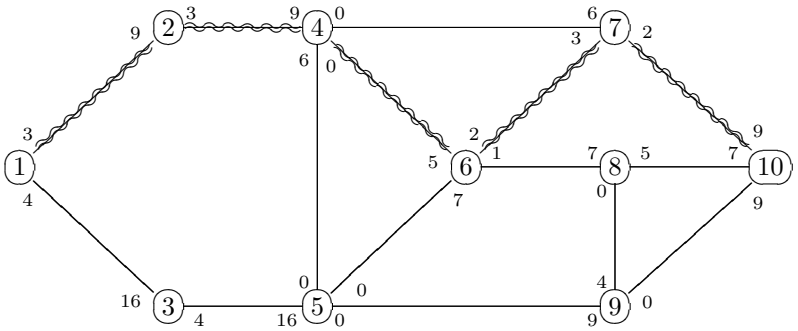
Figur 5: Efter anden iteration

Vi finder nu igen en mulig vej fra source til sink, hvor der kan transporteres en positiv mængde gods. Vi vælger vejen 1-3-5-6-8-10, der samlet kan transportere $c^* = 7$, da mængden begrænses af strækningen mellem knuderne 5 og 6. Derefter nedjusterer vi de residuale kapaciteter på de anvendte strækninger i positiv retning og opjusterer dem i negativ retning, som vist på Figur 6:



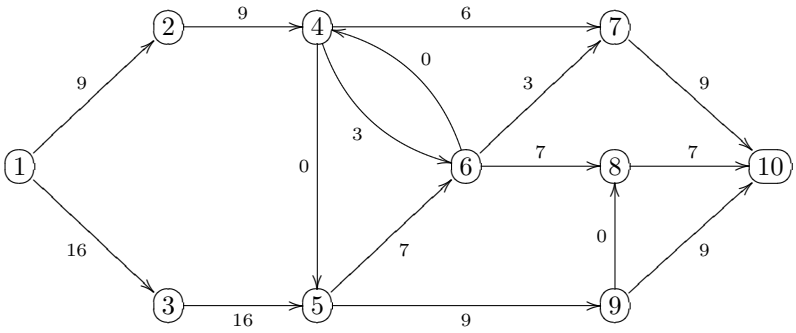
Figur 6: Efter tredje iteration

Vi finder nu igen en mulig vej fra source til sink, hvor der kan transporteres en positiv mængde gods. Vi vælger vejen 1-2-4-6-7-10, der samlet kan transportere $c^* = 3$, da mængden begrænses af strækningen mellem knuderne 4 og 6. Derefter nedjusterer vi de residuale kapaciteter på de anvendte strækninger i positiv retning og opjusterer dem i negativ retning, som vist på Figur 7:



Figur 7: Efter fjerde iteration

Vi bemærker nu, at vi ikke længere har mulighed for at sende varer gennem systemet fra 1 til 10 (vi kan ikke komme længere end 4 eller 5), derfor stopper algoritmen her, og vores maximum flow bliver 9+6+7+3=25. Til slut vil vi så gengive en oversigtsgraf, der viser, hvor meget der skal sendes mellem hver knude:



Figur 8: Sidste oversigtsgraf til Opgave 3

(b)

Vi skal nu opskrive ligningssystemet for problemet fra (a). Først konstaterer vi, at vi skal maksimere outputtet fra node 1 (ad de to kanter, der går ud derfra). Dette bliver så vores objektfunktion. Vi har desuden en max-grænse for hver af kanterne, for hvor meget der kan sendes igennem den pågældende kant. Denne constraint er (noget sammenskrevet!) vist i (1). Derudover skal vi sørge for, at der sendes lige så meget ud fra hver node, som der kommer ind. Dette er der taget hensyn til i (2)-(9). Til sidst skal vores variable selvfølgelig være større end eller lig 0:

$$\text{Maximize } Z{=}x_{1,2} + x_{1,3}$$

$$\text{subject to:}$$

$$(x_{1,2}, x_{1,3}, x_{2,4}, x_{3,5}, x_{4,5}x_{4,6}, x_{4,7}, x_{5,6}\ ,\ x_{5,9}, x_{6,4}, x_{6,7}, x_{6,8}, x_{7,10}, x_{8,10}, x_{9,8}, x_{9,10})^T$$

$$\leq (12, 20, 12, 20, 6, 3, 6\,, 7, 9, 2, 5, 8, 11, 12, 4, 9)^T \tag{1}$$

$$-x_{1,2}{-}x_{2,4} = 0 \tag{2}$$

$$-x_{1,3}{+}x_{3,5} = 0 \tag{3}$$

$$-x_{2,4} - x_{6,4} + x_{4,5} + x_{4,6}{+}x_{4,7} = 0 \tag{4}$$

$$-x_{3,5} - x_{4,5} + x_{5,6}{+}x_{5,9} = 0 \tag{5}$$

$$-x_{4,6} - x_{5,6} + x_{6,4} + x_{6,7}{+}x_{6,8} = 0 \tag{6}$$

$$-x_{4,7} - x_{6,7}{+}x_{7,10} = 0 \tag{7}$$

$$-x_{6,8} - x_{9,8}{+}x_{8,10} = 0 \tag{8}$$

$$-x_{5,9} + x_{9,8}{+}x_{9,10} = 0 \tag{9}$$

$$x_{ij} \geq 0,\ i = 1, \ldots, 9\,,\ j = 2, \ldots, 10.$$