The hub-and-spoke model

A Tutorial

introducing the reader to the subject matters of transportation networks, linear programming (the direct program, the dual program and complementary slackness), programming with 0-1 variables, and coding in GAMS (general algebraic modeling system).

Written in August 2005 in Carvoeiro, Portugal, by Sten Thore and Fedele Iannone.

Contents.

1. Introduction. In the popular press, the pros and cons of the hub-and-spoke model has been discussed for a good many years as applied to the airline industry, to shipping, to overnight express delivery, and many other activities of transportation. For instance, American Airlines at an early point established hubs in the airports at Dallas, Fort Worth and Kennedy, New York and a few other large cities across the nation. Huge aircraft served the trunk-lines between the hubs, and smaller aircraft (including the adjunct airline American Eagle) served the spokes radiating from each hub to smaller cities. More recently, the same model has come under criticism as discount airlines like Easy Jet and Ryanair in Europe fly passengers directly nonstop from Manchester and Bristol to Faro or Alicante, thus avoiding the established hubs at London, Lisboa and Madrid trafficced by BEA and Iberia.

Operations analyst have long distinguished a series of "model types" in economic logistics, such as the transportation model, the network model, and so on. It will here be suggested that among these, the hub-and-spoke model deserves separate recognition and study. To demonstrate its main characteristics, small stylized numerical examples are presented.

Figure 1 illustrates the case of a single homogeneous commodity being transported via a hub-and-spoke network. The network features three kind of *nodes:* supply nodes (nodes 1 and 2), hub nodes (nodes 3 and 4) and demand nodes (nodes 5 and 6). In addition, although not shown here, there could be one or several transshipment nodes interposed between a supply node and a hub, between two or several hub nodes, or between a hub and a demand node. The network further consists of *directed arcs* or *links* connecting pairs of nodes. The traffic in Figure 1 flows along four links (1,3), (1,4), (1,5), (1,6) originating from node 1, four links (2,3), (2,4), (2,5), (2,6) originating from node 2, three links (3,4), (3,5), (3,6) originating from node 3, and two links (4,5) and (4,6) originating from node 4.

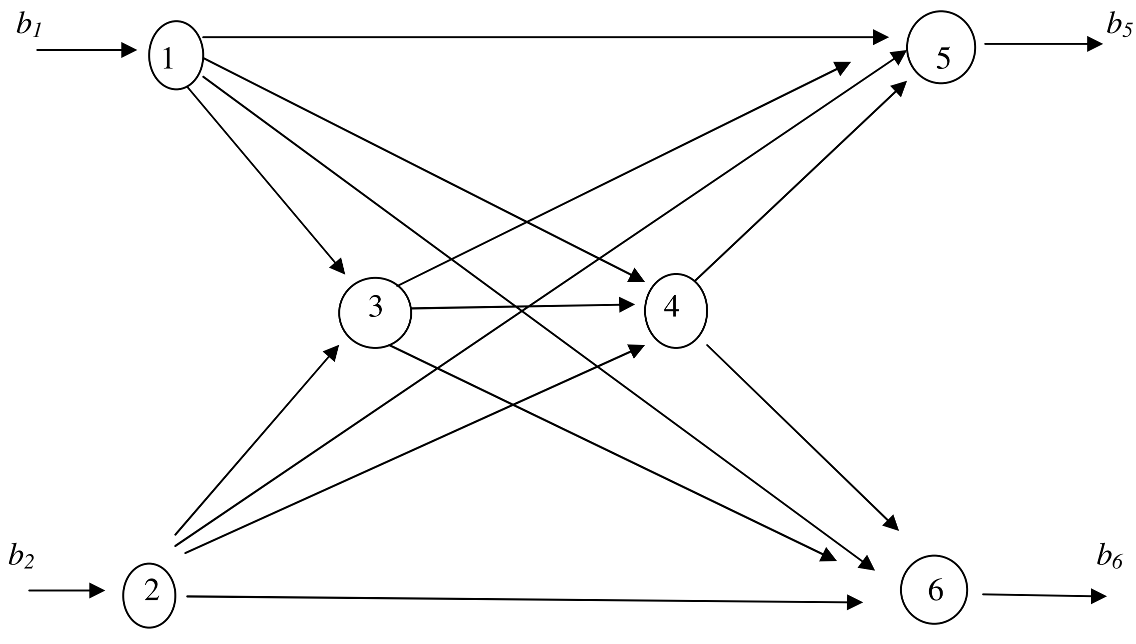

Figure 1. Hub-and-spoke model. Stylized example.

The two links (1,3) and (2,3) are *spokes* leading into hub 3, and the two links (3,5) and (3,6) are spokes leaving hub 3. The link (3,4) is an *inter-hub connection*. The two links (1,4) and (2,4) are spokes leading into hub 4. The four links (1,5) (1,6) (2,5) (2,6) are *direct destination links*. Finally, the two links (4,5) and (4,6) are spokes radiating from node 4.

There are also four *half links:* (b_1) and (b_2) representing supplies of the commodity entering the system at nodes 1 and 2, and (b_5) and (b_6) representing demands of the commodity leaving the system from nodes 5 and 6. We will associate constant supplies and demands with these half links.

Figure 1 only illustrates the traffic flow from left to right in the network. Only traffic emanating from nodes 1 and 2 is shown here. In addition, of course, there could be supply forthcoming at the hubs themselves (or, indeed, at any additional transshipment node) requiring transportation to the demand nodes. Also, only traffic with the ultimate destination of nodes 5 and 6 is shown. In addition, there could be demand present at the hubs themselves (or at any transshipment node). Traffic flow from right to left is not exhibited in Figure 1. To deal with this, a separate model would have to be constructed and solved, being a kind of mirror image of the left-to-right model. The network in Figure 1 illustrates *possible* pathways from supplies to demands via hubs and spokes. But, typically, only a subset of these links and hubs will actually be utilized in any practical application. The task ahead consists in determining the *optimal* choice out of all these possible choices. Precisely what these optimal pathways will look like, is too early to say right now. For instance, the optimal transportation system could be any of the three below:

(i) Direct nonstop transportation directly from the supply nodes to the demand nodes, circumventing the hubs entirely. See Fig.1a.

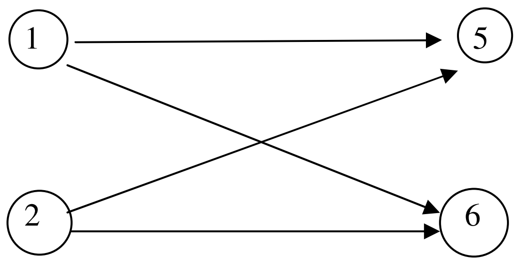

Fig. 1a. All traffic nonstop from supply nodes to demand nodes.
No utilization of hubs.

(ii) Sending all traffic via hub 3. See Fig 1b. (Or, sending all traffic via hub 4 only).

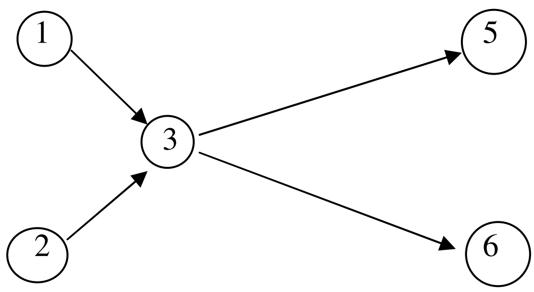

Fig. 1b. All traffic routed via a single hub.

(iii) All shipments passing first node 3, and then node 4. See Fig 1c.

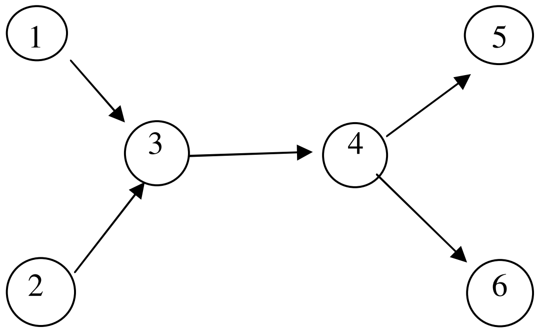

Fig. 1c. All traffic routed via both hubs.

To repeat, the possible configurations 1a-1c are just examples of the general structure exhibited in Figure 1.

Next, returning to the full-fledged Figure 1, we define *the node-link incidence matrix* of the network. It has one column for each link and one row for each node. Thus, there are 13 columns and 6 rows. See Figure 2 below.

		links												
		(1,3)	***(1,4)***	***(1,5)***	***(1,6)***	***(2,3)***	***(2,4)***	***(2,5)***	***(2,6)***	***(3,4)***	***(3.5)***	***(3.6)***	***(4,5)***	***(4,6)***
n	***1***	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0
o	***2***	0	0	0	0	-1	-1	-1	-1	0	0	0	0	0
d	***3***	1	0	0	0	1	0	0	0	-1	-1	-1	0	0
e	***4***	0	1	0	0	0	1	0	0	1	0	0	-1	-1
s	***5***	0	0	1	0	0	0	1	0	0	1	0	1	0
	6	0	0	0	1	0	0	0	1	0	0	1	0	1

Figure 2. The incidence matrix for the network in Figure 1.

There are exactly two non-zero entries in each column: -1 at the originating node of the link and +1 at the destination node. For example, in the first column labelled link (1,3) there is a -1 in row 1 and a +1 in row 3; all other entries are in the column 0.

Now introduce mathematical notation as follows:

i,j =	indices used to designate the nodes 1,2,3,4,5,6
M	incidence matrix (see Figure 2; the dimensions are 6×13)
$b = [b_i]$	a six component column vector of supplies and demands, in the present instance $b = (-b_1, -b_2, 0, 0, b_5, b_6)$. Supplies are entered with a minus sign, demands with a plus sign.
$x = [x_{ij}]$	a thirteen component column vector of shipments along the links, that is $x = (x_{13}, x_{14}, x_{15}, x_{16}, x_{23}, x_{24}, x_{25}, x_{26}, x_{34}, x_{35}, x_{36}, x_{45}, x_{46})$

It is then possible to write the *conservation of flow conditions* as in equation (1) below:

(1) $$Mx \geq b$$

If these equations are written out, it is easy to see that there is one balancing condition for each node, and that the condition simply states that the sum of all flows into a given node cannot fall short of the sum of all flows leading out of that node.

Note in particular the meaning of the inequality signs (≥). For the two first nodes, the balancing conditions say that the supply at the node must suffice to cover the flow leaving the node. The standard situation is no doubt that each supply exactly equals the quantities routed from the supply node further on into the network. But there is also the possibility that an excess supply may be present, a build-up of inventory at the supply node that just stays there. For the hubs, the balancing conditions say that the flow entering a each hub must suffice to cover the flow leaving the hub. Again, there is the possibility of a build-up of inventory at the node. Finally, for each demand node, the balancing conditions say that the deliveries forthcoming at each demand node must suffice to cover the demand. If it exceeds demand, a build-up of inventory at the demand node will occur.

2. A Linear Programming Model. We show a simple linear programming model that determines the optimal routing throughout the hub-and-spoke network, minimizing total costs. To that end, use the notation

$c = [c_{ij}]$ a thirteen component row vector of unit costs of shipments along the links, that is $c = (c_{13}, c_{14}, c_{15}, c_{16}, c_{23}, c_{24}, c_{25}, c_{26}, c_{34}, c_{35}, c_{36}, c_{45}, c_{46})$

The total cost cumulating along all links is then cx (the thirteen component row vector c is premultiplied by a thirteen component column vector x).

The linear programming model then reads

(2) *minimize* cx
subject to
$Mx \geq b,$
$x \geq 0$

The compact notation in (2) can be spelled out in more detail by writing the so-called *data box* of the problem, as in Fig. 3 overleaf.

The top row lists all the unknowns of the problem. These are the flows x_{ij} along the links (i,j), listed in order. The main body of the data box consists of the incidence matrix. Reading the top row and the incidence matrix together, they form the six left hand sides appearing in the constraint set $Mx \geq b$. The last column lists the b vector. Together, the seven first rows of the box exhibit the entire constraint set $Mx \geq b$.

The bottom row lists the unit costs. Reading the top row and the bottom row together, they form the minimand cx.

	$x_{13,}$	x_{14}	x_{15}	x_{16}	x_{23}	x_{24}	x_{25}	x_{26}	x_{34}	x_{35}	x_{36}	x_{45}	x_{46}	
u_1	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	$\geq -b_1$
u_2	0	0	0	0	-1	-1	-1	-1	0	0	0	0	0	$\geq -b_2$
u_3	1	0	0	0	1	0	0	0	-1	-1	-1	0	0	≥ 0
u_4	0	1	0	0	0	1	0	0	1	0	0	-1	-1	≥ 0
u_5	0	0	1	0	0	0	1	0	0	1	0	1	0	$\geq b_5$
u_6	0	0	0	1	0	0	0	1	0	0	1	0	1	$\geq b_6$
	$c_{13,}$	c_{14}	c_{15}	c_{16}	c_{23}	c_{24}	c_{25}	c_{26}	c_{34}	c_{35}	c_{36}	c_{45}	c_{46}	

Figure 3. Data box of linear programming problem.

Why are the two first constraints entered withy minus signs as exhibited? Would it not be more natural to multiply through with -1 and enter those to first inequalities with ≤ signs instead? The answer is that in a *minimization* problem we prefer to write all inequalities with a ≥ *s*ign. One says that the ≥ sign is *the right-way sign* in a minimization problem. The opposite sign ≤ is the *wrong-way* sign in a minimization problem. Conversely, the ≤ sign is *right-way* sign in a *maximization* problem, and the ≥ sign is the *wrong-way.*

As already explained, the inequality signs open the door for the possible buildup of inventories, at the supply nodes, at the hubs, or at the demand nodes. Such buildup will obviously occur at the supply nodes, if the sum total of all demand falls short of all supply to the system. But total demand cannot exceed total supply. To see this, add all the balancing constraints together, and one finds

(3) $0 \geq -b_1 - b_2 + b_5 + b_6$

Example 1. We show a simple stylized example. It is so simple so that the solution is obvious, from direct inspection. But the example is structured enough to convey some of the characterics of a big real-life problem.

Let the given supplies be $b_1 = 100$ and $b_2 = 150$, and the desired demands $b_5 = 125$ and $b_6 = 125$. Let the unit shipping costs be given as in the table below (only links actually present in the network Figure 1 are entered).

From \ to	***j = 1***	***j = 2***	***j = 3***	***j = 4***	***j = 5***	***j = 6***
i = 1			8	5	20	23
i = 2			10	7	18	25
i = 3				12	18	20
i = 4					4	9
i = 5						
i = 6						

The programming problem (2) then reads:

(4) $$\begin{aligned} \textit{minimize}\ \ & 8x_{13,} + 5x_{14} + 20x_{15} + 23x_{16} + 10x_{23} + 7x_{24} + 18x_{25} + 25x_{26} + 12x_{34} \\ & + 18x_{35} + 20x_{36} + 4x_{45} + 9x_{46} \end{aligned}$$

subject to

$$\begin{array}{lr} -x_{13} - x_{14} - x_{15} - x_{16} & \geq -100 \\ -x_{23} - x_{24} - x_{25} - x_{26} & \geq -150 \\ x_{13,} + x_{23} - x_{34} - x_{35} - x_{36} & \geq 0 \\ x_{14} + x_{24} + x_{34} - x_{45} - x_{46} & \geq 0 \\ x_{15} + x_{25} + x_{35} + x_{45} & \geq 125 \\ x_{16} + x_{26} + x_{36} + x_{46} & \geq 125 \\ x_{13}, x_{14}, x_{15}, x_{16}, x_{23}, x_{24}, x_{25}, x_{26}, x_{34}, x_{35}, x_{36}, x_{45}, x_{46} & \geq 0 \end{array}$$

or, using the data-box format Figure 4:

	$x_{13,}$	x_{14}	x_{15}	x_{16}	x_{23}	x_{24}	x_{25}	x_{26}	x_{34}	x_{35}	x_{36}	x_{45}	x_{46}	
u_1	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	≥ -100
u_2	0	0	0	0	-1	-1	-1	-1	0	0	0	0	0	≥ -150
u_3	1	0	0	0	1	0	0	0	-1	-1	-1	0	0	≥ 0
u_4	0	1	0	0	0	1	0	0	1	0	0	-1	-1	≥ 0
u_5	0	0	1	0	0	0	1	0	0	1	0	1	0	≥ 125
u_6	0	0	0	1	0	0	0	1	0	0	1	0	1	≥ 125
	8	*5*	*20*	*23*	*10*	*7*	*18*	*25*	*12*	*18*	*20*	*4*	*9*	

Figure 4. Data box of example 1.

In this simple numerical example, it is easy to find the optimal solution to program (4) by direct inspection. Note that

- the lowest-cost route from node 1 to node 5 is via nodes 1–4-5 and the unit cost is 9;
- the lowest-cost route from node 1 to node 6 is via nodes 1–4-6 and the unit cost is 14;
- the lowest-cost route from node 2 to node 5 is via nodes 2-4–5 and the unit cost is 11;
- the lowest-cost route from node 2 to node 6 is via nodes 2–4–6 and the unit cost is16.

Hence, the optimal solution obviously is to send all 100 units from node 1 to node 5 via nodes 1-4-5. Hence $x_{14}{}^* = 100$ (the asterisk denotes the optimal solution value). As to the supplies coming available at node 2, 25 units will be sent to node 5 via nodes 2-4-5 and the remaining 125 units will be sent to node 6 via nodes 2-4-6. In other words, $x_{24}^* = 150$, $x_{45}^* = 125$ and $x_{46}^* = 125$. All other unknowns are zero. See Figure 5 overleaf.

The total minimal cost is then $5x_{14} + 7x_{24} + 4x_{45} + 9x_{46} = 5\times 100 + 7\times 150 + 4\times 125 + 9\times 125 = 3175.$

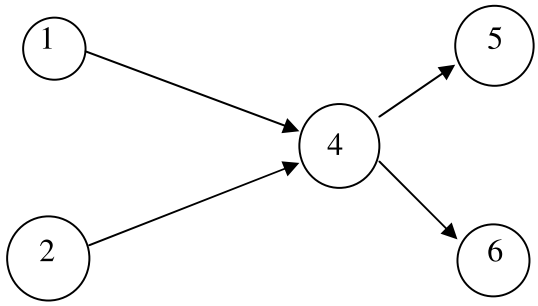

Fig.5. Optimal solution to numerical example. All traffic routed via hub 4.

What is the purpose of the data box? Would it not be much easier just to write down the programming version (4) right away, without bothering about the incidence matrix? The constraints in program (4) after all have their own immediate interpretation: they state, in order, the flow conservation condition for each node of the network (the total inflow into a given node must suffice to cover the total outflow from that node).

The answer is that the data box provides access to the corresponding dual program and a powerful saddle-point principle to which we now turn.

3. The dual program and the conditions of complementary slackness. Refer now to the entries in the first column of the data box Figure 3. We shall call them the *shadow prices* of the traffic flow implied at the six nodes of the network. There is one shadow price at each originating node, at each hub, and at each destination node. Arrange the six shadow prices as the row vector $u = (u_1, u_2\ u_3, u_4, u_5, u_6)$.

A surprising property of any *direct* linear program such as program (4) is that there exists a corresponding *dual* program – a program in an entirely different set of unknown variables but using the same data box. The numerical solution to this dual program yields important insight into the nature of direct problem. In the history of linear programming, the discovery of the dual program ranks as one of the great mathematical discoveries of the 20^{th} century.

When the direct problem is one of minimization of costs, the dual program is one of maximizing value. In the present instance, we shall be seeking to maximize the total appreciation of *shadow value* (= flow volume times the shadow price) cumulating in the network. It is obtained as the shadow value of the transport exit flow $125u_5 + 125u_6$ calculated at the destination nodes minus the shadow costs of the flow entering the origins $100\ u_1\ + 150\ u_2$. So, the net appreciation of value is $125u_5 + 125\ u_6 - 100u_1\ – 150u_2$.

Furthermore, there are the dual constraints. There will be one dual constraint for each and every link in the transportation network. It will state an important principle that is referred to as *exhaustion of value*. It states that along any positive flow in the network, *the increase in shadow price must be exactly exhausted by the unit transportation cost*. That is, all cumulating shadow price must have a source that can be accounted for. Speaking in terms of values, all cumulating value can be accounted for. No free value can arise.

Consider for example the link from origin 1 to hub 3. The shadow price of the flow at node 1 is u_1; the shadow price of the flow at hub 3 is u_3. The exhaustion of value condition then states that $-u_1 + u_3 \leq 8$. In plain English, the shadow price at the hub cannot exceed the shadow cost at node 1 plus the unit shipping cost 8. Actually, if there is a positive transportation flow along the link, the shadow price at the hub 3 must equal the shadow price at node 1 plus the unit shipping cost. A hypothetical shipper shipping goods from node 1 to hub 3 will then *break exactly even*. But there is also the possibility that the shadow value at hub falls short of the shadow cost at node 1 plus the shipping cost 8. In that case, the shipper would suffer an i*mputed loss* and the shipment is not worth his while. No shipment will take place.

(In the real world, profits occur all the time, of course. But even so, profits can always be ascribed to some cause, to some tangible or untangible production factor that the shipper commands. This factor will then have an imputed shadow price, so that the total shadow value accumulating can always formally be traced back to a list of contributing factors. The dual program establishes such a value-exhausting imputation scheme).

In general, the dual constraint for link *(i,j)* reads:

$$-u_i + u_j \leq c_{ij}$$

(Note that we arrange matters so that the inequality sign is oriented as $\leq$. In a maximization problem, the *right way* signs are $\leq$. But remember that in a minimization problem, the *right way* signs are $\geq$).

The entire dual program to (4) now reads:

(5) $\quad maximize\ -100\,u_1 - 150\,u_2 + 125\,u_5 + 125\,u_6$
subject to

- $-u_1 + u_3 \leq 8$
- $-u_1 + u_4 \leq 5$
- $-u_1 + u_5 \leq 20$
- $-u_1 + u_6 \leq 23$
- $-u_2 + u_3 \leq 10$
- $-u_2 + u_4 \leq 7$
- $-u_2 + u_5 \leq 18$
- $-u_2 + u_6 \leq 25$
- $-u_3 + u_4 \leq 12$
- $-u_3 + u_5 \leq 18$
- $-u_3 + u_6 \leq 20$
- $-u_4 + u_5 \leq 4$
- $-u_4 + u_6 \leq 9$
- $u_1, u_2, u_3, u_4, u_5, u_6 \geq 0$

In words, the program asks us to maximize the total net appreciation of shadow value over the network, subject to the conditions that such appreciation created along each and every link be exhausted by its costs. Finally, all the unknown dual variables have to be nonnegative. What this means is that the shipped commodity at some nodes may be *scarce*, i.e. command a positive shadow price. Or the commodity may be a *free good*, in which case the shadow price is zero. This happens if there is a buildup of unwanted inventory at the node, so that an excess availability is created. But the shadow price cannot be negative.

Just as we could solve the simple direct problem by inspection, we can also immediately figure out the numerical solution to the dual problem (5). Remember that we have already from the direct problem found that the only active traffic links of the network are the links (1,4), (2,4), (4,5) and (4,6). Along these links, and these links only, the buildup of unit shadow value will then equal the unit cost. (Along all other links, the buildup of shadow value will fall short of the unit cost, so that a hypothetical shipper will not find it worth while to undertake the transportation in question). So, buildup of unit shadow value along link (1,4) will be 5, the buildup along link (2,4) will be 7, the buildup along link (4,5) will be 4 and the buildup along link (4,6) will be 9.

A moment of afterthought will show that these conditions only determine the increase in unit shadow value along the links of the network, but not their absolute size. Denote the unit shadow value at node 2 by the letter a, standing for an arbitrary positive constant. Then the unit shadow value at node 1 is $a + 2$ so that the resulting shadow value at node 4 is $a + 7$. The resulting unit shadow value at node 5 is then $a + 11$, and the value at node 6 is $a + 16$. So, here we learn something important: the optimal solution to the dual program need not be *unique.* In the present case, there exist an infinity of optimal solutions, one for each nonnegative value assigned to the constant a.

The optimal value of the dual program is in any case $-100\,(a + 2) - 150a + 125\,(a + 11) + 125\,(a + 16) = 3175$. The constant a cancels. The *value* of the dual program (5) is unique. Furthermore, we also notice that the optimal value of the dual program equals the optimal value of the direct program (4). Here we encounter the principle of exhaustion of value again: The total increase in shadow value over the network equals the total transportation cost. This is the famous *duality theorem of linear programming*: the optimal value of the direct problem equals the optimal value of the dual problem.

Clearly, the direct variables and the dual variables are intimately tied to each other. These *complementing* properties can be stated in the form of the following four propositions of *complementary slackness,* the nature of which we have already indicated:

- If it turns out that there are excess deliveries of flow at any node, the shadow price at that node vanishes. The transported commodity is then a free good at the node.
- But if the commodity at the node is scarce, i.e. if it commands a positive shadow price, then outflow from the node exactly equals the total inflow into the node.
- If the appreciation of shadow price along a link in the network falls short of the unit transportation cost, the flow along this link is zero. (A hypothetical shipper would suffer a unit loss so he withdraws).
- But if a positive flow occurs along the link, then the appreciation of shadow price must exactly equal the unit cost. (A hypothetical shipper finds that his unit costs are exactly covered).

Since these propositions are so important, we also state them mathematically. The two first propositions are contained in the conditions, to hold at the point of optimum:

(6)
$$
\begin{aligned}
&- x_{13} - x_{14} - x_{15} - x_{16} + 100 \geq 0 \quad \text{and} \quad u_1(- x_{13} - x_{14} - x_{15} - x_{16} + 100) = 0\\
&- x_{23} - x_{24} - x_{25} - x_{26} + 150 \geq 0 \quad \text{and} \quad u_2(- x_{23} - x_{24} - x_{25} - x_{26} + 150) = 0\\
&x_{13,} + x_{23} - x_{34} - x_{35} - x_{36} \geq 0 \quad \text{and} \quad u_3(x_{13} + x_{23} - x_{34} - x_{35} - x_{36}) = 0\\
&x_{14} + x_{24} + x_{34} - x_{45} - x_{46} \geq 0 \quad \text{and} \quad u_4(x_{14} + x_{24} + x_{34} - x_{45} - x_{46}) = 0\\
&x_{15} + x_{25} + x_{35} + x_{45} - 125 \geq 0 \quad \text{and} \quad u_5(x_{15} + x_{25} + x_{35} + x_{45} - 125) = 0\\
&x_{16} + x_{26} + x_{36} + x_{46} - 125 \geq 0 \quad \text{and} \quad u_6(x_{16} + x_{26} + x_{36} + x_{46} - 125) = 0\\
&u_1, u_2, u_3, u_4, u_5, u_6 \geq 0
\end{aligned}
$$

The two latter propositions of complementary slackness are contained in the conditions, to hold at the point of optimum:

(7) $8 + u_1 - u_3 \geq 0$ and $x_{13}(8 + u_1 - u_3) = 0$
– $5 + u_1 - u_4 \geq 0$ and $x_{14}(5 + u_1 - u_4) = 0$
– $20 + u_1 - u_5 \geq 0$ and $x_{15}(20 + u_1 - u_5) = 0$
– $23 + u_1 - u_6 \geq 0$ and $x_{16}(23 + u_1 - u_6) = 0$
– $10 + u_2 - u_3 \geq 0$ and $x_{23}(10 + u_2 - u_3) = 0$
– $7 + u_2 - u_4 \geq 0$ and $x_{24}(8 + u_2 - u_4) = 0$
– $18 + u_2 - u_5 \geq 0$ and $x_{25}(18 + u_2 - u_5) = 0$
– $25 + u_2 - u_6 \geq 0$ and $x_{26}(8 + u_2 - u_6) = 0$
– $12 + u_3 - u_4 \geq 0$ and $x_{34}(12 + u_3 - u_4) = 0$
– $18 + u_3 - u_5 \geq 0$ and $x_{35}(18 + u_3 - u_5) = 0$
– $20 + u_3 - u_6 \geq 0$ and $x_{36}(20 + u_3 - u_6) = 0$
– $4 + u_4 - u_5 \geq 0$ and $x_{45}(4 + u_4 - u_5) = 0$
– $9 + u_4 - u_6 \geq 0$ and $x_{46}(9 + u_4 - u_6) = 0$

– $x_{13}, x_{14}, x_{15}, x_{16}, x_{23}, x_{24}, x_{25}, x_{26}, x_{34}, x_{35}, x_{36}, x_{45}, x_{46} \geq 0$

The first set of constraints in (6) are simply the constraints of the direct program. The first set of constraints in (7) are simply the dual constraints. As to the the interpretation of the set of equalities in both (6) and (7), note that if two numbers A and B satisfy $AB = 0$ and $A,B \geq 0$ then the following conclusion can be drawn: (i) if $A > 0$ then $B = 0$, and (ii) if $B > 0$ then $A = 0$.

A final remark. What is the purpose of the data box Figure 4? Answer: It permits us to read off the dual program directly. In effect, the maximand in program (5) can be read off the first column, and the constraints from the following columns, read one by one. So, the data box is a tool for writing down the dual program quite mechanically, without having to check the meaning of every single dual constraint.

4. The GAMS program. To solve linear programs numerically, we turn to the program package GAMS (=general algebraic modeling system). Refer to the program HUBSPOKE below which translates the linear program (4) into GAMS statements.

To prepare for later GAMS hub and spoke programs, the nodes 1-6 have here been labeled Bristol, Chester, London, Rome, Naples and Messina. (Naples is called NP rather than NA because NA is a forbidden symbol in GAMS- - it means "not ascertained"). Here, our passenger traffic originates at Bristol and Chester and is bound for Naples and Messina. This may seem a rather contrived example but we shall later allow for traffic originating in London and traffic destined for Rome as well. Further, we shall also – in the section on "multipage programs" - recognize that traffic originating at any originating node of course has its own mix of destinations. That is, one really needs to specify the number of passengers that desire transportation from Bristol to Naples and from Bristol to Messina (rather than the overall traffic ending up in those two places).

The GAMS notation should be self-explanatory but note the following points: there is free format so that the placement of GAMS statements on the page does not matter. All required statements are here printed upper case, and comment statements lower case. Statements starting with an asterisk in the first position are comment lines. Each group of commands (like SETS, ALIAS, PARAMETERS, TABLE and so on) need to end with a semicolon. The POSITIVE VARIABLES section lists the *nonnegative* variables, not the positive variables. (I have been told that the authors of GAMS once hoped to make GAMS available in high schools and the expression "nonnegative" was thought to be too sophisticated). The EQUATIONS section names the minimand and each constraint. Next, each EQUATION is spelled out mathematically. Strangely enough, the colon following the name of each EQUATION is written (..) rather than (:). The equality sign is noted =E=, and the two inequality signs are denoted =L= (equal to or less than) and =G= (equal to or greater than). Finally, the model is named, and the computer is told that ALL equations listed should participate in the model.

```
*Program HUBSPOKE solves the numerical example of the linear program (4).
*Here the nodes will be identified with three cities in England and three cities
*in Italy.

SETS
   N cities /BR    Bristol
             CH    Chester
             LO    London
             RO    Rome
             NP    Naples
             ME    Messina/
   I(N)  origins              /BR,CH/
   J(N)  hubs                 /LO,RO/
   K(N)  destinations         /NP,ME/;

ALIAS
   (N,N1)
   (J,J1)   ;

PARAMETERS
   SUPPLY(N) vector of supplies
   /BR      100
    CH      150
    LO        0
    RO        0
    NP        0
    ME        0/

   DEMAND(N) vector of demands
   /BR        0
    CH        0
    LO        0
    RO        0
    NP      125
    ME      125/ ;

TABLE
   UNITCOST(N,N1)
                BR      CH      LO      RO      NP      ME
       BR       99      99       8       5      20      23
       CH       99      99      10       7      18      25
       LO       99      99      99      12      18      20
       RO       99      99      99      99       4       9
       NP       99      99      99      99      99      99
       ME       99      99      99      99      99      99  ;
*The unit costs 99 are large enough to make sure that these links will not be
*used(they do not exist)
```

```
VARIABLES
   X(N,N1)  traffic flow from N to N1
   OBJ      objective function in minimization program;
POSITIVE VARIABLES
   X;

EQUATIONS
   COST          defines total cost
   ORIGIN(I)     observe supply limits at origin I
   TRANS(J)      transshipment at hub J
   DESTIN(K)     cover demand at destination K;

COST..           OBJ =E= SUM((N,N1),UNITCOST(N,N1)*X(N,N1));
ORIGIN(I)..      - SUM(N,X(I,N))                =G= -SUPPLY(I);
TRANS(J)..       SUM(N,X(N,J)) - SUM(N,X(J,N))  =G= DEMAND(J) - SUPPLY(J);
DESTIN(K)..      SUM(N,X(N,K))                  =G= DEMAND(K);

MODEL HUBSPOKE /ALL/;
SOLVE HUBSPOKE USING LP MINIMIZING OBJ;
```

5. A multi-page format. Environmental constraints, and the shadow prices of such constraints. The next model introduces several features of the real world that have been neglected so far. First, we introduce the shipping of goods (or people) where each shipped unit has *a specific origin and a specific destination.* The demand requirements are then not just that a certain total arrive at each destination (irrespective of their origins), but that all individual customers are satisfied, both at the origins and at the destinations.

Example 2. To illustrate, assume this time that the supplies and demands are as exhibited in the table below:

Origins\ destinations	*Node 4*	*Node 5*	*Node 6*	*Sum supplies at each origin*
Node 1	40	*60*	*40*	140
Node 2	30	*65*	*85*	180
Node 3	100	100	20	220
Sum demands at each destination	170	225	145	540

Figure 6. Data for Example 2.

The supply at node 1 is 140. Out of this total, 40 units are looking for transportation to hub 4 to cover local demand there, 60 units are looking for transportation to node 5 and 40 units to node 6. The supply at node 2 is 180. Out of this total, 30 units are looking for transportation to hub 4 to cover local demand 65 units are demanded at node 5 and 85 units are demanded at node 6. Finally, there is now also a local supply at hub 3 requiring transportation to nodes 4, 5 and 6; 100 units are demanded locally at hub 4, 100 units are earmarked for node 5 and 20 units are earmarked for node 6.

To deal with this situation, we formulate not one linear program but *three* linear programs, one for each originating node. The data box for the first program (dealing with the traffic originating at node 1) is shown in Table 7 below.

	$x_{13,}$	x_{14}	x_{15}	x_{16}	x_{23}	x_{24}	x_{25}	x_{26}	x_{34}	x_{35}	x_{36}	x_{45}	x_{46}	
	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	*≥ - 140*
	0	0	0	0	-1	-1	-1	-1	0	0	0	0	0	*≥ 0*
	1	0	0	0	1	0	0	0	0	-1	-1	0	0	*≥ 0*
	0	1	0	0	0	1	0	0	1	0	0	-1	-1	*≥ 40*
	0	0	1	0	0	0	1	0	0	1	0	1	0	*≥ 60*
	0	0	0	1	0	0	0	1	0	0	1	0	1	*≥ 40*
	8	*5*	*20*	*23*	*10*	*7*	*18*	*25*	*12*	*18*	*20*	*4*	*9*	

Figure 7. Data box for all traffic originating at node 1.

The data box for the second program (dealing with the traffic originating at node 2) is displayed in Table 8. The incidence matrix and the unit costs are the same as before, but the unknowns are now denoted by the letter *y* rather than the letter *x*.

	$y_{13,}$	y_{14}	y_{15}	y_{16}	y_{23}	y_{24}	y_{25}	y_{26}	y_{34}	y_{35}	y_{36}	y_{45}	y_{46}	
	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	*≥ 0*
	0	0	0	0	-1	-1	-1	-1	0	0	0	0	0	*≥ - 180*
	1	0	0	0	1	0	0	0	0	-1	-1	0	0	*≥ 0*
	0	1	0	0	0	1	0	0	1	0	0	-1	-1	*≥ 30*
	0	0	1	0	0	0	1	0	0	1	0	1	0	*≥ 65*
	0	0	0	1	0	0	0	1	0	0	1	0	1	*≥ 85*
	8	*5*	*20*	*23*	*10*	*7*	*18*	*25*	*12*	*18*	*20*	*4*	*9*	

Figure 8. Data box for all traffic originating at node 2.

The data box for the third program (dealing with the traffic originating at hub 3) is shown in Figure 9. The incidence matrix and the unit costs are still the same, but the unknowns are now denoted by the letter *z*.

	$z_{13,}$	z_{14}	z_{15}	z_{16}	z_{23}	z_{24}	z_{25}	z_{26}	z_{34}	z_{35}	z_{36}	z_{45}	z_{46}	
	-1	-1	-1	-1	0	0	0	0	0	0	0	0	0	*≥ 0*
	0	0	0	0	-1	-1	-1	-1	0	0	0	0	0	*≥ 0*
	1	0	0	0	1	0	0	0	0	-1	-1	0	0	*≥ - 220*
	0	1	0	0	0	1	0	0	1	0	0	-1	-1	*≥ 100*
	0	0	1	0	0	0	1	0	0	1	0	1	0	*≥ 100*
	0	0	0	1	0	0	0	1	0	0	1	0	1	*≥ 20*
	8	*5*	*20*	*23*	*10*	*7*	*18*	*25*	*12*	*18*	*20*	*4*	*9*	

Figure 9. Data box for all traffic originating at hub 3.

Rather than dealing with these three separate programs, it is also possible to join them together into one big master program, formed by *minimizing total global transportation costs* (that is, the sum of the three minimands exhibited along the bottom row of Figures 7, 8 and 9). The master program has three sets of unknowns, the *x* variables, the *y* variables and the *z* variables. There are 18 constraints all together, 6 constraints for the *x* variables, 6 for the *y* variables and 6 for the *z* variables. The data box of the entire global problem has a characteristic diagonal structure: it is a *multi-page program* with one "page" of constraints for each group of unknowns.

To solve this simple multi-page program, note that it actually breaks up into its constituent three parts. Therefore, the optimal solution of the multi-page program is identical to the joint solutions of the three individual page programs. To minimize global costs or to minimize each local page cost are identical propositions.

This time the GAMS program was written a little more compactly (using the incidence matrix) as below. Actually, this GAMS program is entirely different. One and the same mathematical program can be coded in GAMS in many different ways.

The sets of origins and destinations are now named ORIG and DEST, and the letter K is this time used to designate the links. Note the use of citation marks (“ ”) in the three constraints CONSTRX, CONSTRY, CONSTRZ. The quotation marks address a particular element of a list, rather than the list itself.

```
SETS
I cities             / BR, CH, LO, RO, NP, ME/
ORIG(I) origins      / BR, CH, LO/
DEST(I) destinations / RO, NP, ME/
K links/BR-LO, BR-RO, BR-NP, BR-ME, CH-LO, CH-RO, CH-NP, CH-ME, LO-RO,
                LO-NP, LO-ME, RO-NP, RO-ME/ ;

PARAMETER
C(K) unit shipping costs
  /BR-LO    8
   BR-RO    5
   BR-NP   20
   BR-ME   23
   CH-LO   10
   CH-RO    7
   CH-NP   18
   CH-ME   25
   LO-RO   12
   LO-NP   18
   LO-ME   20
   RO-NP    4
   RO-ME    9 /


TABLE
DEMANDS(I, ORIG) traffic requirements
            BR    CH      LO

  BR      -140     0       0
  CH         0  -180       0
  LO         0     0    -220
  RO        40    30     100
  NP        60    65     100
  ME        40    85      20 ;
```

```
TABLE INCIDENCE(I,K) incidence matrix
            BR-LO  BR-RO  BR-NP  BR-ME  CH-LO  CH-RO  CH-NP  CH-ME
       BR    -1     -1     -1     -1      0      0      0      0
       CH     0      0      0      0     -1     -1     -1     -1
       LO     1      0      0      0      1      0      0      0
       RO     0      1      0      0      0      1      0      0
       NP     0      0      1      0      0      0      1      0
       ME     0      0      0      1      0      0      0      1

       +    LO-RO  LO-NP  LO-ME  RO-NP  RO-ME
       BR     0      0      0      0      0
       CH     0      0      0      0      0
       LO    -1     -1     -1      0      0
       RO     1      0      0     -1     -1
       NP     0      1      0      1      0
       ME     0      0      1      0      1    ;

VARIABLES
X(K)    traffic originating from BR
Y(K)    traffic originating from CH
Z(K)    traffic originating from LO
OBJ     objective function         ;

POSITIVE VARIABLES
X,Y,Z ;

EQUATIONS
COST  global cost
CONSTRX(I) constraints on x variables
CONSTRY(I) constraints on y variables
CONSTRZ(I) constraints on z variables;

COST..             OBJ=E=SUM(K, C(K)*X(K)) +SUM(K, C(K)*Y(K))
                        +SUM(K, C(K)*Z(K)) ;
CONSTRX(I)..       SUM(K,INCIDENCE(I,K)* X(K)) =G= DEMANDS(I,"BR");
CONSTRY(I)..       SUM(K,INCIDENCE(I,K)* Y(K)) =G= DEMANDS(I,"CH");
CONSTRZ(I)..       SUM(K,INCIDENCE(I,K)* Z(K)) =G= DEMANDS(I,"LO");

MODEL MULTPAGE/ALL/;
SOLVE MULTPAGE USING LP MINIMIZING OBJ;
```

Solving, one finds the following optimal transportation routes:

- Traffic originating at node 1: Send 140 units to node 4; of these 40 units will stay, 60 units will be shipped on to node 5 and 40 units to node 6;
- Traffic originating at node 2: Send 180 units to node 4, of these, 30 units will stay, 65 units will continue to node 5 and 85 units to node 6;
- Traffic originating at node 3: Send 200 units to node 4, of these 100 units will stay at node 4 and 100 units will continue to node 5, furthermore, 20 units will be shipped directly from node 3 to node 6.

Note that this solution spells out in individual detail the gross shipments solved for in Example 1.

Capacitating constraints. In the real world, one encounters constraints on the maximal possible traffic all the time. Such capacitating constraints may be present both along individual links (the maximum traffic that a link can handle) and at individual nodes (the maximal traffic that can pass through a node). Some capacitating constraints are set by physical constraints (the size and number of available containers, the loading capacity of available ships, the number of runways at an airport etc.), other constraints may be imposed by environmental considerations (maximal number of trucks per day that the firefighting authorities will allow to pass through the Mont Blanc tunnel, noise-abatement limits set at an airport, maximal releases of carbon dioxide gas by all sorts of combustion engines calculated to meet the Kyoto accord, and so on).

Example 3. To illustrate how such capacitating constraints at traffic hubs may impact on the optimal transportation patterns and on the operation of hubs, consider the following stylized example. Returning to Example 2, assume that there is now present an environmental constraint set on the traffic passing through hubs 3 and 4, to the effect that:

(8a) $x_{34} + x_{35} + x_{36} + y_{34} + y_{35} + y_{36} + z_{34} + z_{35} + z_{36} \le 220$

(8b) $x_{45} + x_{46} + y_{45} + y_{46} + z_{45} + z_{46} \le 275$

That is, there is an upper limit of 220 on all traffic leaving hub 3 and and upper limit of 275 on all traffic leaving hub 4.

Clearly, these restrictions are not fulfilled by the traffic of Example 2. Starting with node 3 (London) there were a total of 220 *z*-units desiring to leave the airport. These can still be acommodated, but there is no margin for accepting any additional traffic leaving London. Turning to node 4 (Rome), there was in Example 2 a total of 100 *x*-units leaving the node, a total of 150 *y*-units leaving the node, and 100 *z*-units leaving the node - - adding up to a grand total of 350 units leaving the node, thus widely overshooting the maximal limit of 275 laid down in (8b).

Let us take a brief look at the mathematical structure of the problem presently at hand. It is a multipage program, with three pages - - one page for the *x*-traffic (leaving Bristol), one page for the *y*-traffic (leaving Chester) and one page for the *z*-traffic (leaving London). Each page consists of 6 constraints, one balancing condition for each node of the network. In addition, there are the two constraints (8) which clearly tie the *x*-traffic, the *y*-traffic and the *z*-traffic together. So, this time, the global master problem *does not* break up into three separate pages. Instead, the constraints (8) are in the nature of *coupling conditions*. The term "multi-page program" is still illuminating, conjuring up the vision of a book containing many pages, and with the binding or the spine of the book tying the individual pages together.

Adding the two new constraints (8a) and (8b) to the GAMS program and solving again, one find the following optimal solution (see table Figure 10):

- Traffic originating at node 1: send 140 units to node 4; of these 40 units will stay, 60 units will be shipped on to node 5 and 40 units to node 6.
- Traffic originating at node 2: send 180 units to node 4, of these, 30 units will stay, 65 units will continue to node 5 and 85 units to node 6.
- Traffic originating at node 3: send 125 units to node 4, of these 100 units will stay at node 4 and 25 units will continue to node 5. In addition, 75 units will be shipped directly from node 3 to node 5, and 20 units will be shipped directly from node 3 to node 6.

So, the introduction of the two capacitating constraints (8a) and (8b) have the following consequences. The inter-hub traffic from node 3 (London) to node 4 (Rome) has now been reduced from 200 to 125. The outbound traffic from node 4 destined to node 5 (Naples) is curtailed correspondingly, from 100 to 25. As a result, the traffic pressure on node 4 (Rome) is eased as required. Instead, the shortfall of 75 units at node 5 (Naples) is covered by direct shipments from node 3 (thus circumvening the crowded hub at Rome). Refer to Figure 10 for a listing of the optimal flows.

Further insight into the nature of these adjustments of the traffic are delivered by the dual solution. Refer to Figure 11 for a listing of the dual variables.

Remember that there is a dual variable or a shadow price for each constraint in the direct problem. In the present instance, there is a dual variable to each of the 6 constraints of each "page". The column headed "*x*-traffic" in Figure 11 lists the dual variables or shadow prices of all traffic leaving Bristol. We have already found that this traffic is routed directly for Rome for further shipment to Naples and Messina. The shadow price is 4 at Bristol and climbs to 9 at Rome, to 15 at Naples and to 20 at Messina. The interpretation of the two remaining columns in Figure 11 is similar.

links	***x-traffic***	***y-traffic***	***z-traffic***
BR-LO	0	0	0
BR-RO	140	0	0
BR-NP	0	0	0
BR-ME	0	0	0
CH-LO	0	0	0
CH-RO	0	180	0
CH-NP	0	0	0
CH-ME	0	0	0
LO-RO	0	0	125
LO-NP	0	0	75
LO-ME	0	0	20
RO-NP	60	65	25
RO-ME	40	85	0

Figure 10. Optimal solution to Example 3.

nodes	***x-traffic***	***y-traffic***	***z-traffic***
node 1 (BR)	4	2	7
node 2 (CH)	2	0	5
node 3 (LO)	0	0	0
node 4 (RO)	9	7	12
node 5 (NP)	15	13	18
node 6 (ME)	20	18	20

Figure 11. Example 3: optimal dual values. See text.

In addition, the two coupling constraints (8) will also have dual variables. They are listed below:

shadow price of capacity constraint at London (eq. 8a) = 0
shadow price of capacity constraint at Rome (eq. 8b) = 2

To interpret these numerical results, let us now look at the *exhaustion of shadow value* along each link of the network. Figure 12 lists the *increase in shadow prices* occurring along each link, calculated as the difference between the shadow price at the destination node and the shadow price at the origin node (brought from the table in Figure 11). Figure 13 lists the total shadow unit cost along each link, calculated as the sum of the unit transportation cost and the shadow cost of the capacity constraint, if applicable.

Both tables also indicate the optimal traffic pattern, shaded in green. Let us first take a look at all links colored in green. Along each of these links, *a hypothetical shipper breaks exactly even*: the increase in shadow price exactly equals the unit transportation cost plus the shadow cost of the capacitating condition.

links	*x-traffic*	*y-traffic*	*z-traffic*
BR-LO	0-4 = -4	0-2 = -2	0-7 = -7
BR-RO	9-4 = 5	7-2 = 5	12-7 = 5
BR-NP	15-4 = 11	13-2 = 11	18-7 = 11
BR-ME	20-4 = 16	18-2 = 16	20-7 = 13
CH-LO	0-2 = -2	0-0 = 0	0-5 = -5
CH-RO	9-2 = 7	7-0 = 7	12-5= 7
CH-NP	15-2 = 13	13-0 = 13	18-5 = 13
CH-ME	20-2 = 18	18-0 = 18	20-5 = 15
LO-RO	9-0 = 9	7-0 = 7	12-0 = 12
LO-NP	15-0 = 15	13-0 = 13	18-0 = 18
LO-ME	20-0= 20	18-0 = 18	20-0 = 20
RO-NP	15-9 = 6	13-7 = 6	18-12 = 6
RO-ME	20-9 = 11	18-7 = 11	20-12 = 8

Figure 12. Example 3: Increases in shadow price along each link
Optimal traffic proceeds along links colored in green. No traffic along uncolored links.

links	*x-traffic*	*y-traffic*	*z-traffic*
BR-LO	8	8	8
BR-RO	5	5	5
BR-NP	20	20	20
BR-ME	23	23	23
CH-LO	10	10	10
CH-RO	7	7	7
CH-NP	18	18	18
CH-ME	25	25	25
LO-RO	12+0 = 12	12+0 = 12	12+0 = 12
LO-NP	18+0 = 18	18+0 = 18	18+0 = 18
LO-ME	20+0 = 20	20+0 = 20	20+0 = 20
RO-NP	4+2 = 6	4+2 = 6	4+2 = 6
RO-ME	9+2 = 11	9+2 = 11	9+2 = 11

Figure 13. Example 3: Direct unit transportation costs plus shadow cost of capacitating conditions along each link. Optimal traffic proceeds along links colored in green. No traffic along uncolored links.

Next, the reader is asked to inspect the non-colored links. Along most of these links, it turns out that the increase in shadow price is smaller than the unit transportation cost plus the shadow cost of the capacitating condition. *A hypothetical shipper would suffer a unit loss*. Hence, by complementary slackness, no shipments are carried out along these links.

There are also a few non-colored links where the increase in shadow price happens to equal the unit transportation cost plus the shadow cost of the capacitating condition. But this observation, by itself, does not imply anything. It does not follow that a positive shipment takes place. (Remember the mathematical structure of the conditions of complementary slackness: if A and B are nonnegative numbers and $AB = 0$ then the following conclusion can be drawn: (i) if $A > 0$ then $B = 0$, and (ii) if $B > 0$ then $A = 0$. But if $A = 0$ no conslusion can be drawn abnout the magnitude of B. Similarly, if $B = 0$, no conclusion can be drawn about the magnitude of A - - see p. 11 above)

To sum up, if the shadow price of the environmental constraint is included in the standard cost calculation, the principle of exhaustion of the shadow price by unit costs holds as before.

6. Programming with 0-1 variables. The only costs discussed so far are variable costs, calculated as a unit charge. In reality, there are also fixed costs of course, but they do not affect the optimization procedure. Fixed costs like purchase price and depreciation of transportation facilities (ships, airplanes), salaries of permanent staff, and other overhead costs have to be paid regardless of the scheduling of the traffic network.

In the hub-and-spoke network, however, there exists an intermediate category of costs that now needs our attention: the presence of fixed costs of establishing and operating a hub that do not apply if the hub is never set up. Such installation or *set-up costs* may actually be crucial to management in determining whether a hub should be built or not. One of the advantages of low-cost budget airlines flying passengers directly and nonstop to resorts and vacation locations is obviously that they do not have to operate costly hub facilities (paying for landing slots at high-traffic airports, maintaining check-in facilities, renting counter space etc.)

Mathematically, such costs can be handled *by the presence of 0-1 variables - - so-called binary variables* - - taking either the value 1 (a facility with attendant fixed costs has been set up) or the value 0 (the facility has not been set up). When carrying out the optimization, the mathematical program considers all possible combinations of operating individual facilities that may be set up or not be set up, and selects the combined alternative generating the lowest total cost (the highest shadow value).

Returning to Example 1 above, involving three departure nodes (called Bristol, London and Chester) and three arrival nodes (called Rome, Naples and Messina) suppose now that the hub at node 2 (London) requires separate set-up costs if it is going to be operated. Let these costs be S. That is, if the carrier decides to route traffic via the hub, the costs apply. If the routing avoids the hub entirely, so that the hub need not be set up in the first place, there are no setup costs. Introducing the 0-1 variable w, total setup costs can then be written as Sw. If the hub is setup, $Sw = K$, If the hub is not set up, $Sw = 0$.

Variable costs with a constant unit charge are conventionally illustrated diagrammatically as a ray through the origin, rising proportionally with the level of operation. Fixed costs are drawn as a horisontal line. But the setup costs Sw have a peculiar geometric property. They can be illustrated by

- a horisontal line indicating the total cost S, if the hub has been set up
- the origin itself if the hub has not been set up.

In other words, this diagram displays a *discontinuity* at the origin, where total costs jump from 0 to K.

Example 4. This example is the same as Example 1, but allows for the possibilities of either setting up the hub at node 4 (Rome) or not setting it up. The setup cost is S = 2500. Amending program (2), the mathematical program with the setup variable w and setup costs Sw then reads:

(9) $$\begin{aligned} &\textit{minimize } cx + 2500w \\ &\textit{subject to} \\ &Mx \geq b, \\ &x_{14} + x_{24} + x_{34} + x_{45} + x_{46} \leq 99{,}999w \\ &x \geq 0,\ w = 0, 1 \end{aligned}$$

The essential feature of this program is the new constraint $x_{14} + x_{24} + x_{34} + x_{45} + x_{46} \leq 99{,}999w$. What does it mean? The left hand side of the constraint just adds together all links passing through node 4 (Rome). The right hand side is $99{,}999w$. The number $99{,}999$ has been chosen to signify a "very large" positive number. Now, if node 4 has been set up so that $w = 1$, then the constraint just reads $x_{14} + x_{24} + x_{34} + x_{45} + x_{46} \leq 99{,}999$ which will always be satisfied. Hence, the constraint is redundant and can simply be dropped. On the other hand, if node 4 has *not* been set up so that $w = 0$ then the constraint reads $x_{14} + x_{24} + x_{34} + x_{45} + x_{46} \leq 0$. Since each variable on the left hand side is nonnegative, it follows that they are all zero. No traffic can pass through node 4.

The GAMS program reads (to give the reader the benefit of some practice, I am rearranging the statements of the basic program once more):

```
SETS
I cities              / BR, CH, LO, RO, NP, ME/
ORIG(I) origins       / BR, CH, LO/
DEST(I) destinations / RO, NP, ME/
K links/BR-LO, BR-RO, BR-NP, BR-ME, CH-LO, CH-RO, CH-NP, CH-ME, LO-RO,
                LO-NP, LO-ME, RO-NP, RO-ME/ ;

PARAMETER
C(K) unit shipping costs
  /BR-LO    8
   BR-RO    5
   BR-NP   20
   BR-ME   23
   CH-LO   10
   CH-RO    7
   CH-NP   18
   CH-ME   25
   LO-RO   12
   LO-NP   18
   LO-ME   20
   RO-NP    4
   RO-ME    9 /

B(I) supplies and demands
  /BR      -100
   CH      -150
   LO         0
   RO         0
   NP       125
   ME       125/ ;

TABLE INCIDENCE(I,K) incidence matrix
            BR-LO  BR-RO  BR-NP  BR-ME  CH-LO  CH-RO  CH-NP  CH-ME
       BR    -1      -1     -1     -1     0      0      0      0
       CH     0       0      0      0    -1     -1     -1     -1
       LO     1       0      0      0     1      0      0      0
       RO     0       1      0      0     0      1      0      0
       NP     0       0      1      0     0      0      1      0
       ME     0       0      0      1     0      0      0      1

       +    LO-RO  LO-NP  LO-ME  RO-NP  RO-ME
       BR     0      0      0      0      0
       CH     0      0      0      0      0
       LO    -1     -1     -1      0      0
       RO     1      0      0     -1     -1
       NP     0      1      0      1      0
       ME     0      0      1      0      1   ;

VARIABLES
X(K)    traffic along link K
W       0-1 setup variable for hub at Rome
OBJ     objective function ;

POSITIVE VARIABLES X;
BINARY VARIABLES   W;

EQUATIONS
COST  total cost consisting of current costs plus setup costs
CONSFLOW(I) conservation-of-flow conditions
ROME        setup condition at Rome;
```

```
COST..              OBJ=E=SUM(K, C(K)*X(K)) + 2500*W ;
CONSFLOW(I)..       SUM(K,INCIDENCE(I,K)* X(K)) =G= B(I);
ROME..              X("BR-RO") + X("CH-RO") + X("LO-RO")
                       + X("RO-NP") + X("RO-ME") =L= 99999*W;

MODEL SETUP/ALL/;
SOLVE SETUP USING MIP MINIMIZING OBJ;
```

The GAMS statements should be self-explanatory. The variable W is listed as "BINARY". The statement on the last line of the program instructs the GAMS software to use "MIP" (= mixed integer programming) to solve the program.

Solving, one finds that the optimal solution (circumventing node 4 at Rome) requires routing all 100 supply units at node 1 (Bristol) directly nonstop to node 6 (Messina). Likewise, 25 units of the supply forthcoming at node 2 (Chester) should be sent directly to node 6. The remaining 125 units at node 2 (Chester) are shipped directly to node 5 (Naples). In other words, $x_{16} = 100$, $x_{26} = 25$ and $x_{25} = 125$.

The resulting minimal cost is $23x_{16} + 25x_{26} + 18x_{25} = 23 \times 100 + 25 \times 25 + 18 \times 125 = 5175$. This figure should be compared with the solution to Example 1 (permitting transits through Rome) which yielded a total cost of only *3175* (see page 7). Paying the setup cost at Rome, that earlier shipping pattern would now have resulted in a total of $3175 + 2500 = 5675$ which exceeds the present total cost with *500*. So, it does not pay to set up the hub at Rome. The setup cost is prohibitive.

Actually, we can now calculate *the maximal setup cost that the transportation company would be willing to pay, still maintaining the hub operations at Rome*. That maximal setup cost is 2000. If the cost exceeds 2000, the company will never set up the hub at Rome.

*

For further study. The basic network model and the incidence matrix are treated in Thompson and Thore, 1992, Chap. 8 (an elementary text) and in Thore, 1991 (a graduate textbook), Chap. 3.3. For the multi-page formulation of the transportation model, see Charnes and Cooper, Appendix G. Programming with 0-1 variables is described in Thompson and Thore Chaps. 11 and 16. The standard GAMS reference is Brooke, Kendrick, Meeraus, Raman.

A. Brooke, D. Kendrick, A. Meeraus and R. Raman, *GAMS: A User's Guide, GAMS Development Corporation,* 1998. Can be downloaded for free from www.gams.com/docs/document.htm.

A. Charnes and W.W. Cooper, *Management Models and Industrial Applications of Linear Programming*, Wiley 1961, Vols I-II.

G. Thompson and S. Thore, *Computational Economics: Economic Modeling with Optimization Software,* Scientific Press, San Francisco 1992 (to obtain a CD-ROM with the entire text and the accompanying software, consult www.stenthore.info).

S, Thore with W.W. Cooper, *Economic Logistics: The Optimization of Spatial and Sectoral Resource, Production and Distribution Systems,* Quorum Books, Westport, Conn., 1991.