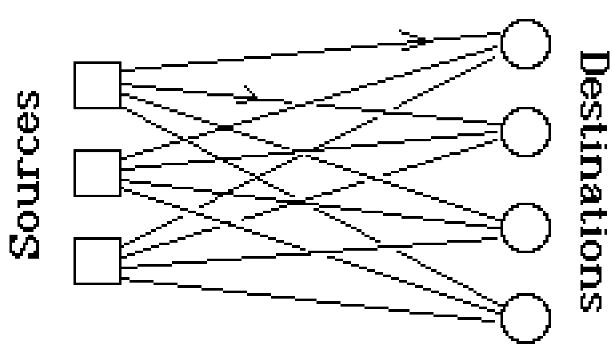
Minimum-Cost Network Flows

This Hypercard stack was prepared by:
Dennis Bricker,
Dept. of Industrial Engineering,
University of Iowa,
Iowa City, Iowa 52242.
e-mail: dennis-bricker@uiowa.edu

author

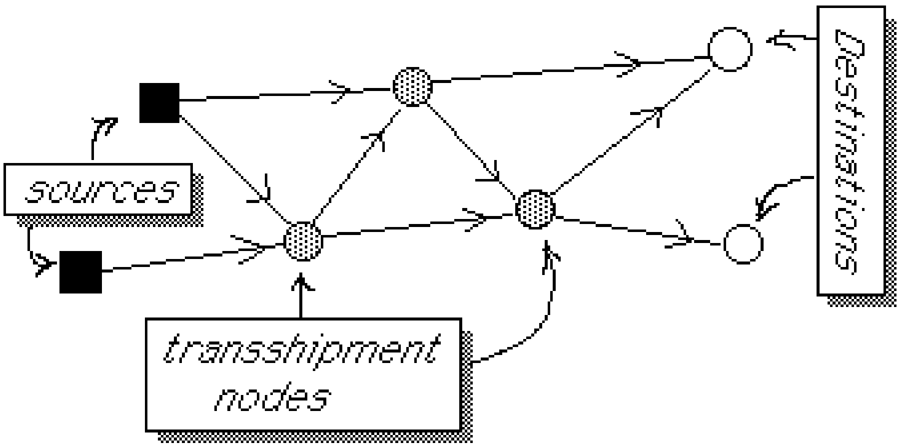
In the transportation model, it is assumed that no route from one source to a destination can pass through other sources or destinations as intermediate points.



The network is "bi-partite", i.e., the nodes may be partitioned into 2 sets, with no arc between 2 nodes of the same set.

"The Transshipment Problem"

We now consider the problem in which "transshipments" through other nodes is allowed.



Conservation of Flow

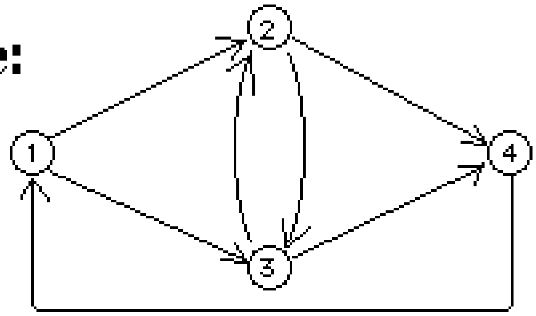
(Material Balance, Kirchoff Equations)

$$\underbrace{\sum_j X_{ij}}_{\text{Total flow out of node i}} - \underbrace{\sum_k X_{ki}}_{\text{Total flow into node i}} = b_i$$

Net flow from node i

Conservation of Flow

Example:



$$
\left\{
\begin{array}{rrrrrrrl}
X_{12} & +X_{13} & & & & & -X_{41} & = b_1 \\
-X_{12} & & +X_{23} & +X_{24} & -X_{32} & & & = b_2 \\
 & -X_{13} & -X_{23} & & +X_{32} & +X_{34} & & = b_3 \\
 & & & -X_{24} & & -X_{34} & +X_{41} & = b_4
\end{array}
\right.
$$

Coefficient Matrix of Kirchoff Eqns

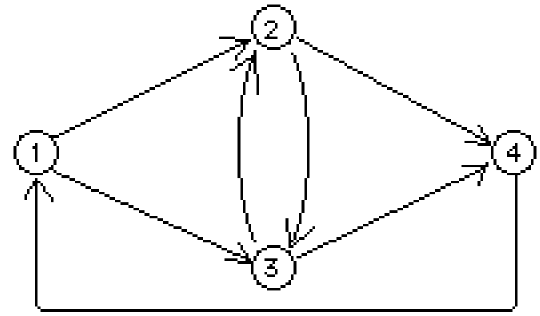
$$
\left\{
\begin{array}{rrrrrrrl}
X_{12} & +X_{13} & & & & & -X_{41} & = b_1 \\
-X_{12} & & +X_{23} & +X_{24} & -X_{32} & & & = b_2 \\
 & -X_{13} & -X_{23} & & +X_{32} & +X_{34} & & = b_3 \\
 & & & -X_{24} & & -X_{34} & +X_{41} & = b_4
\end{array}
\right.
$$

is

$$
\begin{array}{c}
\begin{array}{ccccccc}
(1,2) & (1,3) & (2,3) & (2,4) & (3,2) & (3,4) & (4,1)
\end{array} \\
\begin{bmatrix}
+1 & +1 & & & & & -1 \\
-1 & & +1 & +1 & -1 & & \\
 & -1 & -1 & & +1 & +1 & \\
 & & & -1 & & -1 & +1
\end{bmatrix}
\end{array}
$$

©D.L.Bricker, U. of Iowa, 1998

Node-Arc Incidence Matrix



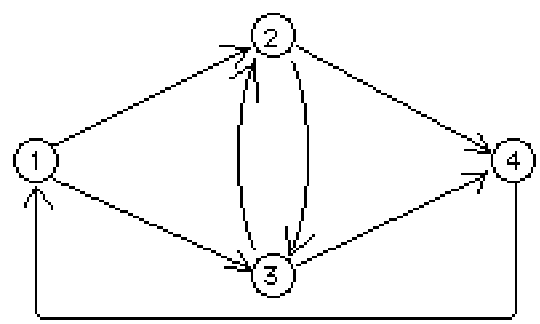
Coefficient matrix of Kirchoff Eq'ns

rows ≈ nodes
columns ≈ arcs

elements are +1, 0, or −1

$$\begin{array}{c}
\begin{matrix} (1,2) & (1,3) & (2,3) & (2,4) & (3,2) & (3,4) & (4,1) \end{matrix} \\
\begin{bmatrix}
+1 & +1 & & & & & -1 \\
-1 & & +1 & +1 & -1 & & \\
& -1 & -1 & & +1 & +1 & \\
& & & -1 & & -1 & +1
\end{bmatrix}
\end{array}$$

Node-Arc Incidence Matrix



column for arc (i,j) has:

$$\begin{cases} +1 \text{ in row } i \\ -1 \text{ in row } j \\ 0 \text{ elsewhere} \end{cases}$$

$$\begin{array}{c}
\begin{matrix} (1,2) & (1,3) & (2,3) & (2,4) & (3,2) & (3,4) & (4,1) \end{matrix} \\
\begin{bmatrix}
+1 & +1 & & & & & -1 \\
-1 & & +1 & +1 & -1 & & \\
& -1 & -1 & & +1 & +1 & \\
& & & -1 & & -1 & +1
\end{bmatrix}
\end{array}$$

Node-Arc Incidence Matrix

Does not have full row rank

(Sum of rows is a row of zeroes, implying linear dependence of the rows!)

Rank is (# rows) - 1

$$
\begin{array}{c}
\begin{array}{ccccccc} (1,2) & (1,3) & (2,3) & (2,4) & (3,2) & (3,4) & (4,1) \end{array} \\
\left[\begin{array}{ccccccc}
+1 & +1 & & & & & -1 \\
-1 & & +1 & +1 & -1 & & \\
 & -1 & -1 & & +1 & +1 & \\
 & & & -1 & & -1 & +1
\end{array}\right]
\end{array}
$$

Node-Arc Incidence Matrix

Exercise:

Draw the network with each node-arc incidence matrix

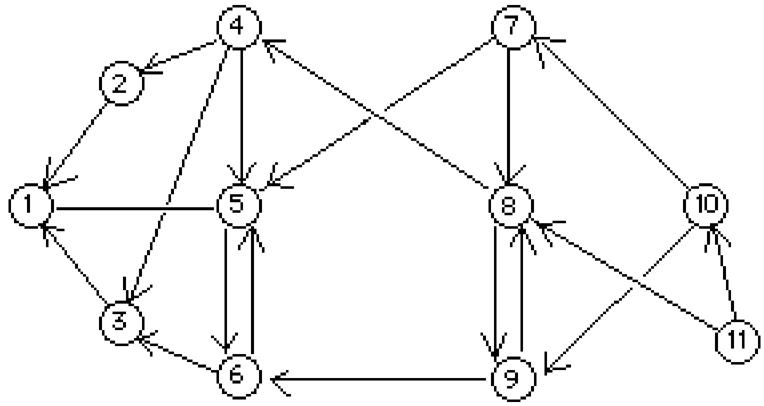
$$
\left[\begin{array}{ccccccc}
1 & 1 & & & & & \\
-1 & & 1 & 1 & -1 & & \\
 & & -1 & & & -1 & \\
 & -1 & & -1 & & & -1 \\
 & & & & 1 & 1 & 1
\end{array}\right]
$$

$$
\left[\begin{array}{cccccccccc}
1 & 1 & 1 & & & & & & & \\
-1 & & & 1 & 1 & 1 & -1 & & & \\
 & -1 & & -1 & & & 1 & 1 & -1 & \\
 & & -1 & & -1 & & & & 1 & -1 \\
 & & & & & -1 & & -1 & & 1
\end{array}\right]
$$

Node-Arc Incidence Matrix

Exercise:

Write the node-arc incidence matrix for the network



Unimodularity

A square integer matrix is called ***unimodular*** if its determinant is ± 1.

$\Longrightarrow$ the inverse of a unimodular matrix has only integer-valued elements

$\Longrightarrow$ if B is unimodular and b is integer-valued, then the solution $x=B^{-1}b$ of the equation $Bx=b$ is integer-valued.

Total Unimodularity

An integer matrix A is ***totally unimodular*** if every square, nonsingular submatrix of A is ***unimodular***.

$\Longrightarrow$ if b is integer-valued, every basic solution of the system $Ax=b$ is integer-valued.

Theorem

Every node-arc incidence matrix is totally unimodular.

$\Longrightarrow$ Every LP whose coefficient matrix is a node-arc incidence matrix and whose RHS is integer-valued will have only integer-valued basic solutions.

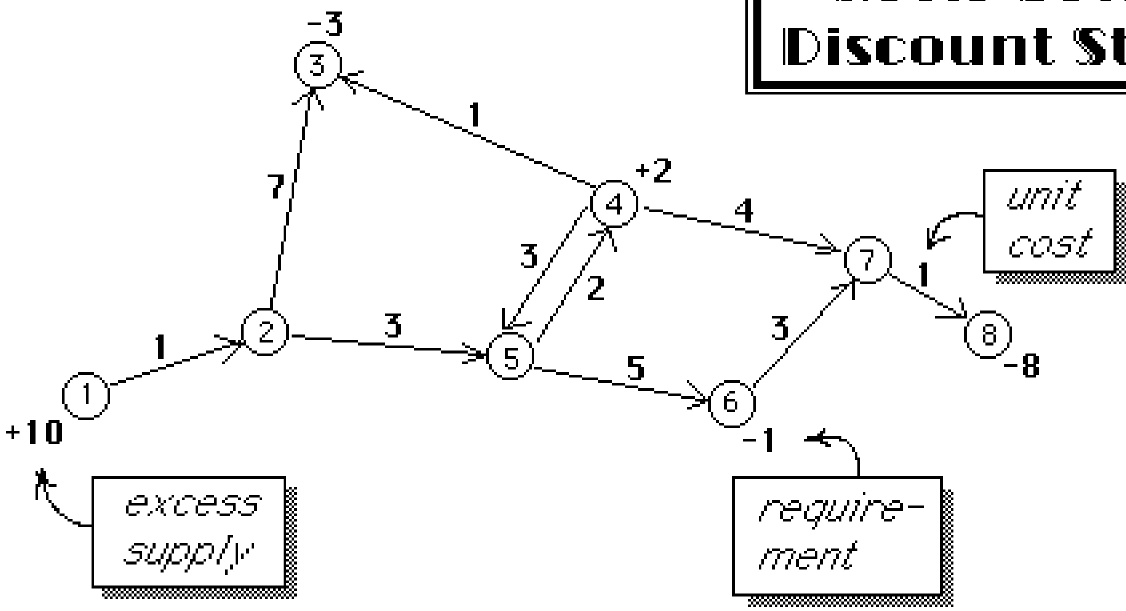
Examples

- ☞ Rock-Bottom Discount Stores
- ☞ Spitzen-Pollish Company
- ☞ Caterer's Problem
- ☞ Opencast Mining
- ☞ Stochastic Transportation Problem

Example: "Rock-Bottom Discount Store:

The company has 8 stores, and is preparing for a promotion of a certain appliance. Some stores have an excess of the product, and others a need for additional units. Given transportation costs for all routes joining the stores, how should the product be re-distributed at minimum cost?

Rock-Bottom Discount Stores



Rock-Bottom Discount Stores

Linear Programming Tableau

	(1,2)	(2,3)	(2,5)	(4,3)	(4,5)	(5,4)	(4,7)	(5,6)	(6,7)	(7,8)		
MIN	1	7	3	1	3	2	4	5	3	1		
1)	1										=	10
2)	-1	1	1								=	0
3)		-1		-1							=	-3
4)				1	1	-1	1				=	2
5)			-1		-1	1		1			=	0
6)								-1	1		=	-1
7)							-1		-1	1	=	0
8)										-1	=	-8

$\mathcal{N}$ = set of nodes of the network
$\mathcal{A}$ = set of arcs of the network
X_{ij} = flow in arc (i,j)
C_{ij} = unit cost of flow in arc (i,j)
L_{ij} = lower bound of flow in arc (i,j)
U_{ij} = upper bound of flow in arc (i,j)

Minimum-cost Network Flow Problem

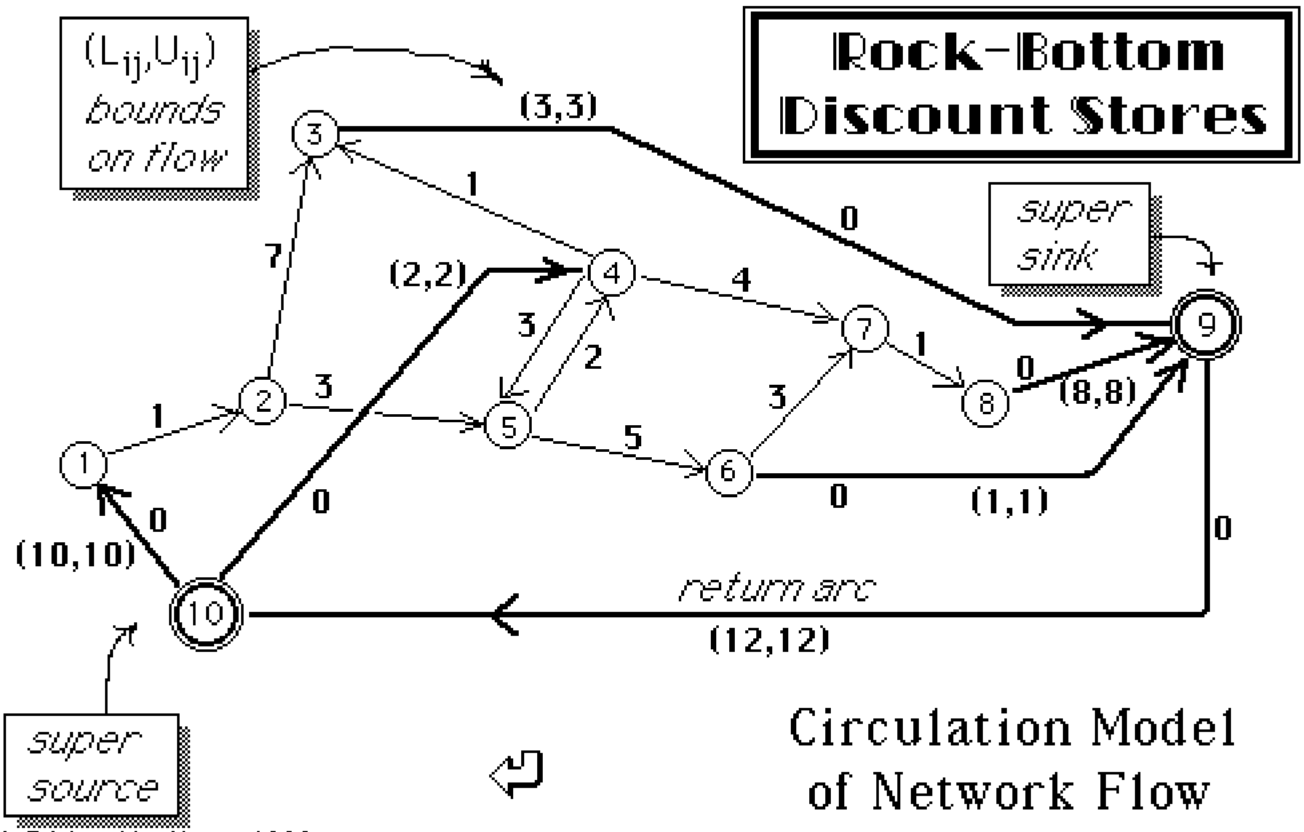
$$\text{Minimize} \quad \sum_{(i,j)\in\mathcal{A}} C_{ij}X_{ij}$$
s.t.
$$\sum_j X_{kj} - \sum_i X_{ik} = 0 \quad \forall\, k\in\mathcal{N}$$
$$L_{ij} \le X_{ij} \le U_{ij} \quad \forall\, (i,j) \in \mathcal{A}$$

$$\text{Minimize} \quad \sum_{(i,j)\in\mathcal{A}} C_{ij}X_{ij}$$
s.t.
$$\sum_j X_{kj} - \sum_i X_{ik} = 0 \quad \forall\, k\in\mathcal{N}$$
$$L_{ij} \le X_{ij} \le U_{ij} \quad \forall\, (i,j) \in \mathcal{A}$$

Assumes:

- no losses or gains in the arcs
- flow is a "circulation" in the network... no accumulation of commodity at a node

Other formulations may have RHS of Kirchoff Eq'ns which are nonzero.



Circulation Model of Network Flow

Example: Crew Scheduling

The Spitzen-Pollish Co. is a contract maintenance firm that provides and supervises semi-skilled manpower for major overhauls of chemical processing equipment.

A standard job frequently requires a thousand or more men, and may extend from one or two weeks to several months.

Since the client's plant often is located in another city, Spitzen-Pollish must transport the workers to the plant and provide on-site housing and meals, etc., in addition to wages.

For a routine job, Spitzen-Pollish can estimate fairly accurately the number of crews required on a day-to-day basis for the job's duration.

The firm may vary the number of crews on-site during the job.... However, there are some costs that do not depend upon how long a crew remains on-site (costs of recruiting, transportation, training, etc.)

The company may therefore find it more economical to retain idle crews on-site if they will be required a few days later.

Spitzen-Pollish Co. LP Formulation

Define: X_{ij} = # of crews beginning work on-site at beginning of period i and returning at end of period (j−1), i.e., beginning of period j.

C_{ij} = Total operating cost of such a crew.

(Assume $C_{ij} \leq C_{hk}$ if $h \leq i < j \leq k$)

R = # of crews required during period k

n = length of job (# periods) + 1

i.e., job ends at beginning of period n

Spitzen-Pollish LP Model

$$\text{Minimize} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} C_{ij} X_{ij}$$

subject to

$$\sum_{j=2}^{n} X_{1j} = R_1$$

$$\sum_{i=1}^{k} \sum_{j=k+1}^{n} X_{ij} \geq R_k \quad \text{for } k=2, 3, 4, \ldots n-2$$

$$\sum_{i=1}^{n-1} X_{in} = R_{n-1}$$

$$X_{ij} \in \{0,1,2,3,\ldots\} \quad \text{for all } i,j$$

LP Tableau (for n=6)

	(1,2)	(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)	(5,6)	S_2	S_3	S_4		rhs
min	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	0	0	0		
1)	1	1	1	1	1														=	R_1
2)		1	1	1	1	1	1	1	1							-1			=	R_2
3)			1	1	1		1	1	1	1	1	1					-1		=	R_3
4)				1	1			1	1		1	1	1	1				-1	=	R_4
5)					1				1			1		1	1				=	R_5

$\underbrace{S_2 \; S_3 \; S_4}_{\text{surplus variables}}$

Note: subscripts of C were omitted for convenience.

Not a node-arc incidence matrix!

min	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	0	0	0		
1)	1	1	1	1	1														=	R_1
2)		1	1	1	1	1	1	1	1							-1			=	R_2
3)			1	1	1		1	1	1	1	1	1					-1		=	R_3
4)				1	1			1	1		1	1	1	1				-1	=	R_4
5)					1				1			1		1	1				=	R_5

Make the following transformation:

subtract row 1 from row 2 to obtain row 2'
" " 2 " " 3 " " " 3'
" " 3 " " 4 " " " 4'
" " 4 " " 5 " " " 5'

The equations obtained in this way are implied by the original set of equations.

Many of the "1"s are eliminated by this transformation, and some "−1"s are introduced:

subtract row 1 from row 2 to obtain row 2'

1)	1	1	1	1	1														=	R_1
2)		1	1	1	1	1	1	1	1							-1			=	R_2
2')	-1					1	1	1	1							-1			=	R_2-R_1

min	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	0	0	0		
1)	1	1	1	1	1														=	R_1
2')	-1					1	1	1	1							-1			=	R_2-R_1
3')		-1				-1				1	1	1				1	-1		=	R_3-R_2
4')			-1				-1			-1			1	1			1	-1	=	R_4-R_3
5')				-1				-1			-1		-1		1			1	=	R_5-R_4

The resulting tableau, equivalent to the original, has a constraint coefficient matrix very nearly that of a node-arc incidence matrix (i.e., +1 and −1 in all but 5 columns, which have a +1 but no −1)!

min	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	0	0	0		
1)	1	1	1	1	1														=	R_1
2')	-1					1	1	1	1							-1			=	R_2-R_1
3')		-1				-1				1	1	1				1	-1		=	R_3-R_2
4')			-1				-1			-1			1	1			1	-1	=	R_4-R_3
5')				-1				-1			-1		-1		1			1	=	R_5-R_4

Sum all of the constraints, and negate both sides of the resulting equation... If a column already has a +1,−1 pair, the sum is zero. Otherwise, we obtain the needed −1:

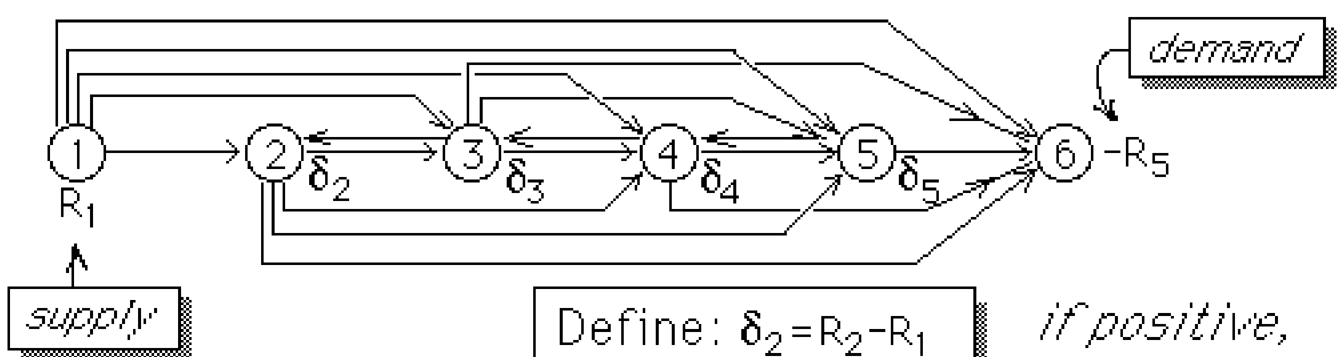
6')					-1				-1			-1		-1	-1				=	$-R_5$

min	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	0	0	0		
1)	1	1	1	1	1														=	R_1
2')	-1					1	1	1	1							-1			=	R_2-R_1
3')		-1				-1				1	1	1				1	-1		=	R_3-R_2
4')			-1				-1			-1			1	1			1	-1	=	R_4-R_3
5')				-1				-1			-1		-1		1			1	=	R_5-R_4
6')					-1				-1			-1		-1	-1				=	$-R_5$

We now have an equivalent formulation of the Spitzen–Pollish problem which is a network problem!

What is the appearance of the network?

The network will have one node per row of the node–arc incidence matrix:



Define: $\delta_2 = R_2 - R_1$

$\delta_3 = R_3 - R_2$

$\delta_4 = R_4 - R_3$

$\delta_5 = R_5 - R_4$

if positive, this will be a supply at node, & if negative, a demand

Because this is a network problem with integer right-hand-side, any basic LP solution (in particular, the optimal LP solution) will be integer-valued.

Example: Caterer's Problem

A catering service must provide napkins for dinners on each of T consecutive days.
The number required on day t is D_t.
Requirements may be met by:

- purchasing new napkins, at cost C_1 each
- laundering napkins soiled at an earlier dinner.

Two types of laundry service are available:

- regular: costs C_3 each, τ days required
- special: costs C_2 each, ν days required

No salvage value for napkins after day T.

Note: $C_3 < C_2 < C_1$
$\nu < \tau$

Sample Data (Caterer's Problem)

T = 4 days
τ = 2 days (one-day service)
ν = 1 day (overnight service)
C = \$2.00 for new napkins
C = \$1.40 for overnight laundry service
C = \$0.90 for regular laundry service

Day t:	Wed	Thurs	Fri	Sat
Rqmt:	450	650	975	850

Decision Variables:

P_t = # napkins purchased on day t
R_t = # napkins sent to regular laundry on day t
S_t = # napkins sent to special laundry on day t
U_t = # soiled napkins stored at end of day t
V_t = # clean napkins stored at end of day t

Constraints

Disposition of clean napkins before dinner:

$$\underbrace{R_{t-2} + S_{t-1} + V_{t-1} + P_t}_{\text{available for use on day } t} = \underbrace{D_t}_{\text{to be used}} + \underbrace{V_t}_{\text{stored clean}}$$

Disposition of soiled napkins after dinner:

$$\underbrace{R_t + S_t}_{\text{sent to laundry}} + \underbrace{U_t}_{\text{stored dirty}} = \underbrace{D_t + U_{t-1}}_{\text{soiled napkins}}$$

Constraint Matrix:

P_1	R_1	S_1	U_1	V_1	P_2	R_2	S_2	U_2	V_2	P_3	S_3	U_3	V_3	P_4	U_4	rhs
1				−1												450
		1		1	1				−1							650
	1						1		1	1			−1			975
						1					1		1	1		850
	1	1	1													450
			−1			1	1	1								650
								−1			1	1				975
												−1			1	850

Not a node-arc incidence matrix.... Can it be manipulated to produce one?

Constraint Matrix:

Negate both sides of the top portion of the matrix:

P_1	R_1	S_1	U_1	V_1	P_2	R_2	S_2	U_2	V_2	P_3	S_3	U_3	V_3	P_4	U_4	rhs
−1				1												− 450
		−1		−1	−1				1							− 650
	−1						−1		−1	−1			1			− 975
						−1					−1		−1	−1		− 850
	1	1	1													450
			−1			1	1	1								650
								−1			1	1				975
												−1			1	850

The result is very nearly a node-arc incidence matrix!

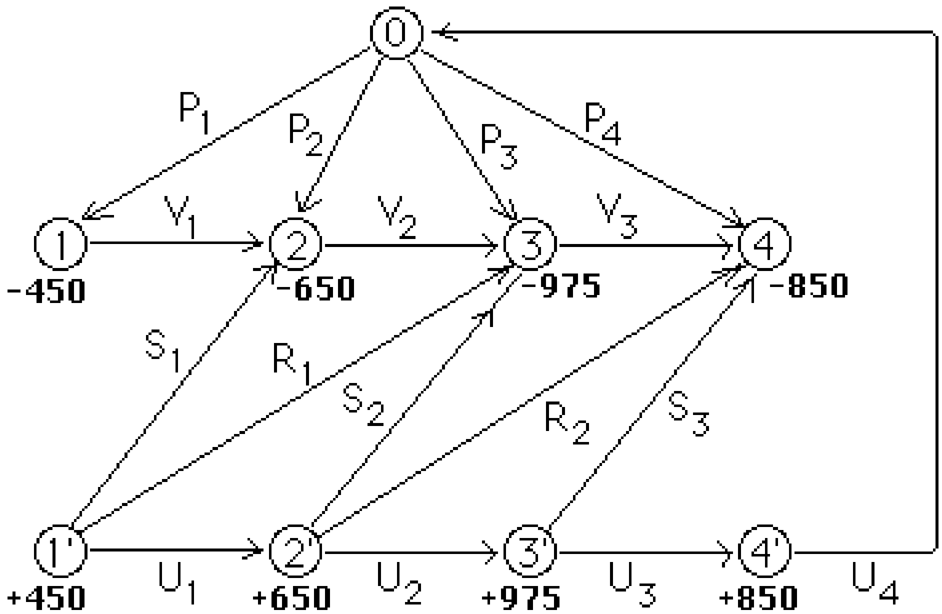
Constraint Matrix:

Append a new row obtained by negating sum of other rows:

row	P_1	R_1	S_1	U_1	V_1	P_2	R_2	S_2	U_2	V_2	P_3	S_3	U_3	V_3	P_4	U_4	rhs
1	−1				1												− 450
2			−1		−1	−1				1							− 650
3		−1						−1		−1	−1			1			− 975
4							−1					−1		−1	−1		− 850
1'		1	1	1													450
2'				−1			1	1	1								650
3'									−1			1	1				975
4'													−1			1	850
0	1					1					1				1	−1	0

... a node-arc incidence matrix!

Caterer's Problem: Network Model

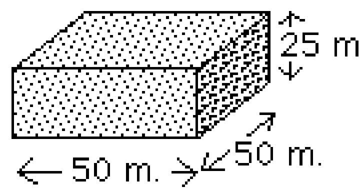


Note: the caterer's problem originated in a military context:

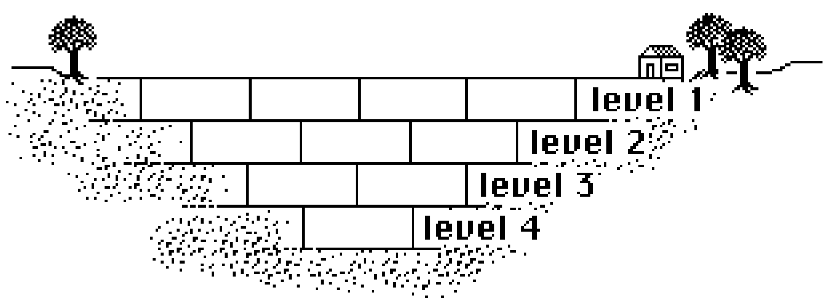
- Airplane engines must be serviced after every mission (or replaced)
- Engine service can be performed overnight at higher cost, otherwise is performed the next day
- The number of daily missions has been planned far in advance

Opencast Mining Problem

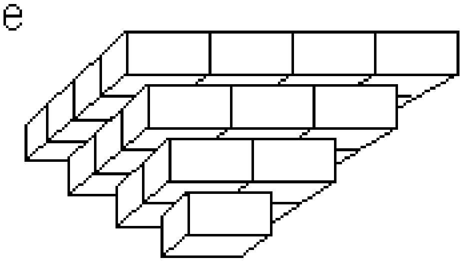
- A company has obtained permission to opencast mine ("strip mine") within a square plot 200 meters on each side.
- Angle of slip of soil is such that sides of excavation may not be steeper than 45^0
- Company decides to consider the problem as one of extracting rectangular blocks

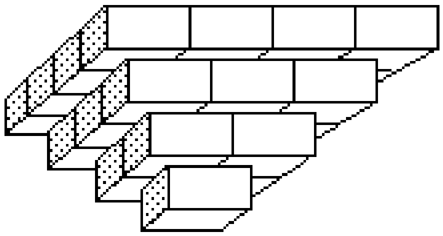


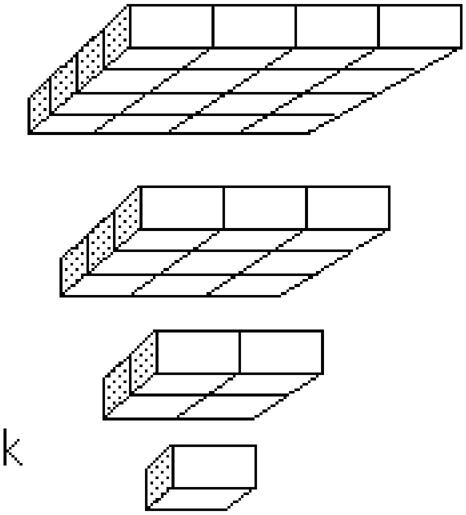
The blocks are selected to lie above one another like so:



Restrictions imposed by the angle of slip means that it is possible only to excavate blocks forming an "inverted pyramid"







The company has estimates for the value of the ore in various places at various depths.
Using these estimates, each block has a certain net income
= (revenue from sale of ore) − (cost of excavating, extracting, & refining)

Which blocks should be excavated?

LEVEL 1

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

LEVEL 2

17	18	19
20	21	22
23	24	25

LEVEL 3

26	27
28	29

LEVEL 4

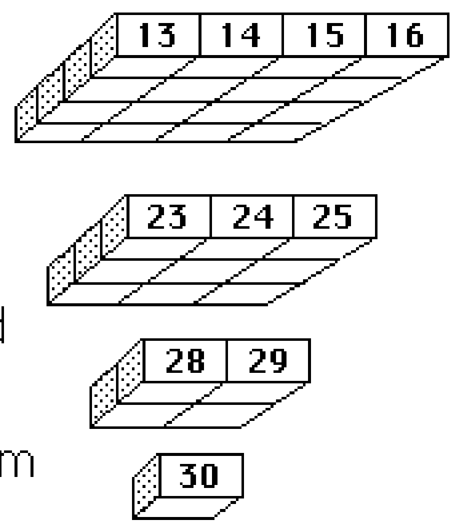
30

First, number the blocks:

Define:

$$Y_i = \begin{cases} 1 & \text{if block } i \text{ is excavated} \\ 0 & \text{otherwise} \end{cases}$$

R_i = net income from block i



$$\boxed{\text{Objective: Maximize } \sum_{i=1}^{n} R_i Y_i}$$

Example Data

Block Numbers

LEVEL 1

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

LEVEL 2

17	18	19
20	21	22
23	24	25

LEVEL 3

26	27
28	29

LEVEL 4

30

Excavation Cost /Block

Level	Cost
1	3
2	6
3	8
4	10

Revenue

LEVEL 1

0	0	0	$-3/2$
0	1	0	$-3/2$
-1	-1	$-3/2$	-2
$-3/2$	$-3/2$	-2	$-5/2$

LEVEL 2

2	2	-2
0	0	-4
-2	-2	-5

LEVEL 3

16	4
2	0

LEVEL 4

20

Constraints

LEVEL 1

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

LEVEL 2

17	18	19
20	21	22
23	24	25

LEVEL 3

26	27
28	29

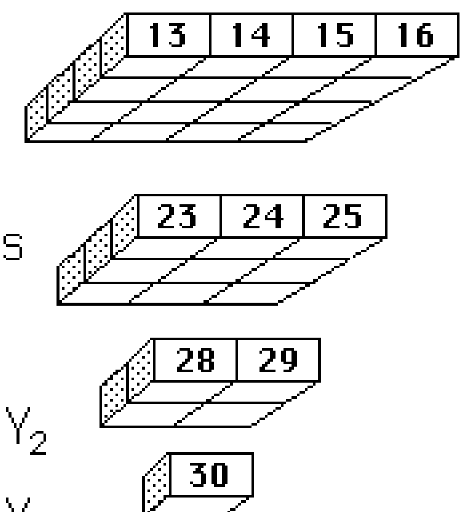
LEVEL 4

30

Example:

Block 17 cannot be excavated unless blocks 1,2,5,&6 are excavated:

$$\begin{cases} Y_{17} \leq Y_1, & Y_{17} \leq Y_2 \\ Y_{17} \leq Y_5, & Y_{17} \leq Y_6 \end{cases}$$



Likewise, for each block in levels 2, 3, & 4, we obtain 4 such constraints.

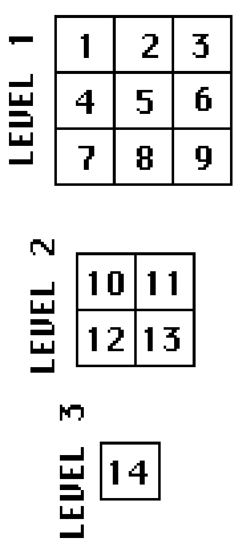
Size of Problem:

\# of Variables: 30 integer (binary) variables
\# of Constraints: $4 \times 14 = 56$ inequalities

If the number of blocks and number of levels were increased by using a smaller grid, the number of binary variables and constraint increases dramatically!

Solution as an integer programming problem quickly becomes exorbitantly expensive to compute!

Let's use a smaller version to study the structure of the problem:



$$\begin{cases} Y_{10} \le Y_1 & Y_{10} \le Y_2 \\ Y_{10} \le Y_4 & Y_{10} \le Y_5 \end{cases}$$
$$\begin{cases} Y_{11} \le Y_2 & Y_{11} \le Y_3 \\ Y_{11} \le Y_5 & Y_{11} \le Y_6 \end{cases}$$
$$\begin{cases} Y_{12} \le Y_4 & Y_{12} \le Y_5 \\ Y_{12} \le Y_7 & Y_{12} \le Y_8 \end{cases}$$
$$\begin{cases} Y_{13} \le Y_5 & Y_{13} \le Y_6 \\ Y_{13} \le Y_8 & Y_{13} \le Y_9 \end{cases}$$
$$\begin{cases} Y_{14} \le Y_{10} & Y_{14} \le Y_{11} \\ Y_{14} \le Y_{12} & Y_{14} \le Y_{13} \end{cases}$$

©D.L.Bricker, U. of Iowa, 1998

Constraint Matrix

i:	1	2	3	4	5	6	7	8	9	10	11	12	13	14		
	-1									1					≤	0
		-1								1					≤	0
				-1						1					≤	0
					-1					1					≤	0
		-1									1				≤	0
			-1								1				≤	0
					-1						1				≤	0
						-1					1				≤	0
				-1								1			≤	0
					-1							1			≤	0
							-1					1			≤	0
								-1				1			≤	0
					-1								1		≤	0
						-1							1		≤	0
								-1					1		≤	0
									-1				1		≤	0
										-1				1	≤	0
											-1			1	≤	0
												-1		1	≤	0
													-1	1	≤	0

no apparent network structure!

... but consider the dual of the LP relaxation!

plus $Y_i \in \{0,1\}\ \forall\ i$

Dual LP:

MIN	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1		
1	-1																				1														≥	R_1
2		-1			-1																	1													≥	R_2
3						-1																	1												≥	R_3
4			-1						-1															1											≥	R_4
5				-1			-1			-1			-1												1										≥	R_5
6								-1						-1												1									≥	R_6
7											-1																1								≥	R_7
8												-1			-1													1							≥	R_8
9																-1													1						≥	R_9
10	1	1	1	1													-1													1					≥	R_{10}
11					1	1	1	1										-1													1				≥	R_{11}
12									1	1	1	1							-1													1			≥	R_{12}
13													1	1	1	1				-1													1		≥	R_{13}
14																	1	1	1	1														1	≥	R_{14}

This is ALMOST a node-arc incidence matrix!

derived from $Y_i \leq 1$

We will obtain a node-arc incidence matrix if we

- ***subtract surplus variables to convert to equations***
- ***add a row = negative of sum of all constraints***

MIN	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1		
1	-1																				1														≥	R_1
2		-1			-1																	1													≥	R_2
3						-1																	1												≥	R_3
4			-1						-1															1											≥	R_4
5				-1			-1			-1			-1												1										≥	R_5
6								-1						-1												1									≥	R_6
7											-1																1								≥	R_7
8												-1			-1													1							≥	R_8
9																-1													1						≥	R_9
10	1	1	1	1													-1													1					≥	R_{10}
11					1	1	1	1										-1													1				≥	R_{11}
12									1	1	1	1							-1													1			≥	R_{12}
13													1	1	1	1				-1													1		≥	R_{13}
14																	1	1	1	1														1	≥	R_{14}

sum of these rows will be zero!

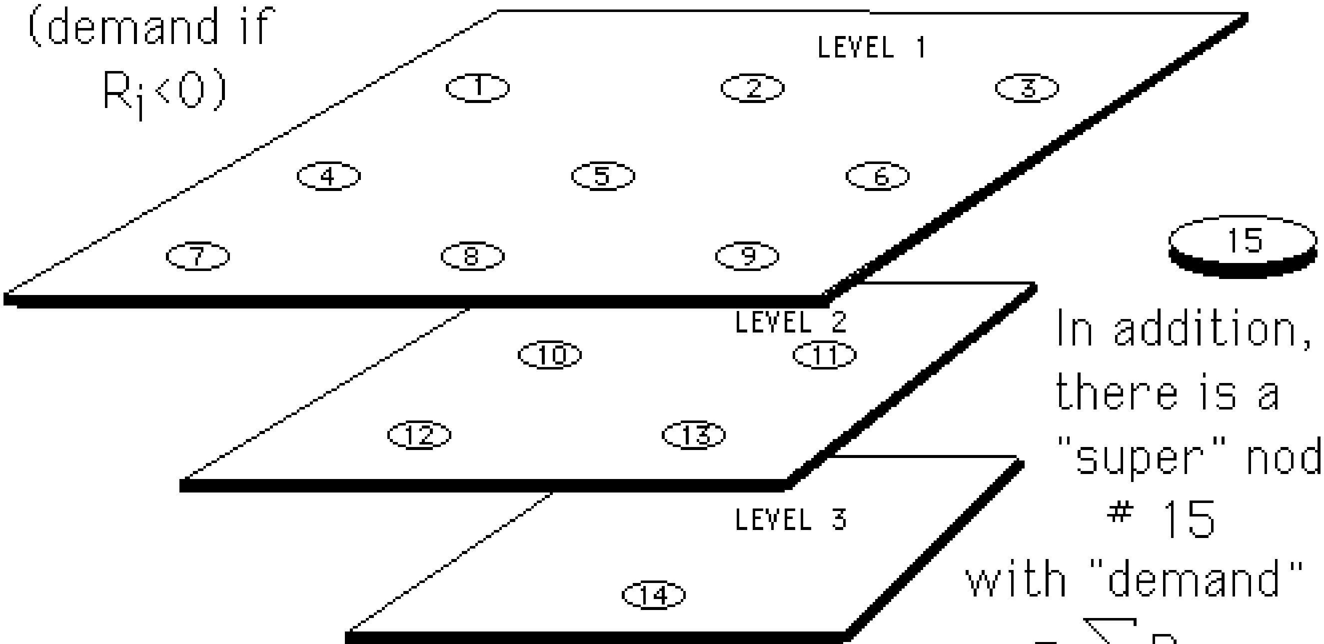
For what network is this the node-arc incidence matrix?

MIN	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1		
1	-1																				1															R_1
2		-1			-1																	1														R_2
3						-1																	1													R_3
4			-1						-1															1											•	R_4
5				-1			-1			-1			-1												1										•	R_5
6								-1						-1												1									•	R_6
7											-1																1								etc.	R_7
8												-1			-1													1								R_8
9																-1													1							R_9
10	1	1	1	1													-1													1					•	R_{10}
11					1	1	1	1										-1													1				•	R_{11}
12									1	1	1	1							-1													1			•	R_{12}
13													1	1	1	1				-1													1			R_{13}
14																	1	1	1	1														1		R_{14}
15																					-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1		$-\Sigma R_i$

These columns are negative of the preceding 14 columns!

For each block, there is a node, whose "supply" is R_i

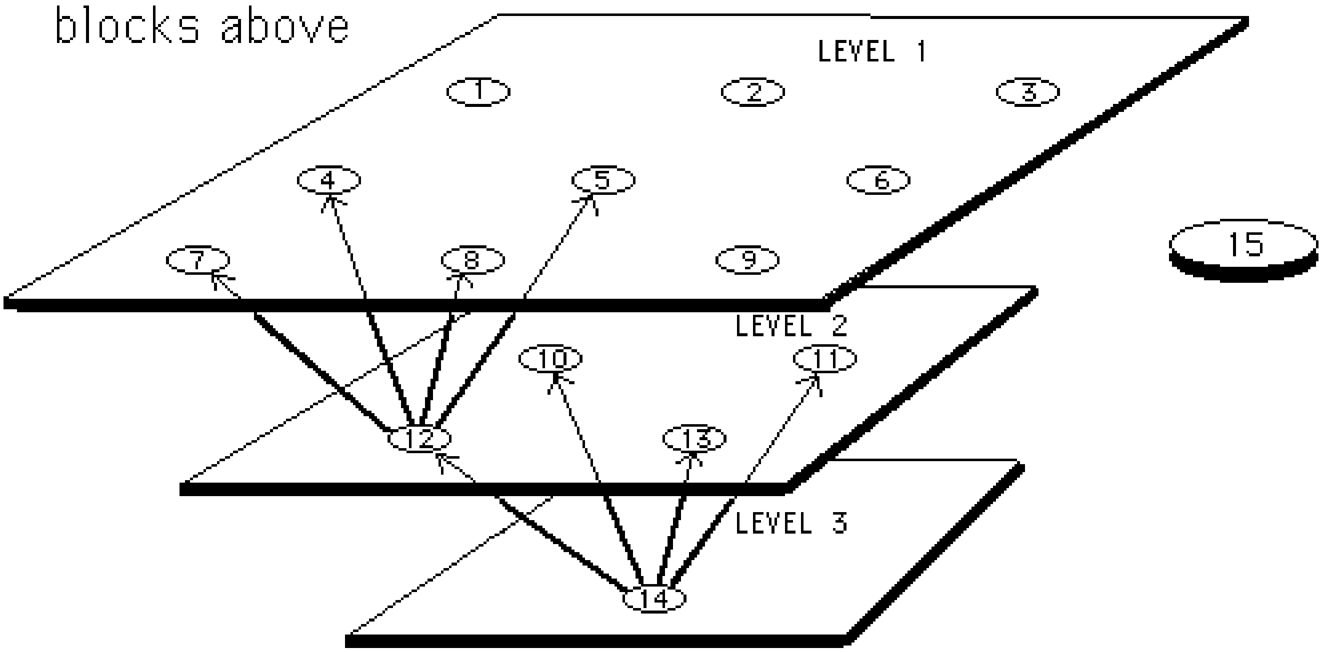
(demand if $R_i<0$)



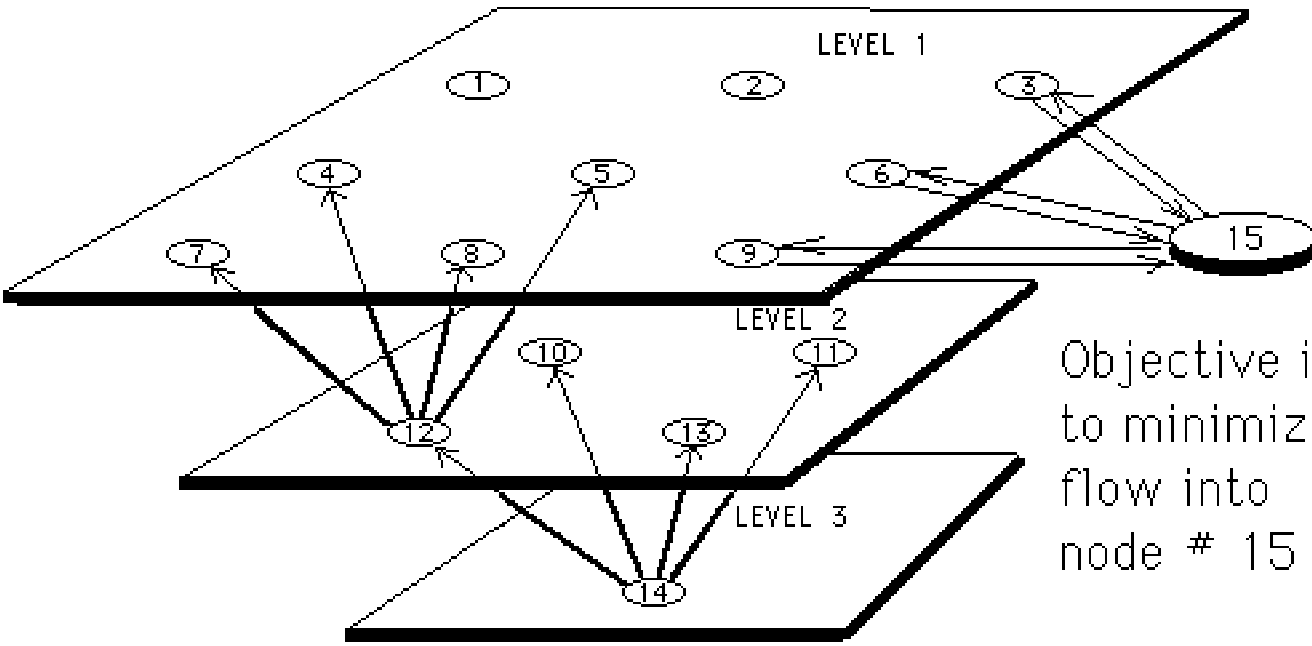
In addition, there is a "super" node # 15 with "demand"

$$= \sum_i R_i$$

There is an arc from each block to each of the 4 blocks above



There is a pair of arcs between each block and # 15



Objective is to minimize flow into node # 15

After solving the network problem, the solution of the original problem is obtained from the dual variables (simplex multipliers).

Because min-cost network flow problems are very efficiently solved by the network simplex method, while general-purpose branch-and-bound algorithms are very time-consuming, large versions of this problem can be solved only as network problems!

Another formulation:

The ***four*** constraints

$$\begin{cases} Y_{17} \le Y_1, & Y_{17} \le Y_2 \\ Y_{17} \le Y_5, & Y_{17} \le Y_6 \end{cases}$$

may be replaced by the ***single*** constraint

$$4Y_{17} \le Y_1 + Y_2 + Y_5 + Y_6$$

since $Y_{17} = 1$ is feasible in this constraint ***only if***

$$Y_1 = Y_2 = Y_5 = Y_6 = 1$$

Using these alternate constraints, our sample problem's formulation is reduced in size from 56 linear constraints to only 14!
However, whereas in the earlier formulation the integer restrictions can be relaxed and the problem solved as a min-cost network flow problem,
the new formulation will require the use of an integer programming algorithm such as branch-and-bound.

The computational effort will be increased by several orders of magnitude!