Christopher Matian
CS325 - Spring 2019
05/13/2019

<u>Problem One: Shortest Path using Linear Programming</u>

Programming Software/Language Used: **Lindo**

The distance between a given node and a direct parent are as follows (ordered from da to dg):

dg = 0	dc - db <= 4	de - dd <= 25
da - df <= 5	dc - df <= 3	de - df <= 2
da - dh <= 4	dd - dc <= 3	df - da <= 10
db - dh <= 9	dd - de <= 9	df - dd <= 18
db - da <= 8	dd - dg <= 2	dh - dg <= 3
db - df <= 7	de - db <= 10	dg - de <= 7

Solution and Code for Part A: The most optimal path to C starting from source G is **<u>16 units</u>**. The path is ***G => H => B => C.***

Input:

```
max dc
ST
       dg = 0
       dc - db <= 4
       de - dd <= 25
       da - df <= 5
       dc - df <= 3
       de - df <= 2
       da - dh <= 4
       dd - dc <= 3
       df - da <= 10
       db - dh <= 9
       dd - de <= 9
       df - dd <= 18
       db - da <= 8
       dd - dg <= 2
       dh - dg <= 3
       db - df <= 7
       de - db <= 10
       dg - de <= 7
END
```

Output

```
 LP OPTIMUM FOUND AT STEP      1
        OBJECTIVE FUNCTION VALUE
        1)      16.00000
```

```
  VARIABLE        VALUE          REDUCED COST
        DC        16.000000           0.000000
        DG         0.000000           0.000000
        DB        12.000000           0.000000
        DE         0.000000           0.000000
        DD         0.000000           0.000000
        DA         4.000000           0.000000
        DF        13.000000           0.000000
        DH         3.000000           0.000000

       ROW   SLACK OR SURPLUS     DUAL PRICES
        2)         0.000000           1.000000
        3)         0.000000           1.000000
        4)        25.000000           0.000000
        5)        14.000000           0.000000
        6)         0.000000           0.000000
        7)        15.000000           0.000000
        8)         3.000000           0.000000
        9)        19.000000           0.000000
       10)         1.000000           0.000000
       11)         0.000000           1.000000
       12)         9.000000           0.000000
       13)         5.000000           0.000000
       14)         0.000000           0.000000
       15)         2.000000           0.000000
       16)         0.000000           1.000000
       17)         8.000000           0.000000
       18)        22.000000           0.000000
       19)         7.000000           0.000000
 NO. ITERATIONS=       1
```

Solution and Code for Part B: The following table outlines the shortest path from the source G to a given node in the directed graph:

Node	**Path**	**Distance**
G	Source	0
A	G => H => A	7
B	G => H => B	12
C	G => H => B => C	16
D	G => D	2
E	G => H => A => F => E	19
F	G => H => A => F	17

H	G => H	3

***Input*:**

```
max dc + dg + da + db + de + df + dh
ST
       dg = 0
       dc - db <= 4
       de - dd <= 25
       da - df <= 5
       dc - df <= 3
       de - df <= 2
       da - dh <= 4
       dd - dc <= 3
       df - da <= 10
       db - dh <= 9
       dd - de <= 9
       df - dd <= 18
       db - da <= 8
       dd - dg <= 2
       dh - dg <= 3
       db - df <= 7
       de - db <= 10
       dg - de <= 7
END
```

***Output*:**

```
 LP OPTIMUM FOUND AT STEP      4
        OBJECTIVE FUNCTION VALUE
        1)      74.00000
  VARIABLE        VALUE          REDUCED COST
        DC        16.000000          0.000000
        DG         0.000000          0.000000
        DA         7.000000          0.000000
        DB        12.000000          0.000000
        DE        19.000000          0.000000
        DF        17.000000          0.000000
        DH         3.000000          0.000000
        DD         2.000000          0.000000

       ROW   SLACK OR SURPLUS     DUAL PRICES
        2)         0.000000          7.000000
        3)         0.000000          1.000000
        4)         8.000000          0.000000
        5)        15.000000          0.000000
        6)         4.000000          0.000000
        7)         0.000000          1.000000
        8)         0.000000          3.000000
        9)        17.000000          0.000000
```

```
      10)         0.000000          2.000000
      11)         0.000000          2.000000
      12)        26.000000          0.000000
      13)         3.000000          0.000000
      14)         3.000000          0.000000
      15)         0.000000          0.000000
      16)         0.000000          6.000000
      17)        12.000000          0.000000
      18)         3.000000          0.000000
      19)        26.000000          0.000000
NO. ITERATIONS=        4
```

Problem Two: Product Mix

Programming Software/Language Used: **Excel w/ Excel Solver Plug-in**
Excel File: **Problem2-ProductMix.xlsx**

The ***objective function*** below is what I used within Excel. A3, B3, C3 and D3 values correspond to a cell which holds the calculated value for the number of ties for each type. A3 = Silk, B3 = Polyester, C3 = Blend1, and D3 = Blend2.

Objective Function =
(6.7 * A3) + (3.55 * B3) + (4.31 * C3) + (4.81 * D3) - (0.75 * (A3 + B3 + C3 + D3)) - (0.125 * 20 * A3 + 0.08 * 6 * B3 + (0.05 * 6 + 0.05 * 9) * C3 + (0.03 * 6 + 0.07 * 9) * D3)

Excel Set-up:
Note - Ignore the decision variable results (7,000, … , 8500). They default to 0 before solving the problem using solver.

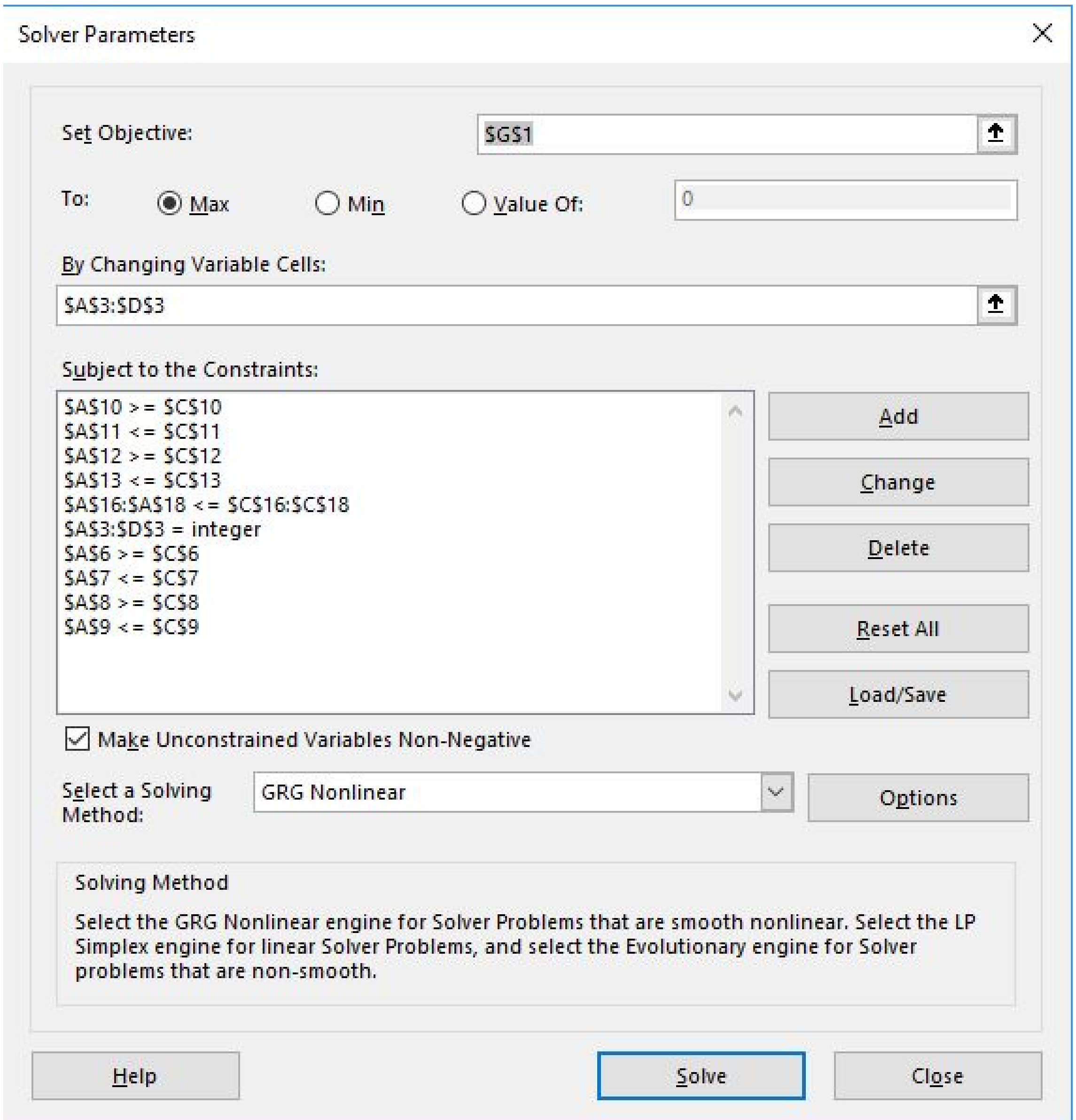
Decision Variable			
S	P	B1	B2
7000	13625	13100	8500
Constraints - From Sale Constraints			
=A3	>=	6000	
=A3	<=	7000	
=B3	>=	10000	
=B3	<=	14000	
=C3	>=	13000	
=C3	<=	16000	
=D3	>=	6000	
=D3	<=	8500	
Yards of Material Available			
=0.125*A3	<=	1000	
=0.08*B3+0.05*C3+0.03*D3	<=	2000	
=0.05*C3+0.07*D3	<=	1250	

Objective Function Set-up:

	F	G
	Objective	=(6.7 * A3) + (3.55 * B3) + (4.31 * C3) + (4.81 * D3) - (0.75 * (A3 + B3 + C3 + D3)) - (0.125 * 20 * A3 + 0.08 * 6 * B3 + (0.05 * 6 + 0.05 * 9) * C3 + (0.03 * 6 + 0.07 * 9) * D3)

Solver Window Set-up:

Note - There is no difference between using the GRG Nonlinear Engine or the Simplex LP engine. Both produce the same results.



Final Answer:
The company would need to produce each tie in quantities provided below:

Silk	**Polyester**	**Blend1**	**Blend2**
7,000	13,625	13,100	8,500

for a total maximized profit of **$120,196**

Problem Three: Transshipment Model

Programming Software/Language Used: **Excel /w Excel Solver Plug-in**
Excel File: **Problem3-TransshipmentModel.xlsx**

--- PART A ---
Note - Refer to the tab "Transshipment Model - Part A" in the excel file for Part A's solution.

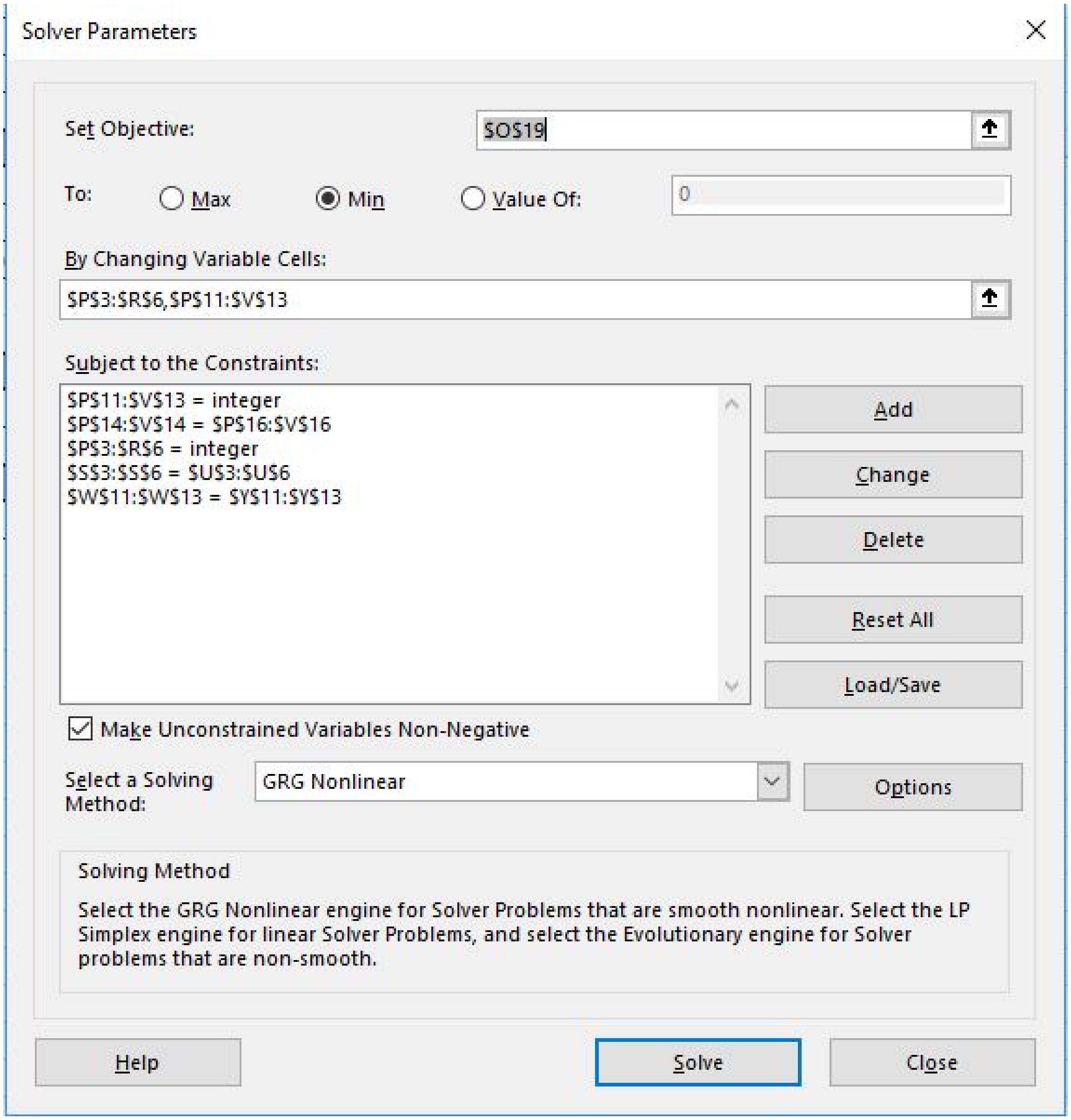
Objective Function: SUMPRODUCT(B2:D5, P3:R6)+SUMPRODUCT(B8:H10, P11:V13)

Excel Set-up:
Note - Increase view size to 200% or more to see the details better.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	cost	W1	W2	W3		Cost of Shipping from Plant -> Warehouse								
2	P1	10	15	100000										
3	P2	11	8	100000										
4	P3	13	8	9										
5	P4	100000	14	8										
6														
7	cost	R1	R2	R3	R4	R5	R6	R7		Cost of shipping from Warehouse -> Retailer				
8	W1	5	6	7	10	100000	100000	100000						
9	W2	100000	100000	12	8	10	14	100000						
10	W3	100000	100000	100000	14	12	12	6						
11														
12														
13	-	P1	P2	P3	P4		Capacity Per Plant							
14	Supply	150	450	250	150									
15														
16	-	R1	R2	R3	R4	R5	R6	R7		Demand Per Retailer				
17	Demand	100	150	100	200	200	150	100						
18														
19														Total Cost
20														Minimize

	O	P	Q	R	S	T	U	V	W	X	Y
1	Decision Variable										
2	Supply to Warehouse	W1	W2	W3	Supply Contraints						
3	P1	150	0	0	150	=	150				
4	P2	200	250	0	450	=	450				
5	P3	0	150	100	250	=	250				
6	P4	0	0	150	150	=	150				
7		350	400	250							
8											
9	Decision Variables										
10	Supply From Warehouse	R1	R2	R3	R4	R5	R6	R7			
11	W1	100	150	100	0	0	0	0	350	=	350
12	W2	0	0	0	200	200	0	0	400	=	400
13	W3	0	0	0	0	0	150	100	250	=	250
14		100	150	100	200	200	150	100			
15		=	=	=	=	=	=	=			
16		100	150	100	200	200	150	100			
17											
18											
19	17100										
20	Objective Function										

Solver Window Set-up & Constraints:



Final Answer for Part A:

Optimal shipping routes for Plants-to-Warehouses:

Supply to Warehouse	W1	W2	W3	Supply Contraints		
P1	150	0	0	150	=	150
P2	200	250	0	450	=	450
P3	0	150	100	250	=	250
P4	0	0	150	150	=	150
	350	400	250			

Optimal shipping routes for Warehouse-to-retailers:

Supply From Warehouse	R1	R2	R3	R4	R5	R6	R7			
W1	100	150	100	0	0	0	0	350	=	350
W2	0	0	0	200	200	0	0	400	=	400
W3	0	0	0	0	0	150	100	250	=	250
	100	150	100	200	200	150	100			
	=	=	=	=	=	=	=			
	100	150	100	200	200	150	100			

*The optimal minimal cost will be **$17,100**.*

-- PART B --

Note - Refer to the tab "Transshipment Model - Part B" in the excel file for Part B's solution.

Objective Function = SUMPRODUCT(B2:C5, P3:Q6)+SUMPRODUCT(B8:H9, P11:V12)

First, there *technically* is an optimal solution where you minimize your shipping costs. However, you have to alter your constraints in order to allow for warehouses to have leftover surplus that doesn't get distributed (or it is distributed to a retailer that doesn't need it which is bad because it increases shipping costs).

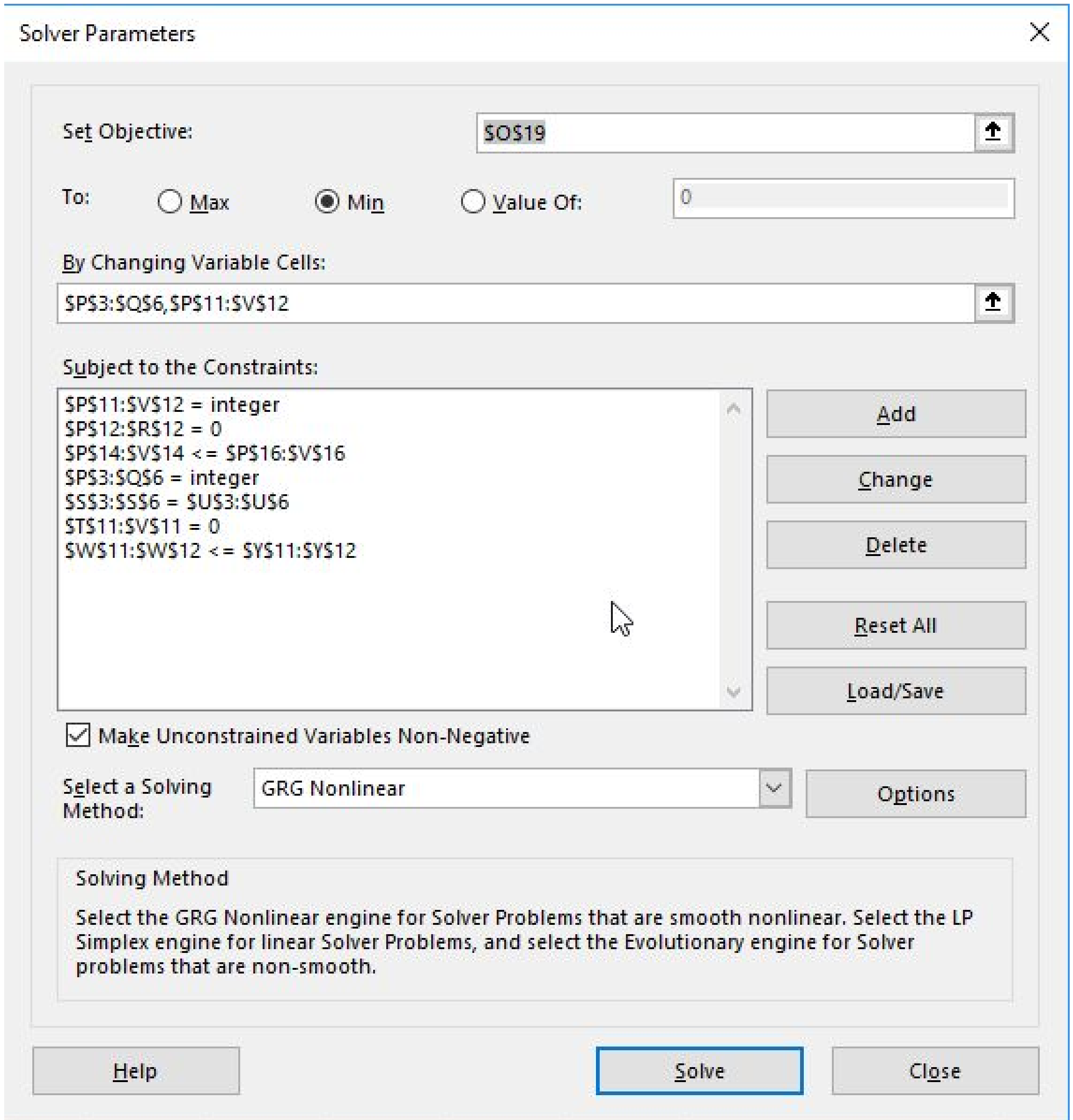
To explain further, the problem with removing warehouse 2 is that you produce a surplus in Warehouse 1. As it goes along each of the retailer's it can provide to (R1...R4), it distributes as much as it can to meet the retailer's demand. However, the moment it hits R4 we hit a situation where it can meet the retailer's demand but it has leftover surplus.

Essentially, the inclusion of Warehouse 2 eliminated the possibility of having over-surplus.

You can see this situation in the provided excel solution here:

Decision Variable										
Supply to Warehouse	W1	W3		Supply Contraints						
P1	150	0		150	=	150				
P2	450	0		450	=	450				
P3	0	250		250	=	250				
P4	0	150		150	=	150				
	600	400								
Decision Variables										
Supply From Warehouse	R1	R2	R3	R4	R5	R6	R7			
W1	100	150	100	200	0	0	0	550	=	600
W3	0	0	0	0	150	150	100	400	=	400
	100	150	100	200	150	150	100			
	=	=	=	=	=	=	=			
	100	150	100	200	200	150	100			

Solver Window Set-up & Constraints:



Final Answer for Part B:

If and only if you allow for surplus in warehouses, your optimal shipping cost will sit at around **$18,200.** Your retailers will still meet their demanded numbers. However, if you require absolute efficiency in the supply line (i.e. no surplus or shortages) you will not be able to achieve it without Warehouse 2.

Please refer to *"Transshipment Model - Part B"* for the full details and code.

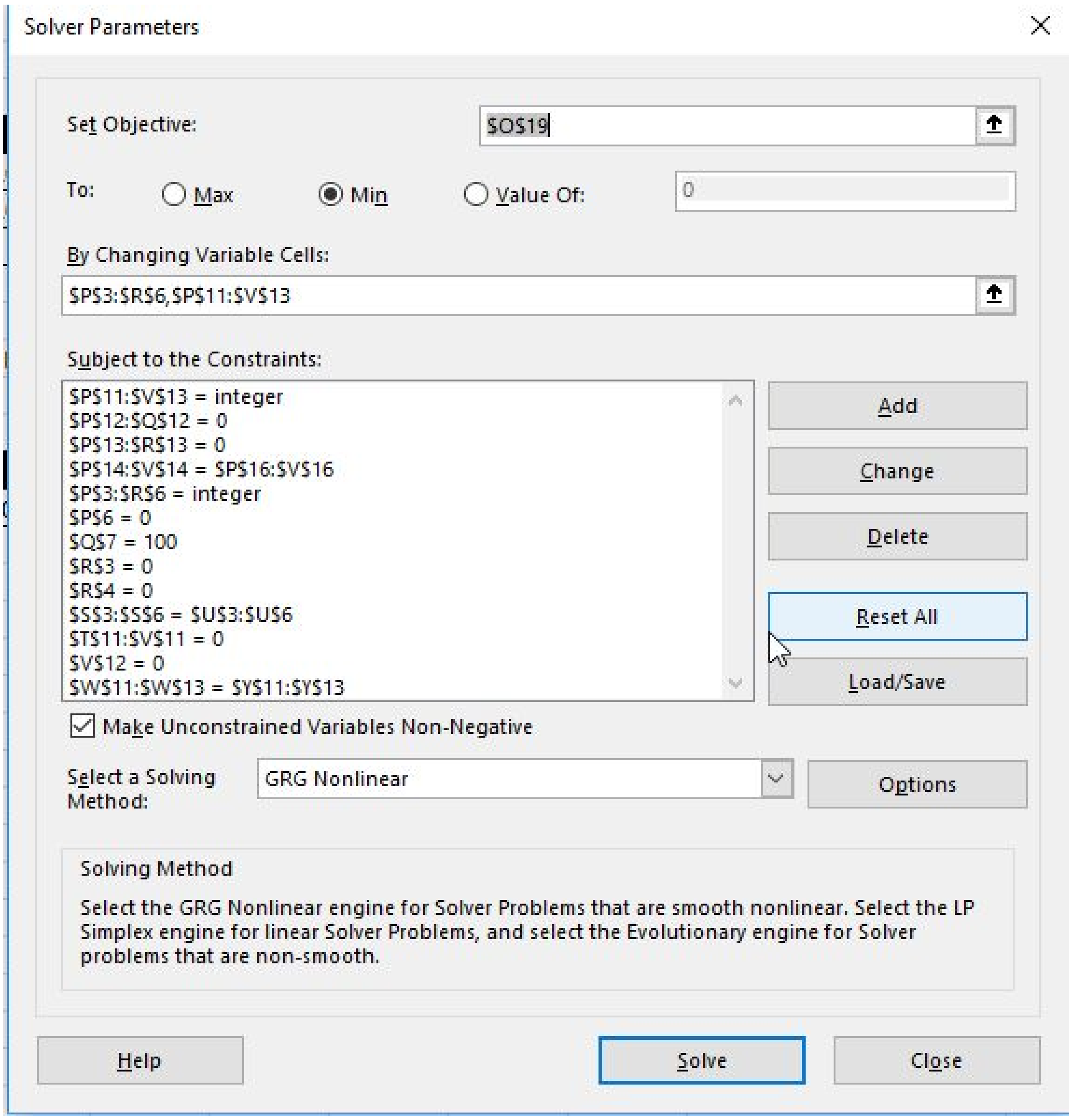
-- PART C --

Note - Refer to the tab "Transshipment Model - Part C" in the excel file for Part C's solution.

Objective Function: SUMPRODUCT(B2:D5, P3:R6)+SUMPRODUCT(B8:H10, P11:V13)

The excel set-up for part C is identical to part A. The only difference is that Warehouse 2's total stock must be equal to 100.

Solver Window Set-up and Constraints:



The constraints from Part A were also tightened up to prevent values from being inserted in routes that aren't viable between nodes.

Final Solution for Part C:
The final answer for part C is that there is indeed an optimal minimum cost when limiting warehouse 2 to 100 units. The optimal, minimal cost for shipping across all routes is **$18,300**.

Problem Four: Making Change

Programming Software/Language Used: **Excel /w Excel Solver Plug-in**
Excel File: **Problem4-MakingChange.xlsx**

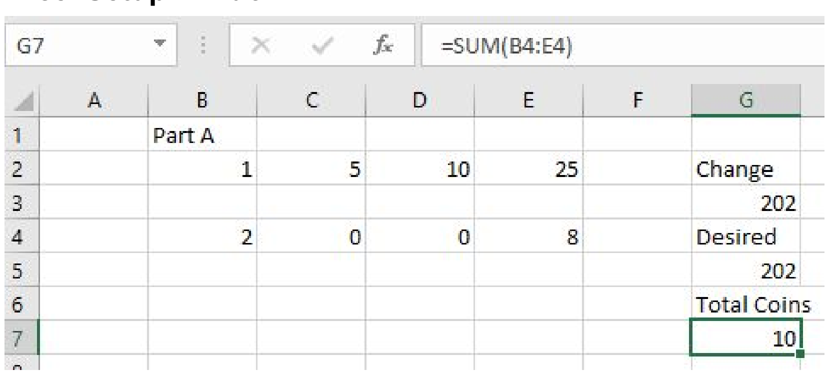
Objective Function: MIN(SUM(V1 + V2 + … + VN))
Note - the objective function above is generalized to encapsulate any number of integers given in a list. In the case of Part A: our objective function would be =SUM(B4:E4) and for Part B it would be =SUM(B4:F4). These 2 values are then used in Excel's Solver to find the ***minimum*** *number of coins.*

-- Part A --
Note - Refer to tab Part A of the relevant spreadsheet for details.

Answer: 10 coins

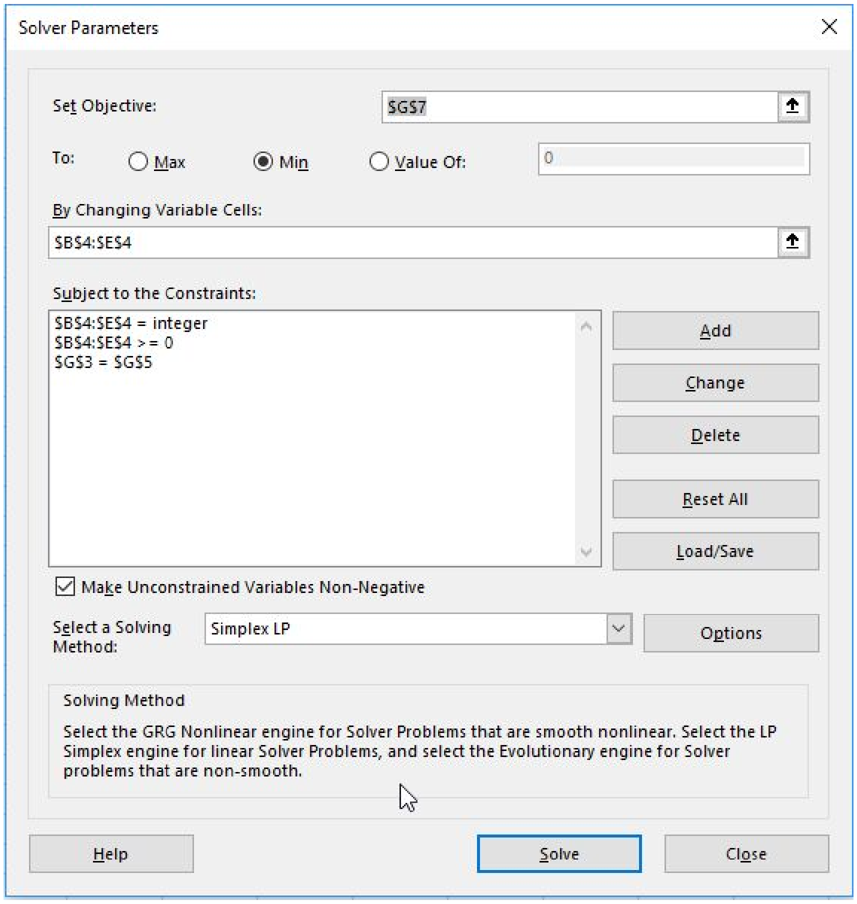
Coin	**1**	**5**	**10**	**25**
Quantity	2	0	0	8

Excel Setup Window:


G7 =SUM(B4:E4)

	A	B	C	D	E	F	G
1		Part A					
2		1	5	10	25		Change
3							202
4		2	0	0	8		Desired
5							202
6							Total Coins
7							10

Solver Setup Window & Constraints:



-- Part B --

Note - Refer to tab Part B of the relevant spreadsheet for details.

Answer: 14 coins

Coin	**1**	**3**	**7**	**12**	**27**
Quantity	0	0	2	3	9

Excel Setup Window:

H7 =SUM(B4:F4)

	A	B	C	D	E	F	G	H
1		Part B						
2		1	3	7	12	27		Change
3								293
4		0	0	2	3	9		Desired
5								293
6								Total Coins
7								14

Solver Setup Window & Constraint