Module 12: Integer Programming: Binary Variables

Reading Material: 12.1 – Introduction

We first saw binary (0/1) variables when creating assignment LP models in the previous module. This module, through selective examples, will further examine some of the many applications of this valuable modeling construct. There are many different scenarios in which yes/no decisions are applicable in practice, so it is impossible to be comprehensive. As before, the examples chosen to help illustrate the 0/1 variable concepts are meant as an in-class exercise for students to experience "Ah-ha" moments themselves.

Reading Material: 12.2 – Example 1: What's in Your Wallet? (or Knapsack)

This type of scenario is a classic starting point for introducing binary variable use in mathematical programming models. This category of problem is often referred to as a "knapsack problem," mimicking the attempt to select an optimal set of items taken in a knapsack (backpack) on a trip, subject to weight or other constraints. This one works really well with props in class. But the SPAM always goes quickly in post-class sharing of goodies.

12.2.1 Problem Description

The following table shows 11 items under consideration by a Boy Scout for his carrying pack for a backpacking trip. The first four items are food items (though one might debate whether some of these items are actually food), the next four drinks, and the final three weapons. The first number is the "Value" parameter for that item, the second entry, the "size." For instance, SPAM, a food (good fried with ketchup), has a value of 22 and a size of 8.

SPAM	22	8	Water	25	10
Vienna Sausage	12	7	Vanilla Coke	12	7
Chicken Spread	14	6			
Fiber Bar	9	4	Stick	6	1
Gatorade	30	12	Gun	38	14
Root Beer	16	7	Knife	17	9

Create a binary (0/1) LP model that will find the optimal items to bring that maximizes the combined value of the items while being less than 25 units of size. Note that you must select one food item, one drink, and one weapon.

12.2.2 Model Creation

Before we go to the spreadsheet, some algebra preliminaries are in order.

First, what is the model deciding? Which of the 11 objects should be brought along on the scout's backpacking trip. Thus, there are 11 decision variables, each one binary (yes/no, 0/1). If equal to 1, the variables indicate that object should be packed. If their value is set equal to 0, the object is not packed.

For simplicity, we will use a two-digit alpha representation for decision variable names in our example: SP, VS, CS, FB, GA, RB, WA, VC, ST, GU, KN.

The objective function and size constraint are straightforward: all 11 terms appear with the corresponding coefficients shown in the table. The size constraint is a less than or equal to constraint with a right-hand side (RHS) of 25.

The "one food," "one drink," and "one weapon" constraints are also fairly straightforward, but perhaps not as obvious.

These three constraints are all equalities – because there must be EXACTLY one from each group.

For food: SP + VS + CS + FB = 1.

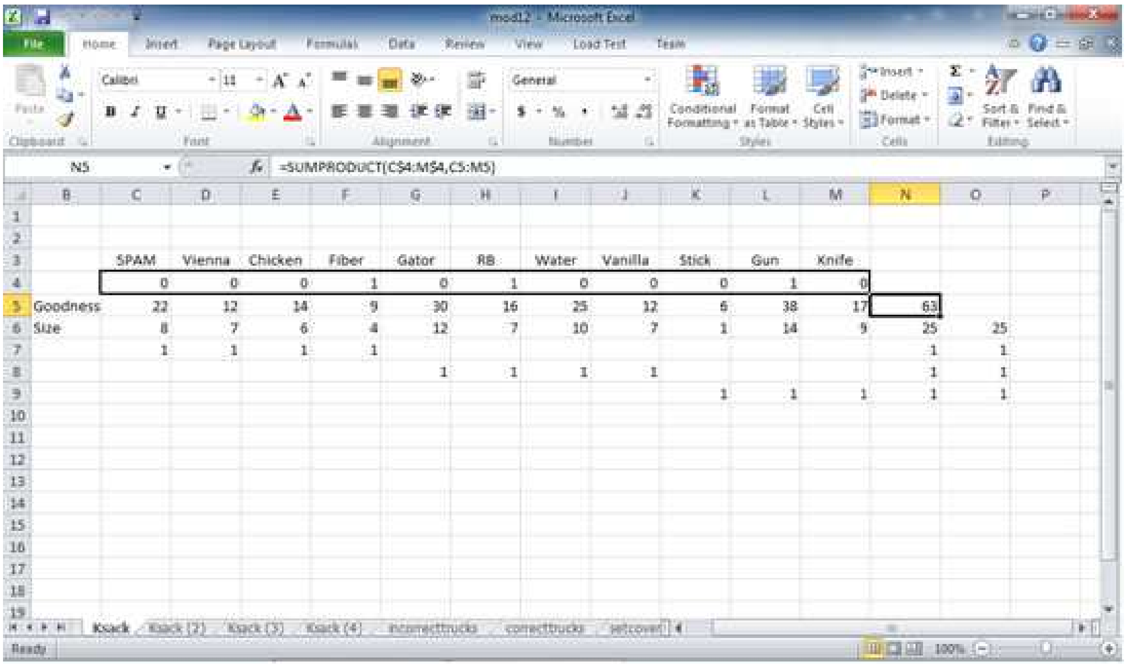
For drink: GA + RB + WA + VC = 1

For weapons: ST + GU + KN = 1

Thus, the object attribute of "type" can be constrained in this manner. This general modeling constraint can be helpful in assignment situations in which individuals involved in an analysis may have certifications, belong to certain defined groups, play certain positions, or any other category type, where such attributes are key in the decision process.

12.2.3 Spreadsheet Implementations

Figures 12.1 and 12.2 show a spreadsheet implementation of this scenario using row/column format. Figure 12.1 is the Excel model, and Figure 12.2 shows the Solver input.



N5 =SUMPRODUCT(C$4:M$4,C5:M5)

	B	C	D	E	F	G	H	I	J	K	L	M	N	O
3		SPAM	Vienna	Chicken	Fiber	Gator	RB	Water	Vanilla	Stick	Gun	Knife		
4		0	0	0	1	0	1	0	0	0	1	0		
5	Goodness	22	12	14	9	30	16	25	12	6	38	17	63	
6	Size	8	7	6	4	12	7	10	7	1	14	9	25	25
7		1	1	1	1								1	1
8						1	1	1	1				1	1
9										1	1	1	1	1

Figure 12.1

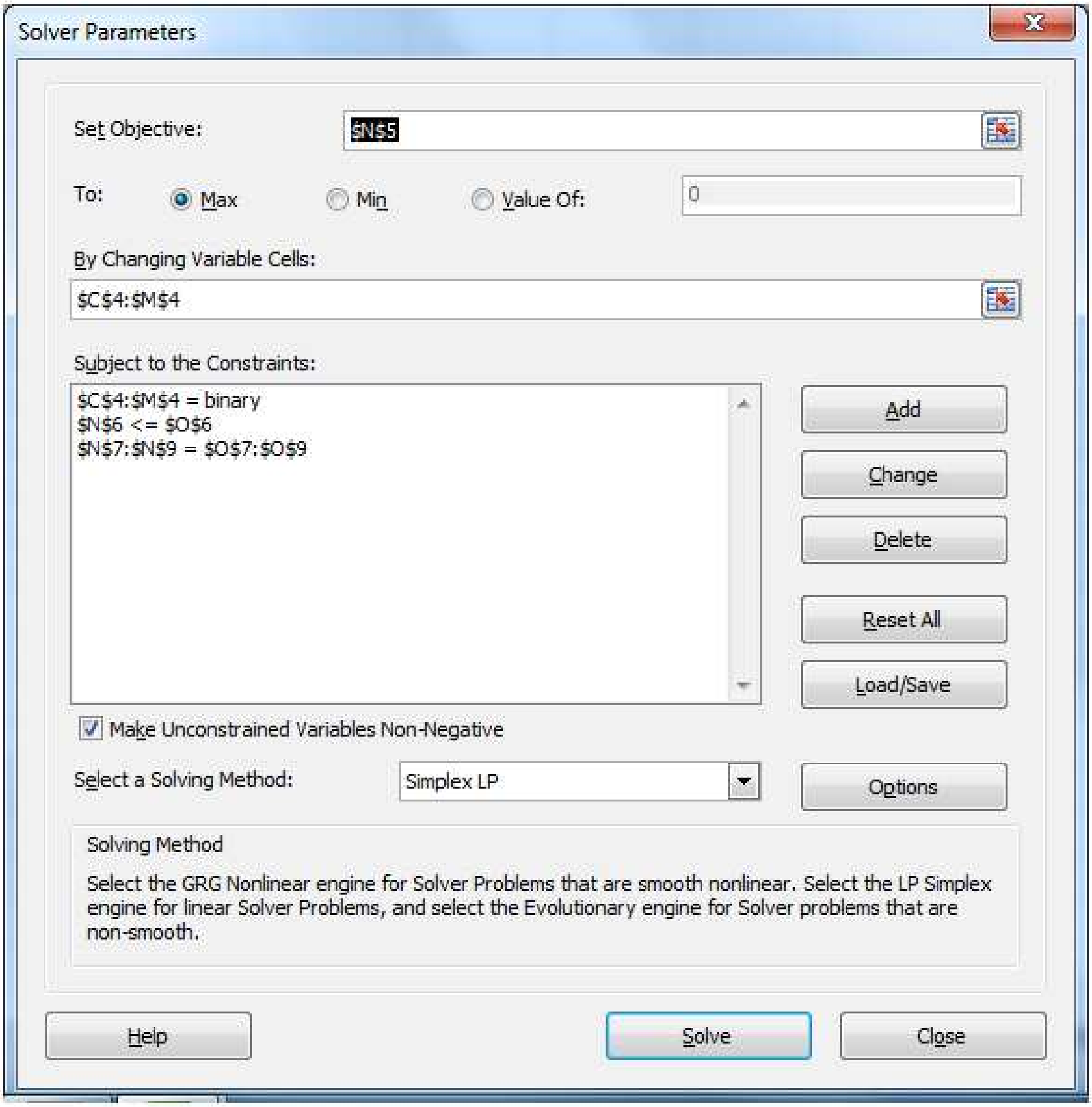


Figure 12.2

Note the optimal composition of the knapsack is a fiber bar, root beer, and a gun, providing a value of 63 units and using all 25 units of size.

Figures 12.3 and 12.4 show an equivalent way of implementing the same ZOLP (ZO = zero/one or binary). Cells C9 through C11 simply sum up the corresponding entries in Row 4; for instance, the formula in C9 is =SUM(C4:F4), the formula in C10 is =SUM(G4:J4), and the formula in C11 is =SUM(K4:M4). These three cells are set equal to 1 in the Solver (see Figure 12.4).

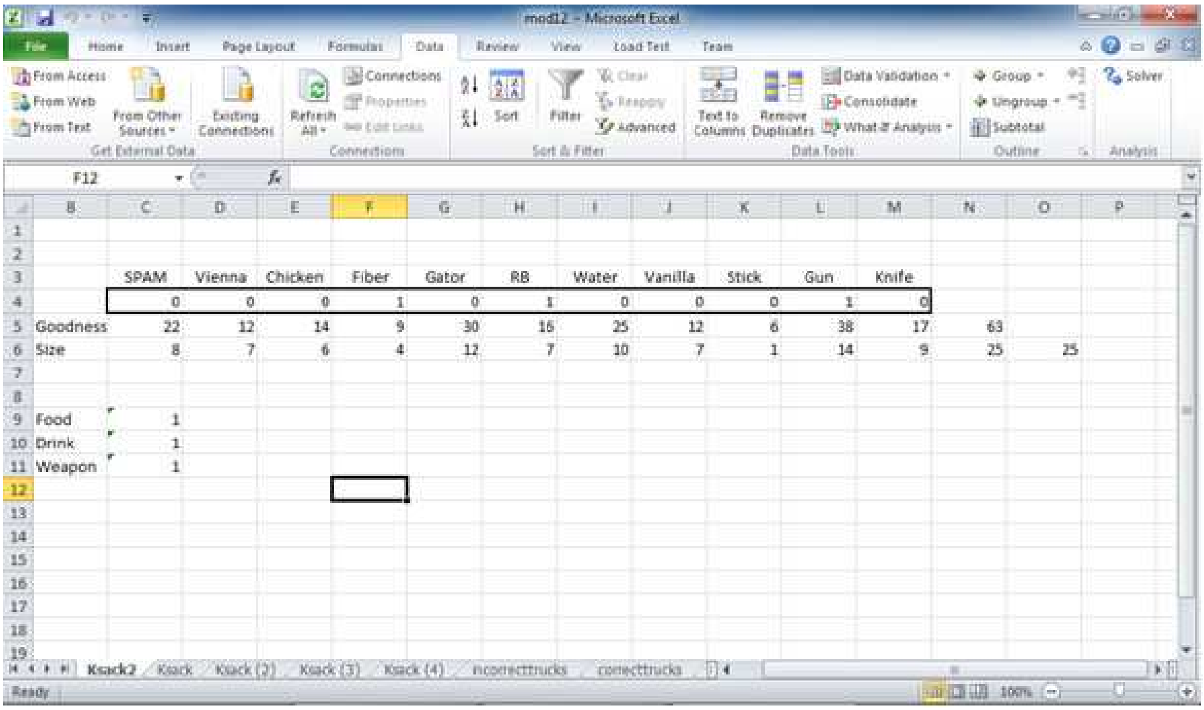

mod12 - Microsoft Excel
File
Home
Insert
Page Layout
Formulas
Data
Review
View
Load Test
Team
F12
SPAM Vienna Chicken Fiber Gator RB Water Vanilla Stick Gun Knife
0 0 0 1 0 1 0 0 0 1 0
Goodness 22 12 14 9 30 16 25 12 6 38 17 63
Size 8 7 6 4 12 7 10 7 1 14 9 25 25
Food 1
Drink 1
Weapon 1
Ksack2
Ksack
Ksack (2)
Ksack (3)
Ksack (4)
incorrecttrucks
correcttrucks
Ready

Figure 12.3

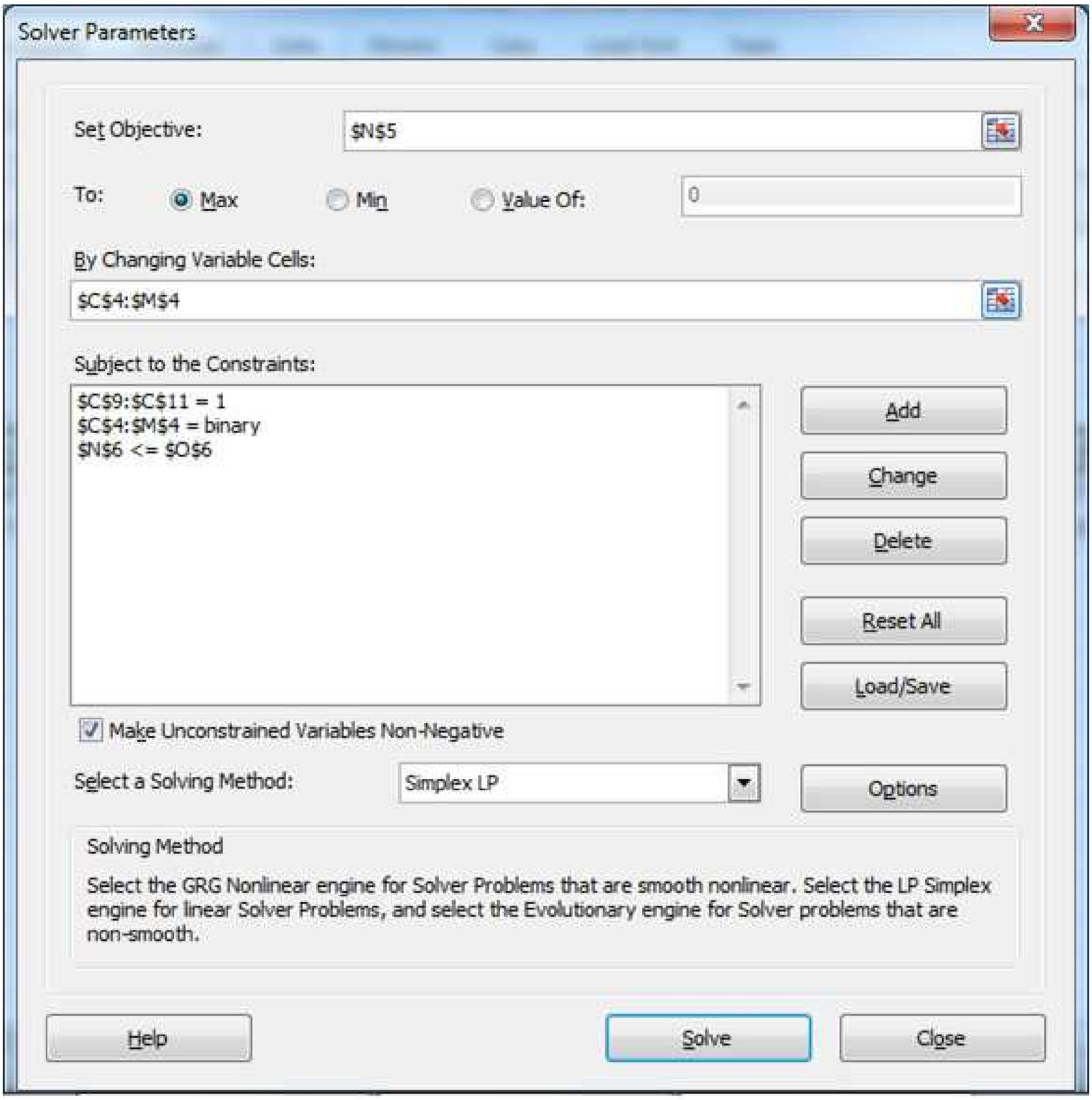


Figure 12.4

The resulting solution is of course the same as before. This is just another "bashing you over the head" with the fact that the same exact model can be implemented in different ways in Excel.

12.2.4 Additional Requirement: Doing Conditions with 0/1 Variables

Suppose some contingencies were involved in our knapsack problem, specifically, that if the model selected root beer, SPAM had to be the food selected. But, if SPAM was selected, root beer was still optional. (I cannot even begin to think of a practical reason why one would combine root beer and SPAM, but let's not worry about it!)

Algebraically, this would be implemented as:

SP >= RB.

Why this way? If root beer is selected by the model, RB=1. This constraint then becomes the equivalent of SP >= 1. Thus, this implies that SPAM must also be selected.

Consider the opposite, that SPAM was selected (SP=1) – what is the impact on root beer? This constraint becomes:

RB <= 1

Since RB is either 0 or 1, this constraint has no actual impact, and other model factors would be utilized to decide whether Root beer should be chosen.

Figure 12.5 shows the constraint added in row/column form. Note that it is added as:

SP –RB >= 0.

Additionally, note the subtle difference between the following two constraints:

SP >= RB
(If root beer is selected, then so must SPAM be, but NOT vice versa.)

and

SP = RB
(If root beer is selected, then so must SPAM be, but if root beer is NOT selected, SPAM cannot be either!).

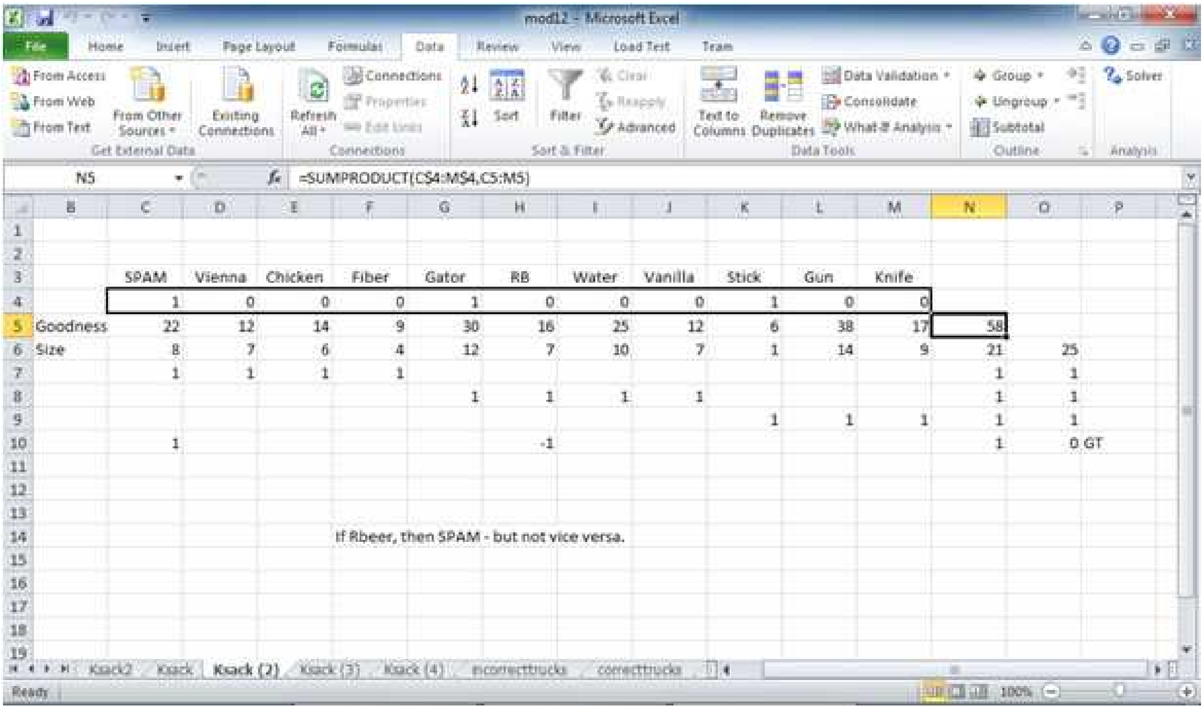


Figure 12.5

This is a minor point but useful as one tries to implement conditions in object selection. A variety of different conditional situations are implementable using 0/1 variables, but we choose to not linger here in too much detail or detract from the main learning goals of the module.

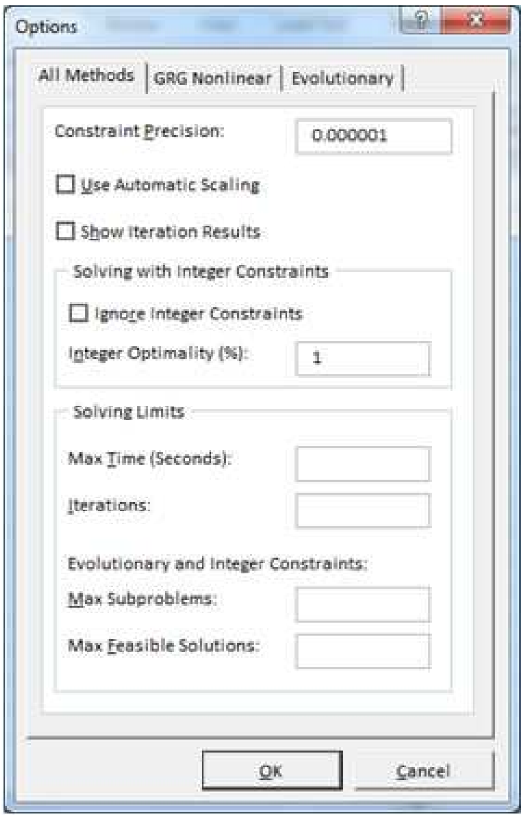


Figure 12.6

The solution shows that SPAM and root beer are not selected; instead a fiber bar, vanilla Coke, and the gun appears as the optimal solution, for a maximized value of 59. So, the condition of linking SPAM to root beer caused the model to change – not to add SPAM but to drop the fiber bar.

A final comment – I almost hate to bring it up, but some of you might be thinking about it. Remember, all models up to this point should be linear in nature. Some of you might be tempted to try to implement our selection "conditions" using IF/THEN/ELSE statements in Excel – DON'T!!!! Use our linear 0/1 related constraints instead. First, Excel IF statements are non-linear. Second, they'll not work as you intend them because they are reactionary in nature, versus using decision variables, which are calculated by the Solver (to ensure optimality). Trust me on this – trying to do solver's job with a non-Solver function will not work at all.

Ok, so it was not a final comment. Another reminder on another topic – make sure your integer programming options are all set up properly as well. Recall that as we transition from Excel 2007 to Excel 2010 we need to make sure that integer requirements are enabled and that the 5% versus 1% integer optimality parameters are understood (see integer programming module).

For reference, Figure 12.6 is the options screen where parameters can be seen and checked/updated. Now, on to the next example illustrating the use of binary variables.

An update on Solver from August, 2013. During the Summer semester, we discovered an odd error in the solution process when a model had binary variables that occurred if one checked (or allowed a check to occur) the "Use Automatic Scaling" box on the Options screen. While the programmers in Nevada were researching it, the fix is to not check the box. So, one more thing to check on this screen. Stay tuned for further details. There really was no reason to use the box in the first place.

Reading Material: 12.3 – Example 2: Taylor's Trucks

12.3.1 Scenario Description

You (Taylor) own two trucks, each with a capacity of 122 units of some unnamed measure of weight. In your business, you pick up large packages from customers and deliver them to FedEx for further mailing.

Today, you have seven potential customers; three of them are repeat customers you already know are "high maintenance" (HIGH column) – implying that they are a little hard to work with. Each customer has an estimated profit (based on rates and cost to travel to the customer's location) and cargo of a certain weight. Trucks are limited by capacity, and each truck, by policy, can pick up packages at no more than 1 high-maintenance customer (this limits the chance of a truck getting delayed).

Obviously, with three high maintenance customers, not every potential customer is going to be served/selected. The customer data is shown in the following table.

High?	Customer	Weight	Profit
1	A	50	22
0	B	60	31
0	C	30	18
1	D	15	9
0	E	70	34
1	F	24	10
0	G	44	25

Your task: Find the best way of loading the trucks subject to capacity and the "high maintenance" requirement that maximizes overall profit.

12.3.2 The Wrong Approach

Having been "around the block" a time or two (not necessarily picking up customer cargo though), sometimes (not always) we can anticipate potential modeling traps. Let's talk about a potential one in this problem situation.

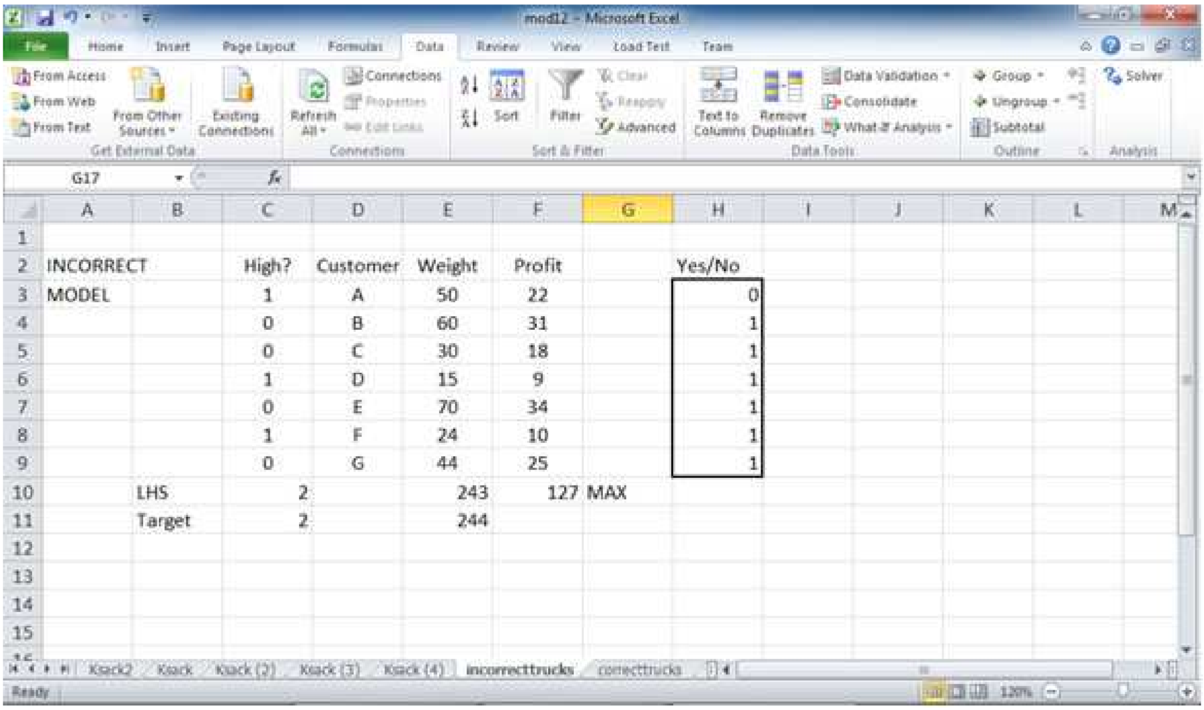


Figure 12.7

For this problem, one might think a simple approach would be to create a knapsack-like model with seven 0/1 variables (select a customer or not), maximize profits, have a weight constraint like the previous example, and then have a composition constraint (like Food, Drink, and Weapon above) for the high maintenance customers. With two trucks, it might seem reasonable to have a single constraint for package weight with an RHS of 2×122 = 244 (twice the capacity of a single truck), and then a similarly coded high maintenance constraint with an RHS of 2.

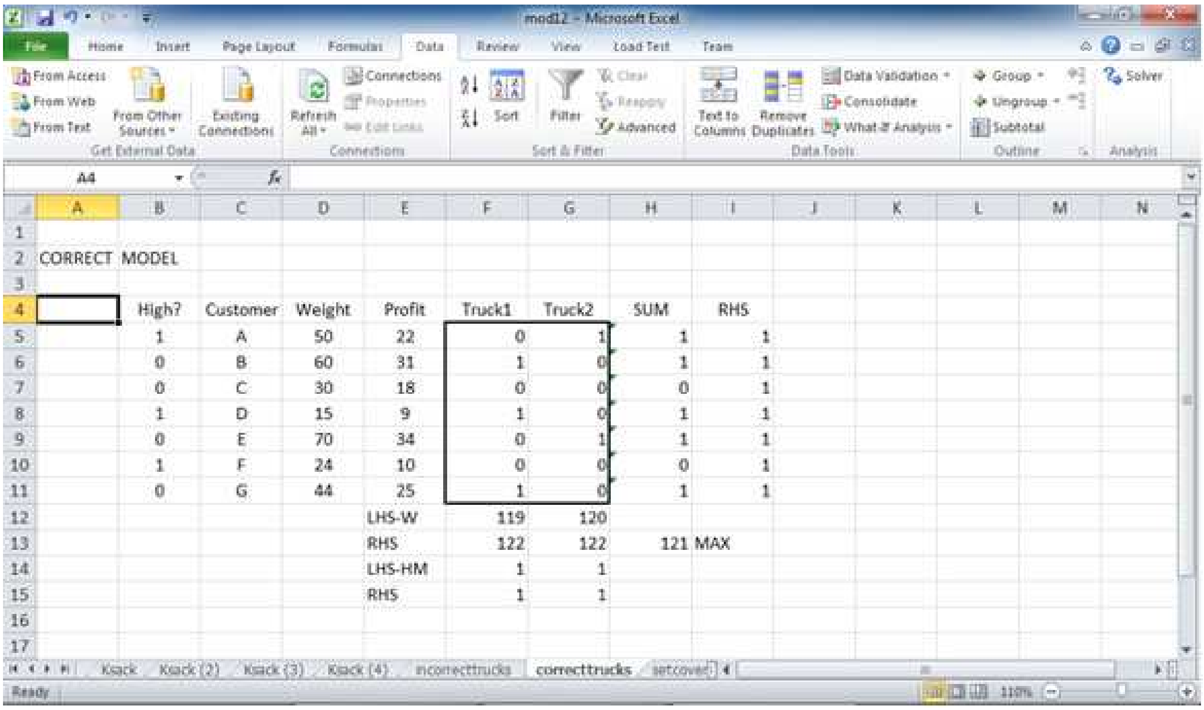
Figure 12.7 implements this simple model of seven decision variables and two constraints. It selects every customer but A for a profit of 127 units, has only two high-maintenance customers and requires 243 units of weight. Seems like a reasonable solution?

The problem: Can the customer cargo be partitioned such that the individual truck limit of 122 is maintained? With this combination of customers, it cannot be done. We have a flaw in our model. (If you don't believe me, please try to get each truck to 122 units).

The fix: We need to explicitly model the individual trucks, even if profit does not depend on the specific truck used. Because capacity is a physical constraint that cannot be altered, this must be explicitly modeled in our LP model.

12.3.3 The Right Approach

Figure 12.8 illustrates an approach that will handle Taylor's Truck tasks. There are seven customers who can be assigned to one of two trucks – thus, a matrix style of decision variables previously used in assignment type models is appropriate. A 7 × 2 matrix will handle the 14 different 0/1 decisions – yes or no, is customer X assigned to Truck Y?



	A	B	C	D	E	F	G	H	I
1									
2	CORRECT MODEL								
3									
4		High?	Customer	Weight	Profit	Truck1	Truck2	SUM	RHS
5		1	A	50	22	0	1	1	1
6		0	B	60	31	1	0	1	1
7		0	C	30	18	0	0	0	1
8		1	D	15	9	1	0	1	1
9		0	E	70	34	0	1	1	1
10		1	F	24	10	0	0	0	1
11		0	G	44	25	1	0	1	1
12					LHS-W	119	120		
13					RHS	122	122	121	MAX
14					LHS-HM	1	1		
15					RHS	1	1		

Figure 12.8

Because of the unique specifics of this scenario, we will subtly modify some of the "always do this" rules of a matrix LP approach. One of our previous mantras – always do row sums and column sums with a matrix style of decision variables.

Column H calculates the variable row sums – this calculation sums up the number of times Customer X is assigned. Obviously, they can be assigned to at most 1 truck. Thus, this family of logical constraints is implemented by Column H being less than or equal to Column I. This constraint allows a customer to NOT be assigned, but does NOT allow the illogical occurrence of a customer being assigned to both trucks.

The model does need to implement a type of column-related constraint, but in this case, it is two column sumproducts, not sums. The specific weight limits of the trucks require consideration of the customer's package weight – so Row 12 is the SUMPRODUCT of Column D (Package Weight) and Column F/G (0/1 variables indicating whether that customer is assigned to the specific truck). The formula for F12: =SUMPRODUCT($D5:$D11,F5,F11).

Row 13 is the maximum amount of package weight per truck – so a second family of constraints is Row 12 <= Row 13, ensuring truck capacity is not exceeded.

Row 14 is similar to Row 12's calculation – in this case, it is a sumproduct of the indicator of whether a customer is high maintenance. This customer attribute is coded in Column B. So Row

14 is the sumproduct of Columns B and F/G. Specifically, the formula for F14: SUMPRODUCT($B5:$B11,F5,F11).

Row 15 is the maximum amount of high-maintenance customers allowed to be assigned to each truck. Thus, the final set of constraints is Row 14 <= Row 15.

Cell H13 is the model's target cell – it is a little different than past objective function cells in that we take advantage of other calculations done in the spreadsheet for other model components. Customer profit, our criteria to maximize, does NOT depend on which truck a customer is assigned, just whether the customer is assigned to any truck. Thus, Column H will indicate whether our model suggests agreeing to pick up a customer's package. Therefore, E times H is the proper sumproduct for determining our profit. The specific formula then for H13: =SUMPRODUCT (E5:E11,H5:H11).

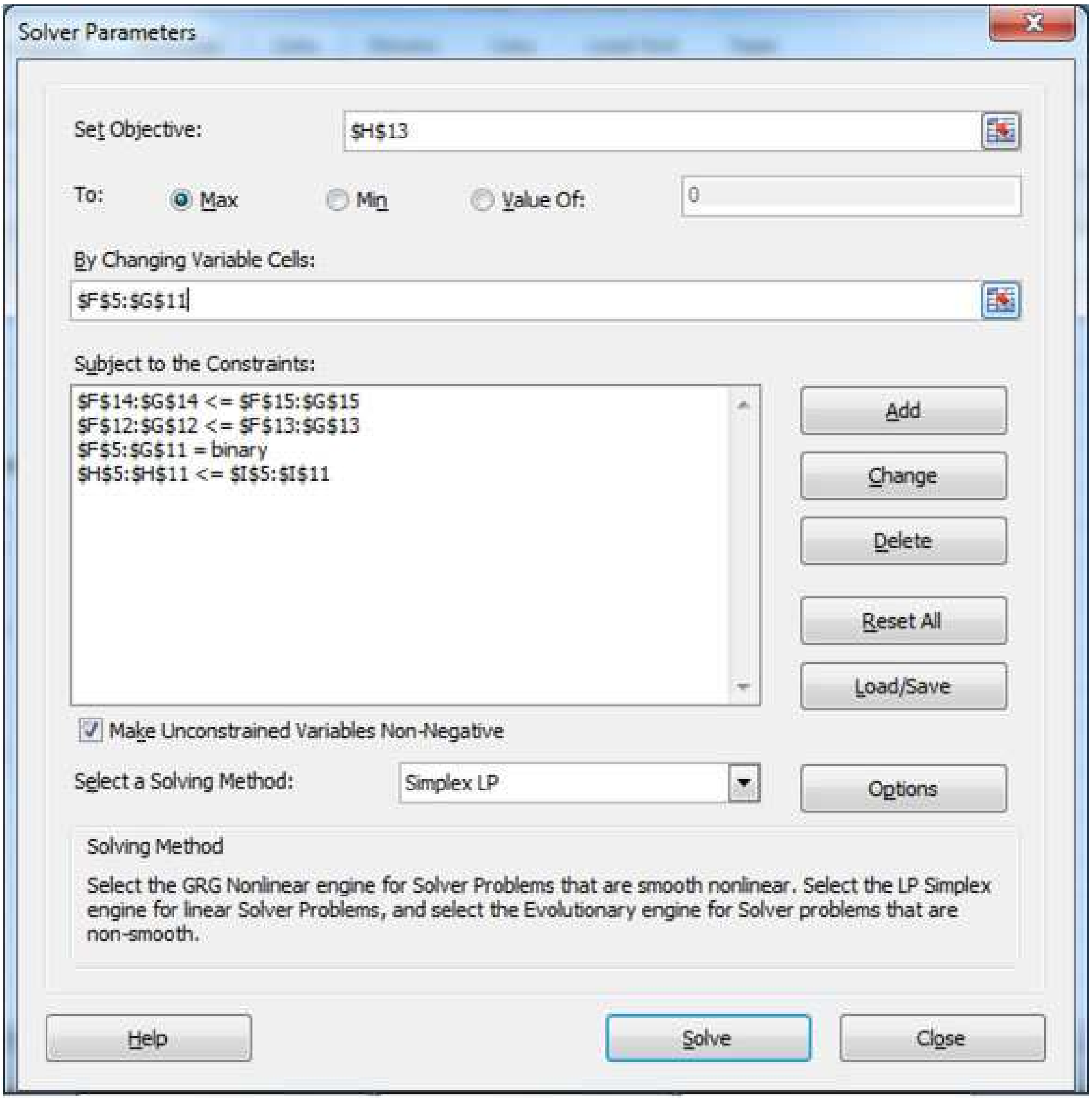


Figure 12.9

Figure 12.9 shows the Solver model, which also obviously includes defining the decision variable matrix as being binary variables. The solution shows that we can achieve a profit of 121 units by assigning Customers B, D, and G to one truck and A and E to another. Capacity and customer attribute constraints are met for each truck. Customers C and F are not desired for today's work.

It might be risky to put in an incorrect model into a book, but this example points out that although we want to have an insightful partnership between decision maker and model, it is best to let the model make the decisions and for us to properly provide the model the true parameters of the scenario.

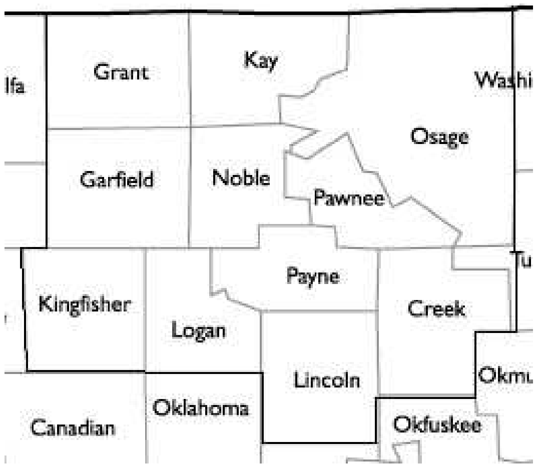
Next, the module presents another classic problem type – the set covering model. It is somewhat of a mix of 0/1 selection models and the staffing/rostering models previously examined. As a core model, it is found in many different applications at the “killer” level – many network design and logistic design problems begin with this type of model as its starting point.

Reading Material: 12.4 – Example 3: National Weather Service (NWS) Location Model – Storm Spotter Warehouses

12.4.1 Problem Description: NWS Problem

The NWS has commissioned the creation of a number of storage warehouses in North Central Oklahoma to store research equipment that trained weather spotters can use to assist in severe weather research. Such equipment includes mobile radars, specific weather balloon technology, Helen Hunt’s phone number, etc.

The 11-county region is shown below. It does not include Canadian, Oklahoma, Okfuskee counties.



There has been some work done in identifying potential locations within each county. One combined value has been calculated that represents a measure of cost/accessibility for each county. Small is better. This information is shown below.

County	Cost	County	Cost	County	Cost
Grant	8	Kay	12	Osage	9
Garfield	14	Noble	18	Pawnee	10
Kingfisher	5	Payne	13	Creek	17
Logan	7	Lincoln	6		

Find the least-cost way to locate NWS storage warehouses that covers all counties; each of the 11 counties should have a warehouse either in it or in a contiguous county.

Assume that the four square counties in the northwest corner are all contiguous to each other. Also, assume that Creek and Osage counties are contiguous. All other relationships should be visually obvious. It is permissible to have overlap, meaning multiple warehouses can be next to a particular county.

Note: Further geospatial research has shown that Creek and Osage county are NOT contiguous (by a matter of feet it seems). No matter, we will continue to assume they are. Cartographers, I apologize for this inaccuracy.

12.4.2 Modeling Preliminaries

As mentioned earlier, this is a classic "set covering" model (don't you just love all the terminology?). One wishes to find the optimal set of warehouse locations that cover all counties. A lot of cell phone tower design algorithms use this basic concept, and you can probably think of other application areas as well.

The decisions to be made: In which of the 11 counties should we place warehouses? Thus, this equates to eleven 0/1 decision variables, indicating yes or no, as to whether a warehouse should be located in a certain county. The objective is to minimize cost – that is a basic formula with 11 terms using the provided cost information.

The covering constraints for these types of problems are the key component. Much like the scheduling models seen previously, in which we wanted to staff shifts such that our hourly requirements were AT LEAST met, here, we want to locate warehouses such that every county has AT LEAST one warehouse either in it or next to it. It is typically not reasonable to expect that we could EXACTLY have one warehouse cover every county (likely infeasible), although minimizing the overlap (where we exceed the minimum required number) might be a secondary objective (or a primary one for that matter). Of course, in the case of cell phone design or even in this hypothetical scenario, we might want design redundancy!

To create these constraints, let us examine Grant County. Grant County is covered if a warehouse is located in Grant County (no kidding!) or in the three other counties that are contiguous – Kay, Noble, or Garfield. Mathematically then, this constraint for Grant County is:

GRANT + KAY + NOBLE + GARFIELD >= 1

Similarly, then, for GARFIELD county:

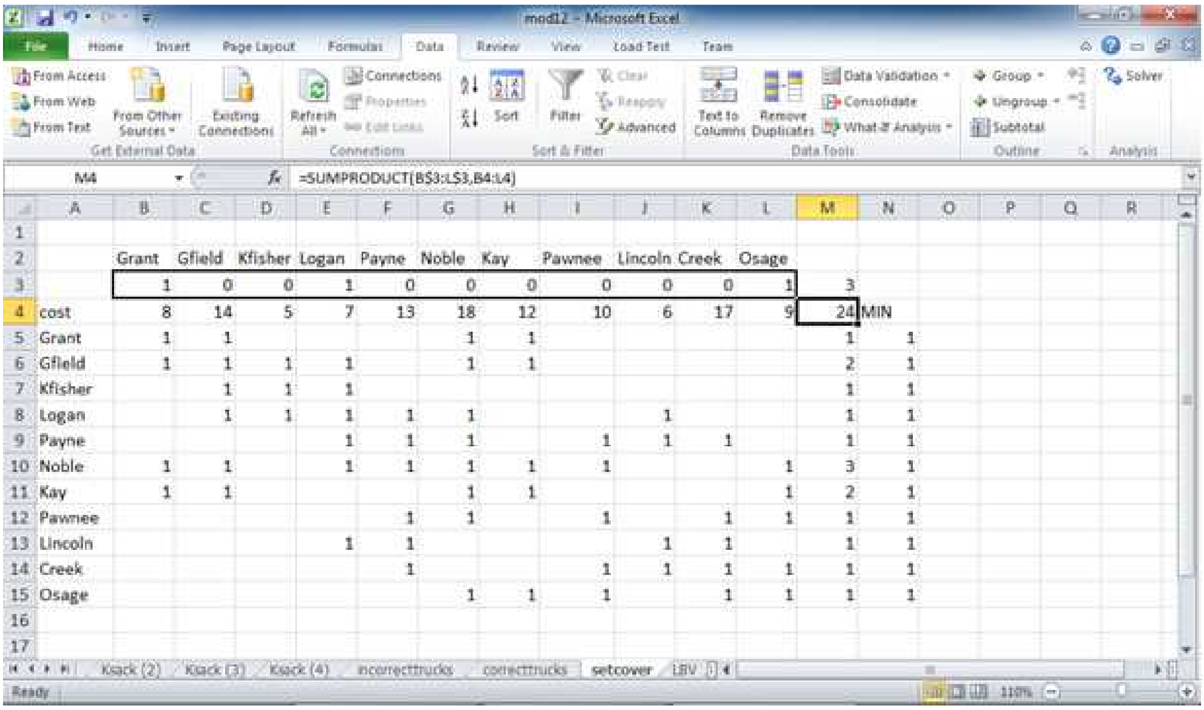
GARFIELD + GRANT + KAY + NOBLE + LOGAN + KINGFISHER >= 1.

I sense you can see the pattern. An expanded knapsack type configuration – we have one constraint for every entity that needs covering. In this case, that would be 11. The fact that we have 11 constraints and 11 decision variables is not required for this type of problem – it stems from the manner in which we are analyzing both the locations and requirements.

12.4.3 Model Solution

Figure 12.10 provides the Excel formulation for the NWS problem, and Figure 12.11 shows the extensive Solver model required (yes, that is me being facetious).

The decision variables are in Row 3 – yes/no, is a warehouse located in the counties (labels in Row 2). Row 4 is the cost for locating a warehouse in a particular county – thus, target Cell M4 is just a sumproduct of Row 3 and Row 4.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1														
2		Grant	Gfield	Kfisher	Logan	Payne	Noble	Kay	Pawnee	Lincoln	Creek	Osage		
3		1	0	0	1	0	0	0	0	0	0	1	3	
4	cost	8	14	5	7	13	18	12	10	6	17	9	24	MIN
5	Grant	1	1				1	1					1	1
6	Gfield	1	1	1	1		1	1					2	1
7	Kfisher		1	1	1								1	1
8	Logan		1	1	1	1	1			1			1	1
9	Payne				1	1	1		1	1	1		1	1
10	Noble	1	1		1	1	1	1	1			1	3	1
11	Kay	1	1				1	1				1	2	1
12	Pawnee					1	1		1		1	1	1	1
13	Lincoln				1	1				1	1		1	1
14	Creek					1			1	1	1	1	1	1
15	Osage						1	1	1		1	1	1	1
16														
17														

Figure 12.10

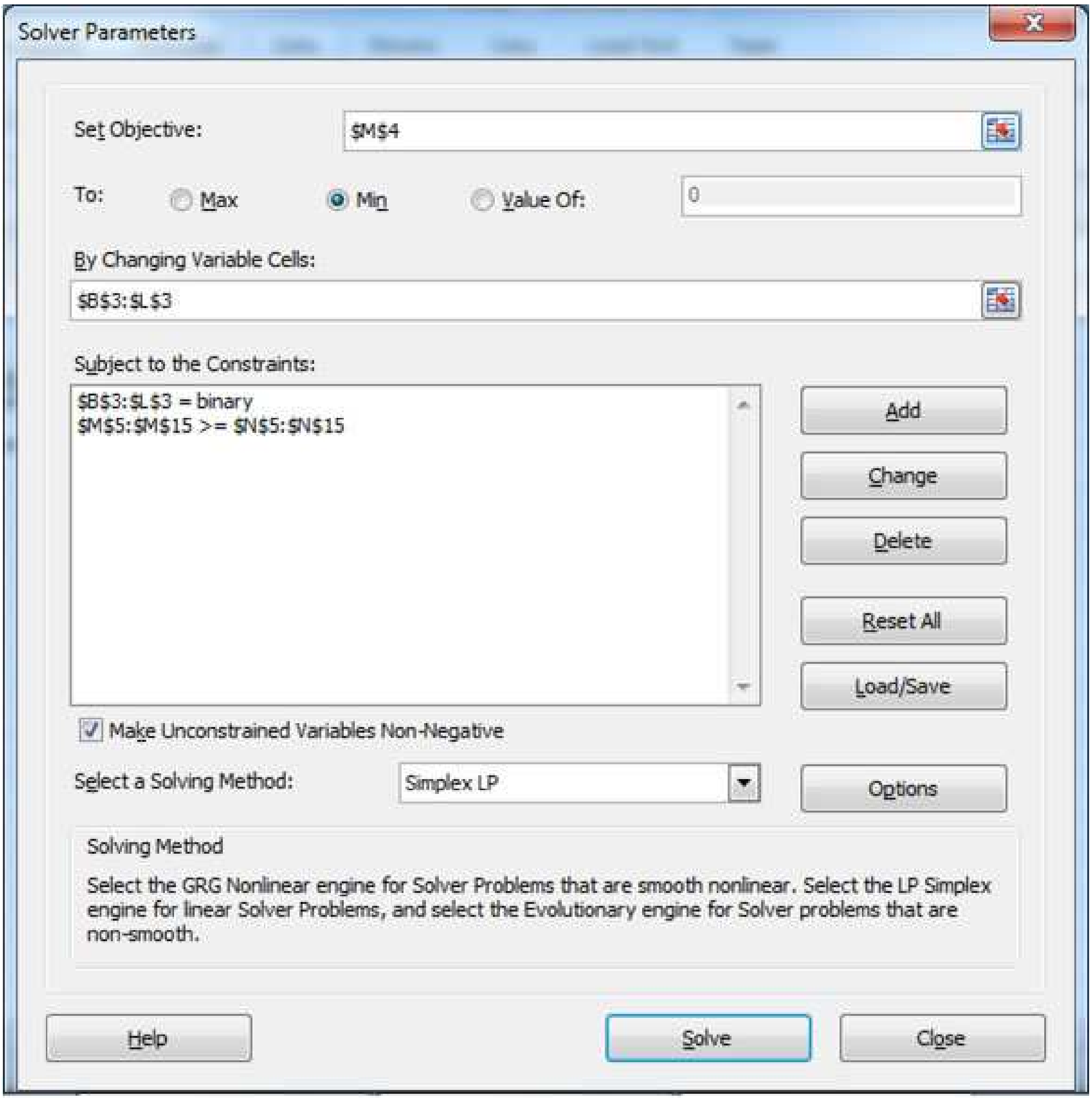


Figure 12.11

Cell M3 is not directly related to our model but is a useful piece of information – it sums the decision variables in Row 3. Thus, it could serve as an alternative objective (minimizing the number of warehouses versus minimizing the cost). You can experiment – interestingly, and worth mentioning, if you minimize the number of warehouses needed to cover the counties, your cost is considerably higher than if you minimize the cost. Although the objectives seem similar, they are decidedly different even in this simple scenario.

Rows 5 through 15 are the constraints for each county. The "1" indicates that the county represented by that column (decision variable indicator for a warehouse) covers that county – a blank implies it does not. Column M is again our action column – it is a sumproduct of Row 3 multiplied by the corresponding row of interest. The formula one would input in Cell M5, to be copied down the column : =SUMPRODUCT(B$3:L$3, B5:L5). The coverage constraints are simply Column M >= Column N.

The solution to the model shows that the least cost way of locating warehouses is to place them in Grant, Logan and Osage County, for a cost of 24 units. All counties are covered by this warehouse arrangement, and three counties are covered by more than one warehouse location (Garfield, Noble and Kay counties).

As an aside, there is a solution to this situation in which you can have EXACTLY one warehouse cover each county. That is a rarity in this type of problem, and if I could change the county configuration of Oklahoma to make this go away so that a student wouldn't accidently get the wrong idea, I would. But the governor turned down my reconfiguration of boundary request. So I mention the anomaly here.

There is one more modeling scenario that I'd like to present that uses binary variables in a unique way – what I call "linking" variables or problems with linking aspects. In this case, by the use of the binary variables, we link different decisions together in one model that allows us to have a more global view of a situation that will lead to a more comprehensive and reasonable suggested solution.

Reading Material: 12.5 – Example 4: The Lady Who Played with Binary Variables

12.5.1 Problem Description

The last book in the trilogy (The Lady Who Sprayed the Wasp Nest) by the late Gregg Larsen is soon to be re-released in paperback. It is anticipated that this will lead to a spike in the sales of the other two books (The Lady with the Rattlesnake Tattoo and The Lady Who Swallowed Fire) in paperback, much like what happened when the book (Wasp's Nest) first appeared in hardcover.

You have identified five different locations that can print The Lady with the Rattlesnake Tattoo – these locations have identified the one-time cost to print books ("opencost" below), and their maximum capacity at that price. You also know how many books you need to get to your three warehouses (demand below W1, W2, W3).

Which locations should you use to print and distribute the books to the warehouses where they are demanded? Assume your objective is to minimize the sum of distribution costs plus the one-time costs to use a printing facility. There is no cost if you do not use a facility. Assume that the production cost is the same at each possible location, and thus we will not include it in our model (to simplify).

shipping		W1	W2	W3		Opencost	Capacity
	Omaha	1	1.3	1.4		2	10
	Des Moines	1.2	1.5	1.6		1.4	8
	Little Rock	2	1.8	1.9		2.3	25
	Wichita	1.6	1.9	1.3		2.1	14
	KC	1.35	2	1.2		2.7	7
Demand		10.5	12	6			

12.5.2 Model Preliminaries

You likely recognize this as a distribution/supply chain problem. Of course, it contains a complication. Let us initially ignore this 'complication' to help us get the model creation started.

If we simply had five locations at which we are printing books and three warehouse locations where they need to be sent, the problem can be solved with a standard transportation model. If that is the case (which it isn't!), there is plenty of spare printing capacity (10 + 8 + 25 + 14 + 7 = 64 units versus 28.5 units demanded). The decisions would be the typical supply chain matrix format – representing the number of books to be sent from print location X to warehouse Y, which results in a total of 15 different combinations.

As before, one would have row sums (capacity used) and column sums (books sent to warehouses). The capacity sums would be constrained by capacity column (<=), whereas the warehouse sums would be forced = to the anticipated demand. The target cell/objective function would be a big sumproduct of the per-unit distribution costs and the decision variables matrix, a collective 5×3 times 5×3 endeavor.

This simplification of the problem is useful – it is STILL the core model for addressing this decision situation. What must be added on to the core model are the simultaneous decisions on where to print books. If we choose to print at Omaha, for instance, we accrue a one-time cost of 2 units regardless of how many books are printed (though we know we cannot print more than 10 units). Likewise, if the model chooses NOT to use the capacity at Omaha, we do not accrue the one-time fixed costs, but we also have to make sure that our model doesn't send books from Omaha to the warehouses BECAUSE WE ARE NOT USING the facilities. So, in essence, the decisions about whether to use Omaha (a yes/no sounding decision!) are linked to the calculation of our overall cost (the fixed cost parameter) and to the optimal distribution of the books (because we don't want to send books from a facility that we are not using!). Thus, the term "linking" model.

12.5.3 Excel Implementation of Solution

The spreadsheet in Figure 12.12 illustrates how to implement these 'linked' sets of decisions.

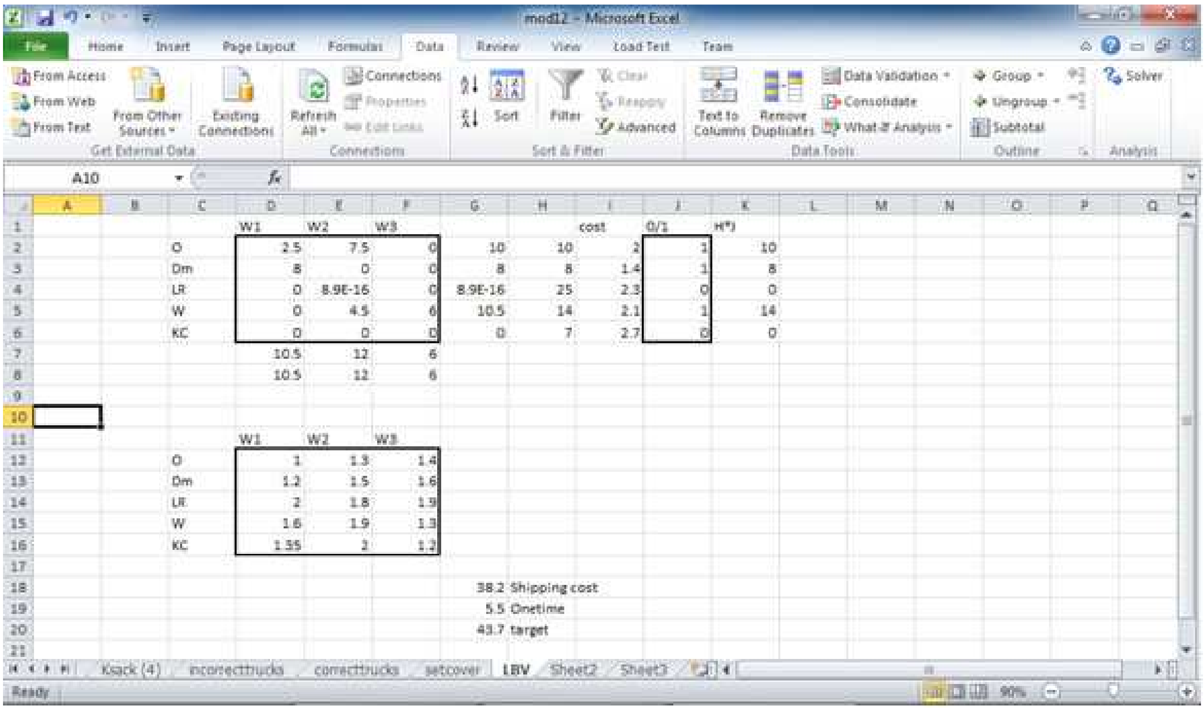


Figure 12.12

First, the familiar portion of the spreadsheet. If you ignore everything from Column I outward, and everything from Row 19 “south,” one sees a standard transportation LP model. D2:F6 are the 15 shipping decision variables, Column G row sums, Row 7 column sums, Column H production capacity, Row 8 demand for books at warehouses. The 5×3 distance/cost matrix is replicated in D12:F16, and the shipping cost sumproduct (=SUMPRODUCT(D2:F6,D12:F16) is in Cell G18. This would typically be the target cell (to minimize) in the basic transportation model.

In a transportation problem (implying all printing locations available), one would have Column G <= Column H and Row 7 = Row 8 as the supply and demand constraints, respectively.

But we know we have to add the use printing/don’t use printing here decisions. Use of a facility comes with a fixed cost, which is shown in Column I. Column J is the location for the binary variables that will indicate, yes or no, whether the printing facility will be used. Thus, J2:J6 are binary variables (as shown in Figure 12.13, the Solver parameters for this spreadsheet).

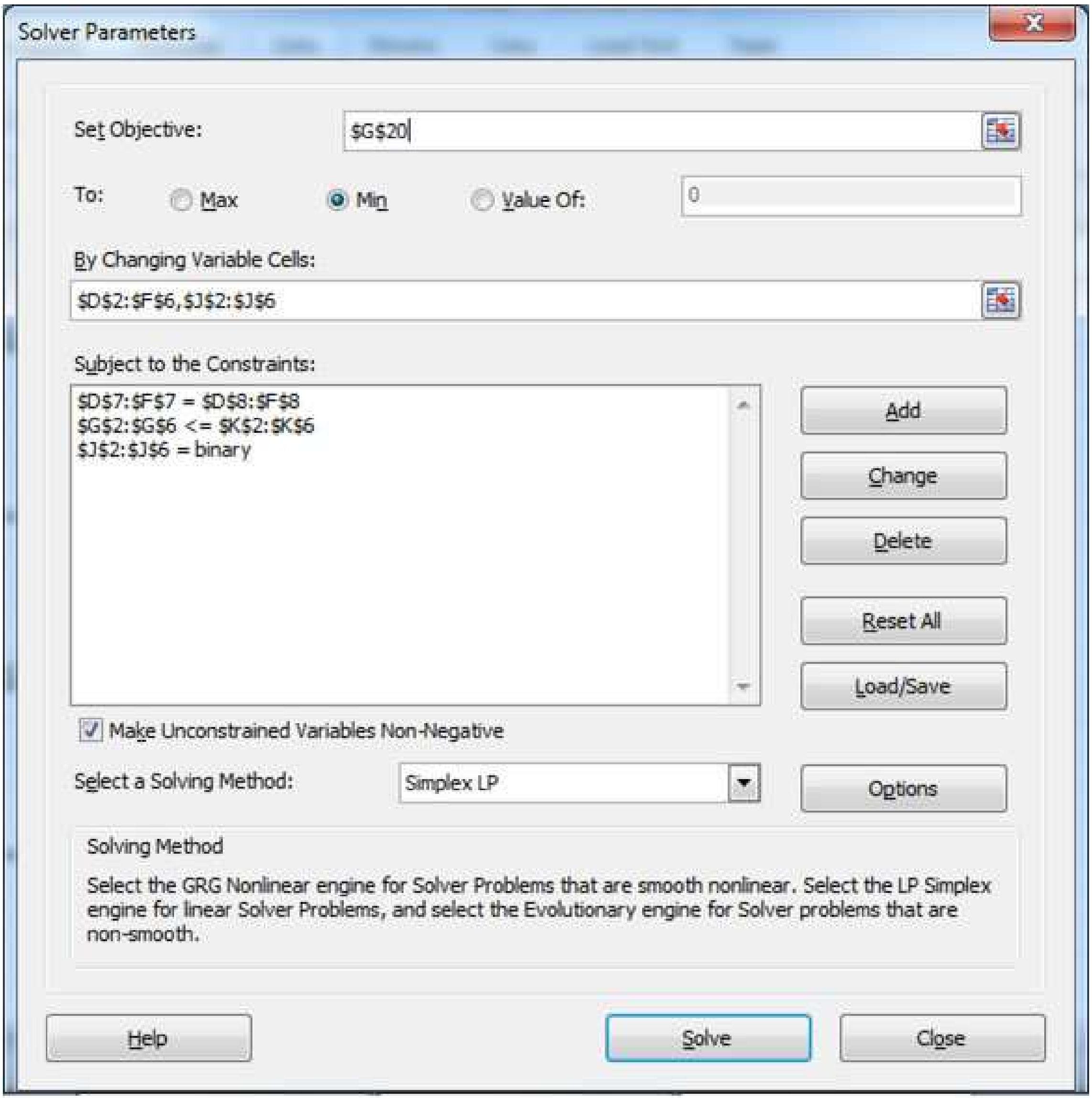


Figure 12.13

Cell G19 calculates the fixed costs of using the printing facilities. This is a simple sumproduct of Column I (fixed costs) times Column J (0/1 – is the facility open or not open!). The cell formula: =SUMPRODUCT(I2:I6,J2:J6). Finally, G20 is the sum of the distribution costs (G18) plus the previously mentioned fixed costs of using the printing facilities (G19). In the situation being modeled, G20 is our objective function/target cell to be minimized.

There is one final critical step to link the 0/1 variables with the distribution variables. The demand constraints from the familiar portion of the scenario can remain as is in the linked portion. The supply constraints, though, need to be altered.

Column H represents the maximum books a printing facility can print IF IT IS OPEN. If the model deems a facility should not be used, the actual capacity is in fact 0. In a Column K formula, Column J (the 0/1 indicator variable) is multiplied by the maximum capacity (Column H) to derive the true capacity of the five printing facilities. It is 0 if closed, but equivalent to Column H if open. Thus, the supply constraint needs to be Column G (row sums) <= Column K (0/1 variable multiplied by the maximum production).

Sometimes, seeing the algebra of this constraint helps the reader – sometimes not, but here it is. Very quickly, consider the Omaha facility. Let O1, O2, and O3 represent the distribution decision variables (books from Omaha to the three warehouses). Let ZO represent the 0/1 variable indicating whether Omaha will be used. The actual supply constraint for Omaha would algebraically be:

O1 + O2 + O3 <= 10*ZO

If ZO =1 (Omaha is used), maximum supply is therefore 10*1 = 10 units of books.

If ZO = 0 (Omaha is not used), maximum supply is therefore 10*0 = 0 units of books.

Thus, the decisions to use/not use the facilities are linked to the supply constraints of the core model.

These kind of linking constraints can be used in many scenarios. This is just one type of scenario, combining supply chain design and operation.

The optimal solution of this model uses Omaha, Des Moines, and Wichita to print books (though there is left over capacity at Wichita). Omaha sends 2.5 units to warehouse 1 and 7.5 units to warehouse 2. Des Moines sends all 8 units made to warehouse 1, while Wichita supplies 4.5 units to warehouse 2 and 6 units to warehouse 3. The total cost of this plan is assessed at 43.7 units (cell G20).

Reading Material: 12.6 – Conclusion

There are so many unique ways that 0/1 variables can be used. It is impossible to cover them all (pun intended). These four examples have historically helped us gain a grasp of the potential use. As always, the practice problems that follow help us gain further modeling power and insight.