Module 15 – Unique Objectives for Math Programming Models

Module 15 – Unique Objectives for Math Programming Models

In the preceding chapter, Goal Programming was shown as one way of incorporating multiple objectives into our 'algebra on steroid' models (making it, of course, LP on steroids). We also mentioned it was not the only wayof incorporating multiple, conflicting objectives that we tend to face in reality.

This module examines two unique objective specifications that can be incorporated in either single or multiple objective modeling scenarios. These two specifications – a maximin and minimax formulation of an objective –represents a small sampling of the many possible ways one can craft models to meet specific situations. As before, our goal is not comprehensive coverage, but to provide the student with a little taste of the novel ways and possibilities of modeling something different than the tradition MAX this or MIN that we've seen to date.

15.2 Maximin and Minimax defined

Maximin is a criteria where decisions are made such that we find the best (maxi) worst case solution of the objective of interest. For instance, if we want to assign workers to project teams, and we can relate the 'utility' of each unique worker to team assignment, we might want to assign workers such that we "maximize the minimum' utility of assignments from a worker perspective. This would be different then determining an optimal assignment based upon maximizing the overall sum of the utilities stemming from the assignments. (It could result in the same solution, but likely not). We might find that to get the overall maximum value in the latter case, we might have to have one or more workers earn a very low utility, while others have exceptionally high utility. In some ways then, a Maximin approach may result in a more 'balanced' solution from an individual worker standpoint.

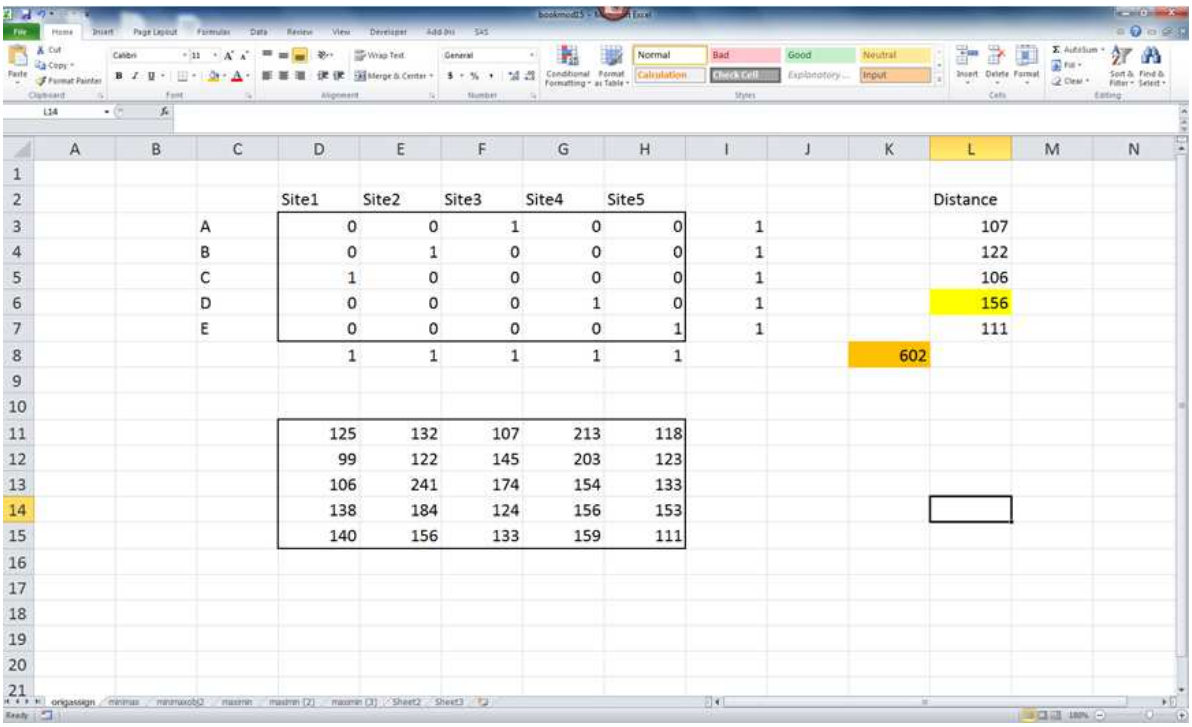
The other type of criteria shown in Module 15 is a minimax. Let us use the same core problem type - assignment problem - to illustrate. Suppose we are interested in finding a way to assign referee teams to football games, with distance traveled by the referrees a criteria of interest. A typical assignment would be to minimize the sum of the overall distances travelled by the referee teams aggregately. However, we might be interested in a solution where we minimized the maximum distance that any one referee team traveled. This kind of situation would be a good use of the minimax objective.

The next section explores how to implement these objectives through some small, 'toy' problems.

15.3 Example 1 – Assigning Employees to Job Sites

15.3.1 – Introduction

Suppose we wish to find the optimal assignment of 5 employees (A through E) to 5 job sites. Figure 15.1 shows the distances between all 25 combinations of employees/sites.



		Site1	Site2	Site3	Site4	Site5		Distance
A		0	0	1	0	0	1	107
B		0	1	0	0	0	1	122
C		1	0	0	0	0	1	106
D		0	0	0	1	0	1	156
E		0	0	0	0	1	1	111
		1	1	1	1	1		602
		125	132	107	213	118		
		99	122	145	203	123		
		106	241	174	154	133		
		138	184	124	156	153		
		140	156	133	159	111		

Figure 15.1 – Original Assignment Model – Min Distance

If we created a model to simply minimize the sum of the distance for all assigned matches, it would be a traditional assignment problem.

Decision variables: Implied 0/1, D3:H7

Objective function: Cell K8, a sumproduct of D3:H7 and D11:H15 (to be minimized).

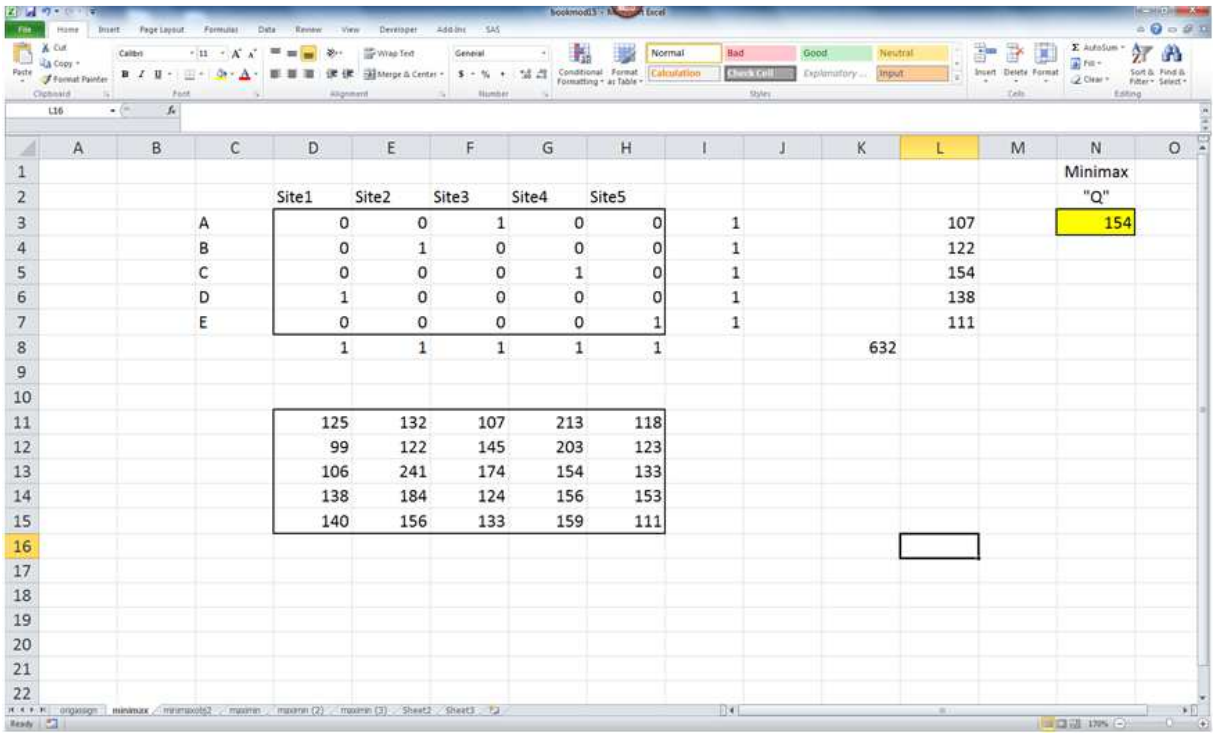
Constraints: Row 8, column sums, set equal to 1.

Column I, row sums, set equal to 1.

The solution to this classic assignment model is shown in Figure 15.1, which shows the minimized total distance traveled at 620 units. Note in this solution, Employee D has the 'worst' assignment, that of 156 units.

15.3.2 Modified model – the Minimax criteria

So, let's change our criteria to finding the assignment that minimizes the maximum distance any one employee must travel. We will see this implementation in Figure 15.2.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1														Minimax	
2				Site1	Site2	Site3	Site4	Site5						"Q"	
3			A	0	0	1	0	0	1			107		154	
4			B	0	1	0	0	0	1			122			
5			C	0	0	0	1	0	1			154			
6			D	1	0	0	0	0	1			138			
7			E	0	0	0	0	1	1			111			
8				1	1	1	1	1			632				
9															
10															
11				125	132	107	213	118							
12				99	122	145	203	123							
13				106	241	174	154	133							
14				138	184	124	156	153							
15				140	156	133	159	111							
16															
17															
18															
19															
20															
21															
22															

Figure 15.2 – Assignment Model with Minimax Objective on Individuals A-E.

To create this specific model, we must make three adjustments to the model described above (the 'classic' assignment model). First, we must create a placeholder for a new decision variable, let's call it Q (Why? Maybe to honor James Bond … the gadget division perhaps?). It is arbitrarily put in cell N3.

Interestingly enough, variable Q is also going to become the new objective function/target cell (it really is quite the gadget!). That's the second change to the model.

Third, we need to add to the core constraints the following: each employee's distance traveled (calculated by a row based sumproduct in column L in Figure 15.2) must be involved in a constraint with Q such that: Q >= Individual Employee distance. This 'links' Q with the criteria

of interest and forces Q to be at least as big as the largest value. Therefore, minimizing Q minimizes this largest value of interest!! Amazing how these linking things work.

To input that into solver, we have to code it this way: L3:L7 <= N3 - it may look backwards, but really, it is saying "Q must be greater than or equal to all the individual distances for the employees".

You can see the resulting solution in Figure 15.2. Q is found to be 154, which is the biggest individual distance that any one employee travels (Employee C in this case). It is a small reduction from what occurred in 15.3.1, and we can see that the overall distance of this solution has increased to 632 units.

15.3.3 Adding a multi-objective twist to the modified model

Let's say that we have two goals we'd like to try to achieve in our assignment, with the first goal being much more important than the second goal.

The first goal is the minimax distance goal we just solved for in 15.3.2. The second goal – to minimize the total distance traveled by the 5 employees.

So, Figure 15.2 really shows the results of attacking goal 1. What is important is how we then modify the output of the first solution step and attack goal 2.

Our strategy – we don't want to undo the achievement of our first goal at the expense of our second goal (a la Goal Programming in Module 14), so we take the results of our first model run (Objective function value =154) and add that to the model as a constraint. Then, we change the objective to reflect the second goal (Cell K8, the overall sum of distances traveled) and resolve.

This process will insure our first goal is maintained, but now the solver looks to see if we can find a solution that better satisfies the second goal – in essence, it is searching our region of multiple optimal solutions for goal 1 to see which extreme point(s) best satisfy goal 2 in the region where goal 1 is optimized.

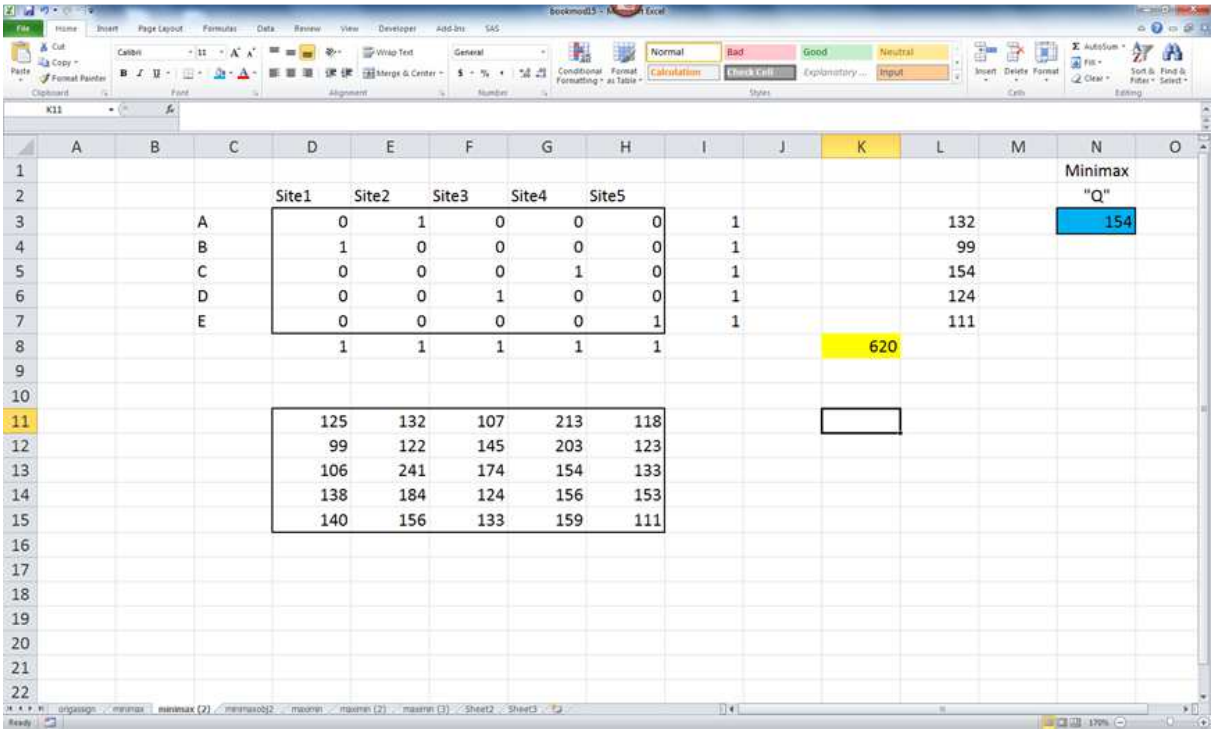
Figure 15.3 shows the solution after both Goal 1 and Goal 2 have been executed. When we set up the model to address goal 2, we changed the following:

Added a constraint: N3 = 154.

Changed objective function value pointer to K8.

Then reran the solver.

One can of course implement a multiple objective approach like this for many goals, turning the results of a more important goal into a constraint, and then solving the model from the next goal's perspective.



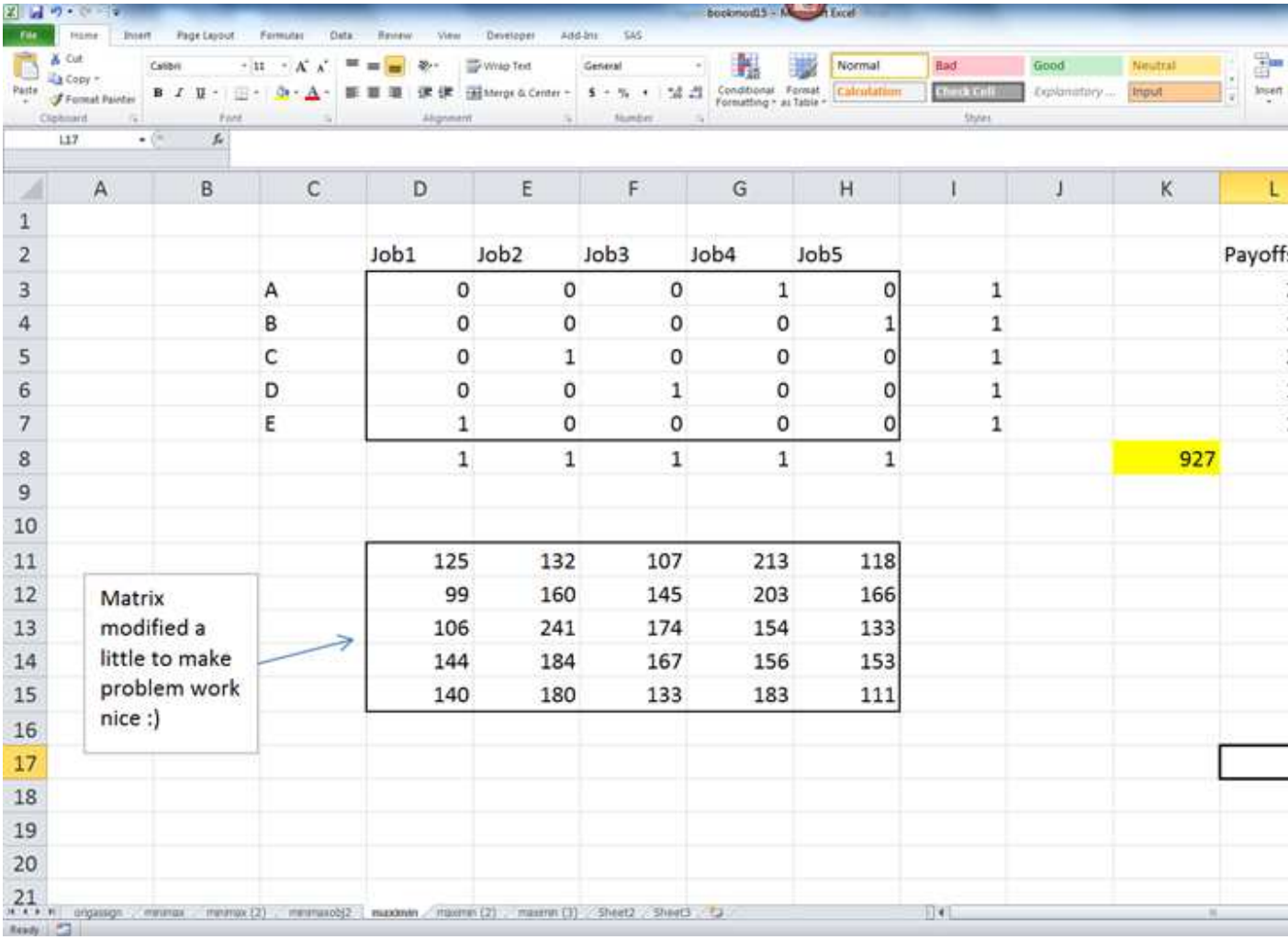
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1														Minimax	
2				Site1	Site2	Site3	Site4	Site5						"Q"	
3			A	0	1	0	0	0	1			132		154	
4			B	1	0	0	0	0	1			99			
5			C	0	0	0	1	0	1			154			
6			D	0	0	1	0	0	1			124			
7			E	0	0	0	0	1	1			111			
8				1	1	1	1	1			620				
9															
10															
11				125	132	107	213	118							
12				99	122	145	203	123							
13				106	241	174	154	133							
14				138	184	124	156	153							
15				140	156	133	159	111							
16															
17															
18															
19															
20															
21															
22															

Figure 15.3 – Multiple Objective – 1st Minimax, 2nd – Min overall distance

15.4 Example 2 – Maximize assignment of employees to jobs based upon ability ("quality")

15.4.1 – Introduction

Suppose we wish to find the optimal assignment of 5 employees (A through E) to 5 jobs. Figure 15.4 shows the ability levels of all 25 combinations of employees/sites. (Note that while Figure 15.4 is very similar to Figure 15.1, there are some subtle differences used to get the learning objectives satisfied in the solution!). We will use the term 'quality' to represent the assignments, implying that we are matching abilities of employees to jobs and trying to maximize the quality of such assignments.



	A	B	C	D	E	F	G	H	I	J	K	L
1												
2				Job1	Job2	Job3	Job4	Job5				Payoff
3			A	0	0	0	1	0	1			
4			B	0	0	0	0	1	1			
5			C	0	1	0	0	0	1			
6			D	0	0	1	0	0	1			
7			E	1	0	0	0	0	1			
8				1	1	1	1	1			927	
9												
10												
11				125	132	107	213	118				
12				99	160	145	203	166				
13				106	241	174	154	133				
14				144	184	167	156	153				
15				140	180	133	183	111				
16												
17												
18												
19												
20												
21												

Matrix modified a little to make problem work nice :)

Figure 15.4 - Original Max Assignment Problem

If we created a model to simply maximize the sum of the 'quality' for all assigned matches, it would be a traditional MAX assignment problem.

Decision variables: Implied 0/1, D3:H7

Objective function: Cell K8, a sumproduct of D3:H7 and D11:H15 (to be maximized).

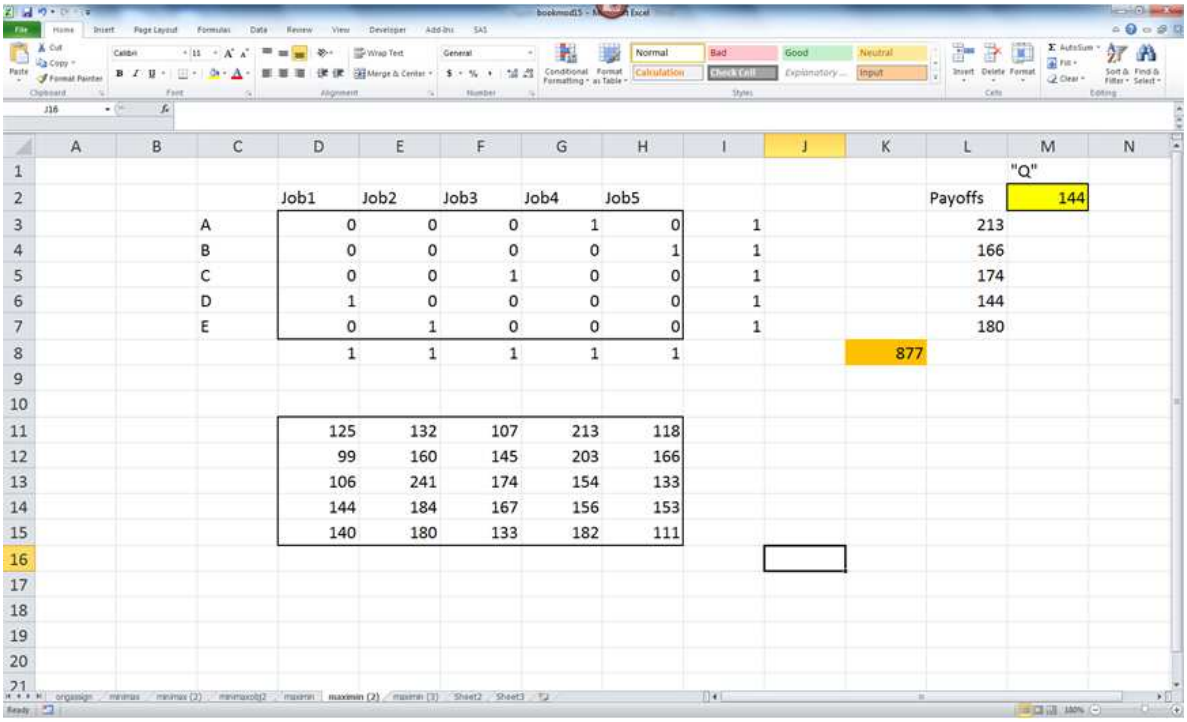
Constraints: Row 8, column sums, set equal to 1.

Column I, row sums, set equal to 1.

The solution to this classic MAX assignment model is shown in Figure 15.4, which shows the maximized total quality points as 927 units. Note in this solution, Employee E has the 'worst' assignment, that of 140 units.

15.4.2 Modified model – the Maximin criteria

So, let us change our criteria to finding the assignment that maximizes the minimum quality any one employee experiences in the assignments to specific jobs. We will see this implementation in Figure 15.5.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1													"Q"	
2				Job1	Job2	Job3	Job4	Job5				Payoffs	144	
3			A	0	0	0	1	0	1			213		
4			B	0	0	0	0	1	1			166		
5			C	0	0	1	0	0	1			174		
6			D	1	0	0	0	0	1			144		
7			E	0	1	0	0	0	1			180		
8				1	1	1	1	1			877			
9														
10														
11				125	132	107	213	118						
12				99	160	145	203	166						
13				106	241	174	154	133						
14				144	184	167	156	153						
15				140	180	133	182	111						
16														
17														
18														
19														
20														
21														

Figure 15.5 – Maximin – Maximize minimum payoff for Employees A-E

As before, to create this specific model, we must make three adjustments to the original model described above. First, we must create a placeholder for a new decision variable, again calling it Q. It is arbitrarily put in cell N3. Second,a s before, it becomes the new objective function too.

Third, we need to add to the core constraints the following: each employee's quality (calculated by a row based sumproduct in column L in Figure 15.5) must be involved in a constraint with Q such that: Q <= Individual Employee quality. This 'links' Q with the criteria of interest and forces Q to be no bigger than the smallest value. Therefore, maximizing Q maximizes this smallest value of interest!!

To input that into solver, we have to code it this way: L3:L7 >= N3 - it may look backwards, but really, it is saying "Q must be no greater than o all the individual qualities for the employee assignments".

You can see the resulting solution in Figure 15.5. Q is found to be 144, which is the smallest quality that an individual experiences in their assignment to jobs (Employee D in this case). It is

a small increase from what occurred in 15.4.1, and we can see that the overall quality of this solution has decreased to 877 units. Thus, the worst case situation is improved, but at the expense of overall measurement of aggregate quality. This doesn't mean one solution is better than the other – they each have different objectives they wish to accomplish.

The examples will stop here, but we could look at other variations for a long, long time.

15.5 Summary

The previous sections have illustrated two different 'types' of objectives as contrasted to the standard MAX profit and Min cost type situations we have seen. They are good compliments to the 'traditional' objectives and, as we saw, can be used in a single objective or multiple objective scenario. Covering goal programming in Module 14, and the way to implement Minimax and Maximin here in Module 15 gives the prescriptive analytic student an idea of how they can do great things in their model specification tool belt. These sections just scratch the surface though on innovative usage of math programming models. The sky is the limit for she that understands prescriptive analytics.

A few practice problems follow, and our final Module is a departure from the modeling topics we've seen up to now. It will be a good stochastic way to cap our course.

Note: Practice problems to be developed.