**Revisiting a few things …**

From Module 8:

*Example*, suppose we are scheduling jobs on a single machine.  
Let ti equal machine time required for job i.  
Let xi equal start time for job i.  
Thus, completion time of job i is equal to ti + xi.

Consider two jobs, j and k, that both need to use the machine.

**Solution:**Completion time is less-than or equal to start time of next job.  
xj + tj ≤ xk OR xk + tk ≤ xj   
xj + tj ≤ xk + My When y=1 ~ this constraint is relaxed.  
xk + tk ≤ xj + M(1-y) When y=0 ~ this constraint is relaxed.  
y is binary.  
Choose "M" large enough to properly relax the constraint, but not too large to cause solution problems.

>>>

***Now:***

Three different items are to be routed through three machines. Each item must be processed first on machine 1, then on machine 2, and finally on machine 3. The sequence of items may differ for each machine. Assume that the times *tij* required to perform the work on item *i* by machine *j* are known and are integers. Our objective is to minimize the total time necessary to process all the items. Formulate the problem as an integer programming problem. Your model must prevent two items from occupying the same machine at the same time; also, an item may not start processing on machine *(j+1)* unless it has completed processing on machine *j*.

>>>

Indexed set:

i = job/item (i = 1, 2, 3), j =machine (j = 1, 2, 3)

Data: t\_ij = time required to perform work on item i by machine j

M=large number=set it equal to the sum of t\_ij’s

Variables:

X\_ij = the starting time of processing i on j

(how many binary variables do we need?) 9 binary variables

Yii’j = binary variable for either or constraints regarding item i and i’ on machine j where i < i’

Y\_121, Y\_131, Y\_231

Y\_122, Y\_132, Y\_232

Y\_123, Y\_133, Y\_233

Objective: Minimize the total time necessary to process all of the items; which is equal to the maximum of the start time plus the time required on the last machine for all products. Thus, this is a min-max problem.

Min{max over i [x13+t13,x23+t23,x33+t33]}

Constraints:

We need to ensure that the products (items or jobs) do not interfere with each other (ie. We are not processing more than one job at a time on the same machine) either x11>= x21+t21 OR x21 >= x11+t11. Thus, continuing for all product and machine combinations.

Machine 1

M\*Y121+x11 >= x21+t21

M\*(1-Y121) + x21 >= x11+t11

M\*Y131+x11 >= x31+t31

M\*(1-Y131) + x31 >= x11+t11

M\*Y231+x21 >= x31+t31

M\*(1-Y231) + x31 >= x21+t21

Machine 2

M\*Y122+x12 >= x22+t22

M\*(1-Y122) + x22 >= x12+t12

M\*Y132+x12 >= x32+t32

M\*(1-Y132) + x32 >= x12+t12

M\*Y232+x22 >= x32+t32

M\*(1-Y232) + x32 >= x22+t22

Machine 3

M\*Y123+x13 >= x23+t23

M\*(1-Y123) + x23 >= x13+t13

M\*Y133+x13 >= x33+t33

M\*(1-Y133) + x33 >= x13+t13

M\*Y233+x23 >= x33+t33

M\*(1-Y233) + x33 >= x23+t23

X\_ij >= 0 for all I,j

Y\_ii’j are binary

W >= x13+t13; W >=x23+t23; W >= x33+t33 (completion time on job i)

When minimize W, that is the minimized total completion time (all jobs on all machines)

Assignment Problem:

(From Module 13)

What if we had this assignment?

7 6 2 8 lambda =2

7 9 4 8 lambda =4

3 3 1 2 lambda =1

9 8 3 7 labmda =3

Step 0: always make sure you have same number of rows than col

5 4 0 6

3 5 0 4

2 2 0 1

6 5 0 4

Mu 2 2 0 1

3 2 0 5

1 3 0 3

0 0 0 0

4 3 0 3

\* smallest number uncovered is 1

2 1 0 4

0 2 0 2

0 0 0 0

3 2 0 2

\* smallest number is 1 still.

2 0 0 3

0 1 0 1

0 0 0 0

2 1 0 1

7+6+3+2 = 18 minimized

What about Max problem?

Negate everything then add a constant to everything to make every cell >=0.

Want to use Hungarian Algorithm – but it only works for minimization problems …

-9 -6 -4 -2

-2 -8 -0 -9

-9 -4 -2 -7

-8 -3 -1 -8

0 3 5 7 lambda = 0

7 1 9 0 lambda = 0

0 5 7 2 lambda = 0

1 6 8 1 lambda = 1

0 3 5 7 lambda = 0

7 1 9 0 lambda = 0

0 5 7 2 lambda = 0

0 5 7 0 lambda = 1

Mu= 0, 1, 5, 0

0 2 0 7

7 0 4 0

0 4 2 2

1 4 2 0

4+8+9+8=29 max satisfaction