

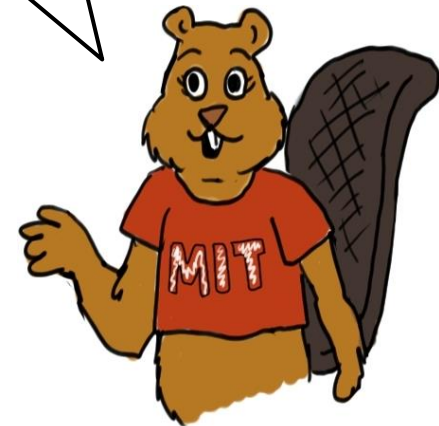
Transformations in Integer Programming

Hi, Mita and I are here to introduce a tutorial on integer programming modeling.



Amit

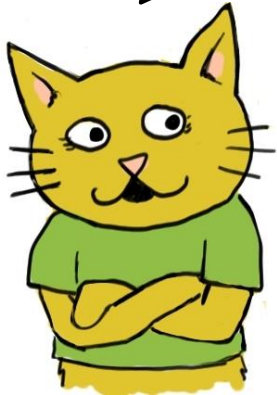
You can think of it as transformations. Our friends from 15.053 will explain how to take constraints that are easily understood and transform them into integer programs.



Mita

This tutorial will include a mixture of techniques as well as lists of transformations.

A more comprehensive document is also available. It is entitled “IP Reference guide for integer programming formulations.” It has the following sections.

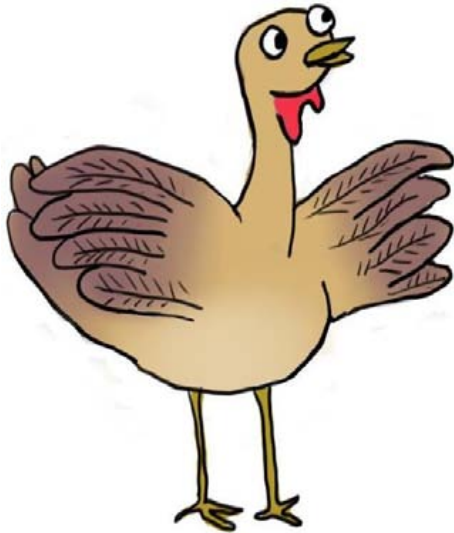


IP Reference Guide

- 1. Subset selection problems.**
- 2. Modular arithmetic**
- 3. Simple logical constraints**
- 4. Other logical constraints and the big M method.**
- 5. Fixed costs**
- 6. Piecewise linear functions and**
- 7. The traveling salesman problem.**

But first, an important question.

Wow! That's a lot of problems. Do you really expect students to learn all of it?



Tom

We don't expect students to memorize the techniques. We'll focus on some problems. For others, we will just refer to the guide. We'll also make subsets of the guide available for quizzes and the second midterm.



Transforming Logical Conditions

$$\text{Max } 16x_1 + 22x_2 + 12x_3 + 8x_4 + 11x_5 + 19x_6$$

$$5x_1 + 7x_2 + 4x_3 + 3x_4 + 4x_5 + 6x_6 \leq 14$$

$$x_j \in \{0,1\} \text{ for each } j = 1 \text{ to } 6$$

Here is the integer program used in the first integer programming lecture. As you recall, it's based on a game show that I was on.

I'm going to pretend to add conditions on what I will choose. We will then model these logical constraints using integer programming.



Nooz, the most trusted name in fox.

$$\text{Max } 16x_1 + 22x_2 + 12x_3 + 8x_4 + 11x_5 + 19x_6$$

$$5x_1 + 7x_2 + 4x_3 + 3x_4 + 4x_5 + 6x_6 \leq 14$$

$$x_j \in \{0, 1\} \text{ for each } j = 1 \text{ to } 6$$

Suppose that I refuse to select item 5 if I have also selected item 1.

In this case, the feasible solutions for Nooz will permit $x_1 = 1$ or $x_5 = 1$, but not both.

This can be modeled via the linear constraint,

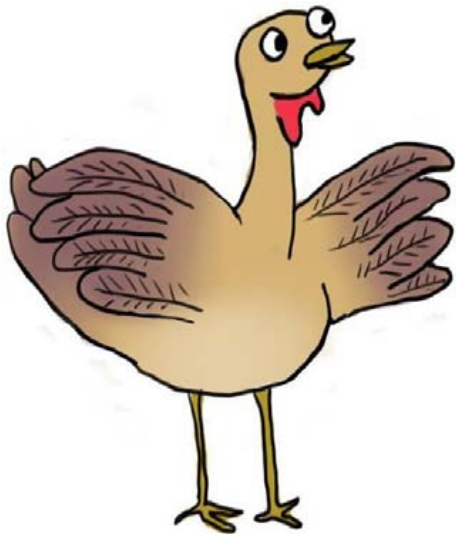
$$x_1 + x_5 \leq 1$$



Ella

I don't get it. The linear constraint doesn't look anything like the logical constraint.

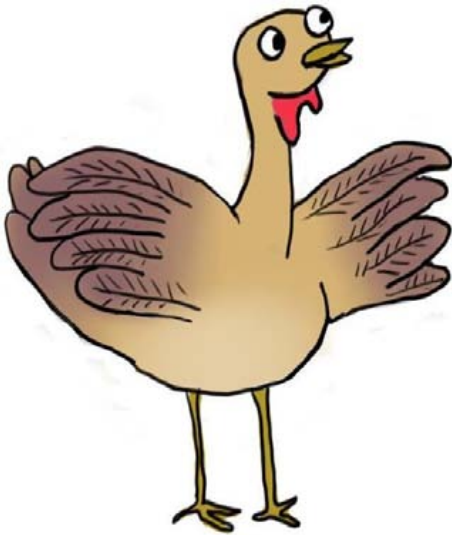
Well, Tom. The important thing isn't whether it "makes sense" as a logical constraint. The important thing is that an optimal solution for the integer program will produce an optimal solution for the original problem.



But how will we know that?

In this case, we need only focus on variables x_1 and x_5 to see that the constraint works. We don't need to think about the other variables.

You see, Nooz just wants to eliminate the possibility that items 1 and 5 are both selected. In other words, we cannot have $x_1 = 1$ and $x_5 = 1$. The linear constraint accomplishes the same thing, taking into account that x_1 and x_5 are both binary.



Another logical constraint

$$\text{Max } 16x_1 + 22x_2 + 12x_3 + 8x_4 + 11x_5 + 19x_6$$

$$5x_1 + 7x_2 + 4x_3 + 3x_4 + 4x_5 + 6x_6 \leq 14$$

$$x_j \in \{0, 1\} \text{ for each } j = 1 \text{ to } 6$$

Suppose in this illustration that I will take item 3 if and only if I take item 4. How do we model that?



In this case, the feasible solutions for Nooz will permit $x_3 = 1$ and $x_4 = 1$, or else $x_3 = 0$ and $x_4 = 0$. We can model it as the linear constraint $x_3 = x_4$.



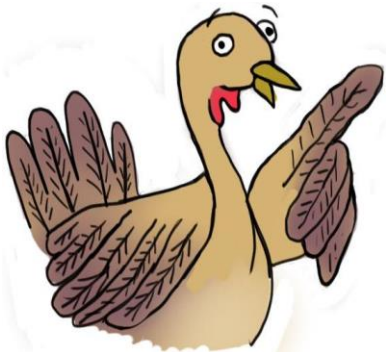
Can you explain it to me. I liked your last explanation.

Well, Tom. Before Nooz added the requirement, there were four possibilities for variables x_3 and x_4 : $x_3 = 0, x_4 = 0$; or $x_3 = 0, x_4 = 1$; or $x_3 = 1, x_4 = 0$; or $x_3 = 1, x_4 = 1$. But after Nooz added his new restriction, there were only two possibilities: $x_3 = 0, x_4 = 0$; or $x_3 = 1, x_4 = 1$;

When we added the linear constraint

$$"x_3 = x_4"$$

we left the same two possible solutions for x_3 and x_4 . So, our integer linear constraints modeled Nooz's new problem.



One more logical constraint

$$\text{Max } 16x_1 + 22x_2 + 12x_3 + 8x_4 + 11x_5 + 19x_6$$

$$5x_1 + 7x_2 + 4x_3 + 3x_4 + 4x_5 + 6x_6 \leq 14$$

$$x_j \in \{0,1\} \text{ for each } j = 1 \text{ to } 6$$

OK. This will be my last practice exercise for my game show problem. Suppose that I add the constraint that I choose exactly 3 items.

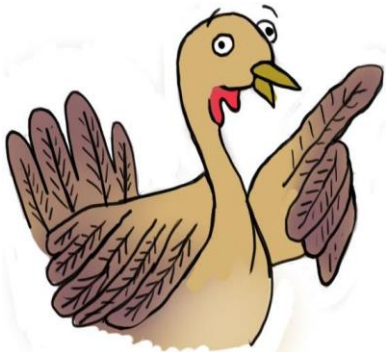


In this case, Nooz' s constraint sounds like a linear constraint. And it can be modeled with the constraint.

$$x_1 + x_2 + x_3 + x_4 + x_5 + x_6 = 3.$$



Ella, this is great. I think I now know how to model logical constraints for integer programming.



Well, Tom. I'm really glad you understand what we've done so far. But for the first examples, we only modeled constraints involving two binary variables. It turns out that other types of logical constraints require other types of modeling techniques. Nooz will show you another couple of examples.



Transforming “Non-Exclusive OR” Constraints

Either $2x_1 + x_2 \geq 5$ or $2x_3 - x_4 \leq 2$ or both

Suppose that we have already modeled a problem as a linear program, and we now want to add the logical constraints

$2x_1 + x_2 \geq 5$ or
 $2x_3 - x_4 \leq 2$ or both

This situation is more complicated, but there is a standard technique for doing it.

We are not assuming here that x_i is binary. In fact, we are not even assuming that it is required to be integer valued. But for our transformation to work, we do need to require it to be bounded. So we will assume that $x_i \leq 100$ for each i .



The Transformation

Either $2x_1 + x_2 \geq 5$ or $2x_3 - x_4 \leq 2$ or both

In order to model this “either-or” constraint, we add a variable y_1 , which is required to be binary, and we add two new constraints to our original linear program.

$$2x_1 + x_2 \geq 5 - My_1$$

$$2x_3 - x_4 \leq 2 + M(1 - y_1)$$

$$y_1 \in \{0, 1\}$$

where M is a constant that is sufficiently large. In this case, we could choose M to be 200 or anything larger.



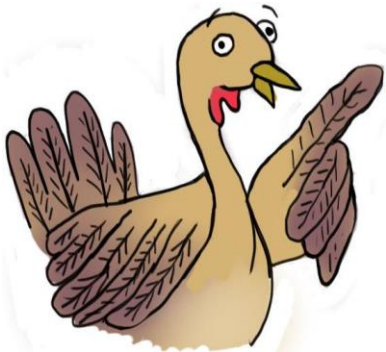
Either $2x_1 + x_2 \geq 5$ or $2x_3 - x_4 \leq 2$ or both

$$2x_1 + x_2 \geq 5 - My_1$$

$$2x_3 - x_4 \leq 2 + M(1 - y_1)$$

$$y_1 \in \{0, 1\}$$

I'm lost again.



Don't worry. I'll explain it to you. Just think about the integer program. The feasible region is the union of the feasible region with $y_1 = 0$ and the feasible region with $y_1 = 1$.

If $y_1 = 0$, the feasible region includes the constraints:

$2x_1 + x_2 \geq 5$ and $2x_3 - x_4 \leq 2 + M$. But because $x_3 \leq 100$ and we will choose $M \geq 200$. The second constraint is automatically satisfied. This leaves us with

$$y_1 = 0, 2x_1 + x_2 \geq 5$$

Either $2x_1 + x_2 \geq 5$ or $2x_3 - x_4 \leq 2$ or both

$$2x_1 + x_2 \geq 5 - My_1$$

$$2x_3 - x_4 \leq 2 + M(1 - y_1)$$

$$y_1 \in \{0, 1\}$$

Go on.

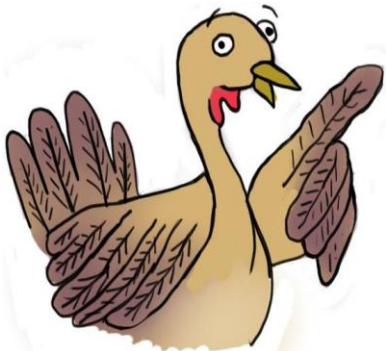
If $y_1 = 1$, the feasible region includes the constraints:

$2x_1 + x_2 \geq 5 - M$ and $2x_3 - x_4 \leq 2$. But because variables are nonnegative and $M \geq 5$, the first constraint is automatically satisfied. This leaves us with

$$y_1 = 1, 2x_3 - x_4 \leq 2.$$

Putting the two cases together, we conclude that the constraints are equivalent to

$2x_1 + x_2 \geq 5$ or $2x_3 - x_4 \leq 2$ or both.



Transforming If-Then Constraints

If $2x_1 + x_2 \leq 5$ then $2x_3 - x_4 \geq 2$.

Suppose that we have a linear program, but we want to add the constraint

“If $2x_1 + x_2 \leq 5$ then $2x_3 - x_4 \geq 2$ “. How can we do this using integer variables plus linear constraints?



This situation is very similar to the previous one, because the “if-then constraint” is equivalent to writing

$2x_1 + x_2 > 5$ or $2x_3 - x_4 \geq 2$ or both.

But there is an added complication. We don't like a strict inequality constraint. So, we will show how to carry out the transformation in the special case that x_1 and x_2 are integer valued. In this case

$2x_1 + x_2 > 5$ if and only if $2x_1 + x_2 \geq 6$.

Transforming the “if-then” constraint.

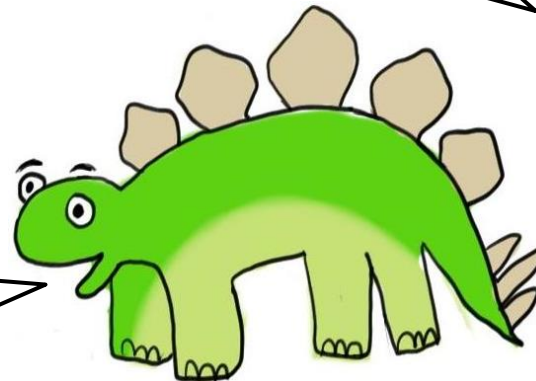
If $2x_1 + x_2 \leq 5$ then $2x_3 - x_4 \geq 2$
is equivalent in this case to
 $2x_1 + x_2 \geq 6$ or $2x_3 - x_4 \geq 2$ or both.

This is equivalent to
 $2x_1 + x_2 \geq 6 - My_1$
 $2x_3 - x_4 \geq 2 - M(1 - y_1)$
 $y_1 \in \{0, 1\}$

But this time, we leave it as an exercise to the student to fill in the details of why it works. As before, we assume that all variables are bounded by 100. In this case, we can let M be 200 or higher.

By the way, if x_1 and x_2 were not integer valued, one cannot model the “if then constraints” correctly using integer and linear constraints.

It's all intuitively obvious to the casual observer.

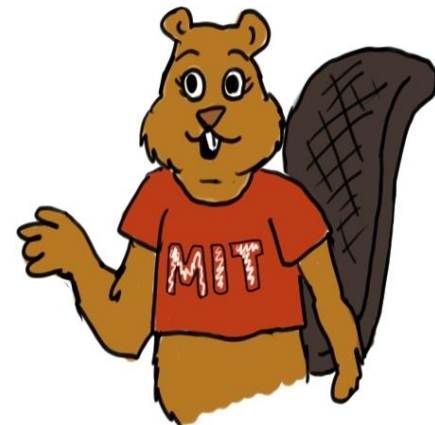
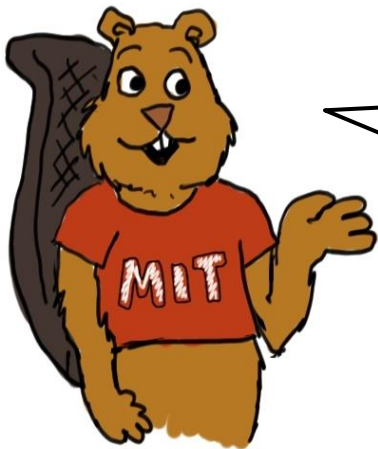


Non-linear Objectives

Another great application of integer programming is non-linear objectives.


Many times in practice, the costs are non-linear. This can be due to “fixed costs” or quantity discounts, or increasing marginal costs or decreasing marginal costs.

Our friends will present a couple of techniques for modeling non-linear objectives.



Zor's original problem

$$\begin{aligned} \text{Maximize} \quad & 52 x_1 + 30 x_2 + 20 x_3 \\ \text{subject to} \quad & 2 x_1 + 4 x_2 + 5 x_3 \leq 100 \\ & 1 x_1 + 1 x_2 + 1 x_3 \leq 30 \\ & 10 x_1 + 5 x_2 + 2 x_3 \leq 204 \\ & x_1, x_2, x_3 \geq 0 \quad \text{integer} \end{aligned}$$



I hope you remember Zor's problem. But even if you don't, you can follow what I'm going to say. We are going to consider a new (revised) problem in which the objective is $f_1(x_1) + f_2(x_2) + f_3(x_3)$, defined as follows.

$$f_1(x_1) = \begin{cases} -500 + 52x_1 & x_1 \geq 1 \\ 0 & x_1 = 0 \end{cases}$$
$$f_2(x_2) = \begin{cases} -400 + 30x_2 & x_2 \geq 1 \\ 0 & x_2 = 0 \end{cases}$$
$$f_3(x_3) = \begin{cases} -300 + 20x_3 & x_3 \geq 1 \\ 0 & x_3 = 0 \end{cases}$$

The new variables represent “fixed costs”. If $x_1 = 0$ and no gold is produced, the cost is 0. Otherwise, Zor has to pay a fixed cost of 500 Euros before doing any production and before getting any revenue.

In order to model fixed costs using integer variables and linear constraints, we create new variables. In this case, we create binary variables w_1 , w_2 , and w_3 . We will then create linear constraints (on the next slide) that ensures that w_1 , w_2 and w_3 take on the values that we want.



$$w_1 = \begin{cases} 1 & x_1 \geq 1 \\ 0 & x_1 = 0 \end{cases}$$

$$w_2 = \begin{cases} 1 & x_2 \geq 1 \\ 0 & x_2 = 0 \end{cases}$$

$$w_3 = \begin{cases} 1 & x_3 \geq 1 \\ 0 & x_3 = 0 \end{cases}$$

Modeling Fixed Charges

We need upper bounds on the variables x_1 , x_2 and x_3 in order to create the IP model. Because of the constraint $x_1 + x_2 + x_3 \leq 30$, we can conclude that $x_i \leq 30$ for $i = 1, 2$ and 3 . We can obtain better (that is, tighter) bounds if we were to analyze the other two linear constraints. But, these bounds are good enough for our purposes.

We then add the following constraints to the integer program:

$$x_1 \leq 30 w_1$$

$$x_2 \leq 30 w_2$$

$$x_3 \leq 30 w_3$$

w_1, w_2, w_3 binary



We just need to add these constraints to the previous IP. These constraints ensure that $w_i = 1$ whenever $x_i > 0$.



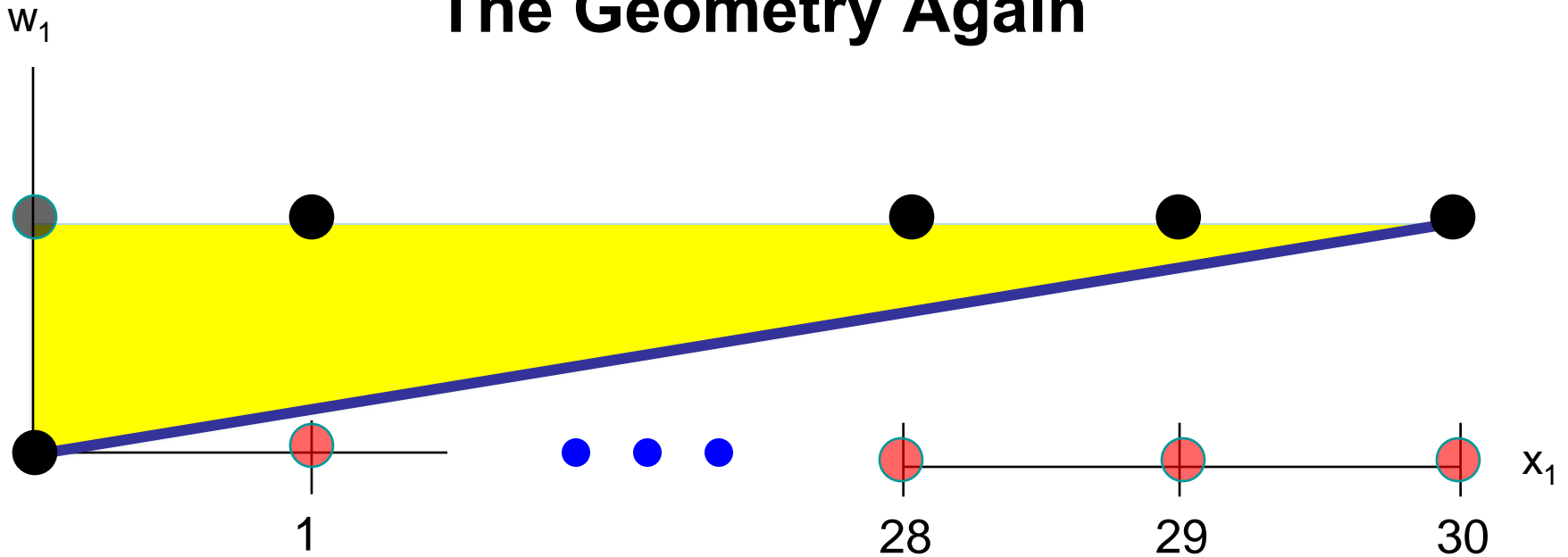
Nooz, it looks like you have an error. I agree that that $w_i = 1$ whenever $x_i > 0$. However, your formulation also permits that $w_i = 1$ even if $x_i = 0$. Your feasible region is too big.



My feasible region may be too big. But the procedure would never produce a solution with $w_i = 1$ and $x_i = 0$. It would always produce a higher profit to have $w_i = 0$ whenever $x_i = 0$.



The Geometry Again



Above is the feasible region for the constraint: If you graph the constraint: $x_1 \leq 30 w_1$. You will notice that the black points are all feasible. These are the points we wanted to be feasible when we defined w_1 , except for the point $(0,1)$. But $(0,0)$ has more profit than $(0,1)$.

The red points are all infeasible, which is also what we wanted.

Zor's problem with fixed costs

$$\begin{aligned} \text{Max} \quad & -500 w_1 + 52 x_1 - 300 w_2 \\ & + 30 x_2 - 200 w_3 + 20 x_3 \end{aligned}$$

$$\text{s.t} \quad 2 x_1 + 4 x_2 + 5 x_3 \leq 100$$

$$1 x_1 + 1 x_2 + 1 x_3 \leq 30$$

$$10 x_1 + 5 x_2 + 2 x_3 \leq 204$$

$$x_1 \leq 30 w_1$$

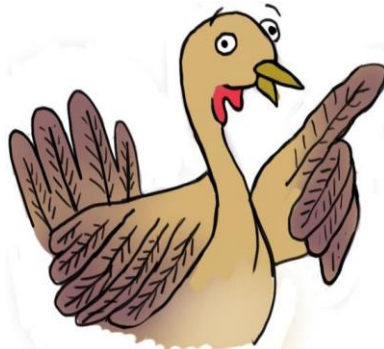
$$x_2 \leq 30 w_2$$

$$x_3 \leq 30 w_3$$

$$x_1, x_2, x_3 \geq 0 \quad \text{integer}$$

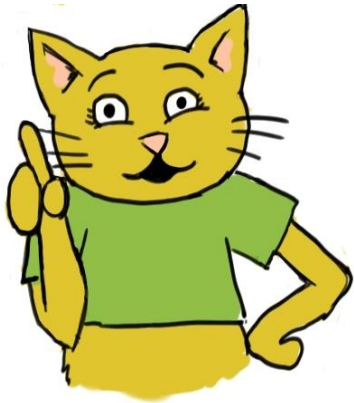
$$w_1, w_2, w_3 \quad \text{binary}$$

We can now put everything together. We have to write the objective in terms of the old and new variables. We need to include all of the original constraints. And we need to include the new constraints. Here is what we get.



It's a little tricky, but I like it.

We're not done yet.
Next we are going to
show you how to model
an even more
complicated non-linear
function. It's a piecewise
linear function.

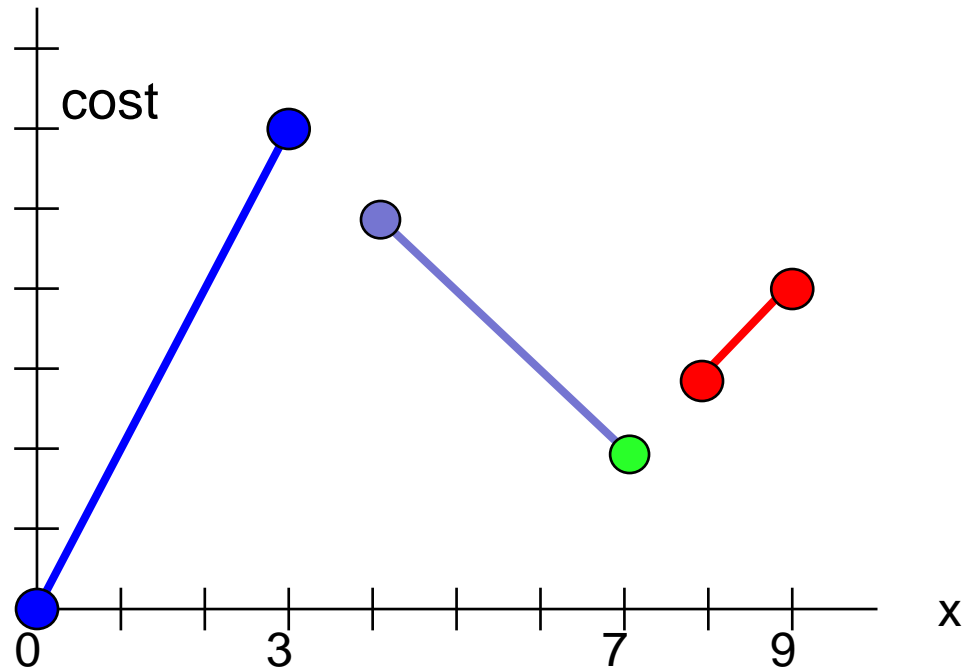


$$y = 2x \quad \text{if } 0 \leq x \leq 3$$

$$y = 9 - x \quad \text{if } 4 \leq x \leq 7$$

$$y = -5 + x \quad \text{if } 8 \leq x \leq 9$$

Assume that x is integer
valued.



$$y = 2x \quad \text{if } 0 \leq x \leq 3$$

$$y = 9 - x \quad \text{if } 4 \leq x \leq 7$$

$$y = -5 + x \quad \text{if } 8 \leq x \leq 9$$

In order to model the non-linear function $f(x)$ using integer and linear variables and using linear constraints, we will introduce 6 new variables. We want them to be defined as follows.

$$w_1 = \begin{cases} 1 & 0 \leq x \leq 3 \\ 0 & \text{otherwise} \end{cases}$$

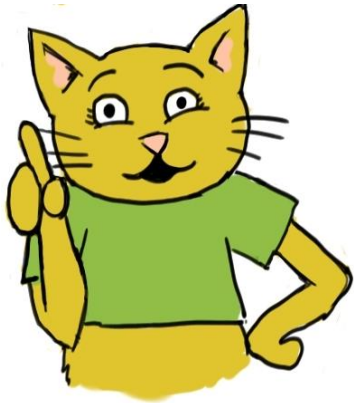
$$x_1 = \begin{cases} x & 0 \leq x \leq 3 \\ 0 & \text{otherwise} \end{cases}$$

$$w_2 = \begin{cases} 1 & 4 \leq x \leq 7 \\ 0 & \text{otherwise} \end{cases}$$

$$x_2 = \begin{cases} x & 4 \leq x \leq 7 \\ 0 & \text{otherwise} \end{cases}$$

$$w_3 = \begin{cases} 1 & 8 \leq x \leq 9 \\ 0 & \text{otherwise} \end{cases}$$

$$x_3 = \begin{cases} x & 8 \leq x \leq 9 \\ 0 & \text{otherwise} \end{cases}$$



And here are the constraints that are equivalent to the definitions of the six variables.

$$w_1 + w_2 + w_3 = 1$$

$$0 \leq x_1 \leq 3 w_1$$

$$4w_2 \leq x_2 \leq 7 w_2$$

$$8w_3 \leq x_3 \leq 9 w_3$$

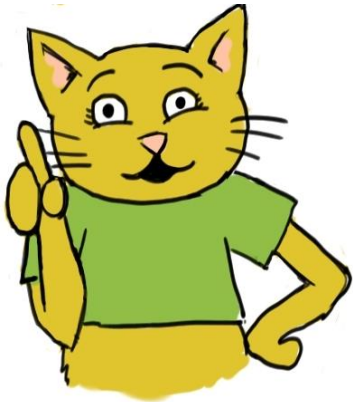
$$x = x_1 + x_2 + x_3$$

$$w_1, w_2, w_3 \text{ binary, } x_1, x_2, x_3 \text{ integer}$$

$$w_1 = \begin{cases} 1 & 0 \leq x \leq 3 \\ 0 & \text{otherwise} \end{cases} \quad x_1 = \begin{cases} x & 0 \leq x \leq 3 \\ 0 & \text{otherwise} \end{cases}$$

$$w_2 = \begin{cases} 1 & 4 \leq x \leq 7 \\ 0 & \text{otherwise} \end{cases} \quad x_2 = \begin{cases} x & 4 \leq x \leq 7 \\ 0 & \text{otherwise} \end{cases}$$

$$w_3 = \begin{cases} 1 & 8 \leq x \leq 9 \\ 0 & \text{otherwise} \end{cases} \quad x_3 = \begin{cases} x & 8 \leq x \leq 9 \\ 0 & \text{otherwise} \end{cases}$$



To complete the model, we need to take the variable y , which was a nonlinear function of x , and model it linearly using these 6 variables.

$$y = 2x_1 + (9w_2 - x_2) + (-5w_3 + x_3)$$

$$w_1 + w_2 + w_3 = 1$$

$$0 \leq x_1 \leq 3 w_1$$

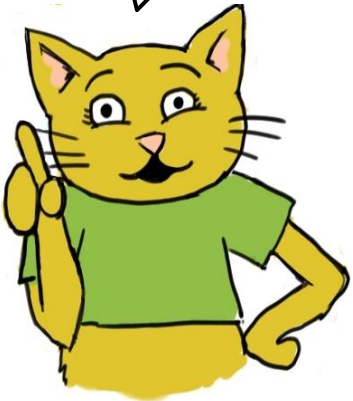
$$4w_2 \leq x_2 \leq 7 w_2$$

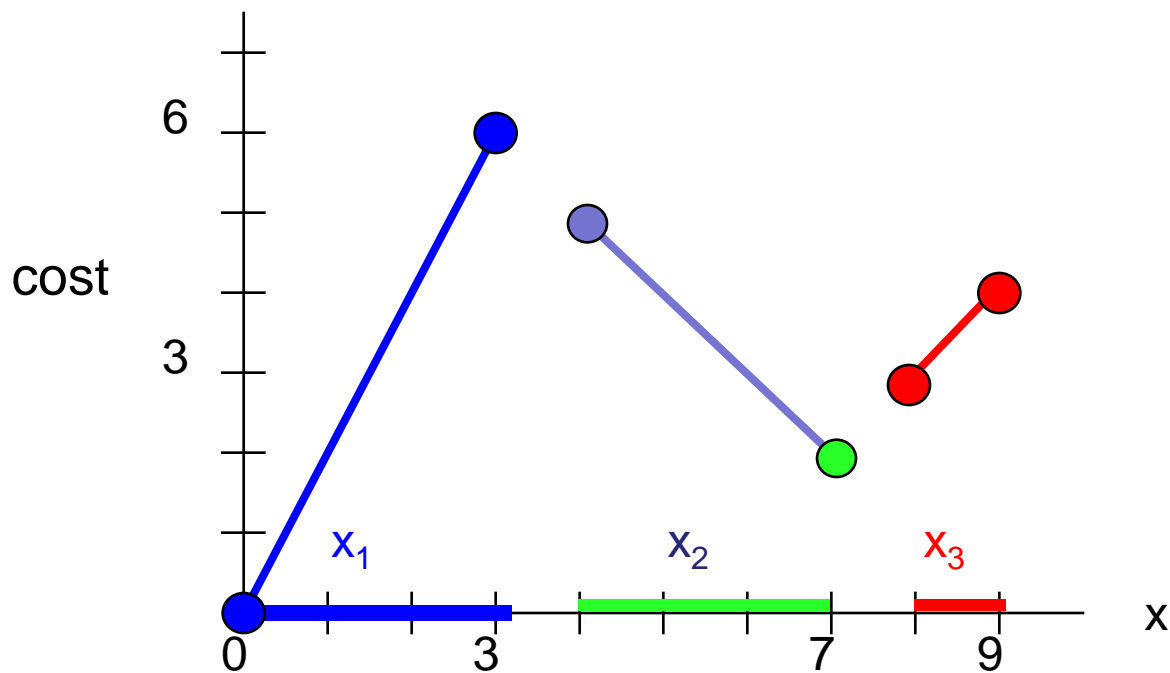
$$8w_3 \leq x_3 \leq 9 w_3$$

$$X = x_1 + x_2 + x_3$$

w_1, w_2, w_3 binary,

$x_1, x_2, x_3 \geq 0$ and integer





Case 1: $w_1 = 1$

Case 2: $w_2 = 1$

Case 3: $w_3 = 1$

$$w_1 + w_2 + w_3 = 1$$

$$0 \leq x_1 \leq 3 w_1$$

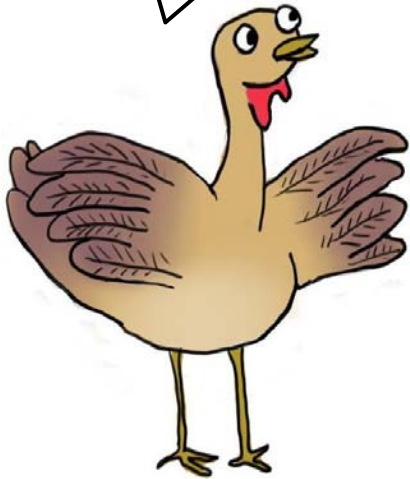
$$4w_2 \leq x_2 \leq 7 w_2$$

$$8w_3 \leq x_3 \leq 9 w_3$$

$$z = 2x_1 + (9w_2 - x_2) + (-5w_3 + x_3)$$

$$x = x_1 + x_2 + x_3$$

Cathy. Integer programming makes me nervous. I think that I understand the formulations. But there is no way that I could possibly have come up with the formulations! Will I learn how to model using integer programming?

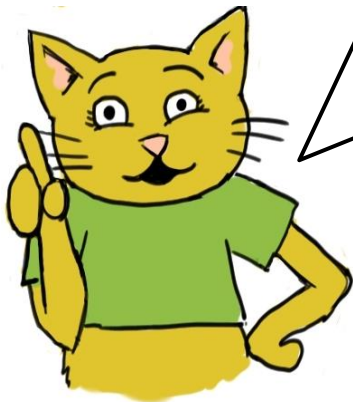


Tom, it's better than you fear. The same tricks get used over and over again. And with some practice, you get used to it. In addition, I recommend that you look at the reference guide for IP formulations that is with the course materials.



It's time for one last formulation. We will consider the problem of forming 10 committees of 10 persons each from a group of 100 people so that

1. Each person is on one committee and
2. No committee has two persons who hate each other, and
3. We ask everyone for their first five choices of committees. We assign everyone to one of their first five choices and try to give as many people their first choice as possible.



First, we need some notation for the data of the problem.

Let $A = \{(i, j) : \text{persons } i \text{ and } j \text{ hate each other}\}$

Let $B(k) = \{i : \text{person } i \text{ can serve on committee } k\}$ (This occurs if i has chosen k as one of the top five choices.)

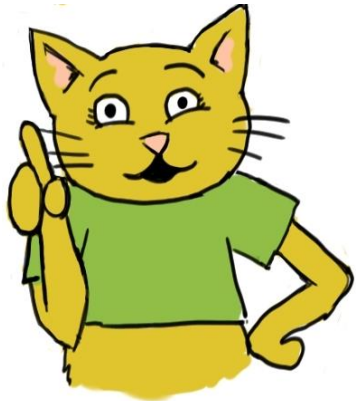
Let $d_{ik} = 1$ if committee k is person i 's first choice.
 $d_{ik} = 0$ otherwise.

Now for the decision variables. These require what we refer to as “assignment variables.”

Let $x_{ik} = 1$ if person i is assigned to committee k .

$x_{ik} = 0$ otherwise.

We now have all the notation we need to formulate the problem.



$$\text{Max} \quad \sum_{i=1}^{100} \sum_{k=1}^{10} d_{ik} x_{ik}$$

$$\text{s.t.} \quad \sum_{k=1}^{10} x_{ik} = 1 \quad \text{for } i = 1 \text{ to } 100$$

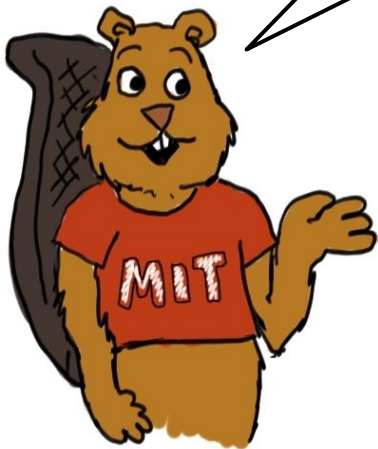
$$\sum_{i=1}^{100} x_{ik} = 10 \quad \text{for } k = 1 \text{ to } 10$$

$$x_{ik} + x_{jk} \leq 1 \quad \text{for } (i, j) \in A, \text{ and } k = 1 \text{ to } 10$$

$$x_{ik} \in \{0, 1\} \quad \text{for } i \in B(k), k = 1 \text{ to } 10$$

$$x_{ik} = 0 \quad \text{for } i \notin B(k), k = 1 \text{ to } 10$$

That's it for this tutorial.
We hope it was helpful.



Bye.



MIT OpenCourseWare
<http://ocw.mit.edu>

15.053 Optimization Methods in Management Science
Spring 2013

For information about citing these materials or our Terms of Use, visit: <http://ocw.mit.edu/terms>.