

15.053

Thursday, April 19

- **Introduction to Integer Programming**
Integer programming models

Quotes of the Day

Somebody who thinks logically is a nice contrast to the real world.

-- The Law of Thumb

“Take some more tea,” the March Hare said to Alice, very earnestly.

“I’ve had nothing yet,” Alice replied in an offended tone, “so I can’t take more.”

“You mean you can’t take less,” said the Hatter. “It’s very easy to take more than nothing.”

-- Lewis Carroll in *Alice in Wonderland*

Integer Programs

Integer programs: a linear program plus the additional constraints that some or all of the variables must be integer valued.

We also permit “ $x_j \in \{0,1\}$,” or equivalently, “ x_j is **binary**”

This is a shortcut for writing the constraints:

$$0 \leq x_j \leq 1 \text{ and } x_j \text{ integer.}$$

3

Integer programs are limited in form in that they can only include:

1. a linear objective
2. linear inequalities and equalities
3. Some or all of the variables can be required to be integer valued.
4. Since it is so easy to model that $x_j \in \{0, 1\}$, we permit this type of constraint as well.

A 2-Variable Integer program

maximize $3x + 4y$

subject to $5x + 8y \leq 24$

$x, y \geq 0$ and integer

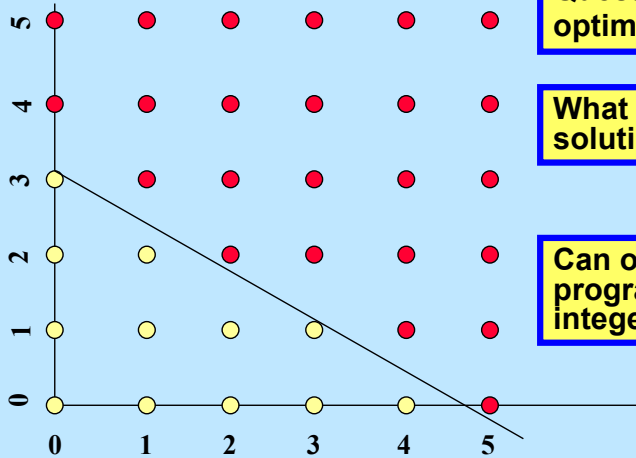
- What is the optimal solution?

4

Integer programs are notoriously hard to solve.

We illustrate this by pointing out that even a relatively simple integer program offers some challenges.

The Feasible Region



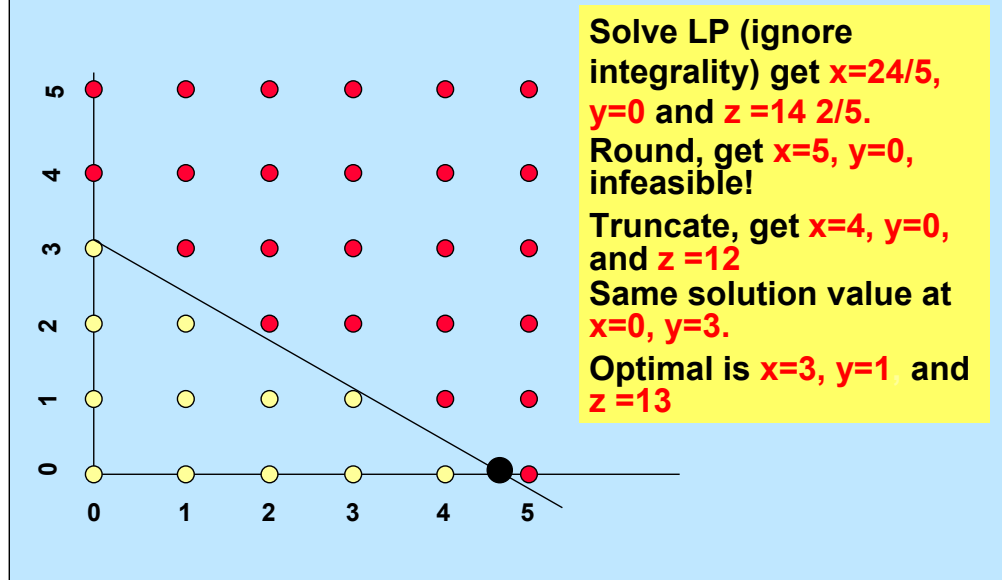
Question: What is the optimal integer solution?

What is the optimal linear solution?

Can one use linear programming to solve the integer program?

The feasible region here consists of the yellow circles only.

A rounding technique that sometimes is useful, and sometimes not.



Often we will consider the “linear programming relaxation” of an integer program. This relaxation is obtained by dropping the integrality constraints. In this case, if we solve the linear programming relaxation, we get the solution $x = 24/5$, $y = 0$. It’s somewhat close to the optimal integer solution. However, in this case, it doesn’t really help us to obtain the optimum integer solution.

Why integer programs?

- **Advantages of restricting variables to take on integer values**
 - **More realistic**
 - **More flexibility**
- **Disadvantages**
 - **More difficult to model**
 - **Can be much more difficult to solve**

7

In practice, most problems have at least some features that prevent linear programming from being a good enough model. For this reason, integer programming models are much more practical. They can model logical constraints, nonlinear functions, and much more.

While it is a robust and excellent modeling tool, it is much harder to solve than linear programs.

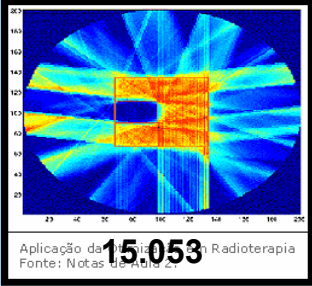
Trading for Profit

Nooz is a contestant on Trading for Profits. Its main slogan is

“I ♥ Trading for Profit”

Nooz has just won 14 IHTFP points. We now join the quiz show to see what the 14 points are worth.

We illustrate integer programming on a contestant choosing prizes in a game show.

iPod	server	brass rat
5 points	7 points	4 points
gift certificate	Tutoring in 6.041	
3 points	4 points	6 points

I got a bit carried away last year in developing these voiceovers. I hope they play on your computer.

In any case, the products are

1. An iPod plus speaker
2. A server for holding vast amounts of data
3. A “brass rat”
4. A \$500 gift certificate to Au Bon Pan
5. Tutoring in 6.041.
6. Dinner with Professor Orlin and the TAs of 15.053.

iPod	server	brass rat	gift certificate
5 points	7 points	4 points	3 points
16 utils	22 utils	12 utils	8 utils
6.041	15.053		
4 points	6 points		
11 utils	19 utils		

Nooz determines what each prize is worth to him. He measure everything in "utils" on a scale from 1 to 25.

Nooz converts each product into Utils. You may note that each IHTFP point is worth approximately 3 utils, but the actual numbers vary from product to product.

Prize	iPod 1	server 2	brass rat 3	gift certificate 4	6.041 5	15.053 6
Points	5	7	4	3	4	6
Utility	16	22	12	8	11	19

Budget: 14 IHTFP points.

Write Nooz's problem as an integer program.

Let $x_i = \begin{cases} 1 & \text{if prize } i \text{ is selected} \\ 0 & \text{otherwise} \end{cases}$

11

Nooz's problem is to maximize the total utility subject to using at most 14 IHTFP points.

The first step is choosing decision variables. The rule of thumb in integer programming is as follows: whenever it makes sense at all, try to use binary decision variables. Binary decision variables are great for modeling decisions where one of two choices can be made. As we shall see in the next lecture, they are also useful when one needs to make a decision of one of many choices.

In this case, we let $x_j = 1$ or 0 according as prize j is selected or is not selected.

Integer Programming Formulation

Objective and Constraints?

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

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

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

We will solve this problem in two lectures.

The objective and constraints show up in slideshow mode.

Knapsack or Capital Budgeting

- You have n items to choose from to put into your knapsack.
- Item i has weight w_i , and it has value c_i .
- The maximum weight your knapsack (or you) can hold is b .
- Formulate the knapsack problem.

13

The structure of this problem is as follows:

1. a linear objective
2. a single linear constraint
3. each variable is binary.

This problem arises so frequently that it is given a name. The most common name is the knapsack problem, because it can be used to model the following problem: what is the choice of items to put in your knapsack so as to maximize the total value and so that the weight in the knapsack is at most some specified number b ?

It is often referred to as capital budgeting. In this case, it models the following problem:

Given: a budget b , and a number of different projects, where the j -th project requires a budget of w_j and gives a net present value return of c_j . What is the choice of projects that maximizes the net present revenue without exceeding the budget b ?

The mystery of integer programming

- **Some integer programs are easy (we can solve problems with millions of variables)**
- **Some integer programs are hard (even 100 variables can be challenging)**
- **It takes expertise and experience to know which is which**
- **It's an active area of research at MIT and elsewhere**

14

Researchers have been working on integer programming for nearly 50 years, and have tried to develop great solution techniques. But to this day, there are many problems that are easily solved (despite being of rather large size) and other problems that cannot be solved even approximately (despite having far fewer variables). Researchers believe that this is inherent to integer programming models. Some are hard, and some are easy to solve. Researchers in integer programming have put enormous energies into figuring out which problems are easy and which are not. In addition, they have tried to develop increasingly better solution techniques.

Types of integer programs

- All integer programs have linear equalities and inequalities and some or all of the variables are required to be integer.

A Pure Integer Program: all variables are required to be integral.

$$\begin{aligned} \text{Max} \quad & 6x_1 + 2x_2 + 2x_3 + 8x_4 \\ & 5x_1 + 3x_2 + 4x_3 + 3x_4 \leq 25 \\ & x_1 + x_2 + x_3 + x_4 \leq 8 \\ & x_j \geq 0 \text{ and integer for each } j = 1 \text{ to } 4 \end{aligned}$$

15

We next point out that there are several varieties of integer programs.

Pure integer programs permit integer variables that can take on more values than 0 and 1. These problems are typically very hard to solve.

Binary (or 0-1) Integer Program:
all variables are required to be 0 or 1

$$\begin{aligned} \text{Max} \quad & 8x_1 + 4x_2 + 2x_3 + 3x_4 \\ & 5x_1 + 3x_2 + 4x_3 + 3x_4 \leq 10 \\ & x_1 + x_2 + x_3 + x_4 \leq 3 \\ & x_j \in \{0,1\} \text{ for each } j = 1 \text{ to } 4 \end{aligned}$$

Recall: the constraint

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

is equivalent to

$$0 \leq x_j \leq 1 \text{ and } x_j \text{ is integer}$$

16

Binary integer programs are typically hard to solve, but easier than pure integer programs.

Mixed Linear Integer Program (MILP)

- Some but not necessarily all variables are required to be integer. Other variables are permitted to be fractional.

$$\begin{aligned} \text{Max} \quad & 6x_1 + 2x_2 + 2x_3 + 4y \\ & 5x_1 + 3x_2 + 4x_3 + 3y \leq 25 \\ & x_1 + x_2 + x_3 + y \leq 8 \\ & x_j \geq 0 \text{ and integer for each } j = 1 \text{ to } 3 \\ & y \geq 0 \end{aligned}$$

17

Mixed linear integer programs are typically the hardest problems to solve of all of them.

**Objective for the rest of this lecture:
introduce modeling in integer programming**

- **Logical Constraints**
- **Fixed charge Problems**
- **The Game of Fiver**

A Standard Approach for IP Formulations

Your model
in the form
of a “word
problem”



A Math Model
sometimes
with logical
constraints and
non-linear functions

The transformation to an IP
sometimes uses cleverness,
sometimes is obvious, and
sometimes, you just look up to see
how it is done.



An Integer Programming
Model

19

I strongly urge students to formulate these challenging problems first using any math that is convenient. So, we take a word problem and convert it into a mathematical optimization problem. Subsequently, we convert the mathematical optimization problem into an integer program.

iPod	server	brass rat	gift certificate
5 points	7 points	4 points	3 points
16 utils	22 utils	12 utils	8 utils

6.041	15.053
4 points	6 points
11 utils	19 utils

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

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

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

Logical Constraints on 0-1 Variables

- Consider Nooz's problem. Suppose he said:
 - If I select the iPod, then I won't bother selecting the 6.041 tutoring, since I'll hope that they podcast the homework solutions.
 - How would we model this?

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

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

$$x_1 + x_5 \leq 1$$

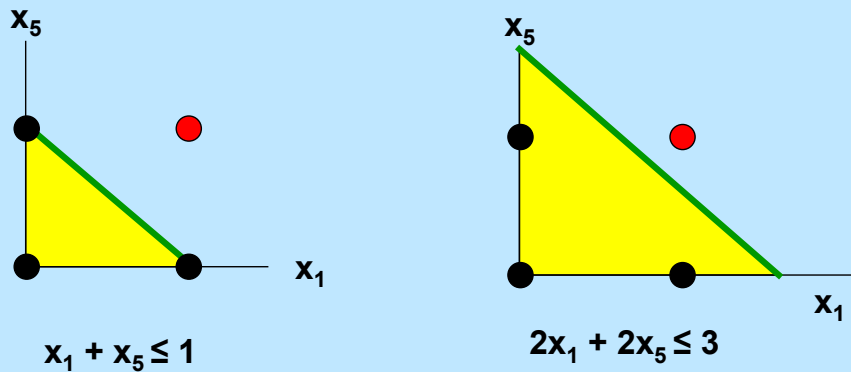
$$x_j \in \{0,1\} \text{ for each } j = 1 \text{ to } 6$$

21

The constraint $x_1 + x_5 \leq 1$ does not look like the original word problem formulation. But it is equivalent. This is more apparent on the next slide.

On logical constraints on binary variables

- With logical constraints involving only two variables, one can easily draw the feasible region, and determine what linear constraint is needed. (Actually, lots of different constraints will work).



22

With only two variables, it is easy to graph the feasible region. In fact, the feasible region consists of 3 points. We want to add a linear constraint that “separates” the infeasible red point from the three feasible black points.

There are an infinite number of lines that separates the three feasible points from the infeasible point $(1, 1)$. The simplest and best of these constraints is $x_1 + x_5 \leq 1$. But $2x_1 + 2x_5 \leq 3$ is also valid. And so is $1000x_1 + x_5 \leq 1000$, assuming that we also constrain x_5 to be 0 or 1.

Mental Break

- MIT Jeopardy



Logical Constraints 1

$x_1 = 1$ if iPod

$x_5 = 1$ if 6.041

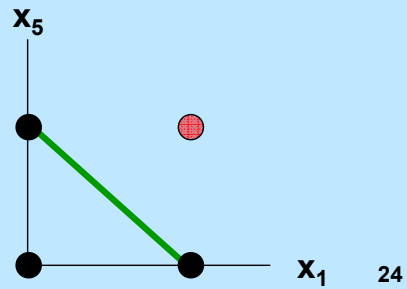
Constraint 1. If you select the iPod, you cannot select 6.041

MIP Constraint:

$$x_1 + x_5 \leq 1$$

Feasibility Table

x_1	x_5	Feasible?
0	0	yes
0	1	yes
1	0	yes
1	1	no



Logical Constraints 2

$x_1 = 1$ if iPod

$x_2 = 1$ if server

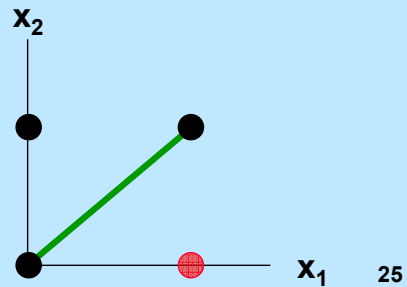
Constraint 2. If Prize 1 is selected then Prize 2 must be selected.

MIP Constraint:

$$x_1 \leq x_2$$

Feasibility Table

x_1	x_2	Feasible?
0	0	yes
0	1	yes
1	0	no
1	1	yes



Once the feasible region is drawn, it is easy to see that there are an infinite number of constraints that separate the three feasible points from the infeasible point $(1, 0)$. The simplest and best of these is $x_1 \leq x_2$.

Logical Constraints 3

$x_1 = 1$ if iPod

$x_2 = 1$ if server

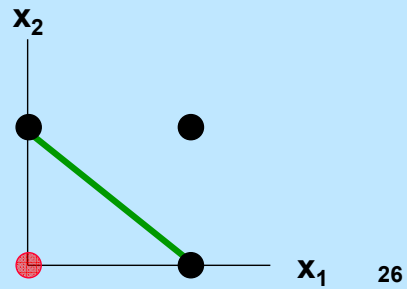
Constraint 3. You must select Prize 1 or Prize 2 or both

MIP Constraint:

$$x_1 + x_2 \geq 1$$

Feasibility Table

x_1	x_2	Feasible?
0	0	
0	1	
1	0	
1	1	



The answer is revealed in Slide Show mode. But you may want to figure out the constraint first.

Logical Constraints 4

$x_1 = 1$ if iPod

$x_2 = 1$ if server

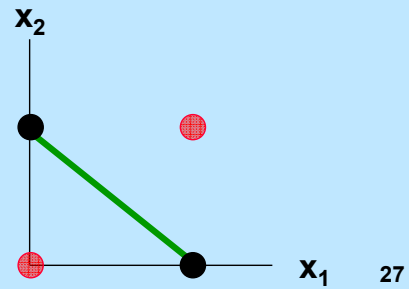
Constraint 4. You must select Prize 1 or Prize 2 but not both.

MIP Constraint:

$$x_1 + x_2 = 1$$

Feasibility Table

x_1	x_2	Feasible?
0	0	
0	1	
1	0	
1	1	



This one is a bit trickier since the constraint that separates the feasible points from the infeasible points is an equality constraint.

Integer programs and non-linear objectives

IPs are great for modeling non-linear objectives

- **Fixed charge problems**
 - you need to pay an up front price for producing anything. Costs are linear after that.
- **General non-linear functions of one variable**

Fixed Charge Problems:

- Data for Zor's problem.

	Gold	Silver	Bronze	Available
TA labor	2	4	5	100
lead	1	1	1	30
pixie dust	10	5	2	204
Profit (Euros)	52	30	20	

Cost of set up	500	400	300	
----------------	-----	-----	-----	--

Suppose that Zor is unable to get any of his reactions going without an expensive set up. After that there is no additional cost beyond material and labor.

29

Here we consider the problem of Zor trying to optimize the production of gold, silver, and bronze using TA labor, pixie dust, and lead. But we have introduced a new feature called a “fixed charge.” In order to make a product, Zor must set up the production at a fixed cost. For example, to make any gold, Zor needs to pay 500 Euros for the set up. After that, there is no additional cost. Similarly, Zor has a fixed charge for producing silver and bronze. Zor does not have to pay the fixed charge if he doesn't produce the metal.

Zor's original problem

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

New variables

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

Part of the math model.

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

We need to transform this into an IP model.

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

30

In order to model Zor's problem as an IP, we create three new binary variables. The variable w_1 indicates whether Zor produces gold or not. The variables w_2 and w_3 indicate whether Zor produces silver or bronze.

The description of the variables on this slide is a valid mathematical formulation of the variables, but it is not part of the integer program itself. We need to write integer linear constraints that will ensure the variables w are correctly defined.

Modeling Fixed Charges

Note: $x_1 \leq 20$ (constraint 3),
 $x_2 \leq 30$ (constraint 2)
 $x_3 \leq 20$ (constraint 1).

We need upper bounds on variables in order to create the IP model.

New variables

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

$$x_1/20 \leq w_1 \leq x_1 \quad w_1 \in \{0, 1\}$$

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

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

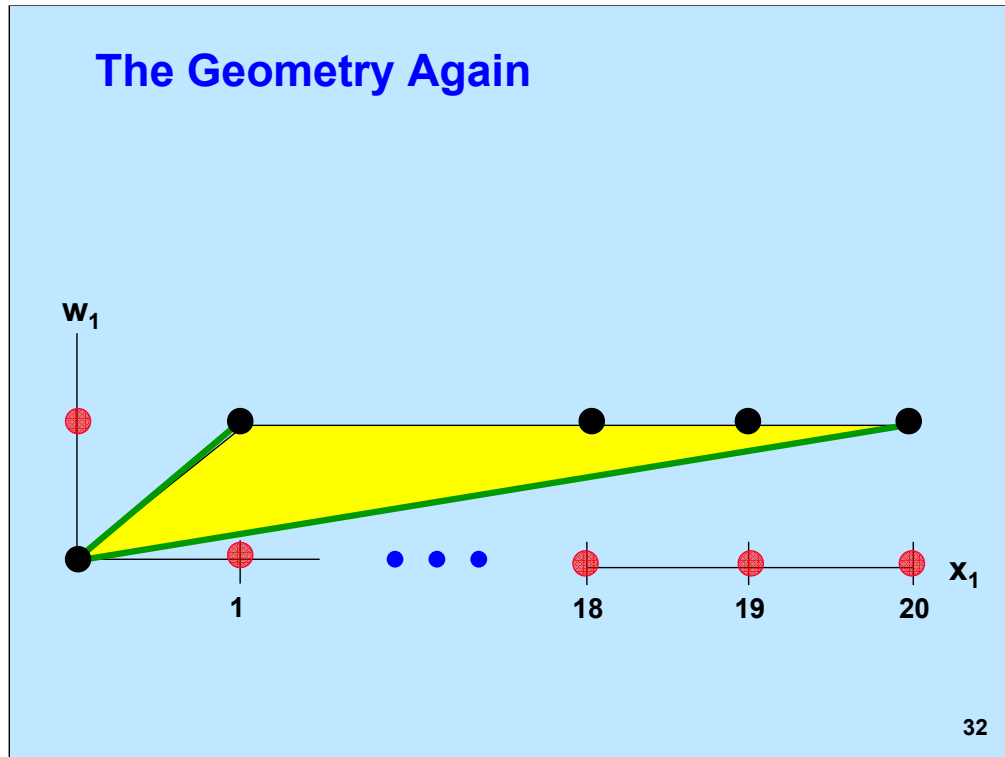
31

In order to write the constraints of the IP, it turns out that we need to have upper bounds on x_1 , x_2 , and x_3 . This is not obvious yet, but you will see why the upper bounds are needed.

We know that x_1 is between 0 and 20. We want to force w_1 to be 0 whenever x_1 is 0. And we want to force w_1 to be 1 whenever x_1 is between 1 and 20. This is accomplished as $x_1/20 \leq w_1 \leq x_1$. You can verify that w_1 will be defined as we want with these constraints, assuming we also add the constraint that w_1 is binary.

We can constrain w_2 and w_3 similarly.

The Geometry Again



This is a two dimensional portrayal of the constraints $x_1/20 \leq w_1 \leq x_1$.

Zor's modified Problem

$$\begin{aligned} \text{Max} \quad & 52 x_1 + 30 x_2 + 20 x_3 - 500 w_1 - 400 w_2 - 300 w_3 \\ \text{s.t} \quad & 2 x_1 + 4 x_2 + 5 x_3 \leq 100 \\ & 1 x_1 + 1 x_2 + 1 x_3 \leq 30 \\ & 10 x_1 + 5 x_2 + 2 x_3 \leq 204 \\ & x_1/20 \leq w_1 \leq x_1 \quad w_1 \in \{0, 1\} \\ & x_2/30 \leq w_2 \leq x_2 \quad w_2 \in \{0, 1\} \\ & x_3/20 \leq w_3 \leq x_3 \quad w_3 \in \{0, 1\} \\ & x_1, x_2, x_3 \geq 0 \text{ integer} \end{aligned}$$

33

As of this point, we have not modeled Zor's problem. We have only modeled w_1 , w_2 , w_3 . But the only addition we need is to subtract out the fixed charges for producing gold, silver and bronze. So, the model on this slide is the complete model.

General Approach for fixed charges for IPs

Suppose that the cost of production for variable j is

$$c(x_j) = \begin{cases} 0 & \text{if } x_j = 0 \\ f_j + c_j x_j & \text{if } x_j > 0 \end{cases}$$

Let $w_j = \begin{cases} 1 & x_j \geq 1 \\ 0 & x_j = 0 \end{cases}$

Suppose that $x_j \leq M$ for all j .
 M may be very large.
Also assume that x_j is integral.

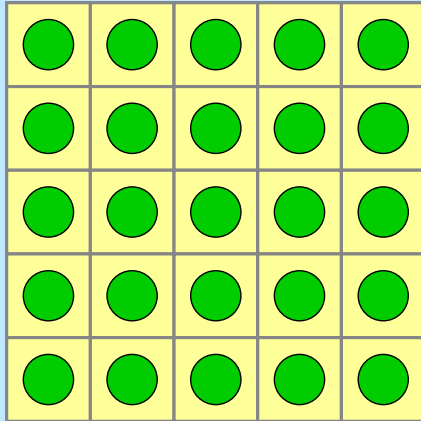
$$\begin{array}{ll} \text{minimize} & f_j w_j + c_j x_j \\ \text{subject to} & x_j / M \leq w_j \leq x_j \quad w_j \in \{0, 1\}. \end{array}$$

34

This is a general approach for modeling fixed charges. You will note that we still need an upper bound on x_j . Often upper bounds on values are represented by the letter M . It is suggestive that the value is quite large, although it need not be so in any particular model.

It turns out that we can omit the constraint $w_j \leq x_j$, and we would still get a valid model. Consider Zor's problem, and suppose that we dropped the constraint $w_j \leq x_j$. This would be equivalent to permitting Zor to pay for a setup of gold even if he does not produce any. (Zor would not have to pay for the set up. But he would have this option.) Normally, having more options will change an optimization problem. However, in this case, we are granting Zor an option that would never be taken in an optimal solution. So, the option of setting up for gold and not producing it is a worthless option. Put another way, dropping the constraint $w_1 \leq x_1$ would not change Zor's optimal solution. And we could also drop the constraints $w_2 \leq x_2$ and $w_3 \leq x_3$.

The Game of Fiver.

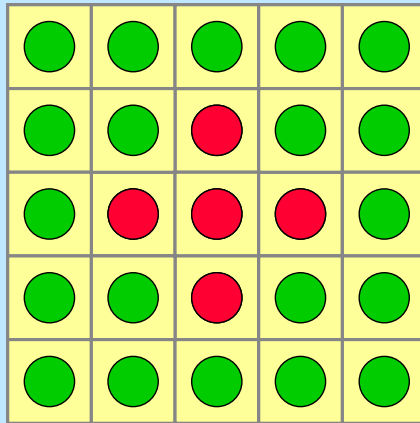


Click on a circle, and flip its color and that of adjacent colors.

Can you make all of the circles red?

To play Fiver, you can find it in multiple places on the web including <http://www.mazeworks.com/fiver/index.htm>

The game of fiver.

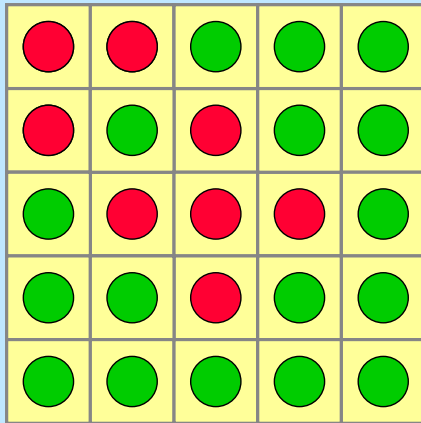


Click on (3, 3)

36

Clicking on a circle flips the color of the circle as well as the colors of adjacent circles.

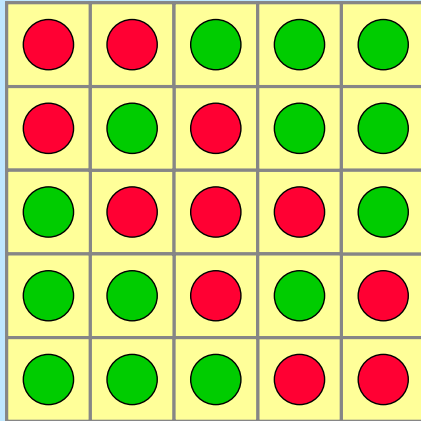
The game of fiver.



Click on (1, 1)

Click on (5, 5)

The game of fiver.



Next: an optimization problem whose solution solves the problem in the fewest moves.

On forming Integer programs

1. First select the set of decision variables.

It turns out that timing does not matter in this game. All that matters is what square are clicked on.

2. Write the objective.
3. Write the constraints. If it is easier to express it using non-linear constraints, or logical constraints, then do this first.

Optimizing the game of fiver.

	1	2	3	4	5
1	●	●	●	●	●
2	●	●	●	●	●
3	●	●	●	●	●
4	●	●	●	●	●
5	●	●	●	●	●

Let $x(i,j) = 1$ if I click on the square in row i and column j .

$x(i,j) = 0$ otherwise.

40

Ultimately, we want each circle to flip from green to red. Consider for example the circle in location $(3, 2)$. If we click on $(3, 1)$ or $(3, 2)$ or $(3, 3)$ or $(2, 2)$, or $(4, 2)$, it will flip the color of the circle $(3, 2)$. If we click on these five different circles an odd number of times in total, then $(3, 2)$ will end up red. If we click on these circles an even number of times in total, then $(3, 2)$ will end up green.

Interestingly, it doesn't matter when each click occurs.

Let's write the formulation

$x(i,j) = 1$ if I click on the square in row i and column j .

$x(i,j) = 0$ otherwise.

We let $x(i, j)$ be the decision variable indicating whether we click on circle (i, j) or not. There is no advantage to clicking on a circle two times since this is equivalent to not clicking on it at all.

Optimizing the game of fiver

Minimize $\sum_{i,j=1 \text{ to } 5} x(i, j)$

s.t. $x(i, j) + x(i, j-1) + x(i, j+1) + x(i-1, j) + x(i+1, j)$ is odd for all $i, j = 1$ to 5 .

$x(i, j)$ is 0 or 1 for all $i, j = 1$ to 5 .

$x(i, j) = 0$ otherwise.

- This (with a little modification) is an **integer program**.

42

I want to write that given any square (i, j) that we should click an odd number of times as restricted to

1. (i, j)
2. $(i, j-1)$, the circle to the left
3. $(i, j+1)$, the circle to the right
4. $(i-1, j)$, the circle above it
5. $(i+1, j)$, the circle below it.

The notation is not perfect because it is possible that i or j is 1 or 5, in which case (i, j) is not adjacent to four other circles. We bypass this by saying that $x(i, j)$ is 0 whenever $i = 0$ or 6 or $j = 0$ or 6 .

Optimizing the game of fiver

$$\text{Minimize } \sum_{i,j=1 \text{ to } 5} x(i, j)$$

$$\text{s.t. } x(i, j) + x(i, j-1) + x(i, j+1) + x(i-1, j) + x(i+1, j) - 2y(i, j) = 1 \quad \text{for all } i, j = 1 \text{ to } 5.$$

$$x(i, j) \text{ is } 0 \text{ or } 1 \text{ for all } i, j = 1 \text{ to } 5.$$

$$x(i, j) = 0 \text{ otherwise}$$

$$0 \leq y(i, j) \leq 2; \quad y(i, j) \text{ integer for } i, j = 1 \text{ to } 5.$$

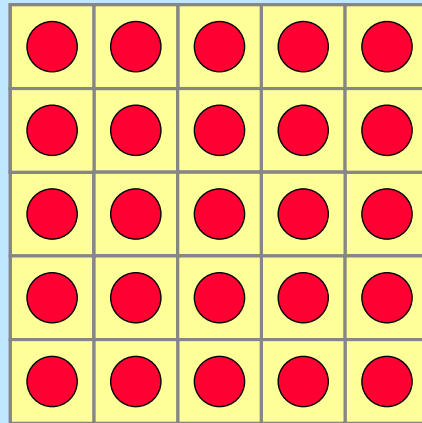
This is an integer program.

43

The previous slide is not an integer program. It has a constraint “ $x(i, j) + x(i, j-1) + x(i, j+1) + x(i-1, j) + x(i+1, j)$ is odd”. But if a number is odd, it means that we can subtract a non-negative even integer and get 0. We use this fact to create an integer program on this slide.

So this model illustrates one more trick in modeling mathematical constraints as integer programming constraints.

Should I give away the solution?



44

In slide show mode, you can see the solution. I pretend in class to be clicking as though I am really playing the game of fiver. But this is a PowerPoint slide, and it does not matter where you click.

Using Excel Solver to Solve Integer Programs

- Add the integrality constraints (or add that a variable is binary)
- Set the Solver Tolerance. (The tolerance is the percentage deviation from optimality allowed by solver in solving Integer Programs.)
 - The default is 5%
 - The default is way too high
 - It often finds the optimum for small problems

Some Comments on IP models

- There are often multiple ways of modeling the same integer program.
- Solvers for integer programs are *extremely* sensitive to the formulation. (not true for LPs)

Summary on Integer Programming

- **Dramatically improves the modeling capability**
 - Economic indivisibilities
 - Logical constraints
 - capital budgeting
 - games
- **Not as easy to model**
- **Not as easy to solve.**
- **Next lecture: more IP formulations**