Math 340 Lecture 33

An integer programming example. Consider the following two-variable linear program.

- Maximize $y$ subject to
  - $y \leq 3x$
  - $y \leq 2 - x$
  - $x, y \geq 0$.

This is simple enough, and you could plug it into the simplex method and find the optimum is at $x = 0.5$ and $y = 1.5$. A picture makes it especially clear what region we’re looking at and it’s obvious where the maximum is. (The scanned page with the picture I drew in class is at the end of this document).

Now, what if I asked you this as an integer programming problem instead: the same objective function and constraints, but to find the maximum where $x, y$ are integers? In this case the picture is very helpful and we can see that the point $x = 1$ and $y = 1$ will be the integer optimum. But how could we find that algebraically or algorithmically without looking at a picture? We’ve said a bit about rounding, and I guess if you round $x$ up and $y$ down you get what you want, but that’s kind of cheating. (If you rounded both down or both up, you’d get an infeasible point!)

The key idea is something a bit more sophisticated than rounding, but still a similar idea. We see our general optimum is at $x = 0.5$ - so we do kind of want to focus on the nearest integer values of $x = 0$ and $x = 1$. However it’s not guaranteed that the optimum would be either - but it is trivially true that the integer optimum will satisfy either $x \leq 0$ or $x \geq 1$! So we can just cut out the entire corridor of $0 < x < 1$ where our fractional optimum was, and look at what’s on either side. This will result in two linear programs to consider, one for each side:

- Maximize $y$ subject to
  - $y \leq 3x$
  - $y \leq 2 - x$
  - $x \leq 0$
  - $x, y \geq 0$

and

- Maximize $y$ subject to
  - $y \leq 3x$
  - $y \leq 2 - x$
  - $x \geq 1$
  - $x, y \geq 0$

The largest integer solution to either will be the largest integer solution to our original LP. We can then go and solve these and find the optimum of the first is $x = y = 0$ (objective value 0) and the optimum of the second is $x = y = 1$ (objective value 1). Both of these are actually integer solutions now, and the solution $x = y = 1$ has the larger objective value of the two, so we can conclude $x = y = 1$ is the integer optimum of the original LP.
Integer programming in general - branch and bound. In general an integer programming problem is a LP where we want to optimize over integer values of the variables. A mixed programming problem is one where some of the variables are restricted to being integers and others aren’t. Both types of problems come up plenty in the real world, because there’s lots of things you might have as variables that only make sense in whole quantities!

The simplex method by itself can’t do anything with integer programming problems - it really depends on a bunch of linear algebra that requires fractions. But the example above suggests we can build off of the simplex method: solve the original “relaxed” LP, and if we have a variable that’s not an integer value add in constraints that kick out that region between two integers. So roughly speaking the strategy is

- Solve the original relaxed LP for its optimum solution \(x_1^*, \ldots, x_n^*\).
- If all variables \(x_i^*\) are integers, we’re done because our optimum is actually an integer solution. If not, pick out some variable that isn’t an integer, so it’s between two consecutive integers: \(M < x_i^* < M + 1\).
- Then solve the following two integer programming problems:
  - Our original LP with the constraint \(x_i^* \leq M\) added.
  - Our original LP with the constraint \(x_i^* \geq M + 1\) added.
- Compare the two optimum integer solutions of these problems; the larger one is the optimum integer solution to the original LP.

At first glance this might seem ridiculous - we’re just going from one integer programming problem we don’t know how to solve to two of them. But the point is that we’re carving up the feasible region and making it smaller in both cases. Fact: If you use this algorithm recursively, eventually it will actually terminate because you’ll reach LPs that either have their optimal solutions being integral (or are infeasible).

This is called a branching approach, since for each LP we have we split into two “smaller” ones to solve, creating a binary tree. The fact stated above tells us that we won’t branch forever, and every path down will eventually end (at an LP that’s either infeasible or has an integer solution as its optimum).

But, while the binary tree will be finite, it might be really big because of the exponential growth from all of the branches we might have to do. Fortunately sometimes we can cut off a lot of these without having to actually evaluate them. Once I’ve found an integer solution with some maximum value, I can cut off any branches where the LP has a smaller maximum (whether or not that maximum is obtained at an integer).

An example. Rather than write out the procedure for this, it’s easier to illustrate with an example (and LINDO to solve all of the LPs for us).

- Maximize \(8x_1 + 5x_2 + 13x_3\) subject to
  - \(x_1 + 2x_2 + 3x_3 \leq 13\)
  - \(2x_1 + x_2 + 7x_3 \leq 22\)
  - \(3x_1 + 2x_2 + 4x_3 \leq 16\)
  - \(x_1, x_2, x_3 \geq 0\) integers

Step 1 is to solve the completely relaxed LP, without any integrality constraints. This just uses the simplex method, and the solution is \(z^* \approx 48.77\), \(x^* \approx (1.8, 0, 2.6)\). This is the root of our tree, which I’ll label (1). (The tree diagram is scanned and added to the end of the notes).

The “branch” idea is now to pick one of the variables that isn’t an integer and see what happens when we push it to either side. Generally a reasonable strategy is to pick the one that’s closest to being an integer, so in this case \(x_1 \approx 1.8\). We imagine that \(x_1 = 2\) is the best integer approximation so the \(x_1 \geq 2\) side seems more interesting, but we can’t neglect the \(x_1 \leq 1\) side. This gives us two branches, node (2) where we impose \(x_1 \geq 2\) (with an optimum of \(z^* = 48.5\) at \(x^* = (2.0, 2.5)\)) and node (3) where we impose \(x_1 \leq 1\) (with an optimum of \(z^* = 48.6\) at \(x^* = (1.1, 1.2)\)).
Of these two, node (3) has a larger optimum, so let’s branch from there. Here $x_2$ is the closest to an integer so we branch on it, putting in node (4) where we add the constraint $x_2 \leq 1$ (giving $z^* \approx 48.29$ at $x^* \approx (1,1,2.7)$) and node (5) where we add $x_2 \geq 2$ (giving $z^* \approx 48.46$ at $x^* \approx (0.3,2,2.8)$).

Now our open nodes are (2), (4), and (5); of these (2) has the largest objective value so we branch down from there. The only non-integer variable in the solution there is $x_3 = 2.5$, so we branch on it. Get node (6) for $x_3 \leq 2$ with $z^* = 47.3$ at $x^* = (2.7,0,2)$, and node (7) for $x_3 \geq 3$ turns out to be infeasible so we terminate there.

Our open nodes now are (4), (5), and (6), and (5) has the largest objective value. For that one we branch on $x_3 \approx 2.8$; node (8) for $x_3 \geq 3$ turns out to be infeasible, and node (9) for $x_3 \leq 2$ gives $z^* = 46.5$ at $x^* = (1,2.5,2)$.

This leaves (4), (6), and (9) open, and (4) has the largest objective value. The only noninteger variable here is $x_3 = 2.7$, so we branch on it: adding the constraint $x_3 \geq 3$ gives an integer solution $z^* = 44$ and $x^* = (0,1,3)$ at node (10). Adding $x_3 \leq 2$ gives an integer solution $z^* = 39$ and $x^* = (1,1,2)$ at node (11), but this is worse than what we have at node (10).

Now we’re down to nodes (6) and (9) open. Node (6) has a higher objective value so let’s go back over there. For it we need to branch on $x_1 = 2.7$; adding $x_1 \geq 3$ gives node (12) with $z^* = 46.75$ at $x^* = (3,0,1.75)$, and adding $x_1 \leq 2$ gives node (13) with $z^* = 47$ at $x^* = (2,1,2)$, another integer solution! This is better than the integer solutions we found on the other branch.

At this point we have nodes (9) and (12) open, but both have objective values that are less than 47 - so nothing coming off of them could possibly beat node (13). So we can truncate both of them, and conclude that we’ve finished our branch and bound problem and that $x^* = (2,1,2)$ giving $z^* = 47$ is the integer optimum!
**INTEGER PROGRAMMING**

**EXAMPLE:** MAXIMIZE \( y \) SUBJECT TO:
- \( y \leq 3x \)
- \( y \leq 2-x \)
- \( x, y \geq 0 \) INTEGERS.

**OPTIMAL FRACTIONAL SOLUTION:** \( x = \frac{1}{2}, y = \frac{3}{2} \)

**OPTIMAL INTEGER SOLUTION:** \( x = 1, y = 1 \).

Q: HOW DO WE DO THIS SYSTEMATICALLY IN GENERAL?
IDEA: CUT OUT REGION \( 0 \leq x \leq 1 \). SPLIT INTO TWO LPs.

1. \( \text{MAX } y, \)
   - \( y \leq 3x \)
   - \( y \leq 2-x \)
   - \( x \leq 0 \)
   - \( x, y \geq 0 \) INTEGERS
   
   **OPTIMUM:** \( x = 0, y = 0 \) (AS LP)

2. \( \text{MAX } y, \)
   - \( y \leq 3x \)
   - \( y \leq 2-x \)
   - \( x \geq 1 \)
   - \( x, y \geq 0 \) INTEGERS

   **OPTIMUM:** \( x = 1, y = 1 \) (AS LP)