INTEGER LINEAR PROGRAMMING: SOLUTION TECHNIQUES

Prof. Stephen Graves

• For linear programming, the simplex method is a very robust solution algorithm. For any kind of linear program, it will solve the linear program very quickly regardless of the problem context or the mathematical structure.

• For integer programming there is no robust solution procedure. There is no one method that will work well for all or most problems. In fact, most integer programming problems are time-consuming to solve and can be enormously difficult to solve, depending on the mathematical structure.

• There are many very sophisticated and complicated methods designed to solve efficiently specific classes or types of integer programs, e. g., the traveling salesman problem or the knapsack problem.

• Four general purpose methods for solving integer programs:

a. **Rounding.** The idea is to solve the linear program version of the problem, and then round the solution to be integer. The resulting answer may or may not be optimal. It may or may not be feasible. It may or may not be near optimal. For some problems the decision of how to do the rounding is obvious; for some others it is not.

b. **Enumeration.** The idea is to enumerate all possible integer solutions. This is a good idea only when the number of integer variables is fairly small, or your computer budget is large: if there are 10 binary variables, there will be $2^{10} = 1024$ possible solutions, etc.

c. **Cutting Plane Methods.** The idea here is to cut away the feasible region until all of the remaining corner points are integer points. This is done by adding constraints (cuts) to the original problem, and then solving the resulting problem as a linear program. This method is increasing in importance for some specialized classes of problems.

d. **Branch and Bound.** The idea here is to divide and conquer. The method proceeds by partially enumerating solutions and solving linear programs over smaller feasible regions, gaining information and being able to ignore a huge number of possible solutions in the process. This is the most successful general purpose integer programming technique available. Solving integer programs with up to 100 binary integer variables is relatively routine with desktop codes, such as LINDO or CPLEX.

Branch and Bound

• Key idea is divide and conquer. Solve ILP as LP, then partition problem into two or more pieces. (this is called *branching*) Solve the LP version of the sub-problems and, if necessary, continue partitioning. (solving the LP version yields a *bound* on the ILP)

• The efficiency of approach comes when it is possible to "fathom" a sub-problem. Fathoming occurs when we have determined that we do not need to explore any further the possible solutions within a sub-problem.

We can fathom a sub-problem if any of the following conditions are discovered when we solve the LP version of the sub-problem:

a. there is no feasible solution to the sub-problem;

b. the optimal LP solution satisfies the integer requirements of the ILP;

c. the optimal objective value from the sub-problem LP is worse than the best known solution to the ILP (the incumbent solution).

Example:

MAX Z = X + Y

s. t.
X + 2Y <= 16
2X + Y <= 14.9
X, Y >= 0, INTEGER

Solve LP relaxation: X* = 4.6; Y* = 5.7; Z = 10.3

Partition into two sub-problems to try to force one of the variables (Y) to be integer:

MAX Z = X + Y	MAX Z = X + Y
s. t. X + 2Y <= 16	s. t. X + 2Y <= 16
2X + Y <= 14.9	2X + Y <= 14.9
Y >= 6	Y <= 5

Solve each LP relaxation:

X* = 4; Y* = 6; Z = 10	X* = 4.95; Y* = 5; Z = 9.95

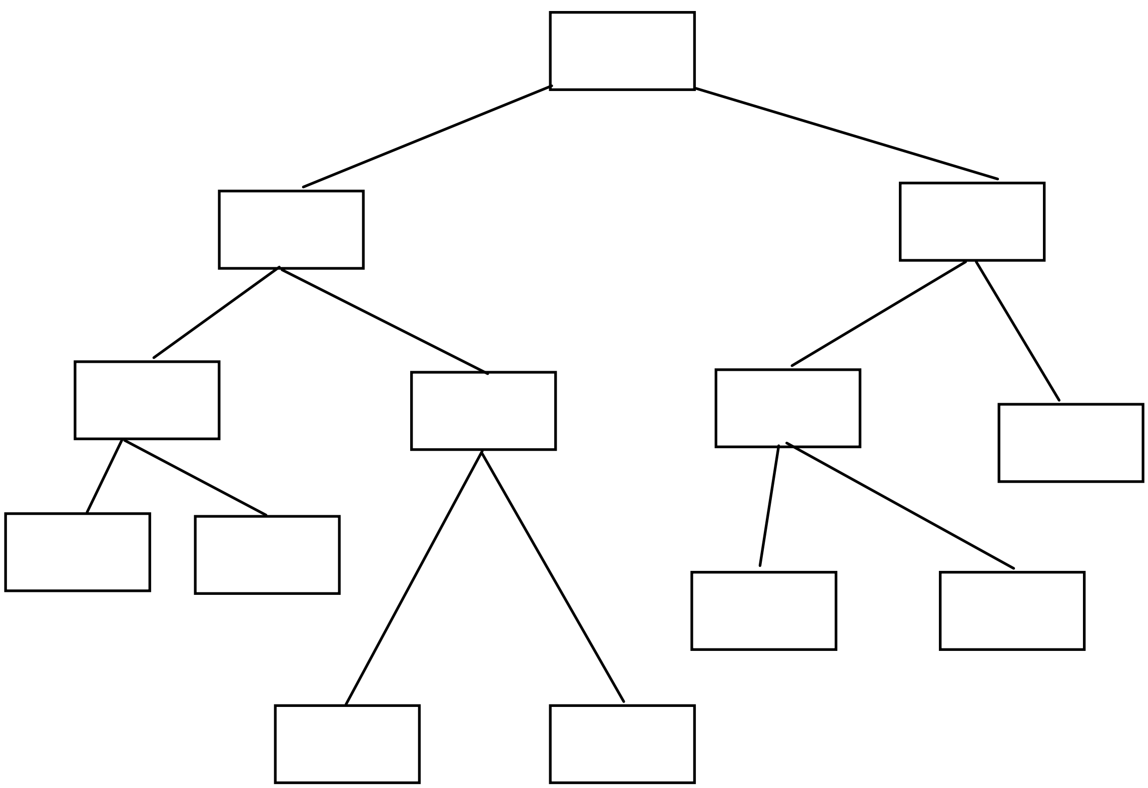
BRANCH AND BOUND EXAMPLE

$$\text{MAX} \quad Z = 60\,X1 + 60\,X2 + 40\,X3 + 10\,X4 + 20\,X5 + 3\,X6$$

SUBJECT TO

$$3\,X1 + 5\,X2 + 4\,X3 + X4 + 4\,X5 + X6 \leq 10$$

$$X1, X2, X3, X4, X5, X6 = 0 \text{ or } 1$$



BRANCH AND BOUND EXAMPLE

MAX $Z = 60\,X1 + 60\,X2 + 40\,X3 + 10\,X4 + 20\,X5 + 3\,X6$

SUBJECT TO

$3\,X1 + 5\,X2 + 4\,X3 + X4 + 4\,X5 + X6 \le 10$

$X1, X2, X3, X4, X5, X6 = 0 \text{ or } 1$