COMS21103: Integer Programming

Dima Damen
Dima.Damen@bristol.ac.uk

Bristol University, Department of Computer Science
Bristol BS8 1UB, UK

December 9, 2015

Integer Program

- An integer program is encountered when the optimum output is expected as integer values
- The book shipment example from the last lecture is an integer program
- This is because the output is not meaningful unless it is in integers
- In the book shipment example, even though the solution was actually given as integers (200,0,200,600), solving a linear program cannot guarantee that the solution is in integers

Integer Program

- An integer program is encountered when the optimum output is expected as integer values
- The book shipment example from the last lecture is an integer program
- This is because the output is not meaningful unless it is in integers
- In the book shipment example, even though the solution was actually given as integers (200,0,200,600), solving a linear program cannot guarantee that the solution is in integers
- In formulating this as a linear program, we ignored the 'integer' constraint
- This is referred to as **linear program relaxation of an integer program**

Integer Program

- An integer program is encountered when the optimum output is expected as integer values
- The book shipment example from the last lecture is an integer program
- This is because the output is not meaningful unless it is in integers
- In the book shipment example, even though the solution was actually given as integers (200,0,200,600), solving a linear program cannot guarantee that the solution is in integers
- In formulating this as a linear program, we ignored the ‘integer’ constraint
- This is referred to as **linear program relaxation of an integer program**
- One way around this is to round numbers... this can cause serious problems

Integer Program

An integer program thus has the standard form:

$$
\begin{array}{lll}
\text{maximise} & \displaystyle\sum_{j=1}^{n} c_j x_j & \\
\text{subject to} & \displaystyle\sum_{j=1}^{n} a_{ij} x_j \leq b_i & \text{for } i = 1, 2, .., m \\
 & x_j \geq 0 & \text{for } j = 1, 2, .., n \\
 & x_j \in \mathbb{Z}(\mathit{integer}) & \text{for } j = 1, 2, .., n
\end{array}
$$

Integer Program

- An integer linear program is a linear program further constrained by the integrality restrictions
- In integer programs all variables are restricted to be integers

Integer Program

- An integer linear program is a linear program further constrained by the integrality restrictions
- In integer programs all variables are restricted to be integers
- In mixed-integer programs, some variables only are restricted to be integers
- Mixed-integer programs are out of the scope in these slides

Integer Program

- An integer linear program is a linear program further constrained by the integrality restrictions
- In integer programs all variables are restricted to be integers
- In mixed-integer programs, some variables only are restricted to be integers
- Mixed-integer programs are out of the scope in these slides
- Binary programs are programs where variables are restricted to the binary $\in \{0, 1\}$.
- These are special cases of integer programs and could be solved in the same way by adding inequalities $x_j \leq 1$ for $j = 1, 2, .., n$

Integer Program

- An integer linear program is a linear program further constrained by the integrality restrictions
- In integer programs all variables are restricted to be integers
- In mixed-integer programs, some variables only are restricted to be integers
- Mixed-integer programs are out of the scope in these slides
- Binary programs are programs where variables are restricted to the binary $\in \{0, 1\}$.
- These are special cases of integer programs and could be solved in the same way by adding inequalities $x_j \leq 1$ for $j = 1, 2, .., n$
- Integer programming has applications in planning, scheduling and networking

Integer Program - Case Study

Work of Raymond Kwan, University of Leeds
Scheduling crew shifts for train companies

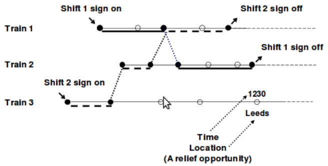


Fig. 1 Train work and crew shifts

R. Kwan (2009). Case Studies of Successful Train Crew Scheduling Optimization. Multidisciplinary International Conference on Scheduling: Theory and Applications.

Integer Program - Case Study

Work of Raymond Kwan, University of Leeds
Scheduling crew shifts for train companies

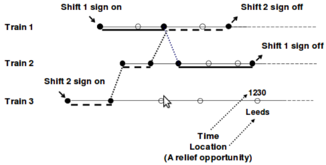


Fig.1 Train work and crew shifts

Minimise $$W_1 \sum_{j=1}^{n} c_j x_j + W_2 \sum_{j=1}^{n} x_j$$

Subject to

$$\sum_{j=1}^{n} a_{ij} x_j \geq 1 \qquad i = 1,2,\ldots\ldots,m$$

$$x_j = 0 \; or \; 1 \qquad j = 1,2,\ldots\ldots,n$$

where

n is the number of candidate shifts
m is the number of work pieces

x_j is a shift variable, $x_j = \begin{cases} 1 & \text{if shift } j \text{ is selected} \\ 0 & \text{otherwise} \end{cases}$

c_j is the cost of shift j

$$a_{ij} = \begin{cases} 1 & \text{if work piece } i \text{ is covered by shift } j \\ 0 & \text{otherwise} \end{cases}$$

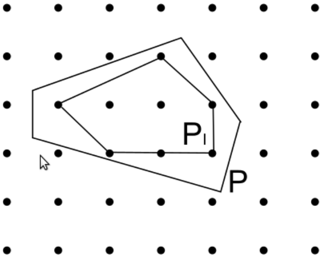
W_1 and W_2 are weight constants

Fig.2 Set covering ILP for train crew scheduling

R. Kwan (2009). Case Studies of Successful Train Crew Scheduling Optimization. Multidisciplinary International Conference on Scheduling: Theory and Applications.

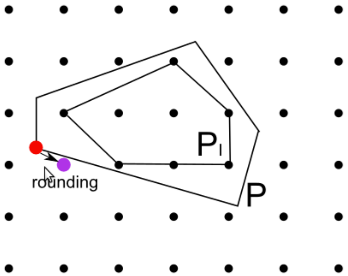
Integer Programming and Linear Programming

- P_I: the space of feasible solutions of the integer program
- P: the space of feasible solutions of the *relaxed* linear program
- $P_I \subseteq P$



Integer Programming and Linear Programming

- P_I: the space of feasible solutions of the integer program
- P: the space of feasible solutions of the *relaxed* linear program
- $P_I \subseteq P$
- If rounding is used, the integer solution could be infeasible.



Integer Programming and Linear Programming

- Whereas the simplex method is effective for solving linear programs, there is no single technique for solving integer programs.

Integer Programming and Linear Programming

- Whereas the simplex method is effective for solving linear programs, there is no single technique for solving integer programs.
- We thus first relax the integer program by removing integer constraints on the variables
- The *relaxed* linear program is then solved using the simplex method

Integer Programming and Linear Programming

- Whereas the simplex method is effective for solving linear programs, there is no single technique for solving integer programs.
- We thus first relax the integer program by removing integer constraints on the variables
- The *relaxed* linear program is then solved using the simplex method
- The integer solution is then found using one of three strategies:
  - Enumeration techniques (e.g. Branch-and-bound)
  - Cutting plane techniques
  - Group-theoretic techniques
- We will discuss the first two strategies next,

Integer Programming and Linear Programming

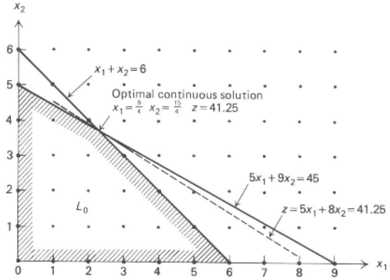
- Whereas the simplex method is effective for solving linear programs, there is no single technique for solving integer programs.
- We thus first relax the integer program by removing integer constraints on the variables
- The *relaxed* linear program is then solved using the simplex method
- The integer solution is then found using one of three strategies:
  - Enumeration techniques (e.g. Branch-and-bound)
  - Cutting plane techniques
  - Group-theoretic techniques
- We will discuss the first two strategies next,
- These approaches assume a solution to the *relaxed* linear program has been found
- If the relaxed linear program is infeasible, the integer program is definitely infeasible

Integer Programming - Ex.

$$\begin{array}{ll} \text{maximise} & 5x_1 + 8x_2 \\ \text{subject to} & x_1 + x_2 \leq 6 \\ & 5x_1 + 9x_2 \leq 45 \\ & x_1, x_2 \geq 0 \text{ and integer} \end{array}$$

Integer Programming - Ex.

$$\begin{array}{ll} \text{maximise} & 5x_1 + 8x_2 \\ \text{subject to} & x_1 + x_2 \leq 6 \\ & 5x_1 + 9x_2 \leq 45 \\ & x_1, x_2 \geq 0 \text{ and integer} \end{array}$$



Integer Programming - Ex.

$$
\begin{array}{ll}
\text{maximise} & 5x_1 + 8x_2 \\
\text{subject to} & x_1 + x_2 \leq 6 \\
 & 5x_1 + 9x_2 \leq 45 \\
 & x_1, x_2 \geq 0 \text{ and integer}
\end{array}
$$

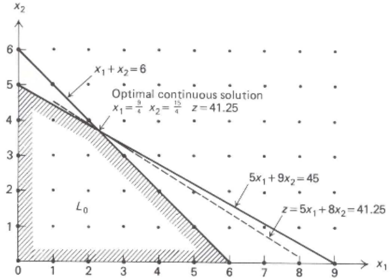


Table 9.1 Problem features.

	Continuous optimum	*Round off*	*Nearest feasible point*	*Integer optimum*
x_1	$\frac{9}{4} = 2.25$	2	2	0
x_2	$\frac{15}{4} = 3.75$	4	3	5
z	41.25	Infeasible	34	40

Integer Programming - Ex.

$$\begin{array}{ll} \text{maximise} & 5x_1 + 8x_2 \\ \text{subject to} & x_1 + x_2 \leq 6 \\ & 5x_1 + 9x_2 \leq 45 \\ & x_1, x_2 \geq 0 \text{ and integer} \end{array}$$

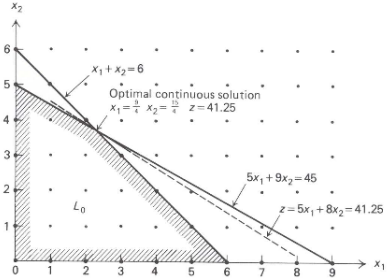


Table 9.1 Problem features.

	Continuous optimum	*Round off*	*Nearest feasible point*	*Integer optimum*
x_1	$\frac{9}{4}=2.25$	2	2	0
x_2	$\frac{15}{4}=3.75$	4	3	5
z	41.25	Infeasible	34	40

- the upper bound on $z^* \leq z^0 = 41\frac{1}{4}$, and since z^* must be integral then $z^* \leq 41$.

Method 1: Branch and Bound

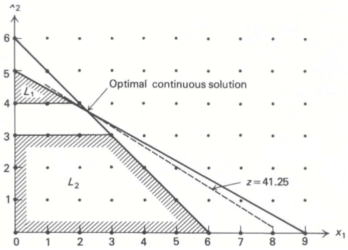
- Branch and bound is another “divide and conquer” strategy.
- systematically subdivide the linear programming feasible region
- make assessments of the integer-programming problem based upon these subdivisions.

Method 1: Branch and Bound

- The linear programming solution is $x_1 = 2\frac{1}{4}, x_2 = 3\frac{3}{4}$
- We can divide the feasible region by *attempting* to make either x_1 or x_2 integer.
- Assume we attempt to make $x2$ integer $\rightarrow x_2 \geq 4$ or $x_2 \leq 3$

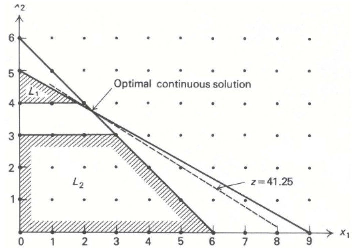
Method 1: Branch and Bound

- The linear programming solution is $x_1 = 2\frac{1}{4}, x_2 = 3\frac{3}{4}$
- We can divide the feasible region by *attempting* to make either x_1 or x_2 integer.
- Assume we attempt to make $x2$ integer $\rightarrow x_2 \geq 4$ or $x_2 \leq 3$



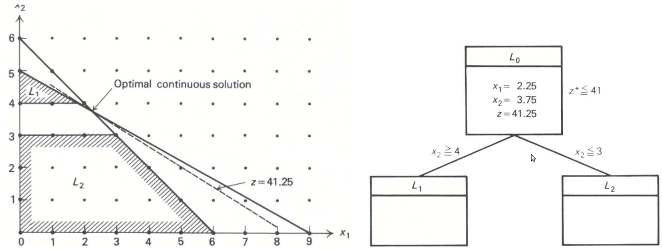
Method 1: Branch and Bound

- The linear programming solution is $x_1 = 2\frac{1}{4}, x_2 = 3\frac{3}{4}$
- We can divide the feasible region by *attempting* to make either x_1 or x_2 integer.
- Assume we attempt to make $x2$ integer $\rightarrow x_2 \geq 4$ or $x_2 \leq 3$
- We can visualise this by an **enumeration tree**.



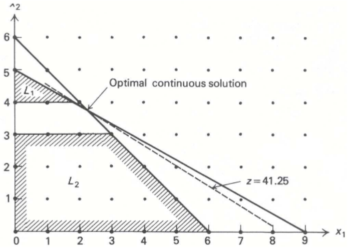
Method 1: Branch and Bound

- The linear programming solution is $x_1 = 2\frac{1}{4}, x_2 = 3\frac{3}{4}$
- We can divide the feasible region by *attempting* to make either x_1 or x_2 integer.
- Assume we attempt to make $x2$ integer $\rightarrow x_2 \geq 4$ or $x_2 \leq 3$
- We can visualise this by an **enumeration tree**.



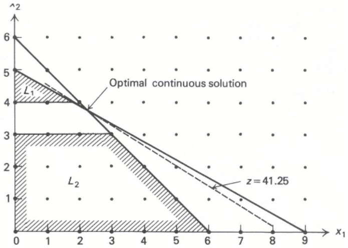
Method 1: Branch and Bound

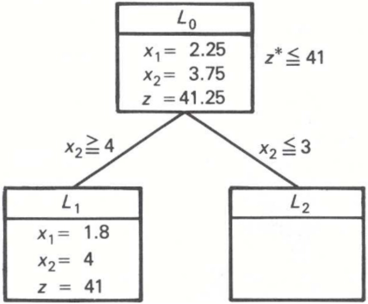
- Follow the branch and solve for L_1



Method 1: Branch and Bound

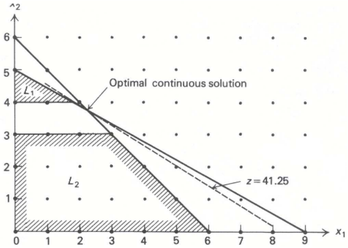
- Follow the branch and solve for L_1

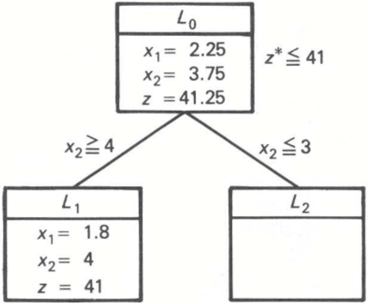




Method 1: Branch and Bound

- Follow the branch and solve for L_1

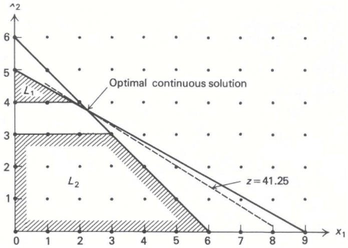


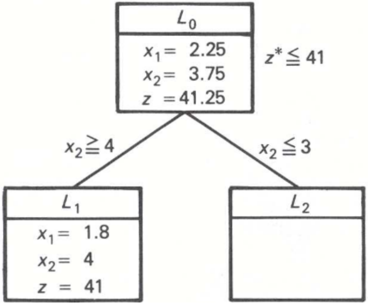


- L_1 is feasible and x_2 is integer.

Method 1: Branch and Bound

- Follow the branch and solve for L_1





- L_1 is feasible and x_2 is integer.
- Shift to explore changing x_1 to integer.

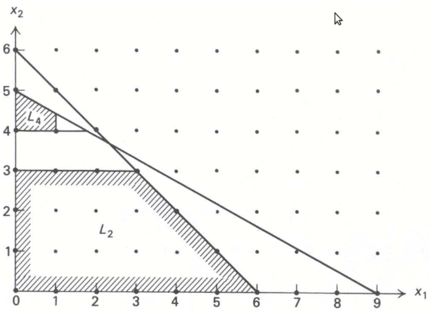
Dima Damen
Dima.Damen@bristol.ac.uk

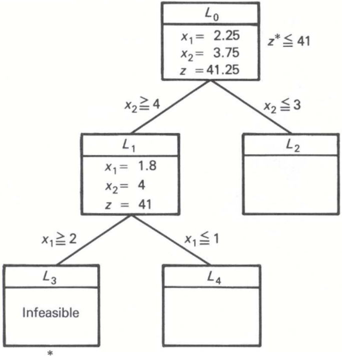
Method 1: Branch and Bound

- Subdivide L_1 further into regions L_3 with $x_1 \geq 2$ and L_4 with $x_1 \leq 1$

Method 1: Branch and Bound

- Subdivide L_1 further into regions L_3 with $x_1 \geq 2$ and L_4 with $x_1 \leq 1$

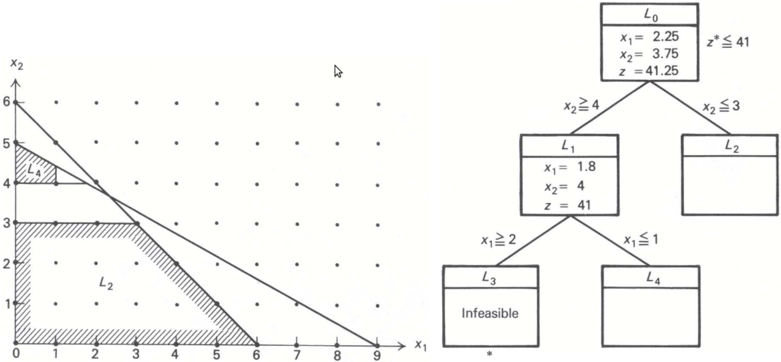




Dima Damen
Dima.Damen@bristol.ac.uk

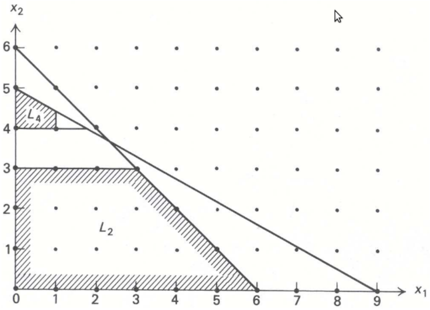
Method 1: Branch and Bound

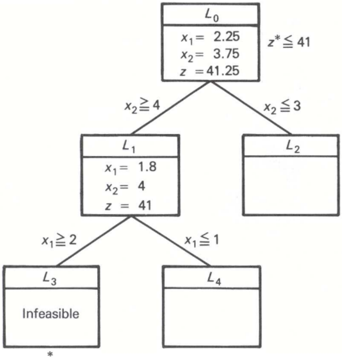
- Subdivide L_1 further into regions L_3 with $x_1 \geq 2$ and L_4 with $x_1 \leq 1$
- L_3 is an infeasible problem



Method 1: Branch and Bound

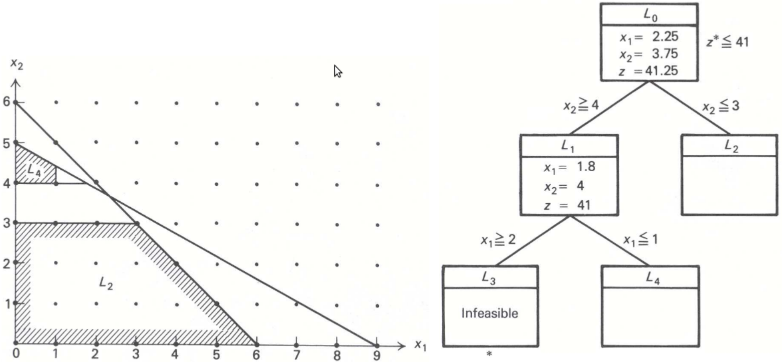
- Subdivide L_1 further into regions L_3 with $x_1 \geq 2$ and L_4 with $x_1 \leq 1$
- L_3 is an infeasible problem
- The branch in the enumeration tree does not need to be explored further





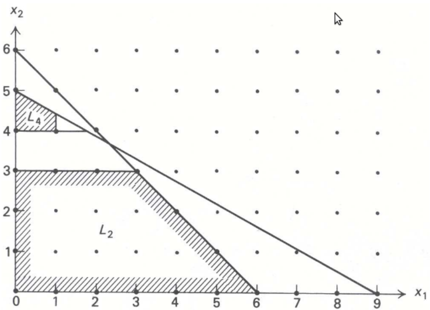
Method 1: Branch and Bound

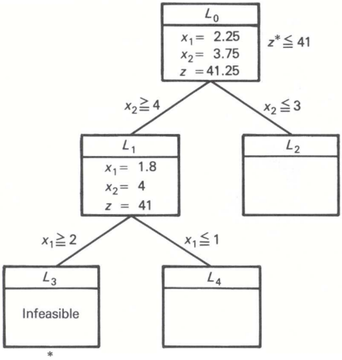
- Subdivide L_1 further into regions L_3 with $x_1 \geq 2$ and L_4 with $x_1 \leq 1$
- L_3 is an infeasible problem
- The branch in the enumeration tree does not need to be explored further
- At this point we can explore L_2 or L_4



Method 1: Branch and Bound

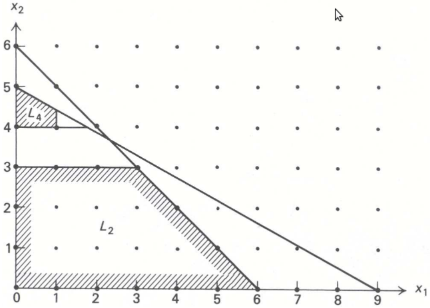
- Subdivide L_1 further into regions L_3 with $x_1 \geq 2$ and L_4 with $x_1 \leq 1$
- L_3 is an infeasible problem
- The branch in the enumeration tree does not need to be explored further
- At this point we can explore L_2 or L_4
- The decision can be done arbitrarily, or using some heuristics

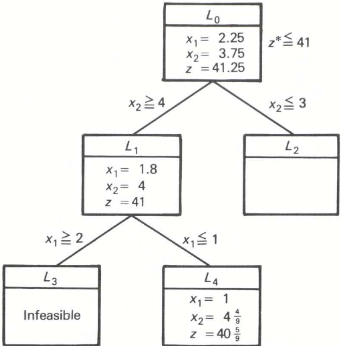




Method 1: Branch and Bound

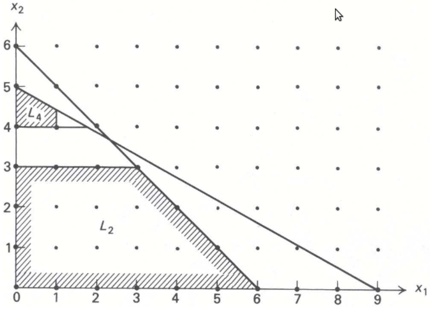
- Let's choose L_4, solution to L_4 is $(x_1, x_2) = (1, 4\frac{4}{9})$.

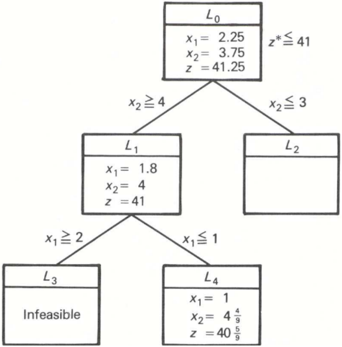




Method 1: Branch and Bound

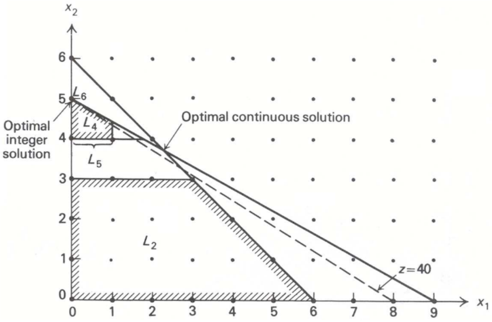
- Let's choose L_4, solution to L_4 is $(x_1, x_2) = (1, 4\frac{4}{9})$.
- x_2 is not yet an integer $\rightarrow$ explore $x_2 \leq 4$ and $x_2 \geq 5$



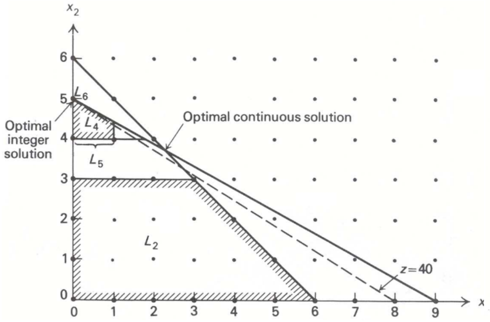


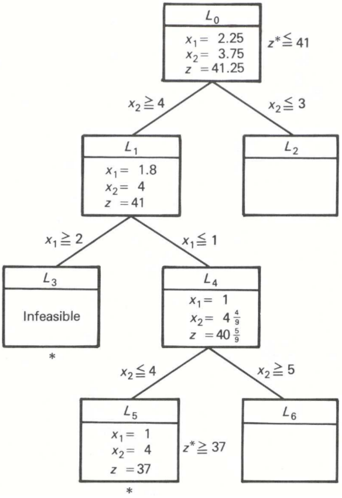
Dima Damen
Dima.Damen@bristol.ac.uk

Method 1: Branch and Bound



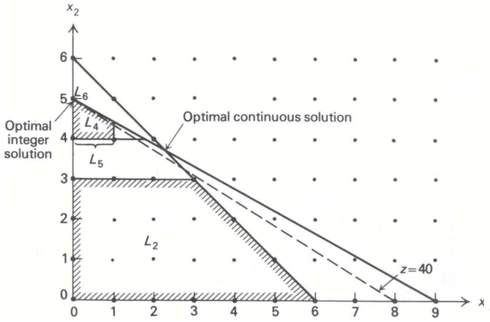
Method 1: Branch and Bound

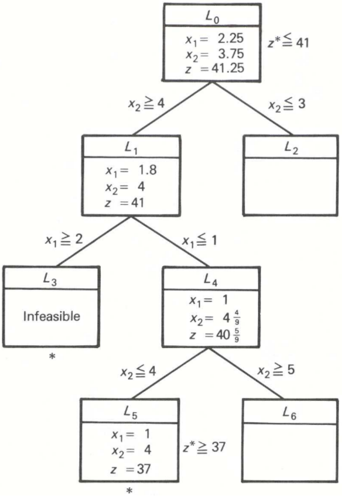




Method 1: Branch and Bound

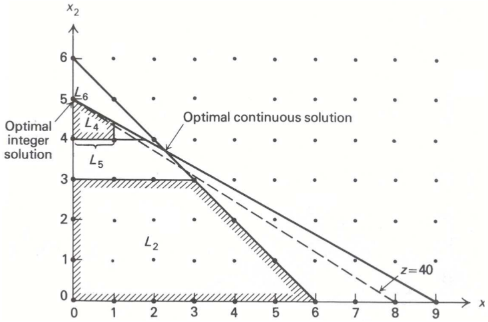
- L_5 is a feasible problem with the optimal solution $(x_1, x_2) = (1, 4)$ and $z_{L_5} = 37$.

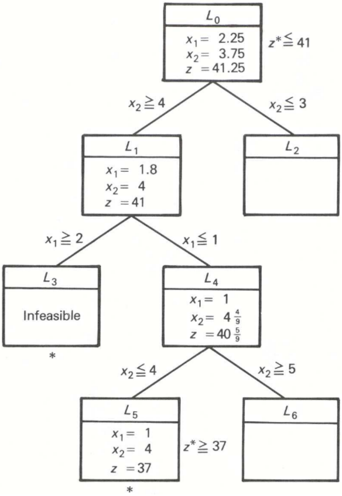




Method 1: Branch and Bound

- L_5 is a feasible problem with the optimal solution $(x_1, x_2) = (1, 4)$ and $z_{L_5} = 37$.
- We known that $37 \leq z^* \leq 41$.

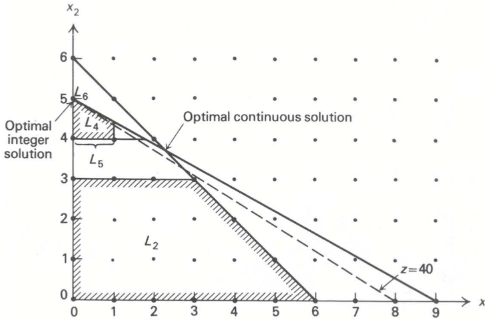


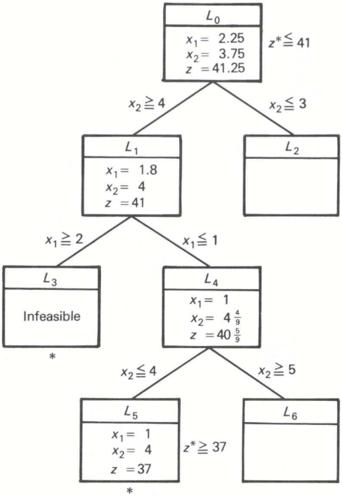


Dima Damen
Dima.Damen@bristol.ac.uk

Method 1: Branch and Bound

- L_5 is a feasible problem with the optimal solution $(x_1, x_2) = (1, 4)$ and $z_{L_5} = 37$.
- We known that $37 \leq z^* \leq 41$.
- L_2 or L_6 might contain a better answer



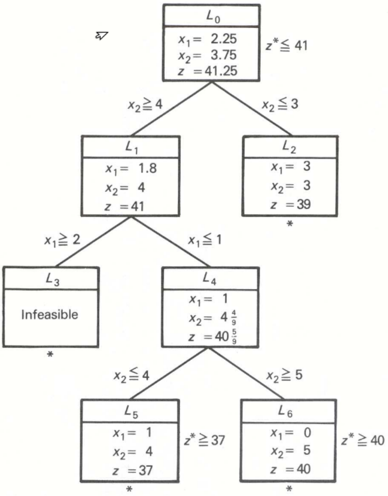


Method 1: Branch and Bound

- Solving for L_2 and L_6 accordingly
  optimal answers are integers

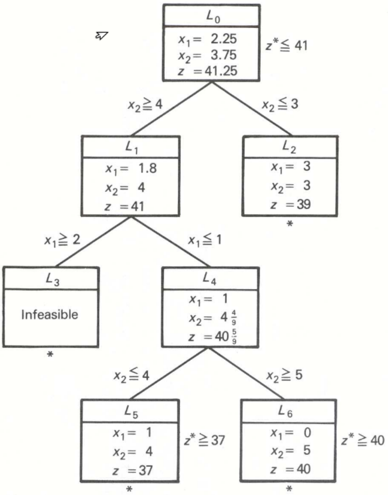
Method 1: Branch and Bound

- Solving for L_2 and L_6 accordingly optimal answers are integers
- L_6 has the optimal solution $z^* = 40$



Method 1: Branch and Bound

- Solving for L_2 and L_6 accordingly optimal answers are integers
- L_6 has the optimal solution $z^* = 40$
- The optimum answer to the integer program is $(x_1, x_2) = (0, 5)$



Method 2: Cutting Planes

- Cutting-plane algorithm solves integer programs by modifying *relaxed* linear-programming solutions until the integer solution is obtained.
- It does not partition the feasible region, as in brand-and-bound
- It instead works with a single linear program, which is refined by adding new constraints.
- The new constraints successively reduce the feasible region until an integer optimal solution is found.
- In practice, the branch-and-bound procedures almost always outperform the cutting-plane algorithm.
- Cutting-plane was though the first algorithm developed for integer programming that could be proven to converge in a finite number of steps.

Method 2: Cutting Planes

$$\boxed{\begin{array}{ll} \text{maximise} & 5x_1 + 8x_2 \\ \text{subject to} & x_1 + x_2 \leq 6 \\ & 5x_1 + 9x_2 \leq 45 \\ & x_1, x_2 \geq 0 \text{ and integer} \end{array}}$$

change to slack form

$$\boxed{\begin{array}{ll} z = & 5x_1 + 8x_2 \\ s_1 = & 6 - x_1 - x_2 \\ s_2 = & 45 - 5x_1 - 9x_2 \end{array}}$$

Method 2: Cutting Planes

$$\begin{array}{ll} \text{maximise} & 5x_1 + 8x_2 \\ \text{subject to} & x_1 + x_2 \leq 6 \\ & 5x_1 + 9x_2 \leq 45 \\ & x_1, x_2 \geq 0 \text{ and integer} \end{array}$$

change to slack form

$$\begin{array}{ll} z = & 5x_1 + 8x_2 \\ s_1 = & 6 - x_1 - x_2 \\ s_2 = & 45 - 5x_1 - 9x_2 \end{array}$$

Solving using the Simplex method results in:

$$\begin{array}{llllll} z = & 41\frac{1}{4} & - & \frac{5}{4}s_1 & - & \frac{3}{4}s_2 \\ x_1 = & \frac{9}{4} & - & \frac{9}{4}s_1 & + & \frac{1}{4}s_2 \\ x_2 = & \frac{15}{4} & + & \frac{5}{4}s_1 & - & \frac{1}{4}s_2 \end{array}$$

Method 2: Cutting Planes

$$\boxed{\begin{array}{llllll} z = & 41\frac{1}{4} & - & \frac{5}{4}s_1 & - & \frac{3}{4}s_2 \\ x_1 = & \frac{9}{4} & - & \frac{9}{4}s_1 & + & \frac{1}{4}s_2 \\ x_2 = & \frac{15}{4} & + & \frac{5}{4}s_1 & - & \frac{1}{4}s_2 \end{array}}$$

Method 2: Cutting Planes

$$
\begin{array}{llclcl}
z = & 41\frac{1}{4} & - & \frac{5}{4}s_1 & - & \frac{3}{4}s_2 \\
x_1 = & \frac{9}{4} & - & \frac{9}{4}s_1 & + & \frac{1}{4}s_2 \\
x_2 = & \frac{15}{4} & + & \frac{5}{4}s_1 & - & \frac{1}{4}s_2
\end{array}
$$

Rewrite as

$$
\begin{array}{ccccccccc}
-z & & & - & \frac{5}{4}s_1 & - & \frac{3}{4}s_2 & = & -41\frac{1}{4} \\
 & x_1 & & + & \frac{9}{4}s_1 & - & \frac{1}{4}s_2 & = & \frac{9}{4} \\
 & & x_2 & - & \frac{5}{4}s_1 & + & \frac{1}{4}s_2 & = & \frac{14}{4}
\end{array}
$$

Method 2: Cutting Planes

$$\boxed{\begin{array}{lllll} z = & 41\frac{1}{4} & - & \frac{5}{4}s_1 & - & \frac{3}{4}s_2 \\ x_1 = & \frac{9}{4} & - & \frac{9}{4}s_1 & + & \frac{1}{4}s_2 \\ x_2 = & \frac{15}{4} & + & \frac{5}{4}s_1 & - & \frac{1}{4}s_2 \end{array}}$$

Rewrite as

$$\boxed{\begin{array}{lllllllll} -z & & & - & \frac{5}{4}s_1 & - & \frac{3}{4}s_2 & = & -41\frac{1}{4} \\ & x_1 & & + & \frac{9}{4}s_1 & - & \frac{1}{4}s_2 & = & \frac{9}{4} \\ & & x_2 & - & \frac{5}{4}s_1 & + & \frac{1}{4}s_2 & = & \frac{14}{4} \end{array}}$$

and rewrite as

$$\boxed{\begin{array}{lllllllllllllll} -z & & & - & 2s_1 & - & s_2 & + & 42 & = & \frac{3}{4} & - & \frac{3}{4}s_1 & - & \frac{1}{4}s_2 \\ & x_1 & & + & 2s_1 & - & s_2 & - & 2 & = & \frac{1}{4} & - & \frac{1}{4}s_1 & - & \frac{3}{4}s_2 \\ & & x_2 & - & 2s_1 & & & - & 3 & = & \frac{3}{4} & - & \frac{3}{4}s_1 & - & \frac{1}{4}s_2 \end{array}}$$

Method 2: Cutting Planes

Let's look at the right hand side,

$$\begin{array}{cccccccccccccc}
-z & & & - & 2s_1 & - & s_2 & + & 42 & = & \frac{3}{4} & - & \frac{3}{4}s_1 & - & \frac{1}{4}s_2 \\
 & x_1 & & + & 2s_1 & - & s_2 & - & 2 & = & \frac{1}{4} & - & \frac{1}{4}s_1 & - & \frac{3}{4}s_2 \\
 & & x_2 & - & 2s_1 & & & - & 3 & = & \frac{3}{4} & - & \frac{3}{4}s_1 & - & \frac{1}{4}s_2
\end{array}$$

- These algebraic manipulations
  - isolate integer coefficients to one side of the equalities and fractions to the other
  - constant terms on the righthand side are all nonnegative
  - slack variable coefficients on the righthand side are all nonpositive

Method 2: Cutting Planes

Let's look at the right hand side,

$$\begin{array}{ccccccccccccccc} -z & & & - & 2s_1 & - & s_2 & + & 42 & = & \frac{3}{4} & - & \frac{3}{4}s_1 & - & \frac{1}{4}s_2 \\ & x_1 & & + & 2s_1 & - & s_2 & - & 2 & = & \frac{1}{4} & - & \frac{1}{4}s_1 & - & \frac{3}{4}s_2 \\ & & x_2 & - & 2s_1 & & & - & 3 & = & \frac{3}{4} & - & \frac{3}{4}s_1 & - & \frac{1}{4}s_2 \end{array}$$

Method 2: Cutting Planes

Let's look at the right hand side,

$$\begin{array}{ccccccccccccccc}
-z & & & - & 2s_1 & - & s_2 & + & 42 & = & \frac{3}{4} & - & \frac{3}{4}s_1 & - & \frac{1}{4}s_2 \\
 & x_1 & & + & 2s_1 & - & s_2 & - & 2 & = & \frac{1}{4} & - & \frac{1}{4}s_1 & - & \frac{3}{4}s_2 \\
 & & x_2 & - & 2s_1 & & & - & 3 & = & \frac{3}{4} & - & \frac{3}{4}s_1 & - & \frac{1}{4}s_2
\end{array}$$

- $s_1 \geq 0$ and $s_2 \geq 0$

Method 2: Cutting Planes

Let's look at the right hand side,

$$
\begin{array}{cccccccccccccc}
-z & & & - & 2s_1 & - & s_2 & + & 42 & = & \frac{3}{4} & - & \frac{3}{4}s_1 & - & \frac{1}{4}s_2 \\
 & x_1 & & + & 2s_1 & - & s_2 & - & 2 & = & \frac{1}{4} & - & \frac{1}{4}s_1 & - & \frac{3}{4}s_2 \\
 & & x_2 & - & 2s_1 & & & - & 3 & = & \frac{3}{4} & - & \frac{3}{4}s_1 & - & \frac{1}{4}s_2
\end{array}
$$

- $s_1 \geq 0$ and $s_2 \geq 0$
- Since s_1 and s_2 appear to the right with negative coefficients, then
  $\frac{3}{4} \leq \frac{3}{4}s_1 + \frac{1}{4}s_2$
  $\frac{3}{4} - \frac{3}{4}s_1 - \frac{1}{4}s_2 \leq 0$

Method 2: Cutting Planes

Let's look at the right hand side,

$-z$			-	$2s_1$	-	s_2	+	42	=	$\frac{3}{4}$	-	$\frac{3}{4}s_1$	-	$\frac{1}{4}s_2$
	x_1		+	$2s_1$	-	s_2	-	2	=	$\frac{1}{4}$	-	$\frac{1}{4}s_1$	-	$\frac{3}{4}s_2$
		x_2	-	$2s_1$			-	3	=	$\frac{3}{4}$	-	$\frac{3}{4}s_1$	-	$\frac{1}{4}s_2$

- $s_1 \geq 0$ and $s_2 \geq 0$
- Since s_1 and s_2 appear to the right with negative coefficients, then
  $\frac{3}{4} \leq \frac{3}{4}s_1 + \frac{1}{4}s_2$
  $\frac{3}{4} - \frac{3}{4}s_1 - \frac{1}{4}s_2 \leq 0$

Add a new slack variable $s_3 \geq 0$ where

$$\frac{3}{4} - \frac{3}{4}s_1 - \frac{1}{4}s_2 + s_3 = 0$$

Method 2: Cutting Planes

Let's look at the right hand side,

$$\begin{array}{|ccccccccccccc|}
\hline
-z & & & - & 2s_1 & - & s_2 & + & 42 & = & \frac{3}{4} & - & \frac{3}{4}s_1 & - & \frac{1}{4}s_2 \\
& x_1 & & + & 2s_1 & - & s_2 & - & 2 & = & \frac{1}{4} & - & \frac{1}{4}s_1 & - & \frac{3}{4}s_2 \\
& & x_2 & - & 2s_1 & & & - & 3 & = & \frac{3}{4} & - & \frac{3}{4}s_1 & - & \frac{1}{4}s_2 \\
\hline
\end{array}$$

And similarly for the remaining constraints

$$\begin{array}{llll}
\frac{3}{4} - \frac{3}{4}s_1 - \frac{1}{4}s_2 + s_3 = 0, & s_3 \geq 0 & \text{and} & \text{integer} \\
\frac{1}{4} - \frac{1}{4}s_1 - \frac{3}{4}s_2 + s_4 = 0, & s_4 \geq 0 & \text{and} & \text{integer} \\
\frac{3}{4} - \frac{3}{4}s_1 - \frac{1}{4}s_2 + s_5 = 0, & s_5 \geq 0 & \text{and} & \text{integer}
\end{array}$$

Method 2: Cutting Planes

Let's look at the right hand side,

$$\begin{array}{|ccccccccccccccc|}
\hline
-z & & & - & 2s_1 & - & s_2 & + & 42 & = & \frac{3}{4} & - & \frac{3}{4}s_1 & - & \frac{1}{4}s_2 \\
 & x_1 & & + & 2s_1 & - & s_2 & - & 2 & = & \frac{1}{4} & - & \frac{1}{4}s_1 & - & \frac{3}{4}s_2 \\
 & & x_2 & - & 2s_1 & & & - & 3 & = & \frac{3}{4} & - & \frac{3}{4}s_1 & - & \frac{1}{4}s_2 \\
\hline
\end{array}$$

And similarly for the remaining constraints

$$\begin{array}{llll}
\frac{3}{4} - \frac{3}{4}s_1 - \frac{1}{4}s_2 + s_3 = 0, & s_3 \geq 0 & \text{and} & \text{integer} \\
\frac{1}{4} - \frac{1}{4}s_1 - \frac{3}{4}s_2 + s_4 = 0, & s_4 \geq 0 & \text{and} & \text{integer} \\
\frac{3}{4} - \frac{3}{4}s_1 - \frac{1}{4}s_2 + s_5 = 0, & s_5 \geq 0 & \text{and} & \text{integer}
\end{array}$$

- Notice that first and third constraints are identical.

Method 2: Cutting Planes

Let's look at the right hand side,

$$\begin{array}{ccccccccccccccc}
-z & & & - & 2s_1 & - & s_2 & + & 42 & = & \frac{3}{4} & - & \frac{3}{4}s_1 & - & \frac{1}{4}s_2 \\
 & x_1 & & + & 2s_1 & - & s_2 & - & 2 & = & \frac{1}{4} & - & \frac{1}{4}s_1 & - & \frac{3}{4}s_2 \\
 & & x_2 & - & 2s_1 & & & - & 3 & = & \frac{3}{4} & - & \frac{3}{4}s_1 & - & \frac{1}{4}s_2
\end{array}$$

And similarly for the remaining constraints

$$\begin{array}{llll}
\frac{3}{4} - \frac{3}{4}s_1 - \frac{1}{4}s_2 + s_3 = 0, & s_3 \geq 0 & \text{and} & \text{integer} \\
\frac{1}{4} - \frac{1}{4}s_1 - \frac{3}{4}s_2 + s_4 = 0, & s_4 \geq 0 & \text{and} & \text{integer} \\
\frac{3}{4} - \frac{3}{4}s_1 - \frac{1}{4}s_2 + s_5 = 0, & s_5 \geq 0 & \text{and} & \text{integer}
\end{array}$$

- Notice that first and third constraints are identical.
- The new equations are called **cuts**, because their derivation did not exclude any integer solutions to the problem.

Method 2: Cutting Planes

Let's look at the right hand side,

$$\begin{array}{|lllllllllllllll|}
\hline
-z & & & - & 2s_1 & - & s_2 & + & 42 & = & \frac{3}{4} & - & \frac{3}{4}s_1 & - & \frac{1}{4}s_2 \\
 & x_1 & & + & 2s_1 & - & s_2 & - & 2 & = & \frac{1}{4} & - & \frac{1}{4}s_1 & - & \frac{3}{4}s_2 \\
 & & x_2 & - & 2s_1 & & & - & 3 & = & \frac{3}{4} & - & \frac{3}{4}s_1 & - & \frac{1}{4}s_2 \\
\hline
\end{array}$$

And similarly for the remaining constraints

$$\begin{array}{llll}
\frac{3}{4} - \frac{3}{4}s_1 - \frac{1}{4}s_2 + s_3 = 0, & s_3 \geq 0 & \text{and} & \text{integer} \\
\frac{1}{4} - \frac{1}{4}s_1 - \frac{3}{4}s_2 + s_4 = 0, & s_4 \geq 0 & \text{and} & \text{integer} \\
\frac{3}{4} - \frac{3}{4}s_1 - \frac{1}{4}s_2 + s_5 = 0, & s_5 \geq 0 & \text{and} & \text{integer}
\end{array}$$

- Notice that first and third constraints are identical.
- The new equations are called **cuts**, because their derivation did not exclude any integer solutions to the problem.
- Any integer feasible solution to the original problem must satisfy the cut constraints
- A cut is to cut away the optimal linear-programming solution from the feasible region without excluding any feasbile integer solution.

Method 2: Cutting Planes

$$\begin{array}{llll} \frac{3}{4} - \frac{3}{4}s_1 - \frac{1}{4}s_2 + s_3 = 0, & s_3 \geq 0 & \text{and} & \text{integer} \\ \frac{1}{4} - \frac{1}{4}s_1 - \frac{3}{4}s_2 + s_4 = 0, & s_4 \geq 0 & \text{and} & \text{integer} \end{array}$$

Recall from the slack form that:

$$\boxed{\begin{array}{ll} z = & 5x_1 + 8x_2 \\ s_1 = & 6 - x_1 - x_2 \\ s_2 = & 45 - 5x_1 - 9x_2 \end{array}}$$

By replacing, you achieve two cut inequalities:

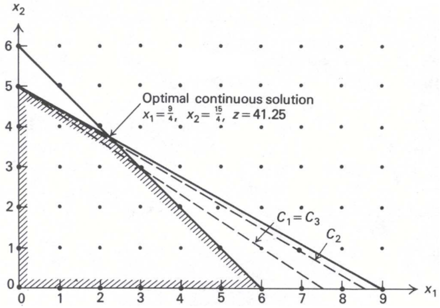
$$\begin{array}{l} 2x_1 + 3x_2 \leq 15 \\ 4x_1 + 7x_2 \leq 35 \end{array}$$

Method 2: Cutting Planes

- The usual strategy is to add cuts (usually only one) to the constraints, then solve the resulting linear program.
- If the optimal values are all integer, they are optimal
- Otherwise, a new cut is derived from the new optimal linear program, and is appended to the constraints

Method 2: Cutting Planes

C_1: $2x_1 + 3x_2 \leq 15$
C_2: $4x_1 + 7x_2 \leq 35$



- In the example before, the first cut leads directly to the optimal solution, but the second does not.
- Also notice that the first cut cuts deeper into the feasible region, but it is difficult to determine which cuts will be deep in this sense.
- If the algorithm chooses a cut that removes very little from the feasible region, the algorithm's performance will be poor

Method 2: Cutting Planes - Time Complexity

- In 1958, R. Gomory made a theoretical break-through
- He showed that integer programs can be solved by *some* linear program (i.e. the relaxed linear program plus added constraints)

Method 2: Cutting Planes - Time Complexity

- In 1958, R. Gomory made a theoretical break-through
- He showed that integer programs can be solved by *some* linear program (i.e. the relaxed linear program plus added constraints)
- Unfortunately, the number of cuts to be added, though finite, is usually large
- This result does not have important practical ramifications

Method 2: Cutting Planes - Time Complexity

- In 1958, R. Gomory made a theoretical break-through
- He showed that integer programs can be solved by *some* linear program (i.e. the relaxed linear program plus added constraints)
- Unfortunately, the number of cuts to be added, though finite, is usually large
- This result does not have important practical ramifications
- In practice, the branch-and-bound procedures almost always outperform the cutting-plane algorithm.

Further Reading

- Applied Mathematical Programming (available online)
  Bradley, Hax and Magnanti.
  Addison-Wesley, 1977.
  - Chapter 9 – Integer Programming