Integer Programming: Theory and Algorithms

Piotr Wojciechowski[1]

[1] LCSEE
West Virginia University
Morgantown, WV USA

April 11, 2015

Outline

Outline

Outline

Reasoning

Reasoning

- Linear Programming allows variables to take non integer values.

Reasoning

- Linear Programming allows variables to take non integer values.
- In many cases rounding of solutions can be performed.

Reasoning

- Linear Programming allows variables to take non integer values.
- In many cases rounding of solutions can be performed.
- Integer Programming allows for the modeling of yes or no choices.

Reasoning

- Linear Programming allows variables to take non integer values.
- In many cases rounding of solutions can be performed.
- Integer Programming allows for the modeling of yes or no choices.
- In these cases non-integer solutions are of little use.

Reasoning

- Linear Programming allows variables to take non integer values.
- In many cases rounding of solutions can be performed.
- Integer Programming allows for the modeling of yes or no choices.
- In these cases non-integer solutions are of little use.

Definition (Integer Program)

Reasoning

- Linear Programming allows variables to take non integer values.
- In many cases rounding of solutions can be performed.
- Integer Programming allows for the modeling of yes or no choices.
- In these cases non-integer solutions are of little use.

Definition (Integer Program)

- An *Integer Program* (IP), like an LP, has linear constraints and linear objective function.

Reasoning

- Linear Programming allows variables to take non integer values.
- In many cases rounding of solutions can be performed.
- Integer Programming allows for the modeling of yes or no choices.
- In these cases non-integer solutions are of little use.

Definition (Integer Program)

- An *Integer Program* (IP), like an LP, has linear constraints and linear objective function.
- However, variables are now restricted to take only integer values.

Reasoning

- Linear Programming allows variables to take non integer values.
- In many cases rounding of solutions can be performed.
- Integer Programming allows for the modeling of yes or no choices.
- In these cases non-integer solutions are of little use.

Definition (Integer Program)

- An *Integer Program* (IP), like an LP, has linear constraints and linear objective function.
- However, variables are now restricted to take only integer values.
- Thus, an IP will have the following form.

$$\max z = \mathbf{c} \cdot \mathbf{x}$$

$$\begin{aligned} \mathbf{A} \cdot \mathbf{x} &\leq \mathbf{b} \\ \mathbf{x} &\geq \mathbf{0} \end{aligned}$$

Reasoning

- Linear Programming allows variables to take non integer values.
- In many cases rounding of solutions can be performed.
- Integer Programming allows for the modeling of yes or no choices.
- In these cases non-integer solutions are of little use.

Definition (Integer Program)

- An *Integer Program* (IP), like an LP, has linear constraints and linear objective function.
- However, variables are now restricted to take only integer values.
- Thus, an IP will have the following form.

$$\max z = \mathbf{c} \cdot \mathbf{x}$$

$$\begin{aligned} \mathbf{A} \cdot \mathbf{x} &\leq \mathbf{b} \\ \mathbf{x} &\geq \mathbf{0} \\ \mathbf{x} &\in \mathbb{Z}^n \end{aligned}$$

Definition (Mixed Integer Linear Program)

Definition (Mixed Integer Linear Program)

- In a *Mixed Integer Linear Program* (MILP) not all variables are restricted to only integer values.

Definition (Mixed Integer Linear Program)

- In a *Mixed Integer Linear Program* (MILP) not all variables are restricted to only integer values.
- Thus both IPs and LPs are restircted forms of MILPs.

Definition (Mixed Integer Linear Program)

- In a *Mixed Integer Linear Program* (MILP) not all variables are restricted to only integer values.
- Thus both IPs and LPs are restircted forms of MILPs.
- An MILP will have the following form.

$$\max z = \mathbf{c} \cdot \mathbf{x}$$
$$\begin{aligned} \mathbf{A} \cdot \mathbf{x} &\leq \mathbf{b} \\ \mathbf{x} &\geq \mathbf{0} \end{aligned}$$

Definition (Mixed Integer Linear Program)

- In a *Mixed Integer Linear Program* (MILP) not all variables are restricted to only integer values.
- Thus both IPs and LPs are restircted forms of MILPs.
- An MILP will have the following form.

$$\begin{aligned} \max z = \mathbf{c} \cdot \mathbf{x} & \\ \mathbf{A} \cdot \mathbf{x} &\leq \mathbf{b} \\ \mathbf{x} &\geq \mathbf{0} \\ x_i &\in \mathbb{Z}, i = 1 \dots p \end{aligned}$$

Outline

Difference from Linear Programming

Difference from Linear Programming

Unlike LPs, IPs and MILPs allow us to model a disjunction of constraints.

Difference from Linear Programming

Unlike LPs, IPs and MILPs allow us to model a disjunction of constraints.

Example

Difference from Linear Programming

Unlike LPs, IPs and MILPs allow us to model a disjunction of constraints.

Example

In an IP we can model $x_1 \leq 0$ *or* $-x_1 \leq -5$.

Difference from Linear Programming

Unlike LPs, IPs and MILPs allow us to model a disjunction of constraints.

Example

In an IP we can model $x_1 \leq 0$ *or* $-x_1 \leq -5$.
To do this we introduce a new $(0, 1)$-variable x_2. The disjunction can now be expressed as:

$$\begin{aligned} x_1 - M \cdot x_2 &\leq 0 \\ -x_1 - M \cdot (1 - x_2) &\leq -5 \end{aligned}$$

Difference from Linear Programming

Unlike LPs, IPs and MILPs allow us to model a disjunction of constraints.

Example

In an IP we can model $x_1 \leq 0$ *or* $-x_1 \leq -5$.
To do this we introduce a new $(0, 1)$-variable x_2. The disjunction can now be expressed as:

$$\begin{aligned} x_1 - M \cdot x_2 &\leq 0 \\ -x_1 - M \cdot (1 - x_2) &\leq -5 \end{aligned}$$

This technique can also be used to model constraints like $x_1 \neq 4$.

Exercise

Exercise

Construct an IP which models the following problem.

Exercise

Construct an IP which models the following problem.

- We wish to invest $\$19,000$.

Exercise

Construct an IP which models the following problem.

- We wish to invest $\$19,000$.
- Investment 1 requires an investment of $\$6,700$ and has a net present value of $\$8,000$.

Exercise

Construct an IP which models the following problem.

- We wish to invest $\$19,000$.
- Investment 1 requires an investment of $\$6,700$ and has a net present value of $\$8,000$.
- Investment 2 requires $\$10,000$ and has a value of $\$11,000$.

Exercise

Construct an IP which models the following problem.

- We wish to invest $\$19,000$.
- Investment 1 requires an investment of $\$6,700$ and has a net present value of $\$8,000$.
- Investment 2 requires $\$10,000$ and has a value of $\$11,000$.
- Investment 3 requires $\$5,500$ and has a value of $\$6,000$.

Exercise

Construct an IP which models the following problem.

- We wish to invest $\$19,000$.
- Investment 1 requires an investment of $\$6,700$ and has a net present value of $\$8,000$.
- Investment 2 requires $\$10,000$ and has a value of $\$11,000$.
- Investment 3 requires $\$5,500$ and has a value of $\$6,000$.
- Investment 4 requires $\$3,400$ and has a value of $\$4,000$.

Solution

Solution

We get the following IP

$$\max 8 \cdot x_1 + 11 \cdot x_2 + 6 \cdot x_3 + 4 \cdot x_4$$

Solution

We get the following IP

$$\begin{aligned} \max 8 \cdot x_1 + 11 \cdot x_2 + 6 \cdot x_3 + 4 \cdot x_4 & & \\ 6.7 \cdot x_1 + 10 \cdot x_2 + 5.5 \cdot x_3 + 3.4 \cdot x_4 &\leq& 19 \\ x_1, x_2, x_3, x_4 &\in& \{0, 1\} \end{aligned}$$

Modeling Restrictions

Modeling Restrictions

We can also use integer programs to model restrictions on investment choices. For example:

Modeling Restrictions

We can also use integer programs to model restrictions on investment choices. For example:

- We can only make two investments.

Modeling Restrictions

We can also use integer programs to model restrictions on investment choices. For example:

- We can only make two investments.
- $x_1 + x_2 + x_3 + x_4 \leq 2$

Modeling Restrictions

We can also use integer programs to model restrictions on investment choices. For example:

- We can only make two investments.
- $x_1 + x_2 + x_3 + x_4 \leq 2$
- If we choose Investment 2, then we must also choose investment 4.

Modeling Restrictions

We can also use integer programs to model restrictions on investment choices. For example:

- We can only make two investments.
- $x_1 + x_2 + x_3 + x_4 \leq 2$
- If we choose Investment 2, then we must also choose investment 4.
- $x_2 - x_4 \leq 0$

Modeling Restrictions

We can also use integer programs to model restrictions on investment choices. For example:

- We can only make two investments.
- $x_1 + x_2 + x_3 + x_4 \leq 2$
- If we choose Investment 2, then we must also choose investment 4.
- $x_2 - x_4 \leq 0$
- We cannot choose both Investment 1 and Investment 3.

Modeling Restrictions

We can also use integer programs to model restrictions on investment choices. For example:

- We can only make two investments.
- $x_1 + x_2 + x_3 + x_4 \leq 2$
- If we choose Investment 2, then we must also choose investment 4.
- $x_2 - x_4 \leq 0$
- We cannot choose both Investment 1 and Investment 3.
- $x_1 + x_3 \leq 1$

Outline

Definition (LP Relaxation)

Definition (LP Relaxation)

When given an MILP we can *relax* it to an LP by removing all integrality requirements.

Definition (LP Relaxation)

When given an MILP we can *relax* it to an LP by removing all integrality requirements. Since the LP is less constrained than the original MILP we have the following.

Definition (LP Relaxation)

When given an MILP we can *relax* it to an LP by removing all integrality requirements. Since the LP is less constrained than the original MILP we have the following.

- The optimal solution to the LP provides us with an upper bound on the solution to the MILP.

Definition (LP Relaxation)

When given an MILP we can *relax* it to an LP by removing all integrality requirements. Since the LP is less constrained than the original MILP we have the following.

- The optimal solution to the LP provides us with an upper bound on the solution to the MILP.
- If the LP is infeasible, then so is the MILP.

Definition (LP Relaxation)

When given an MILP we can *relax* it to an LP by removing all integrality requirements. Since the LP is less constrained than the original MILP we have the following.

- The optimal solution to the LP provides us with an upper bound on the solution to the MILP.
- If the LP is infeasible, then so is the MILP.
- If the optimal solution to the LP has $x_i \in \mathbb{Z}$ for $i = 1 \dots p$, then it is also the optimal solution to the MILP.

Example

Example

Consider the following IP.

Example

Consider the following IP.

$$
\begin{aligned}
\max z = 20 \cdot x_1 + 10 \cdot x_2 + 10 \cdot x_3 & & \\
2 \cdot x_1 + 20 \cdot x_2 + 4 \cdot x_3 &\leq& 15 \\
6 \cdot x_1 + 20 \cdot x_2 + 4 \cdot x_3 &=& 20 \\
x_1, x_2, x_3 &\geq& 0 \\
x_1, x_2, x_3 &\in& \mathbb{Z}
\end{aligned}
$$

Example

Consider the following IP.

$$\begin{aligned} \max z = 20 \cdot x_1 + 10 \cdot x_2 + 10 \cdot x_3 & & \\ 2 \cdot x_1 + 20 \cdot x_2 + 4 \cdot x_3 & \leq & 15 \\ 6 \cdot x_1 + 20 \cdot x_2 + 4 \cdot x_3 & = & 20 \\ x_1, x_2, x_3 & \geq & 0 \\ x_1, x_2, x_3 & \in & \mathbb{Z} \end{aligned}$$

When we relax this to an LP we obtain the following as the optimal solution.

Example

Consider the following IP.

$$\begin{aligned} \max z = 20 \cdot x_1 + 10 \cdot x_2 + 10 \cdot x_3 & & \\ 2 \cdot x_1 + 20 \cdot x_2 + 4 \cdot x_3 & \leq & 15 \\ 6 \cdot x_1 + 20 \cdot x_2 + 4 \cdot x_3 & = & 20 \\ x_1, x_2, x_3 & \geq & 0 \\ x_1, x_2, x_3 & \in & \mathbb{Z} \end{aligned}$$

When we relax this to an LP we obtain the following as the optimal solution.

$$(x_1, x_2, x_3) = (3.\bar{3}, 0, 0) \quad z = 66.\bar{6}$$

Example

Consider the following IP.

$$\begin{aligned} \max z = 20 \cdot x_1 + 10 \cdot x_2 + 10 \cdot x_3 & & \\ 2 \cdot x_1 + 20 \cdot x_2 + 4 \cdot x_3 &\leq& 15 \\ 6 \cdot x_1 + 20 \cdot x_2 + 4 \cdot x_3 &=& 20 \\ x_1, x_2, x_3 &\geq& 0 \\ x_1, x_2, x_3 &\in& \mathbb{Z} \end{aligned}$$

When we relax this to an LP we obtain the following as the optimal solution.

$$(x_1, x_2, x_3) = (3.\bar{3}, 0, 0) \quad z = 66.\bar{6}$$

However when we consider the original IP we obtain the following as the optimal solution.

Example

Consider the following IP.

$$\begin{aligned} \max z = 20 \cdot x_1 + 10 \cdot x_2 + 10 \cdot x_3 & & \\ 2 \cdot x_1 + 20 \cdot x_2 + 4 \cdot x_3 &\leq& 15 \\ 6 \cdot x_1 + 20 \cdot x_2 + 4 \cdot x_3 &=& 20 \\ x_1, x_2, x_3 &\geq& 0 \\ x_1, x_2, x_3 &\in& \mathbb{Z} \end{aligned}$$

When we relax this to an LP we obtain the following as the optimal solution.

$$(x_1, x_2, x_3) = (3.\bar{3}, 0, 0) \quad z = 66.\bar{6}$$

However when we consider the original IP we obtain the following as the optimal solution.

$$(x_1, x_2, x_3) = (2, 0, 2) \quad z = 60$$

Outline

Definition (Branch and Bound)

Definition (Branch and Bound)

Branch and Bound is a technique for solving Integer Programs.

Definition (Branch and Bound)

Branch and Bound is a technique for solving Integer Programs.

- We first utilize LP Relaxation to obtain the linear optimum for the problem.

Definition (Branch and Bound)

Branch and Bound is a technique for solving Integer Programs.

- We first utilize LP Relaxation to obtain the linear optimum for the problem.
- We then *branch* by dividing the solution space.

Definition (Branch and Bound)

Branch and Bound is a technique for solving Integer Programs.

- We first utilize LP Relaxation to obtain the linear optimum for the problem.
- We then *branch* by dividing the solution space.
- A branch is *pruned* under the following conditions.

Definition (Branch and Bound)

Branch and Bound is a technique for solving Integer Programs.

- We first utilize LP Relaxation to obtain the linear optimum for the problem.
- We then *branch* by dividing the solution space.
- A branch is *pruned* under the following conditions.
  - The optimum solution to the branch is integral. (Pruning by *integrality*)

Definition (Branch and Bound)

Branch and Bound is a technique for solving Integer Programs.

- We first utilize LP Relaxation to obtain the linear optimum for the problem.
- We then *branch* by dividing the solution space.
- A branch is *pruned* under the following conditions.
  - The optimum solution to the branch is integral. (Pruning by *integrality*)
  - The solution space for the branch is empty. (Pruning by *infeasibility*)

Definition (Branch and Bound)

Branch and Bound is a technique for solving Integer Programs.

- We first utilize LP Relaxation to obtain the linear optimum for the problem.
- We then *branch* by dividing the solution space.
- A branch is *pruned* under the following conditions.
  - The optimum solution to the branch is integral. (Pruning by *integrality*)
  - The solution space for the branch is empty. (Pruning by *infeasibility*)
  - The optimum value for the branch is lower than that of a branch with an integer optimum. (Pruning by *bounds*)

Example

Example

Let us consider the Integer program described by System 1.

Example

Let us consider the Integer program described by System 1.

$$\begin{aligned} \max z = x_1 + x_2 & & \\ -x_1 + x_2 &\leq& 2 \\ 8 \cdot x_1 + 2 \cdot x_2 &\leq& 19 \\ x_1, x_2 &\geq& 0 \\ x_1, x_2 &\in& \mathbb{Z} \end{aligned} \tag{1}$$

Example

Example

Let us now look at this problem graphically.

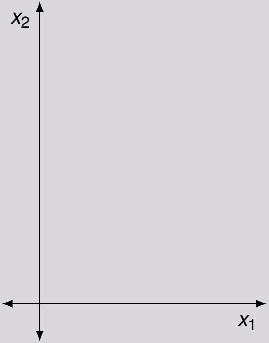

Figure: Graphical Solution

Example

Let us now look at this problem graphically.

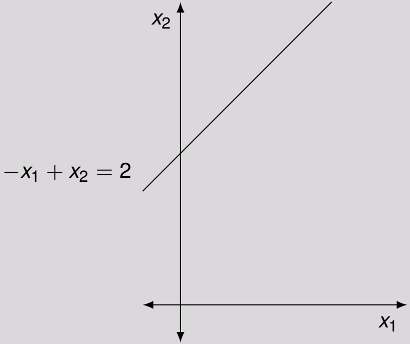

Figure: Graphical Solution

Example

Let us now look at this problem graphically.

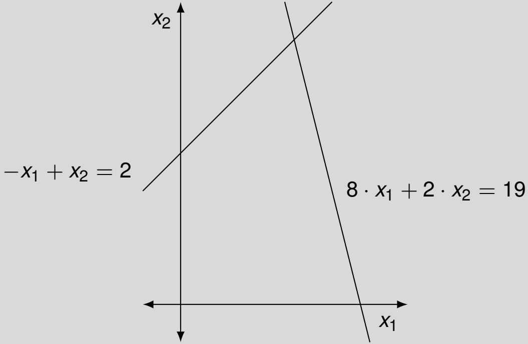

Figure: Graphical Solution

Example

Let us now look at this problem graphically.

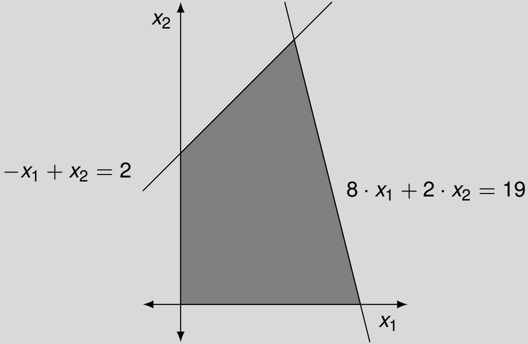

Figure: Graphical Solution

Example

Let us now look at this problem graphically.

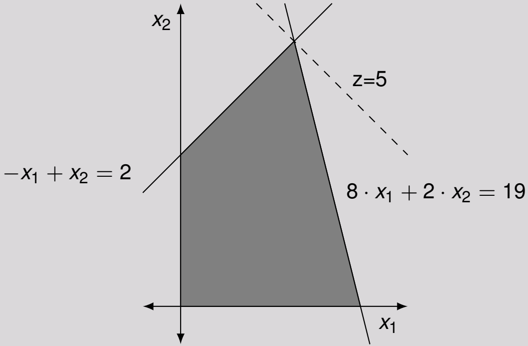

Figure: Graphical Solution

Example

Example

- We have that the optimal solution to System 1 is $(x_1, x_2) = (1.5, 3.5)$ which gives us $z = 5$.

Example

- We have that the optimal solution to System 1 is $(x_1, x_2) = (1.5, 3.5)$ which gives us $z = 5$.
- We can now branch on x_1.

Example

- We have that the optimal solution to System 1 is $(x_1, x_2) = (1.5, 3.5)$ which gives us $z = 5$.
- We can now branch on x_1.
- Any integer solution must have either $x_1 \leq 1$ or $x_1 \geq 2$.

Example

- We have that the optimal solution to System 1 is $(x_1, x_2) = (1.5, 3.5)$ which gives us $z = 5$.
- We can now branch on x_1.
- Any integer solution must have either $x_1 \leq 1$ or $x_1 \geq 2$.
- Thus, we can create two new systems of constraints:

Example

- We have that the optimal solution to System 1 is $(x_1, x_2) = (1.5, 3.5)$ which gives us $z = 5$.
- We can now branch on x_1.
- Any integer solution must have either $x_1 \leq 1$ or $x_1 \geq 2$.
- Thus, we can create two new systems of constraints:
  - one by adding $x_1 \leq 1$

Example

- We have that the optimal solution to System 1 is $(x_1, x_2) = (1.5, 3.5)$ which gives us $z = 5$.
- We can now branch on x_1.
- Any integer solution must have either $x_1 \leq 1$ or $x_1 \geq 2$.
- Thus, we can create two new systems of constraints:
  - one by adding $x_1 \leq 1$
  - one by adding $x_1 \geq 2$

Example

- We have that the optimal solution to System 1 is $(x_1, x_2) = (1.5, 3.5)$ which gives us $z = 5$.
- We can now branch on x_1.
- Any integer solution must have either $x_1 \leq 1$ or $x_1 \geq 2$.
- Thus, we can create two new systems of constraints:
  - one by adding $x_1 \leq 1$
  - one by adding $x_1 \geq 2$
- We can solve these branches graphically.

Example

Example

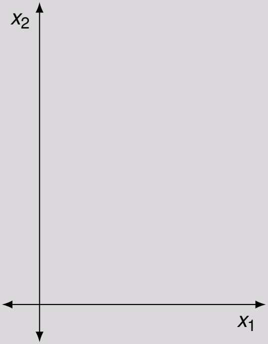

Figure: Graphical Solution

Example

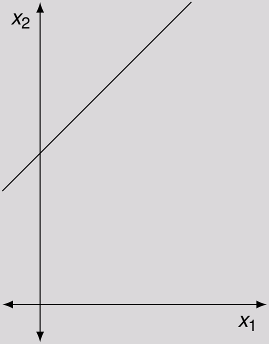

Figure: Graphical Solution

Example

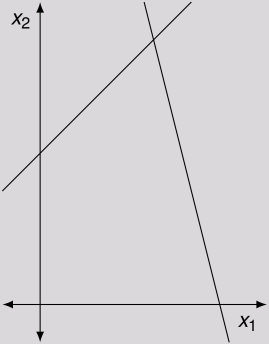

Figure: Graphical Solution

Example

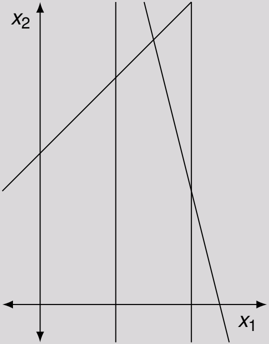

Figure: Graphical Solution

Example

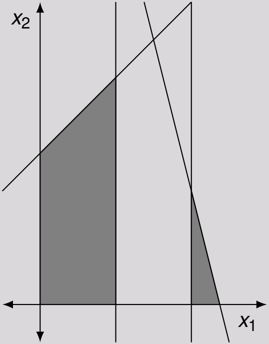

Figure: Graphical Solution

Example

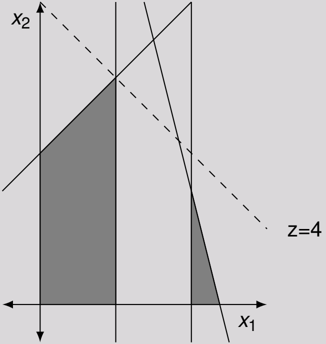

Figure: Graphical Solution

Example

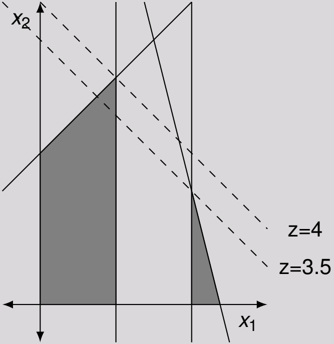

Figure: Graphical Solution

Example

Example

- We have that the optimal solution to System (1) is $(x_1, x_2) = (1.5, 3.5)$ which gave us $z = 5$.

Example

- We have that the optimal solution to System (1) is $(x_1, x_2) = (1.5, 3.5)$ which gave us $z = 5$.
- When we add the constraint $x_1 \leq 1$ we get that the optimum solution is $(x_1, x_2) = (1, 3)$ which gives us $z = 4$.

Example

- We have that the optimal solution to System (1) is $(x_1, x_2) = (1.5, 3.5)$ which gave us $z = 5$.
- When we add the constraint $x_1 \leq 1$ we get that the optimum solution is $(x_1, x_2) = (1, 3)$ which gives us $z = 4$.
  - Since the optimum solution is integeral we prune this branch.

Example

- We have that the optimal solution to System (1) is $(x_1, x_2) = (1.5, 3.5)$ which gave us $z = 5$.
- When we add the constraint $x_1 \leq 1$ we get that the optimum solution is $(x_1, x_2) = (1, 3)$ which gives us $z = 4$.
  - Since the optimum solution is integeral we prune this branch.
- When we add the constraint $x_1 \geq 2$ we get that the optimum solution is $(x_1, x_2) = (2, 1.5)$ which gives us $z = 3.5$.

Example

- We have that the optimal solution to System (1) is $(x_1, x_2) = (1.5, 3.5)$ which gave us $z = 5$.
- When we add the constraint $x_1 \leq 1$ we get that the optimum solution is $(x_1, x_2) = (1, 3)$ which gives us $z = 4$.
  - Since the optimum solution is integeral we prune this branch.
- When we add the constraint $x_1 \geq 2$ we get that the optimum solution is $(x_1, x_2) = (2, 1.5)$ which gives us $z = 3.5$.
  - Since the optimum solution lower than a known integral solution we also prune this branch.

Example

- We have that the optimal solution to System (1) is $(x_1, x_2) = (1.5, 3.5)$ which gave us $z = 5$.
- When we add the constraint $x_1 \leq 1$ we get that the optimum solution is $(x_1, x_2) = (1, 3)$ which gives us $z = 4$.
  - Since the optimum solution is integeral we prune this branch.
- When we add the constraint $x_1 \geq 2$ we get that the optimum solution is $(x_1, x_2) = (2, 1.5)$ which gives us $z = 3.5$.
  - Since the optimum solution lower than a known integral solution we also prune this branch.
- All branches are now pruned and we have that the optimum integer solution is $(x_1, x_2) = (1, 3)$ which gives us $z = 4$.

Example

Example

We can consider the LPs solved as part of branch and bound as forming a binary tree. For System (1) this tree is:

Example

We can consider the LPs solved as part of branch and bound as forming a binary tree. For System (1) this tree is:

$$(x_1, x_2) = (1.5, 3.5)$$

$$z = 5$$

Figure: Branching Tree

Example

We can consider the LPs solved as part of branch and bound as forming a binary tree. For System (1) this tree is:

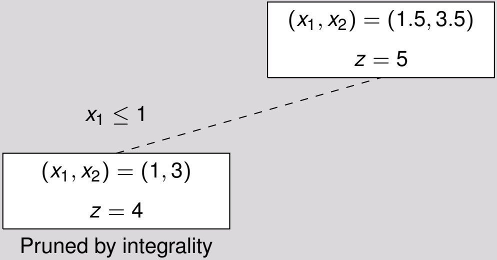

Figure: Branching Tree

Example

We can consider the LPs solved as part of branch and bound as forming a binary tree. For System (1) this tree is:

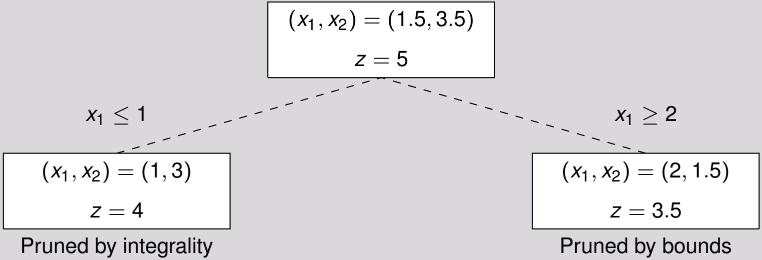

Figure: Branching Tree

Exercise

Exercise

Perform Branch and bound to find the solution to the following IP.

Exercise

Perform Branch and bound to find the solution to the following IP.

$$\max z = 3 \cdot x_1 + x_2$$

$$\begin{aligned} -x_1 + x_2 &\leq 2 \\ 8 \cdot x_1 + 2 \cdot x_2 &\leq 19 \\ x_1, x_2 &\geq 0 \\ x_1, x_2 &\in \mathbb{Z} \end{aligned}$$

Solution

Solution

$$(x_1, x_2) = (1.5, 3.5)$$

$$z = 8$$

Figure: Branching Tree

Solution

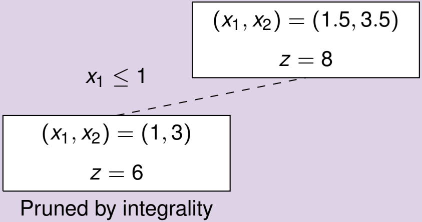

Figure: Branching Tree

Solution

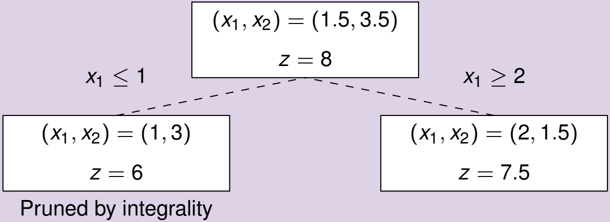

Figure: Branching Tree

Solution

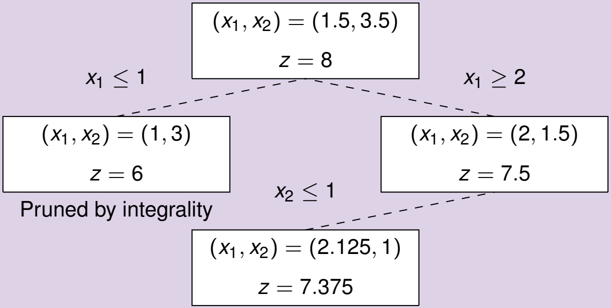

Figure: Branching Tree

Solution

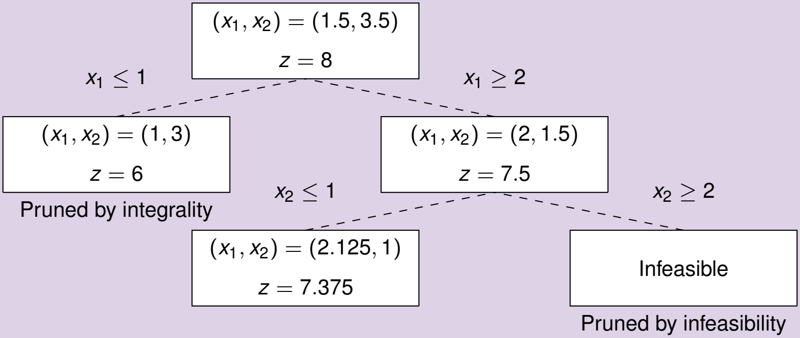

Figure: Branching Tree

Solution

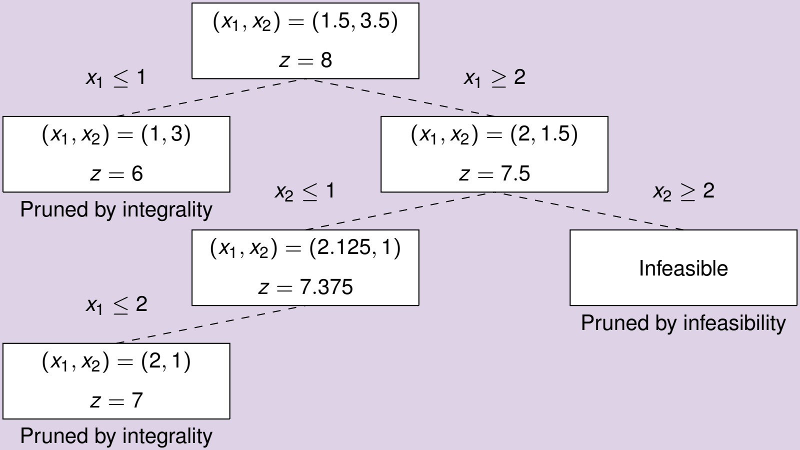

Figure: Branching Tree

Solution

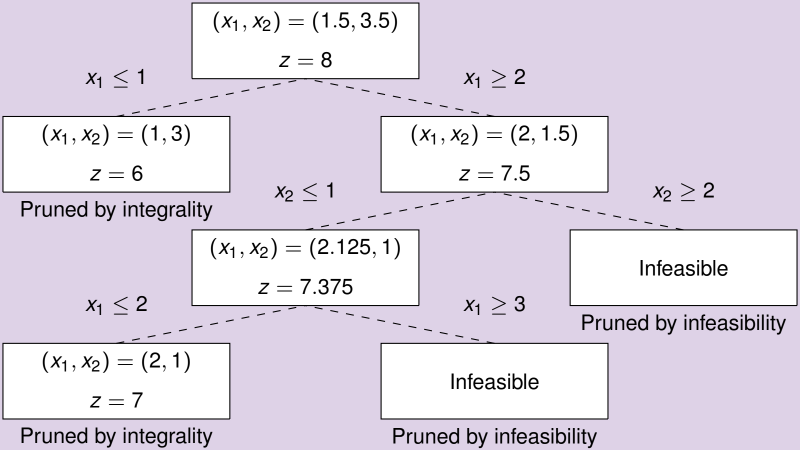

Figure: Branching Tree

Outline

Definition (Cutting Planes)

Definition (Cutting Planes)

- When relaxing an MILP to an LP we can add additional constraints.

Definition (Cutting Planes)

- When relaxing an MILP to an LP we can add additional constraints.
- These added constraints are designed to be satisfied by every solution to the MILP.

Definition (Cutting Planes)

- When relaxing an MILP to an LP we can add additional constraints.
- These added constraints are designed to be satisfied by every solution to the MILP.
- However, they can still eliminate possible solutions to the relaxed LP.

Definition (Cutting Planes)

- When relaxing an MILP to an LP we can add additional constraints.
- These added constraints are designed to be satisfied by every solution to the MILP.
- However, they can still eliminate possible solutions to the relaxed LP.
- Such additional constraints serve to strengthen the LP relaxation and are called *Cutting Planes*.

Definition (Cutting Planes)

- When relaxing an MILP to an LP we can add additional constraints.
- These added constraints are designed to be satisfied by every solution to the MILP.
- However, they can still eliminate possible solutions to the relaxed LP.
- Such additional constraints serve to strengthen the LP relaxation and are called *Cutting Planes*.
  - The optimal solution to the strengthened LP provides us with an upper bound on the solution to the MILP.

Definition (Cutting Planes)

- When relaxing an MILP to an LP we can add additional constraints.
- These added constraints are designed to be satisfied by every solution to the MILP.
- However, they can still eliminate possible solutions to the relaxed LP.
- Such additional constraints serve to strengthen the LP relaxation and are called *Cutting Planes*.
    - The optimal solution to the strengthened LP provides us with an upper bound on the solution to the MILP.
    - If the strengthened LP is infeasible, then so is the MILP.

Definition (Cutting Planes)

- When relaxing an MILP to an LP we can add additional constraints.
- These added constraints are designed to be satisfied by every solution to the MILP.
- However, they can still eliminate possible solutions to the relaxed LP.
- Such additional constraints serve to strengthen the LP relaxation and are called *Cutting Planes*.
  - The optimal solution to the strengthened LP provides us with an upper bound on the solution to the MILP.
  - If the strengthened LP is infeasible, then so is the MILP.
  - If the optimal solution to the strengthened LP has $x_i \in \mathbb{Z}$ for $i = 1 \ldots p$, then it is also the optimal solution to the MILP.

Generating Cutting Planes

Generating Cutting Planes

Let Constraint (2) be satisfied by the solutions to an MILP with $b \notin \mathbb{Z}$.

Generating Cutting Planes

Let Constraint (2) be satisfied by the solutions to an MILP with $b \notin \mathbb{Z}$.

$$\sum_{i=1}^{p} a_i \cdot x_i + \sum_{i=p+1}^{n} a_i \cdot x_i = b \tag{2}$$

Generating Cutting Planes

Let Constraint (2) be satisfied by the solutions to an MILP with $b \notin \mathbb{Z}$.

$$\sum_{i=1}^{p} a_i \cdot x_i + \sum_{i=p+1}^{n} a_i \cdot x_i = b \tag{2}$$

Let $f_0 = b - \lfloor b \rfloor$, and let $f_i = a_i - \lfloor a_i \rfloor$ for $i = 1 \dots p$.

Generating Cutting Planes

Let Constraint (2) be satisfied by the solutions to an MILP with $b \notin \mathbb{Z}$.

$$\sum_{i=1}^{p} a_i \cdot x_i + \sum_{i=p+1}^{n} a_i \cdot x_i = b \qquad (2)$$

Let $f_0 = b - \lfloor b \rfloor$, and let $f_i = a_i - \lfloor a_i \rfloor$ for $i = 1 \ldots p$.
We can now rewrite Constraint (2) as

$$\sum_{i \leq p, f_i \leq f_0} f_i \cdot x_i + \sum_{i \leq p, f_i > f_0} (f_i - 1) \cdot x_i + \sum_{i=p+1}^{n} a_i = k + f_0.$$

Where $k \in \mathbb{Z}$.

Generating Cutting Planes

Generating Cutting Planes

If $k \geq 0$ then Constraint (2) implies that

$$\sum_{i \leq p, f_i \leq f_0} \frac{f_i}{f_0} \cdot x_i - \sum_{i \leq p, f_i > f_0} \frac{1 - f_i}{f_0} \cdot x_i + \sum_{i=p+1}^{n} \frac{a_i}{f_0} \geq 1.$$

Generating Cutting Planes

If $k \geq 0$ then Constraint (2) implies that

$$\sum_{i \leq p, f_i \leq f_0} \frac{f_i}{f_0} \cdot x_i - \sum_{i \leq p, f_i > f_0} \frac{1 - f_i}{f_0} \cdot x_i + \sum_{i=p+1}^{n} \frac{a_i}{f_0} \geq 1.$$

If $k \leq -1$ then Constraint (2) implies that

$$-\sum_{i \leq p, f_i \leq f_0} \frac{f_i}{1 - f_0} \cdot x_i + \sum_{i \leq p, f_i > f_0} \frac{1 - f_i}{1 - f_0} \cdot x_i - \sum_{i=p+1}^{n} \frac{a_i}{1 - f_0} \geq 1.$$

Generating Cutting Planes

If $k \geq 0$ then Constraint (2) implies that

$$\sum_{i \leq p, f_i \leq f_0} \frac{f_i}{f_0} \cdot x_i - \sum_{i \leq p, f_i > f_0} \frac{1 - f_i}{f_0} \cdot x_i + \sum_{i=p+1}^{n} \frac{a_i}{f_0} \geq 1.$$

If $k \leq -1$ then Constraint (2) implies that

$$-\sum_{i \leq p, f_i \leq f_0} \frac{f_i}{1 - f_0} \cdot x_i + \sum_{i \leq p, f_i > f_0} \frac{1 - f_i}{1 - f_0} \cdot x_i - \sum_{i=p+1}^{n} \frac{a_i}{1 - f_0} \geq 1.$$

Since each x_i is non negative we will always have that.

$$\sum_{i \leq p, f_i \leq f_0} \frac{f_i}{f_0} \cdot x_i + \sum_{i \leq p, f_i > f_0} \frac{1 - f_i}{1 - f_0} \cdot x_i + \sum_{i > p, a_i > 0} \frac{a_i}{f_0} - \sum_{i > p, a_i < 0} \frac{a_i}{1 - f_0} \geq 1.$$

Generating Cutting Planes

If $k \geq 0$ then Constraint (2) implies that

$$\sum_{i \leq p, f_i \leq f_0} \frac{f_i}{f_0} \cdot x_i - \sum_{i \leq p, f_i > f_0} \frac{1 - f_i}{f_0} \cdot x_i + \sum_{i=p+1}^{n} \frac{a_i}{f_0} \geq 1.$$

If $k \leq -1$ then Constraint (2) implies that

$$-\sum_{i \leq p, f_i \leq f_0} \frac{f_i}{1 - f_0} \cdot x_i + \sum_{i \leq p, f_i > f_0} \frac{1 - f_i}{1 - f_0} \cdot x_i - \sum_{i=p+1}^{n} \frac{a_i}{1 - f_0} \geq 1.$$

Since each x_i is non negative we will always have that.

$$\sum_{i \leq p, f_i \leq f_0} \frac{f_i}{f_0} \cdot x_i + \sum_{i \leq p, f_i > f_0} \frac{1 - f_i}{1 - f_0} \cdot x_i + \sum_{i > p, a_i > 0} \frac{a_i}{f_0} - \sum_{i > p, a_i < 0} \frac{a_i}{1 - f_0} \geq 1.$$

This constraint is known as the *Gomory mixed integer cut*.

Example

Example

Let us return to System (1).

Example

Let us return to System (1).

Adding slack and surplus variables yields the system

Example

Let us return to System (1).

Adding slack and surplus variables yields the system

$$
\begin{aligned}
\max z = x_1 + x_2 & & \\
-x_1 + x_2 + x_3 &= 2 \\
8 \cdot x_1 + 2 \cdot x_2 + x_4 &= 19 \\
x_1, x_2, x_3, x_4 &\geq 0 \\
x_1, x_2, x_3, x_4 &\in \mathbb{Z}
\end{aligned}
$$

Example

Let us return to System (1).

Adding slack and surplus variables yields the system

$$\begin{aligned} \max z = x_1 + x_2 & & \\ -x_1 + x_2 + x_3 &= 2 \\ 8 \cdot x_1 + 2 \cdot x_2 + x_4 &= 19 \\ x_1, x_2, x_3, x_4 &\geq 0 \\ x_1, x_2, x_3, x_4 &\in \mathbb{Z} \end{aligned}$$

Using the simplex method we get the following at the linear optimum,

Example

Let us return to System (1).
Adding slack and surplus variables yields the system

$$\max z = x_1 + x_2$$
$$\begin{aligned} -x_1 + x_2 + x_3 &= 2 \\ 8 \cdot x_1 + 2 \cdot x_2 + x_4 &= 19 \\ x_1, x_2, x_3, x_4 &\geq 0 \\ x_1, x_2, x_3, x_4 &\in \mathbb{Z} \end{aligned}$$

Using the simplex method we get the following at the linear optimum,

$$\begin{aligned} z &= 5 - 0.6 \cdot x_3 - 0.2 \cdot x_4 \\ x_1 &= 1.5 + 0.2 \cdot x_3 - 0.1 \cdot x_4 \\ x_2 &= 3.5 - 0.8 \cdot x_3 - 0.1 \cdot x_4 \end{aligned}$$

Example

Let us return to System (1).

Adding slack and surplus variables yields the system

$$\begin{aligned} \max z = x_1 + x_2 & & \\ -x_1 + x_2 + x_3 &= 2 \\ 8 \cdot x_1 + 2 \cdot x_2 + x_4 &= 19 \\ x_1, x_2, x_3, x_4 &\geq 0 \\ x_1, x_2, x_3, x_4 &\in \mathbb{Z} \end{aligned}$$

Using the simplex method we get the following at the linear optimum,

$$\begin{aligned} z &= 5 - 0.6 \cdot x_3 - 0.2 \cdot x_4 \\ x_1 &= 1.5 + 0.2 \cdot x_3 - 0.1 \cdot x_4 \\ x_2 &= 3.5 - 0.8 \cdot x_3 - 0.1 \cdot x_4 \end{aligned}$$

We can now use the constraint $x_2 + 0.8 \cdot x_3 + 0.1 \cdot x_4 = 3.5$ to generate a Gomory mixed integer cut.

Example

Example

Use the constraint $x_2 + 0.8 \cdot x_3 + 0.1 \cdot x_4 = 3.5$ to generate a Gomory mixed integer cut we get the constraint

$$\frac{1-0.8}{1-0.5} \cdot x_3 + \frac{0.1}{0.5} \cdot x_4 \geq 1.$$

Example

Use the constraint $x_2 + 0.8 \cdot x_3 + 0.1 \cdot x_4 = 3.5$ to generate a Gomory mixed integer cut we get the constraint

$$\frac{1-0.8}{1-0.5} \cdot x_3 + \frac{0.1}{0.5} \cdot x_4 \geq 1.$$

This is equivalent to the constraint

$$2 \cdot x_3 + x_4 \geq 5.$$

Example

Use the constraint $x_2 + 0.8 \cdot x_3 + 0.1 \cdot x_4 = 3.5$ to generate a Gomory mixed integer cut we get the constraint

$$\frac{1 - 0.8}{1 - 0.5} \cdot x_3 + \frac{0.1}{0.5} \cdot x_4 \geq 1.$$

This is equivalent to the constraint

$$2 \cdot x_3 + x_4 \geq 5.$$

Since $x_3 = 2 + x_1 - x_2$ and $x_4 = 19 - 8 \cdot x_1 - 2 \cdot x_2$ this constraint is equivalent to

$$3 \cdot x_1 + 2 \cdot x_2 \geq 9.$$

Example

Example

Let us now look at this problem graphically.

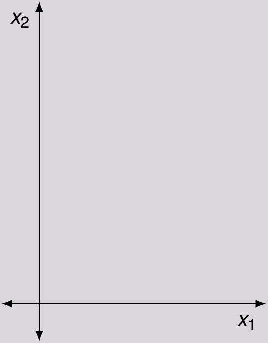

Figure: Graphical Solution

Example

Let us now look at this problem graphically.

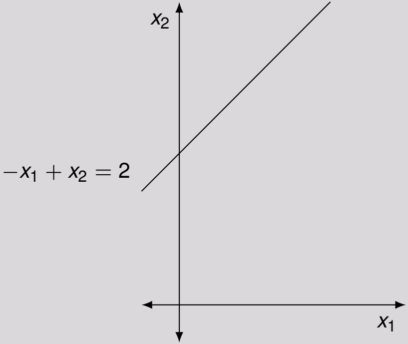

Figure: Graphical Solution

Example

Let us now look at this problem graphically.

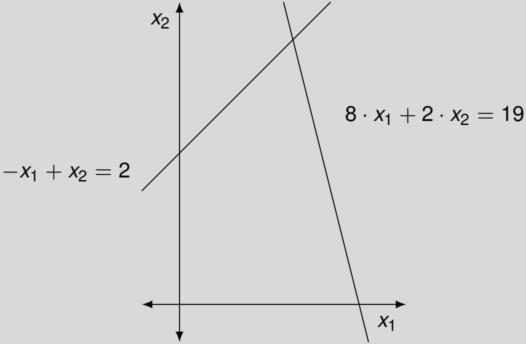

Figure: Graphical Solution

Example

Let us now look at this problem graphically.

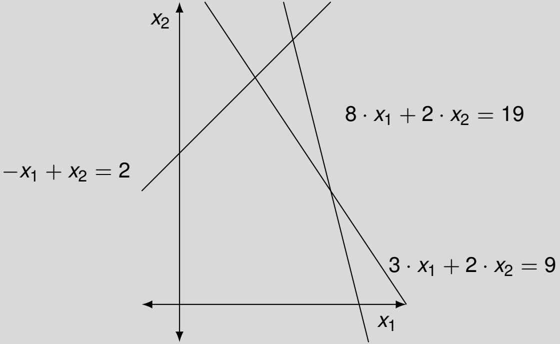

Figure: Graphical Solution

Example

Let us now look at this problem graphically.

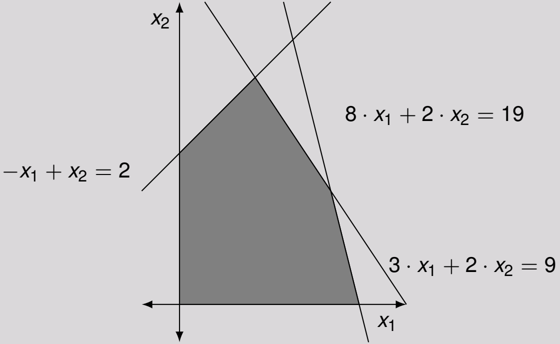

Figure: Graphical Solution

Example

Let us now look at this problem graphically.

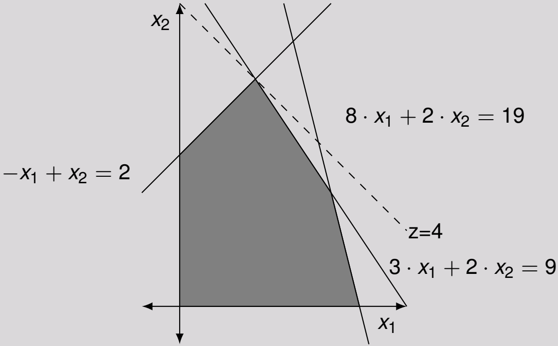

Figure: Graphical Solution

Outline

Definition (Branch and Cut)

Definition (Branch and Cut)

- *Branch an Cut* is a combination of branch and bound and cutting planes.

Definition (Branch and Cut)

- *Branch an Cut* is a combination of branch and bound and cutting planes.
- This method proceeds just like regular branch and bound except that at each branch the LP is strengthened using cutting planes.