Algorithm Foundations of Data Science and Engineering

Lecture 9: Integer Programming

MING GAO

DaSE @ ECNU
(for course related communications)
mgao@dase.ecnu.edu.cn

May 20, 2019

Outline

Combinatorial optimization problem

- **Input:** A description of the data for an instance of the problem;

Combinatorial optimization problem

- **Input:** A description of the data for an instance of the problem;
- **Feasible solutions:** there is a way of determining from the input whether a given solution x' (assignment of values to decision variables) is feasible.

Combinatorial optimization problem

- **Input:** A description of the data for an instance of the problem;
- **Feasible solutions:** there is a way of determining from the input whether a given solution x' (assignment of values to decision variables) is feasible. Typically in combinatorial optimization problems there is a finite number of possible solutions.

Combinatorial optimization problem

- **Input:** A description of the data for an instance of the problem;
- **Feasible solutions:** there is a way of determining from the input whether a given solution x' (assignment of values to decision variables) is feasible. Typically in combinatorial optimization problems there is a finite number of possible solutions.
- **Objective function:** For each feasible solution x' there is an associated objective $f(x')$.

Combinatorial optimization problem

- **Input:** A description of the data for an instance of the problem;
- **Feasible solutions:** there is a way of determining from the input whether a given solution x' (assignment of values to decision variables) is feasible. Typically in combinatorial optimization problems there is a finite number of possible solutions.
- **Objective function:** For each feasible solution x' there is an associated objective $f(x')$.
- Optimization problem
  - **Maximization:** Find a feasible solution x that maximizes $f(\cdot)$ among all feasible solutions;

Combinatorial optimization problem

- **Input:** A description of the data for an instance of the problem;
- **Feasible solutions:** there is a way of determining from the input whether a given solution x' (assignment of values to decision variables) is feasible. Typically in combinatorial optimization problems there is a finite number of possible solutions.
- **Objective function:** For each feasible solution x' there is an associated objective $f(x')$.
- Optimization problem
  - **Maximization:** Find a feasible solution x that maximizes $f(\cdot)$ among all feasible solutions;
  - **Minimization:** Find a feasible solution x that minimizes $f(\cdot)$ among all feasible solutions.

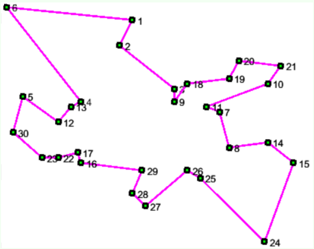
Outline

Traveling salesman problem

Given a set of cities and the cost of travel (or distance) between each possible pairs, the traveling salesman problem (TSP), is to find the best possible way of visiting all the cities and returning to the starting point that minimize the travel cost.

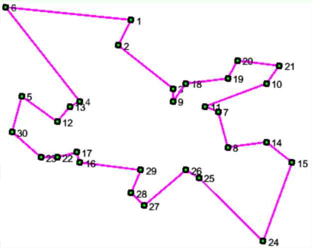
Traveling salesman problem

Given a set of cities and the cost of travel (or distance) between each possible pairs, the traveling salesman problem (TSP), is to find the best possible way of visiting all the cities and returning to the starting point that minimize the travel cost.



Traveling salesman problem

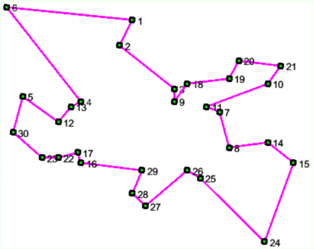
Given a set of cities and the cost of travel (or distance) between each possible pairs, the traveling salesman problem (TSP), is to find the best possible way of visiting all the cities and returning to the starting point that minimize the travel cost.



- **Input:** a set of n points;

Traveling salesman problem

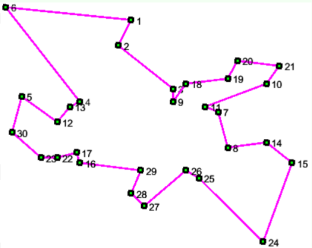
Given a set of cities and the cost of travel (or distance) between each possible pairs, the traveling salesman problem (TSP), is to find the best possible way of visiting all the cities and returning to the starting point that minimize the travel cost.



- **Input:** a set of n points;
- **Feasible solutions:** a tour that passes through each point exactly once, the possible feasible solutions is given as $\frac{(n-1)!}{2}$ for symmetric TSP.

Traveling salesman problem

Given a set of cities and the cost of travel (or distance) between each possible pairs, the traveling salesman problem (TSP), is to find the best possible way of visiting all the cities and returning to the starting point that minimize the travel cost.



- **Input:** a set of n points;
- **Feasible solutions:** a tour that passes through each point exactly once, the possible feasible solutions is given as $\frac{(n-1)!}{2}$ for symmetric TSP.
- **Objective function:** minimize the length of the tour.

It can be applied into scheduling problem, vehicle routing, aircraft routing, etc.

TSP formulation

The TSP can be defined on an undirected graph $G = (V, E)$ if it is symmetric (directed VS. asymmetric), $V = \{1, \cdots, n\}$ is the vertex set, $E \subset V \times V$ is an edge set, and a cost matrix C_{ij} is defined on E.

TSP formulation

The TSP can be defined on an undirected graph $G = (V, E)$ if it is symmetric (directed VS. asymmetric), $V = \{1, \cdots, n\}$ is the vertex set, $E \subset V \times V$ is an edge set, and a cost matrix C_{ij} is defined on E.

We define $x_{ij} = \begin{cases} 1, & \text{edge } e_{ij} \text{ appears in the optimal tour;} \\ 0, & \text{otherwise.} \end{cases}$

TSP formulation

The TSP can be defined on an undirected graph $G = (V, E)$ if it is symmetric (directed VS. asymmetric), $V = \{1, \cdots, n\}$ is the vertex set, $E \subset V \times V$ is an edge set, and a cost matrix C_{ij} is defined on E.

We define $x_{ij} = \begin{cases} 1, & \text{edge } e_{ij} \text{ appears in the optimal tour;} \\ 0, & \text{otherwise.} \end{cases}$

The TSP is formulated as

Minimize: $\sum_{i \neq j} c_{ij} x_{ij}$

TSP formulation

The TSP can be defined on an undirected graph $G = (V, E)$ if it is symmetric (directed VS. asymmetric), $V = \{1, \cdots, n\}$ is the vertex set, $E \subset V \times V$ is an edge set, and a cost matrix C_{ij} is defined on E.

We define $x_{ij} = \begin{cases} 1, & \text{edge } e_{ij} \text{ appears in the optimal tour;} \\ 0, & \text{otherwise.} \end{cases}$

The TSP is formulated as

Minimize: $\sum_{i \neq j} c_{ij} x_{ij}$

Subject to: $\sum_{j=1}^{n} x_{ij} = 1 \qquad (i \in V, i \neq j)$

TSP formulation

The TSP can be defined on an undirected graph $G = (V, E)$ if it is symmetric (directed VS. asymmetric), $V = \{1, \cdots, n\}$ is the vertex set, $E \subset V \times V$ is an edge set, and a cost matrix C_{ij} is defined on E.

We define $x_{ij} = \begin{cases} 1, & \text{edge } e_{ij} \text{ appears in the optimal tour;} \\ 0, & \text{otherwise.} \end{cases}$

The TSP is formulated as

Minimize: $\sum_{i \neq j} c_{ij} x_{ij}$

Subject to: $\sum_{j=1}^{n} x_{ij} = 1 \qquad (i \in V, i \neq j)$

$\sum_{i=1}^{n} x_{ij} = 1 \qquad (j \in V, j \neq i)$

TSP formulation

The TSP can be defined on an undirected graph $G = (V, E)$ if it is symmetric (directed VS. asymmetric), $V = \{1, \cdots, n\}$ is the vertex set, $E \subset V \times V$ is an edge set, and a cost matrix C_{ij} is defined on E.

We define $x_{ij} = \begin{cases} 1, & \text{edge } e_{ij} \text{ appears in the optimal tour;} \\ 0, & \text{otherwise.} \end{cases}$

The TSP is formulated as

Minimize: $\sum_{i \neq j} c_{ij} x_{ij}$

Subject to:

$$\begin{aligned} &\textstyle\sum_{j=1}^{n} x_{ij} = 1 && (i \in V, i \neq j) \\ &\textstyle\sum_{i=1}^{n} x_{ij} = 1 && (j \in V, j \neq i) \\ &\textstyle\sum_{i,j \in S} x_{ij} \leq |S| - 1 && (S \subset V, 2 \leq |S| \leq n-2) \end{aligned}$$

TSP formulation

The TSP can be defined on an undirected graph $G = (V, E)$ if it is symmetric (directed VS. asymmetric), $V = \{1, \cdots, n\}$ is the vertex set, $E \subset V \times V$ is an edge set, and a cost matrix C_{ij} is defined on E.

We define $x_{ij} = \begin{cases} 1, & \text{edge } e_{ij} \text{ appears in the optimal tour;} \\ 0, & \text{otherwise.} \end{cases}$

The TSP is formulated as

$$\begin{array}{ll} \text{Minimize:} & \sum_{i \neq j} c_{ij} x_{ij} \\ & \\ \text{Subject to:} & \sum_{j=1}^{n} x_{ij} = 1 \qquad (i \in V, i \neq j) \\ & \sum_{i=1}^{n} x_{ij} = 1 \qquad (j \in V, j \neq i) \\ & \sum_{i,j \in S} x_{ij} \leq |S| - 1 \qquad (S \subset V, 2 \leq |S| \leq n-2) \\ & x_{ij} \in \{0, 1\} \qquad (i, j \in V) \end{array}$$

Set covering problem: SCP

Input: Universe set $U = \{u_1, u_2, \cdots, u_n\}$
Subsets $S = \{s_i | s_i \subset U, 1 \leq i \leq m\}$
Costs $C = \{c_1, c_2, \cdots, c_m\}$

Set covering problem: SCP

Input: Universe set $U = \{u_1, u_2, \cdots, u_n\}$
Subsets $S = \{s_i | s_i \subset U, 1 \leq i \leq m\}$
Costs $C = \{c_1, c_2, \cdots, c_m\}$

Goal: Find a set $I \subset \{1, 2, \cdots, m\}$ that minimizes $\sum_{i \in I} c_i$,
s.t., $\bigcup_{i \in I} s_i = U$.

Set covering problem: SCP

Input: Universe set $U = \{u_1, u_2, \cdots, u_n\}$
Subsets $S = \{s_i | s_i \subset U, 1 \leq i \leq m\}$
Costs $C = \{c_1, c_2, \cdots, c_m\}$

Goal: Find a set $I \subset \{1, 2, \cdots, m\}$ that minimizes $\sum_{i \in I} c_i$,
s.t., $\bigcup_{i \in I} s_i = U$.

- You must select a minimum number of these subsets s.t. the sets you have picked contain all the elements that are contained in any of the sets in the input;

Set covering problem: SCP

Input: Universe set $U = \{u_1, u_2, \cdots, u_n\}$
Subsets $S = \{s_i | s_i \subset U, 1 \leq i \leq m\}$
Costs $C = \{c_1, c_2, \cdots, c_m\}$

Goal: Find a set $I \subset \{1, 2, \cdots, m\}$ that minimizes $\sum_{i \in I} c_i$, s.t., $\bigcup_{i \in I} s_i = U$.

- You must select a minimum number of these subsets s.t. the sets you have picked contain all the elements that are contained in any of the sets in the input;
- It was one of Karp's NP-complete problems.

Set covering problem: SCP

Input: Universe set $U = \{u_1, u_2, \cdots, u_n\}$
Subsets $S = \{s_i | s_i \subset U, 1 \le i \le m\}$
Costs $C = \{c_1, c_2, \cdots, c_m\}$

Goal: Find a set $I \subset \{1, 2, \cdots, m\}$ that minimizes $\sum_{i \in I} c_i$, s.t., $\bigcup_{i \in I} s_i = U$.

- You must select a minimum number of these subsets s.t. the sets you have picked contain all the elements that are contained in any of the sets in the input;
- It was one of Karp's NP-complete problems.
- It can be applied in the edge covering, vertex covering, text summarization, etc.

Set covering problem: SCP

Input: Universe set $U = \{u_1, u_2, \cdots, u_n\}$
Subsets $S = \{s_i | s_i \subset U, 1 \le i \le m\}$
Costs $C = \{c_1, c_2, \cdots, c_m\}$

Set covering problem: SCP

Input: Universe set $U = \{u_1, u_2, \cdots, u_n\}$
Subsets $S = \{s_i | s_i \subset U, 1 \leq i \leq m\}$
Costs $C = \{c_1, c_2, \cdots, c_m\}$

Goal: Find a set $I \subset \{1, 2, \cdots, m\}$ that minimizes $\sum_{i \in I} c_i$,
s.t., $\bigcup_{i \in I} s_i = U$.

Set covering problem: SCP

Input: Universe set $U = \{u_1, u_2, \cdots, u_n\}$
Subsets $S = \{s_i | s_i \subset U, 1 \le i \le m\}$
Costs $C = \{c_1, c_2, \cdots, c_m\}$

Goal: Find a set $I \subset \{1, 2, \cdots, m\}$ that minimizes $\sum_{i \in I} c_i$,
s.t., $\bigcup_{i \in I} s_i = U$.

- You must select a minimum number of these subsets s.t. the sets you have picked contain all the elements that are contained in any of the sets in the input;

Set covering problem: SCP

Input: Universe set $U = \{u_1, u_2, \cdots, u_n\}$
Subsets $S = \{s_i | s_i \subset U, 1 \leq i \leq m\}$
Costs $C = \{c_1, c_2, \cdots, c_m\}$

Goal: Find a set $I \subset \{1, 2, \cdots, m\}$ that minimizes $\sum_{i \in I} c_i$,
s.t., $\bigcup_{i \in I} s_i = U$.

- You must select a minimum number of these subsets s.t. the sets you have picked contain all the elements that are contained in any of the sets in the input;
- It was one of Karp's NP-complete problems.

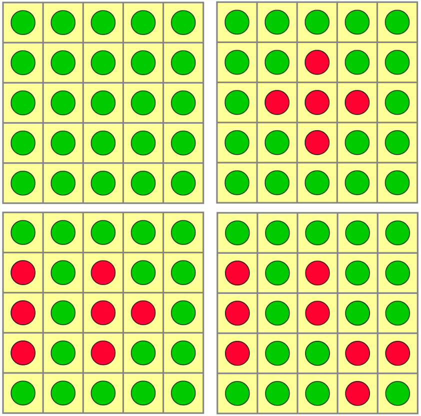
Set covering problem: SCP

Input: Universe set $U = \{u_1, u_2, \cdots, u_n\}$
Subsets $S = \{s_i | s_i \subset U, 1 \leq i \leq m\}$
Costs $C = \{c_1, c_2, \cdots, c_m\}$

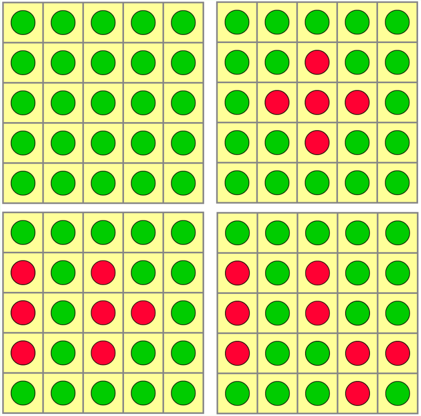
Goal: Find a set $I \subset \{1, 2, \cdots, m\}$ that minimizes $\sum_{i \in I} c_i$,
s.t., $\bigcup_{i \in I} s_i = U$.

- You must select a minimum number of these subsets s.t. the sets you have picked contain all the elements that are contained in any of the sets in the input;
- It was one of Karp's NP-complete problems.
- It can be applied in the edge covering, vertex covering, text summarization, etc.

A game of fiver

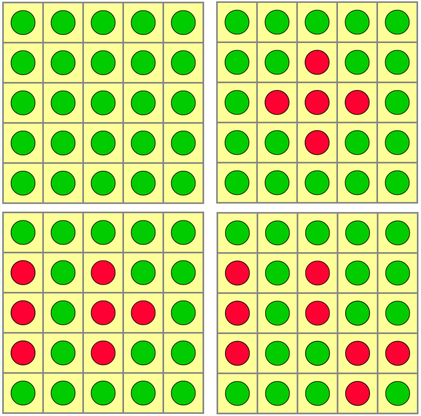


A game of fiver



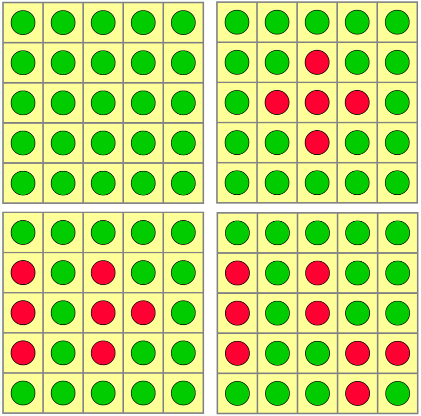
- Click on a circle, and flip its color and that of adjacent colors;

A game of fiver



- Click on a circle, and flip its color and that of adjacent colors;
- Can you make all of the circles red?

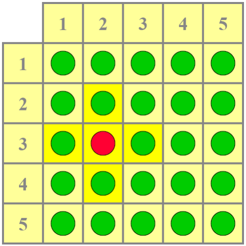
A game of fiver



- Click on a circle, and flip its color and that of adjacent colors;
- Can you make all of the circles red?
- Click on $(3,3),(3,1)$ and $(4,4)$, sequentially.

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

Fiver formulation

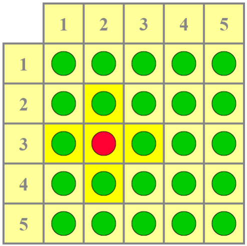


Let

$$x_{ij} \begin{cases} 1, & \text{if row i and column j} \\ & \text{in the square is clicked.} \\ 0, & \text{otherwise.} \end{cases}$$

Minimize: $\sum_i^5 \sum_j^5 x_{ij}$

Fiver formulation



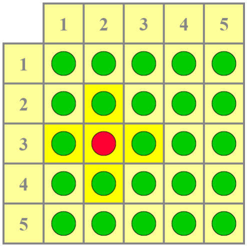
Let

$$x_{ij} \begin{cases} 1, & \text{if row i and column j} \\ & \text{in the square is clicked.} \\ 0, & \text{otherwise.} \end{cases}$$

Minimize: $\sum_i^5 \sum_j^5 x_{ij}$

Subject to: $x_{ij} + x_{i(j-1)} + x_{i(j+1)} + x_{(i-1)j} + x_{(i+1)j}$ is odd for all $1 \leq i, j \leq 5$

Fiver formulation



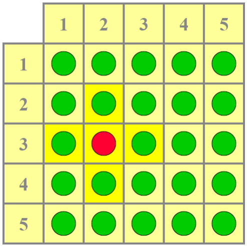
Let

$$x_{ij} \begin{cases} 1, & \text{if row i and column j} \\ & \text{in the square is clicked.} \\ 0, & \text{otherwise.} \end{cases}$$

Minimize: $\sum_i^5 \sum_j^5 x_{ij}$

Subject to: $x_{ij} + x_{i(j-1)} + x_{i(j+1)} + x_{(i-1)j} + x_{(i+1)j}$
is odd for all $1 \leq i, j \leq 5$
$x_{ij} \in \{0, 1\}$ for all $1 \leq i, j \leq 5$

Fiver formulation



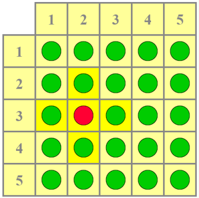
Let

$$x_{ij} \begin{cases} 1, & \text{if row i and column j} \\ & \text{in the square is clicked.} \\ 0, & \text{otherwise.} \end{cases}$$

Minimize: $\sum_i^5 \sum_j^5 x_{ij}$

Subject to: $x_{ij} + x_{i(j-1)} + x_{i(j+1)} + x_{(i-1)j} + x_{(i+1)j}$
is odd for all $1 \leq i, j \leq 5$
$x_{ij} \in \{0, 1\}$ for all $1 \leq i, j \leq 5$
$x_{ij} = 0$ otherwise

Optimizing Fiver formulation

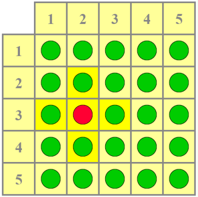


Let

$$x_{ij} \begin{cases} 1, & \text{if row i and column j} \\ & \text{in the square is clicked.} \\ 0, & \text{otherwise.} \end{cases}$$

Minimize: $\sum_{i}^{5} \sum_{j}^{5} x_{ij}$

Optimizing Fiver formulation



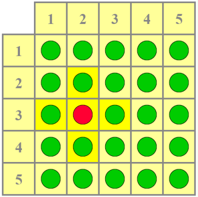
Let

$$x_{ij} \begin{cases} 1, & \text{if row i and column j} \\ & \text{in the square is clicked.} \\ 0, & \text{otherwise.} \end{cases}$$

Minimize: $\sum_i^5 \sum_j^5 x_{ij}$

Subject to: $x_{ij} + x_{i(j-1)} + x_{i(j+1)} + x_{(i-1)j} + x_{(i+1)j} - 2y_{ij} = 1$
for all $1 \leq i, j \leq 5$

Optimizing Fiver formulation



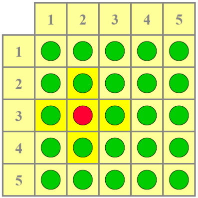
Let

$$x_{ij} \begin{cases} 1, & \text{if row i and column j} \\ & \text{in the square is clicked.} \\ 0, & \text{otherwise.} \end{cases}$$

Minimize: $\sum_i^5 \sum_j^5 x_{ij}$

Subject to: $x_{ij} + x_{i(j-1)} + x_{i(j+1)} + x_{(i-1)j} + x_{(i+1)j} - 2y_{ij} = 1$
for all $1 \leq i, j \leq 5$
$x_{ij} \in \{0, 1\}$ for all $1 \leq i, j \leq 5$

Optimizing Fiver formulation



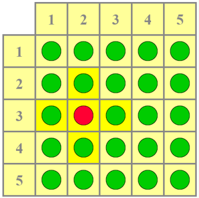
Let

$$x_{ij} \begin{cases} 1, & \text{if row i and column j} \\ & \text{in the square is clicked.} \\ 0, & \text{otherwise.} \end{cases}$$

Minimize: $\sum_i^5 \sum_j^5 x_{ij}$

Subject to: $x_{ij} + x_{i(j-1)} + x_{i(j+1)} + x_{(i-1)j} + x_{(i+1)j} - 2y_{ij} = 1$
for all $1 \leq i, j \leq 5$
$x_{ij} \in \{0, 1\}$ for all $1 \leq i, j \leq 5$
$x_{ij} = 0$ otherwise

Optimizing Fiver formulation



Let

$$x_{ij} \begin{cases} 1, & \text{if row i and column j} \\ & \text{in the square is clicked.} \\ 0, & \text{otherwise.} \end{cases}$$

Minimize: $\sum_i^5 \sum_j^5 x_{ij}$

Subject to: $x_{ij} + x_{i(j-1)} + x_{i(j+1)} + x_{(i-1)j} + x_{(i+1)j} - 2y_{ij} = 1$
for all $1 \leq i, j \leq 5$
$x_{ij} \in \{0, 1\}$ for all $1 \leq i, j \leq 5$
$x_{ij} = 0$ otherwise
$0 \leq y_{ij} \leq 2$, and $y_{ij} \in Z^+$ for all $1 \leq i, j \leq 5$

Integer programming

- **Input:** a set of integer variables $x_1, \cdots, x_n$ and a set of linear inequalities and equalities;

Integer programming

- **Input:** a set of integer variables $x_1, \cdots, x_n$ and a set of linear inequalities and equalities;
- **Feasible solutions:** a solution x' that satisfies all inequalities and equalities as well as the integrality requirements;

Integer programming

- **Input:** a set of integer variables $x_1, \cdots, x_n$ and a set of linear inequalities and equalities;
- **Feasible solutions:** a solution x' that satisfies all inequalities and equalities as well as the integrality requirements;
- **Objective function:** maximize $\sum_i c_i x_i$.

An IP example is formulated as

$$\text{Minimize:} \quad 360 \cdot x_1 + 400 \cdot x_2$$

Integer programming

- **Input:** a set of integer variables $x_1, \cdots, x_n$ and a set of linear inequalities and equalities;
- **Feasible solutions:** a solution x' that satisfies all inequalities and equalities as well as the integrality requirements;
- **Objective function:** maximize $\sum_i c_i x_i$.

An IP example is formulated as

Minimize: $360 \cdot x_1 + 400 \cdot x_2$

Subject to: $20x_1 + 40x_2 \geq 180$

Integer programming

- **Input:** a set of integer variables $x_1, \cdots, x_n$ and a set of linear inequalities and equalities;
- **Feasible solutions:** a solution x' that satisfies all inequalities and equalities as well as the integrality requirements;
- **Objective function:** maximize $\sum_i c_i x_i$.

An IP example is formulated as

$$\begin{array}{ll} \text{Minimize:} & 360 \cdot x_1 + 400 \cdot x_2 \\ & \\ \text{Subject to:} & 20x_1 + 40x_2 \geq 180 \\ & 20x_1 + 10x_2 \geq 110 \end{array}$$

Integer programming

- **Input:** a set of integer variables $x_1, \cdots, x_n$ and a set of linear inequalities and equalities;
- **Feasible solutions:** a solution x' that satisfies all inequalities and equalities as well as the integrality requirements;
- **Objective function:** maximize $\sum_i c_i x_i$.

An IP example is formulated as

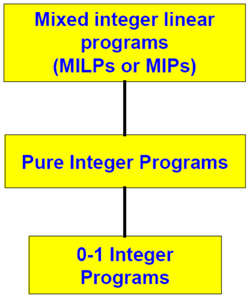
$$
\begin{aligned}
\text{Minimize:} \quad & 360 \cdot x_1 + 400 \cdot x_2 \\
\text{Subject to:} \quad & 20x_1 + 40x_2 \geq 180 \\
& 20x_1 + 10x_2 \geq 110 \\
& 0 \leq x_1, x_2 \in Z
\end{aligned}
$$

20 TVs + 20 laundries
Cost: 360

40 TVs + 10 laundries
Cost: 400

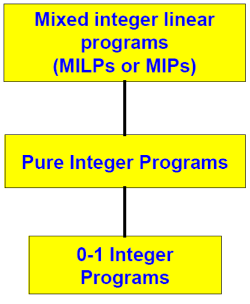
Task: ship 180 TVs and 110 laundries.

Type of Integer programming



$x_i \geq 0$ and integer for some or all i

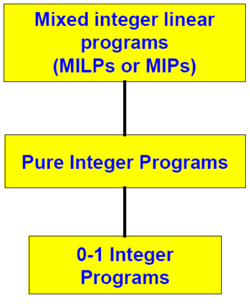
Type of Integer programming



$x_i \geq 0$ and integer for some or all i

$x_i \geq 0$ and integer for every i

Type of Integer programming

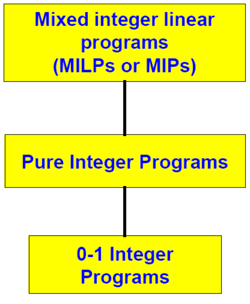


$x_i \geq 0$ and integer for some or all i

$x_i \geq 0$ and integer for every i

$x_i \in \{0, 1\}$ for every i

Type of Integer programming



$x_i \geq 0$ and integer for some or all i

$x_i \geq 0$ and integer for every i

$x_i \in \{0, 1\}$ for every i

Note, pure integer programming instances that are unbounded can have an infinite number of solutions. But they have a finite number of solutions if the variables are bounded.

Outline

Constraint

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

Constraint

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";

Constraint

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";
- That is, $0 \leq x_j \leq 1$ and $x_j \in Z$.

Constraint

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";
- That is, $0 \leq x_j \leq 1$ and $x_j \in Z$.
- Logical constraints $x_i \in \{0,1\}$

Constraint

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";
- That is, $0 \leq x_j \leq 1$ and $x_j \in Z$.
- Logical constraints $x_i \in \{0,1\}$
  - If you select x_1, you cannot select x_2, then $x_1 + x_2 \leq 1$;

Constraint

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";
- That is, $0 \leq x_j \leq 1$ and $x_j \in Z$.
- Logical constraints $x_i \in \{0, 1\}$
  - If you select x_1, you cannot select x_2, then $x_1 + x_2 \leq 1$;
  - If x_1 is selected then x_2 must be selected, then $x_1 \leq x_2$;

Constraint

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";
- That is, $0 \leq x_j \leq 1$ and $x_j \in Z$.
- Logical constraints $x_i \in \{0, 1\}$
  - If you select x_1, you cannot select x_2, then $x_1 + x_2 \leq 1$;
  - If x_1 is selected then x_2 must be selected, then $x_1 \leq x_2$;
  - You must select x_1 or x_2 or both, then $x_1 + x_2 \geq 1$;

Constraint

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";
- That is, $0 \le x_j \le 1$ and $x_j \in Z$.
- Logical constraints $x_i \in \{0, 1\}$
  - If you select x_1, you cannot select x_2, then $x_1 + x_2 \le 1$;
  - If x_1 is selected then x_2 must be selected, then $x_1 \le x_2$;
  - You must select x_1 or x_2 or both, then $x_1 + x_2 \ge 1$;
- Modeling logical constraints that involve non-binary variables is much harder. But we will try to make it as simple as possible.

Logical constraint

If constraint $x \leq 2$ or $x \geq 6$, choose a binary variable w s.t.,

$$w = \begin{cases} 1, & x \leq 2; \\ 0, & x \geq 6. \end{cases}$$

Logical constraint

If constraint $x \leq 2$ or $x \geq 6$, choose a binary variable w s.t.,

$$w = \begin{cases} 1, & x \leq 2; \\ 0, & x \geq 6. \end{cases}$$

When M is a larger number, then it become IP constraints

$$\begin{aligned} & x \leq 2 + M(1 - w) \\ & x \geq 6 - Mw \\ & w \in \{0, 1\} \end{aligned}$$

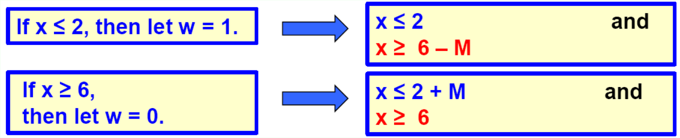
Logical constraint

If constraint $x \leq 2$ or $x \geq 6$, choose a binary variable w s.t.,

$$w = \begin{cases} 1, & x \leq 2; \\ 0, & x \geq 6. \end{cases}$$

When M is a larger number, then it become IP constraints

$$\begin{aligned} x &\leq 2 + M(1 - w) \\ x &\geq 6 - Mw \\ w &\in \{0, 1\} \end{aligned}$$



In both cases, the IP constraints are satisfied.

Modeling "or" constraint

$x_1 + 2x_2 \geq 12$ or
$4x_2 - 10x_3 \leq 1$

Logical constraints

If w = 1, then $x_1 + 2x_2 \geq 12$

If w = 0, then $4x_2 - 10x_3 \leq 1$

$x_1 + 2x_2 \geq 12 - M(1 - w)$ or
$4x_2 - 10x_3 \leq 1 + Mw$
$w \in \{0, 1\}$

IP constraints

Suppose that (x, w) is feasible, for the IP.

Modeling "or" constraint

$x_1 + 2x_2 \geq 12$ or
$4x_2 - 10x_3 \leq 1$

Logical constraints

If w = 1, then $x_1 + 2x_2 \geq 12$

If w = 0, then $4x_2 - 10x_3 \leq 1$

$x_1 + 2x_2 \geq 12 - M(1 - w)$ or
$4x_2 - 10x_3 \leq 1 + Mw$
$w \in \{0, 1\}$

IP constraints

Suppose that (x, w) is feasible, for the IP. Therefore, the logical constraints are satisfied.

Modeling "or" constraint

$x_1 + 2x_2 \geq 12$ or

$4x_2 - 10x_3 \leq 1$

Logical constraints

$x_1 + 2x_2 \geq 12 - M(1 - w)$ or

$4x_2 - 10x_3 \leq 1 + Mw$

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

IP constraints

If w = 1, then $x_1 + 2x_2 \geq 12$

If w = 0, then $4x_2 - 10x_3 \leq 1$

Suppose that (x, w) is feasible, for the IP. Therefore, the logical constraints are satisfied.

If $x_1 + 2x_2 \geq 12$, then let w = 1	$\Rightarrow$	$x_1 + 2x_2 \geq 12$ AND $4x_2 - 10x_3 \leq 1 + M$.
Else $4x_2 - 10x_3 \leq 1$ then let w = 0	$\Rightarrow$	$x_1 + 2x_2 \geq 12 - M$ AND $4x_2 - 10x_3 \leq 1$.

Modeling "or" constraint

$x_1 + 2x_2 \geq 12$ or

$4x_2 - 10x_3 \leq 1$

Logical constraints

$x_1 + 2x_2 \geq 12 - M(1 - w)$ or

$4x_2 - 10x_3 \leq 1 + Mw$

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

IP constraints

If w = 1, then $x_1 + 2x_2 \geq 12$

If w = 0, then $4x_2 - 10x_3 \leq 1$

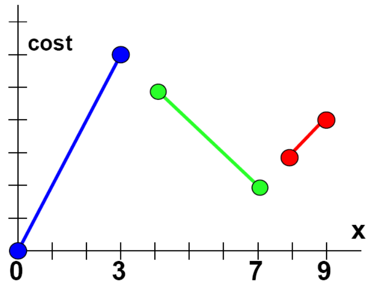
Suppose that (x, w) is feasible, for the IP. Therefore, the logical constraints are satisfied.

If $x_1 + 2x_2 \geq 12$, then let w = 1	$\Rightarrow$	$x_1 + 2x_2 \geq 12$ AND $4x_2 - 10x_3 \leq 1 + M$.
Else $4x_2 - 10x_3 \leq 1$ then let w = 0	$\Rightarrow$	$x_1 + 2x_2 \geq 12 - M$ AND $4x_2 - 10x_3 \leq 1$.

The logical constraints are equivalent to the IP constraints.

Outline

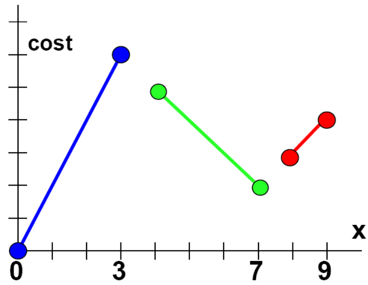
Modeling piecewise linear functions



$$y = \begin{cases} 2x, & \text{if } 0 \leq x \leq 3 \\ 9 - x, & \text{if } 4 \leq x \leq 7 \\ -5 + x, & \text{if } 8 \leq x \leq 9 \end{cases}$$

Assume that x is integer valued. We will create an IP formulation so that the variable y is correctly modeled.

Modeling piecewise linear functions



$$y = \begin{cases} 2x, & \text{if } 0 \le x \le 3 \\ 9 - x, & \text{if } 4 \le x \le 7 \\ -5 + x, & \text{if } 8 \le x \le 9 \end{cases}$$

Assume that x is integer valued. We will create an IP formulation so that the variable y is correctly modeled.

Create new binary and integer variables

$w_1 = \begin{cases} 1 & 0 \le x \le 3 \\ 0 & \text{otherwise} \end{cases}$	$x_1 = \begin{cases} x & 0 \le x \le 3 \\ 0 & \text{otherwise} \end{cases}$
$w_2 = \begin{cases} 1 & 4 \le x \le 7 \\ 0 & \text{otherwise} \end{cases}$	$x_2 = \begin{cases} x & 4 \le x \le 7 \\ 0 & \text{otherwise} \end{cases}$
$w_3 = \begin{cases} 1 & 8 \le x \le 9 \\ 0 & \text{otherwise} \end{cases}$	$x_3 = \begin{cases} x & 8 \le x \le 9 \\ 0 & \text{otherwise} \end{cases}$

Modeling piecewise linear functions Cont'd

$$y = \begin{cases} 2x, & \text{if } 0 \le x \le 3 \\ 9 - x, & \text{if } 4 \le x \le 7 \\ -5 + x, & \text{if } 8 \le x \le 9 \end{cases}$$

$w_1 = \begin{cases} 1 & 0 \le x \le 3 \\ 0 & \text{otherwise} \end{cases}$	$x_1 = \begin{cases} x & 0 \le x \le 3 \\ 0 & \text{otherwise} \end{cases}$
$w_2 = \begin{cases} 1 & 4 \le x \le 7 \\ 0 & \text{otherwise} \end{cases}$	$x_2 = \begin{cases} x & 4 \le x \le 7 \\ 0 & \text{otherwise} \end{cases}$
$w_3 = \begin{cases} 1 & 8 \le x \le 9 \\ 0 & \text{otherwise} \end{cases}$	$x_3 = \begin{cases} x & 8 \le x \le 9 \\ 0 & \text{otherwise} \end{cases}$

$0 \le x_1 \le 3\, w_1$ $w_1 \in \{0, 1\}$
$4w_2 \le x_2 \le 7\, w_2$ $w_2 \in \{0, 1\}$
$8w_3 \le x_3 \le 9\, w_3$ $w_3 \in \{0, 1\}$
$w_1 + w_2 + w_3 = 1$
$x = x_1 + x_2 + x_3$ x_i integer $\forall\ i$

IP constraints

Modeling piecewise linear functions Cont'd

$$y = \begin{cases} 2x, & \text{if } 0 \le x \le 3 \\ 9 - x, & \text{if } 4 \le x \le 7 \\ -5 + x, & \text{if } 8 \le x \le 9 \end{cases}$$

$w_1 = \begin{cases} 1 & 0 \le x \le 3 \\ 0 & \text{otherwise} \end{cases}$	$x_1 = \begin{cases} x & 0 \le x \le 3 \\ 0 & \text{otherwise} \end{cases}$
$w_2 = \begin{cases} 1 & 4 \le x \le 7 \\ 0 & \text{otherwise} \end{cases}$	$x_2 = \begin{cases} x & 4 \le x \le 7 \\ 0 & \text{otherwise} \end{cases}$
$w_3 = \begin{cases} 1 & 8 \le x \le 9 \\ 0 & \text{otherwise} \end{cases}$	$x_3 = \begin{cases} x & 8 \le x \le 9 \\ 0 & \text{otherwise} \end{cases}$

$0 \le x_1 \le 3\, w_1$ $w_1 \in \{0, 1\}$
$4w_2 \le x_2 \le 7\, w_2$ $w_2 \in \{0, 1\}$
$8w_3 \le x_3 \le 9\, w_3$ $w_3 \in \{0, 1\}$
$w_1 + w_2 + w_3 = 1$
$x = x_1 + x_2 + x_3$ x_i integer $\forall\ i$

IP constraints

Suppose that $0 \le x \le 9$, $x \in Z$. If the variables are defined as above, then

$$y = 2x_1 + (9w_2 - x_2) + (-5w_3 + x_3).$$

Modeling piecewise linear functions Cont'd

$$y = \begin{cases} 2x, & \text{if } 0 \le x \le 3 \\ 9 - x, & \text{if } 4 \le x \le 7 \\ -5 + x, & \text{if } 8 \le x \le 9 \end{cases}$$

$w_1 = \begin{cases} 1 & 0 \le x \le 3 \\ 0 & \text{otherwise} \end{cases}$	$x_1 = \begin{cases} x & 0 \le x \le 3 \\ 0 & \text{otherwise} \end{cases}$
$w_2 = \begin{cases} 1 & 4 \le x \le 7 \\ 0 & \text{otherwise} \end{cases}$	$x_2 = \begin{cases} x & 4 \le x \le 7 \\ 0 & \text{otherwise} \end{cases}$
$w_3 = \begin{cases} 1 & 8 \le x \le 9 \\ 0 & \text{otherwise} \end{cases}$	$x_3 = \begin{cases} x & 8 \le x \le 9 \\ 0 & \text{otherwise} \end{cases}$

$0 \le x_1 \le 3\, w_1$ $w_1 \in \{0, 1\}$
$4w_2 \le x_2 \le 7\, w_2$ $w_2 \in \{0, 1\}$
$8w_3 \le x_3 \le 9\, w_3$ $w_3 \in \{0, 1\}$
$w_1 + w_2 + w_3 = 1$
$x = x_1 + x_2 + x_3$ x_i integer $\forall$ i

IP constraints

Suppose that $0 \le x \le 9$, $x \in Z$. If the variables are defined as above, then

$$y = 2x_1 + (9w_2 - x_2) + (-5w_3 + x_3).$$

If (x, w) satisfies the definitions, then it also satisfies the constraints.

Modeling piecewise linear functions Cont'd

$$y = \begin{cases} 2x, & \text{if } 0 \leq x \leq 3 \\ 9 - x, & \text{if } 4 \leq x \leq 7 \\ -5 + x, & \text{if } 8 \leq x \leq 9 \end{cases}$$

$w_1 = \begin{cases} 1 & 0 \leq x \leq 3 \\ 0 & \text{otherwise} \end{cases}$	$x_1 = \begin{cases} x & 0 \leq x \leq 3 \\ 0 & \text{otherwise} \end{cases}$
$w_2 = \begin{cases} 1 & 4 \leq x \leq 7 \\ 0 & \text{otherwise} \end{cases}$	$x_2 = \begin{cases} x & 4 \leq x \leq 7 \\ 0 & \text{otherwise} \end{cases}$
$w_3 = \begin{cases} 1 & 8 \leq x \leq 9 \\ 0 & \text{otherwise} \end{cases}$	$x_3 = \begin{cases} x & 8 \leq x \leq 9 \\ 0 & \text{otherwise} \end{cases}$

$0 \leq x_1 \leq 3\, w_1$ $w_1 \in \{0, 1\}$
$4w_2 \leq x_2 \leq 7\, w_2$ $w_2 \in \{0, 1\}$
$8w_3 \leq x_3 \leq 9\, w_3$ $w_3 \in \{0, 1\}$
$w_1 + w_2 + w_3 = 1$
$x = x_1 + x_2 + x_3$ x_i integer $\forall$ i

IP constraints

Suppose that $0 \leq x \leq 9$, $x \in Z$. If the variables are defined as above, then

$$y = 2x_1 + (9w_2 - x_2) + (-5w_3 + x_3).$$

If (x, w) satisfies the definitions, then it also satisfies the constraints.
If (x, w) satisfies the constraints, then it also satisfies the definitions.

Outline

A 2-variable integer program

Maximize: $\quad z = 3x + 4y$

A 2-variable integer program

Maximize: $z = 3x + 4y$

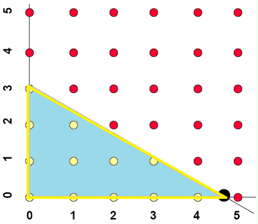
Subject to: $5x + 8y \leq 24$

A 2-variable integer program

Maximize: $z = 3x + 4y$

Subject to: $5x + 8y \leq 24$

$0 \leq x, y \in Z$

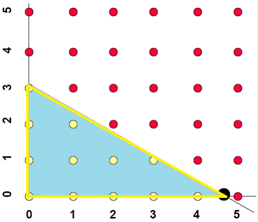


A 2-variable integer program

Maximize: $z = 3x + 4y$

Subject to: $5x + 8y \leq 24$

$0 \leq x, y \in Z$



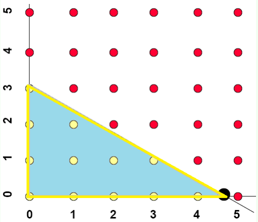
Q1: What is the optimal integer solution?

A 2-variable integer program

Maximize: $z = 3x + 4y$

Subject to: $5x + 8y \leq 24$

$0 \leq x, y \in Z$



Q1: What is the optimal integer solution?

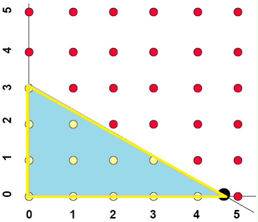
Q2: Can one use linear programming to solve IP problem?

A 2-variable integer program

Maximize: $z = 3x + 4y$

Subject to: $5x + 8y \leq 24$

$0 \leq x, y \in Z$



Q1: What is the optimal integer solution?

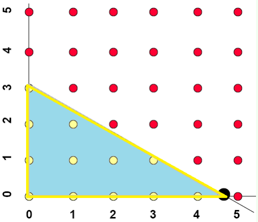
Q2: Can one use linear programming to solve IP problem?

Solve LP (ignore integrality) get $x = \frac{24}{5}, y = 0$ and $z = 14.4$.

A 2-variable integer program

Maximize: $z = 3x + 4y$

Subject to: $5x + 8y \leq 24$
$0 \leq x, y \in Z$



Q1: What is the optimal integer solution?

Q2: Can one use linear programming to solve IP problem?

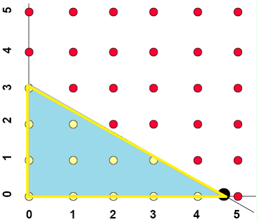
Solve LP (ignore integrality) get $x = \frac{24}{5}, y = 0$ and $z = 14.4$.
Round, get $x = 5, y = 0$, infeasible!

A 2-variable integer program

Maximize: $z = 3x + 4y$

Subject to: $5x + 8y \leq 24$

$0 \leq x, y \in Z$



Q1: What is the optimal integer solution?

Q2: Can one use linear programming to solve IP problem?

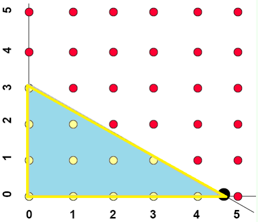
Solve LP (ignore integrality) get $x = \frac{24}{5}, y = 0$ and $z = 14.4$.
Round, get $x = 5, y = 0$, infeasible!
Truncate, get $x = 4, y = 0$, and $z = 12$. Same solution value at $x = 0, y = 3$.

A 2-variable integer program

Maximize: $z = 3x + 4y$

Subject to: $5x + 8y \leq 24$

$0 \leq x, y \in Z$



Q1: What is the optimal integer solution?

Q2: Can one use linear programming to solve IP problem?

Solve LP (ignore integrality) get $x = \frac{24}{5}, y = 0$ and $z = 14.4$.
Round, get $x = 5, y = 0$, infeasible!
Truncate, get $x = 4, y = 0$, and $z = 12$. Same solution value at $x = 0, y = 3$.
Optimal is $x = 3, y = 1$, and $z = 13$.

Feasible region for two constraints

Maximize: $\quad z = 3x + 4y$

Feasible region for two constraints

Maximize: $z = 3x + 4y$

Subject to: $x + y \leq 4$

Feasible region for two constraints

Maximize: $z = 3x + 4y$

Subject to: $x + y \leq 4$
$2x + 3y \leq 9$

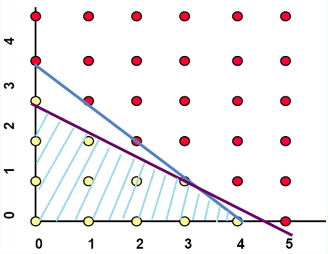
Feasible region for two constraints

Maximize: $z = 3x + 4y$

Subject to: $x + y \leq 4$

$2x + 3y \leq 9$

$0 \leq x, y \in Z$



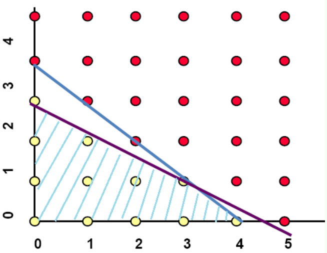
Feasible region for two constraints

Maximize: $z = 3x + 4y$

Subject to: $x + y \leq 4$

$2x + 3y \leq 9$

$0 \leq x, y \in Z$



- More constraints will result in a smaller feasible region;

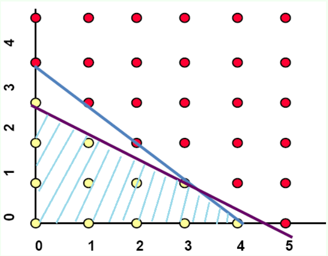
Feasible region for two constraints

Maximize: $z = 3x + 4y$

Subject to: $x + y \leq 4$

$2x + 3y \leq 9$

$0 \leq x, y \in Z$

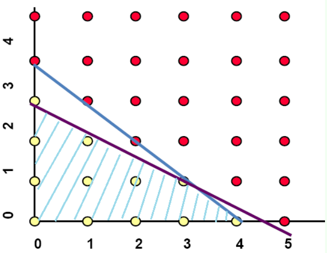


- More constraints will result in a smaller feasible region;
- That is, the search space will be reduced;

Feasible region for two constraints

Maximize: $z = 3x + 4y$

Subject to: $x + y \leq 4$
$2x + 3y \leq 9$
$0 \leq x, y \in Z$



- More constraints will result in a smaller feasible region;
- That is, the search space will be reduced;
- Much, much harder than solving linear programs.

Outline

Complete enumeration

Prize	iPad	server	Brass Rat	Au Bon Pain	6.041 tutoring	15.053 dinner
Points	**5**	**7**	**4**	**3**	**4**	**6**
Utility	**16**	**22**	**12**	**8**	**11**	**19**

Maximize: $16x_1 + 22x_2 + 12x_3 + 8x_4 + 11x_5 + 19x_6$

Complete enumeration

Prize	iPad	server	Brass Rat	Au Bon Pain	6.041 tutoring	15.053 dinner
Points	**5**	**7**	**4**	**3**	**4**	**6**
Utility	**16**	**22**	**12**	**8**	**11**	**19**

Maximize: $16x_1 + 22x_2 + 12x_3 + 8x_4 + 11x_5 + 19x_6$

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

Complete enumeration

Prize	iPad	server	Brass Rat	Au Bon Pain	6.041 tutoring	15.053 dinner
Points	**5**	**7**	**4**	**3**	**4**	**6**
Utility	**16**	**22**	**12**	**8**	**11**	**19**

Maximize: $16x_1 + 22x_2 + 12x_3 + 8x_4 + 11x_5 + 19x_6$

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

$x_i \in \{0, 1\}$ for $1 \leq i \leq 6$

Complete enumeration

Prize	iPad	server	Brass Rat	Au Bon Pain	6.041 tutoring	15.053 dinner
Points	**5**	**7**	**4**	**3**	**4**	**6**
Utility	**16**	**22**	**12**	**8**	**11**	**19**

Maximize: $16x_1 + 22x_2 + 12x_3 + 8x_4 + 11x_5 + 19x_6$

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

$x_i \in \{0, 1\}$ for $1 \leq i \leq 6$

- Systematically considers all possible values of the decision variables, i.e., $n \to 2^n$;

Complete enumeration

Prize	iPad	server	Brass Rat	Au Bon Pain	6.041 tutoring	15.053 dinner
Points	**5**	**7**	**4**	**3**	**4**	**6**
Utility	**16**	**22**	**12**	**8**	**11**	**19**

Maximize: $16x_1 + 22x_2 + 12x_3 + 8x_4 + 11x_5 + 19x_6$

Subject to: $5x_1 + 7x_2 + 4x_3 + 3x_4 + 4x_5 + 6x_6 \leq 14$
$x_i \in \{0, 1\}$ for $1 \leq i \leq 6$

- Systematically considers all possible values of the decision variables, i.e., $n \to 2^n$;
- Usual idea: iteratively break the problem in two. At the first iteration, we consider separately the case that $x_1 \in \{0, 1\}$;

Complete enumeration

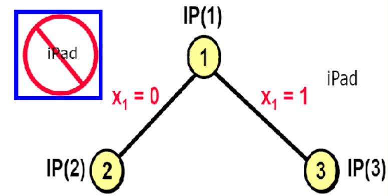
Prize	iPad	server	Brass Rat	Au Bon Pain	6.041 tutoring	15.053 dinner
Points	**5**	**7**	**4**	**3**	**4**	**6**
Utility	**16**	**22**	**12**	**8**	**11**	**19**

Maximize: $16x_1 + 22x_2 + 12x_3 + 8x_4 + 11x_5 + 19x_6$

Subject to: $5x_1 + 7x_2 + 4x_3 + 3x_4 + 4x_5 + 6x_6 \leq 14$
$x_i \in \{0, 1\}$ for $1 \leq i \leq 6$

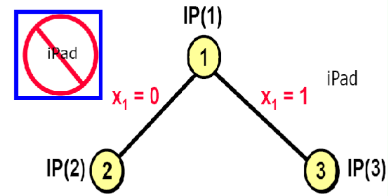
- Systematically considers all possible values of the decision variables, i.e., $n \to 2^n$;
- Usual idea: iteratively break the problem in two. At the first iteration, we consider separately the case that $x_1 \in \{0, 1\}$;
- Each node of the tree represents the original problem plus additional constraints.

An enumeration tree



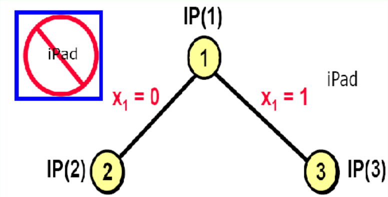
We refer to nodes 2 and 3 as the children of node 1 in the enumeration tree.

An enumeration tree



We refer to nodes 2 and 3 as the children of node 1 in the enumeration tree. Branch and bound is family friendly – so long as you don't mind "pruning" children.

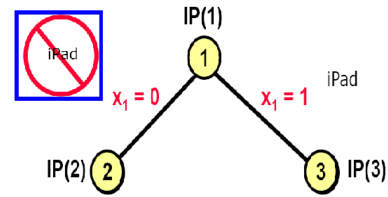
An enumeration tree



We refer to nodes 2 and 3 as the children of node 1 in the enumeration tree. Branch and bound is family friendly – so long as you don't mind "pruning" children.

- IP(1) is the original integer program.

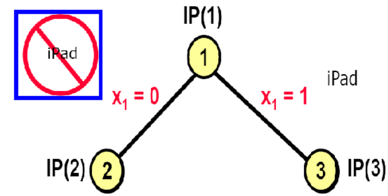
An enumeration tree



We refer to nodes 2 and 3 as the children of node 1 in the enumeration tree. Branch and bound is family friendly – so long as you don't mind "pruning" children.

- IP(1) is the original integer program.
- IP(3) is obtained from IP(1) by adding constraint "$x_1 = 1$";

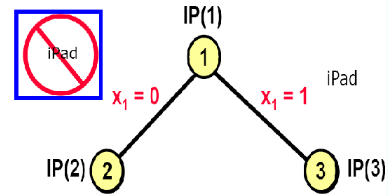
An enumeration tree



We refer to nodes 2 and 3 as the children of node 1 in the enumeration tree. Branch and bound is family friendly – so long as you don't mind "pruning" children.

- IP(1) is the original integer program.
- IP(3) is obtained from IP(1) by adding constraint "$x_1 = 1$";
- An optimal solution for IP(1) can be obtained by taking the best solution from IP(2) and IP(3);

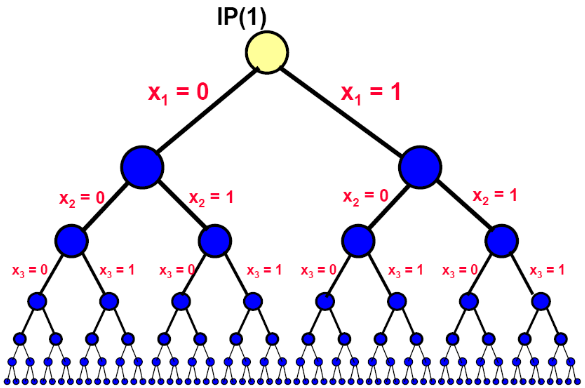
An enumeration tree



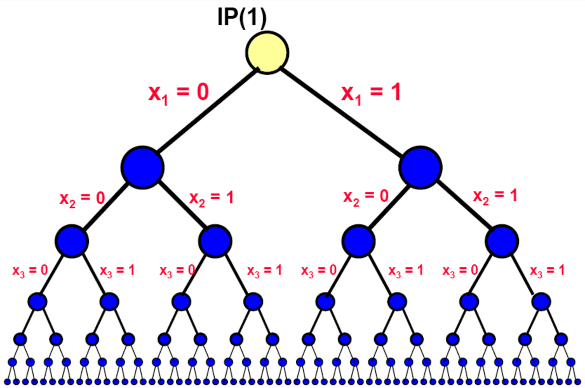
We refer to nodes 2 and 3 as the children of node 1 in the enumeration tree. Branch and bound is family friendly – so long as you don't mind "pruning" children.

- IP(1) is the original integer program.
- IP(3) is obtained from IP(1) by adding constraint "$x_1 = 1$";
- An optimal solution for IP(1) can be obtained by taking the best solution from IP(2) and IP(3);
- It is possible that there is some solution that is feasible for both IP(2) and IP(3).

An enumeration tree

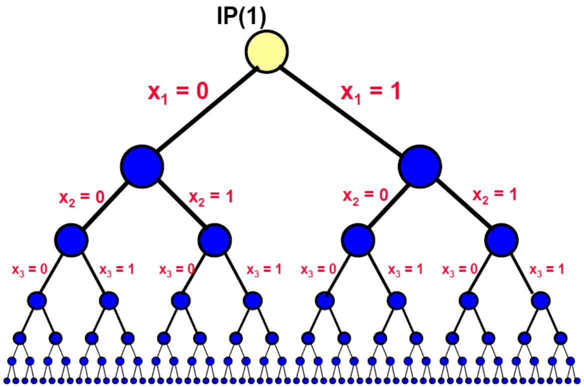


An enumeration tree



- Number of leaves of the tree: 64;

An enumeration tree



- Number of leaves of the tree: 64;
- If there are n variables, the number of leaves is 2^n.

A complete enumeration

- Suppose that we could evaluate 1 billion solutions per second;

A complete enumeration

- Suppose that we could evaluate 1 billion solutions per second;
- Let n = number of binary variables;

n=30	1 sec.	n=60	31 years
n=40	17 minutes	n=70	31,000 years
n=50	11.6 days		

A complete enumeration

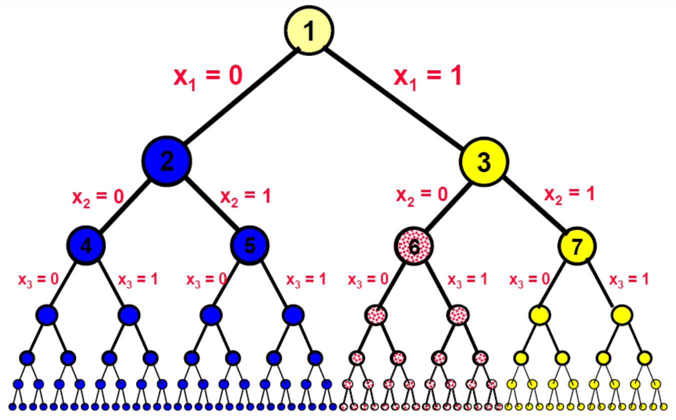
- Suppose that we could evaluate 1 billion solutions per second;
- Let n = number of binary variables;

n=30	1 sec.	n=60	31 years
n=40	17 minutes	n=70	31,000 years
n=50	11.6 days		

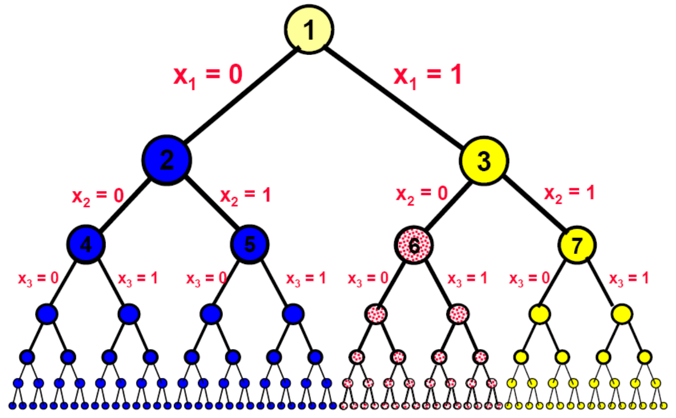
- Suppose that we could evaluate 1 trillion solutions per second, and instantaneously eliminate 99.9999999% of all solutions as not worth considering

n=70	1 sec.	n=100	31 years
n=80	17 minutes	n=110	31,000 years
n=90	11.6 days		

An enumeration tree



An enumeration tree



If we can eliminate an entire subtree in one step, we can eliminate a fraction of all complete solutions at in a single step.

A simpler problem to work with

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

A simpler problem to work with

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

A simpler problem to work with

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_i \in \{0, 1\}$ for $1 \leq i \leq 4$

A simpler problem to work with

$$
\begin{aligned}
\text{Maximize:} \quad & 24x_1 + 2x_2 + 20x_3 + 4x_4 \\
\text{Subject to:} \quad & 8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9 \\
& x_i \in \{0, 1\} \text{ for } 1 \leq i \leq 4
\end{aligned}
$$

- Systematically considers all possible values of the decision variables, i.e., $n \to 2^n$;

A simpler problem to work with

$$\text{Maximize:} \quad 24x_1 + 2x_2 + 20x_3 + 4x_4$$

$$\text{Subject to:} \quad \begin{array}{l} 8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9 \\ x_i \in \{0, 1\} \text{ for } 1 \leq i \leq 4 \end{array}$$

- Systematically considers all possible values of the decision variables, i.e., $n \to 2^n$;
- Usual idea: iteratively break the problem in two. At the first iteration, we consider separately the case that $x_1 \in \{0, 1\}$;

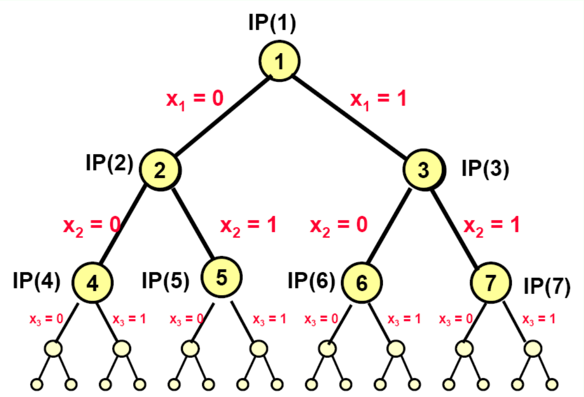
A simpler problem to work with

$$\text{Maximize:} \quad 24x_1 + 2x_2 + 20x_3 + 4x_4$$

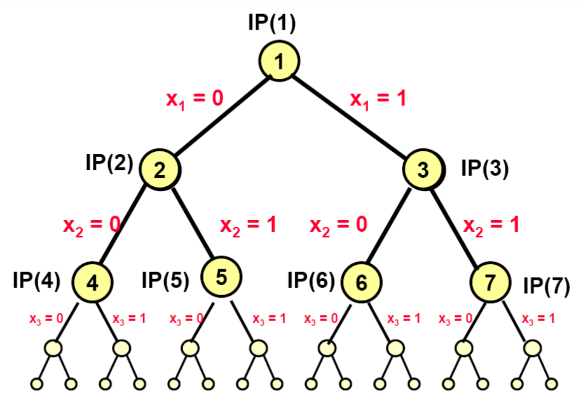
$$\begin{aligned}\text{Subject to:} \quad & 8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9 \\ & x_i \in \{0,1\} \text{ for } 1 \leq i \leq 4\end{aligned}$$

- Systematically considers all possible values of the decision variables, i.e., $n \to 2^n$;
- Usual idea: iteratively break the problem in two. At the first iteration, we consider separately the case that $x_1 \in \{0,1\}$;
- Each node of the tree represents the original problem plus additional constraints.

The entire enumeration tree (16 leaves)

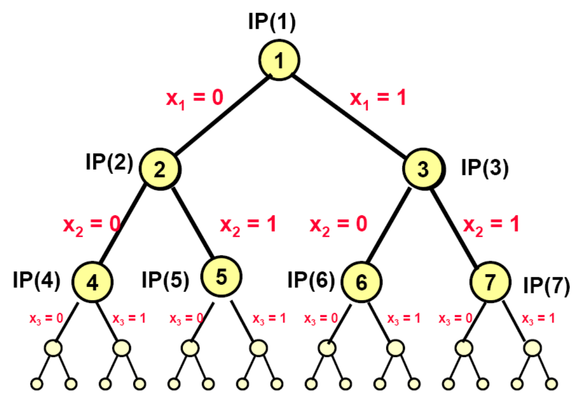


The entire enumeration tree (16 leaves)



- In a branch and bound tree, the nodes represent IPs;

The entire enumeration tree (16 leaves)



- In a branch and bound tree, the nodes represent IPs;
- What is the optimal objective value for IP(4)?

Eliminating subtrees

We eliminate a subtree if

- We have solved the IP for the root of the subtree or;

Eliminating subtrees

We eliminate a subtree if

- We have solved the IP for the root of the subtree or;
- We have proved that the IP solution at the root of the subtree cannot be optimal;

Eliminating subtrees

We eliminate a subtree if

- We have solved the IP for the root of the subtree or;
- We have proved that the IP solution at the root of the subtree cannot be optimal;
- For example, after we solved IP(4), you don't need to look at its children.

Outline

The LP relaxation of the IP

If we drop the requirements that variables be integer, we call it the **LP relaxation of the IP**.

$$\text{Maximize:} \qquad 24x_1 + 2x_2 + 20x_3 + 4x_4$$

The LP relaxation of the IP

If we drop the requirements that variables be integer, we call it the **LP relaxation of the IP**.

$$\text{Maximize:} \quad 24x_1 + 2x_2 + 20x_3 + 4x_4$$

$$\text{Subject to:} \quad 8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$$

The LP relaxation of the IP

If we drop the requirements that variables be integer, we call it the **LP relaxation of the IP**.

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$0 \leq x_i \leq 1$ for $1 \leq i \leq 4$

The LP relaxation of the IP

If we drop the requirements that variables be integer, we call it the **LP relaxation of the IP**.

$$\begin{array}{ll} \text{Maximize:} & 24x_1 + 2x_2 + 20x_3 + 4x_4 \\ \\ \text{Subject to:} & 8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9 \\ & 0 \leq x_i \leq 1 \text{ for } 1 \leq i \leq 4 \end{array}$$

- The LP relaxation of the knapsack problem can be solved using a "greedy algorithm";

The LP relaxation of the IP

If we drop the requirements that variables be integer, we call it the **LP relaxation of the IP**.

$$
\begin{aligned}
\text{Maximize:} \quad & 24x_1 + 2x_2 + 20x_3 + 4x_4 \\
\\
\text{Subject to:} \quad & 8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9 \\
& 0 \leq x_i \leq 1 \text{ for } 1 \leq i \leq 4
\end{aligned}
$$

- The LP relaxation of the knapsack problem can be solved using a "greedy algorithm";
- Think of the objective in terms of dollars, and consider the constraint as a bound on the weight.

The LP relaxation of the IP

If we drop the requirements that variables be integer, we call it the **LP relaxation of the IP**.

$$\text{Maximize:} \quad 24x_1 + 2x_2 + 20x_3 + 4x_4$$

$$\begin{aligned} \text{Subject to:} \quad & 8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9 \\ & 0 \leq x_i \leq 1 \text{ for } 1 \leq i \leq 4 \end{aligned}$$

- The LP relaxation of the knapsack problem can be solved using a "greedy algorithm";
- Think of the objective in terms of dollars, and consider the constraint as a bound on the weight.

item	1	2	3	4
value/lb.	\$3	\$2	\$4	\$1

The LP relaxation solves the IP

item	1	2	3	4
value/lb.	\$3	\$2	\$4	\$1

The LP relaxation solves the IP

item	1	2	3	4
value/lb.	\$3	\$2	\$4	\$1

- Put items into the knapsack in decreasing order of value per pound. What do you get?

The LP relaxation solves the IP

item	1	2	3	4
value/lb.	\$3	\$2	\$4	\$1

- Put items into the knapsack in decreasing order of value per pound. What do you get?
- We get bounds for each IP by solving the LP relaxations.

LP(4)
Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

The LP relaxation solves the IP

item	1	2	3	4
value/lb.	\$3	\$2	\$4	\$1

- Put items into the knapsack in decreasing order of value per pound. What do you get?
- We get bounds for each IP by solving the LP relaxations.

LP(4)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

The LP relaxation solves the IP

item	1	2	3	4
value/lb.	$3	$2	$4	$1

- Put items into the knapsack in decreasing order of value per pound. What do you get?
- We get bounds for each IP by solving the LP relaxations.

LP(4)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 0, x_2 = 0$

The LP relaxation solves the IP

item	1	2	3	4
value/lb.	$3	$2	$4	$1

- Put items into the knapsack in decreasing order of value per pound. What do you get?
- We get bounds for each IP by solving the LP relaxations.

LP(4)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 0, x_2 = 0$

$0 \leq x_i \leq 1$ for $3 \leq i \leq 4$

The LP relaxation solves the IP

item	1	2	3	4
value/lb.	\$3	\$2	\$4	\$1

- Put items into the knapsack in decreasing order of value per pound. What do you get?
- We get bounds for each IP by solving the LP relaxations.

LP(4)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$
$x_1 = 0, x_2 = 0$
$0 \leq x_i \leq 1$ for $3 \leq i \leq 4$

Optimal solution for LP(4): $x_1 = 0, x_2 = 0, x_3 = 1, x_4 = 1, z = 24.$

The LP relaxation solves the IP

item	1	2	3	4
value/lb.	$3	$2	$4	$1

- Put items into the knapsack in decreasing order of value per pound. What do you get?
- We get bounds for each IP by solving the LP relaxations.

LP(4)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$
$x_1 = 0, x_2 = 0$
$0 \leq x_i \leq 1$ for $3 \leq i \leq 4$

Optimal solution for LP(4): $x_1 = 0, x_2 = 0, x_3 = 1, x_4 = 1, z = 24.$

If the optimal solution for LP(k) is feasible for IP(k), then it is also optimal for IP(k).

The LP relaxation solves the IP Cont'd

LP(15)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

The LP relaxation solves the IP Cont'd

LP(15)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

The LP relaxation solves the IP Cont'd

LP(15)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 1, x_2 = 1, x_3 = 1$

The LP relaxation solves the IP Cont'd

LP(15)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 1, x_2 = 1, x_3 = 1$

$0 \leq x_4 \leq 1$

The LP relaxation solves the IP Cont'd

LP(15)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 1, x_2 = 1, x_3 = 1$

$0 \leq x_4 \leq 1$

And occasionally, the LP relaxation is infeasible.

The LP relaxation solves the IP Cont'd

LP(15)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$
$x_1 = 1, x_2 = 1, x_3 = 1$
$0 \leq x_4 \leq 1$

And occasionally, the LP relaxation is infeasible. In this case, the IP is also infeasible.

The LP relaxation solves the IP Cont'd

LP(15)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$
$x_1 = 1, x_2 = 1, x_3 = 1$
$0 \leq x_4 \leq 1$

There is no feasible solution for LP(15).

And occasionally, the LP relaxation is infeasible. In this case, the IP is also infeasible.

The LP relaxation solves the IP Cont'd

LP(15)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 1, x_2 = 1, x_3 = 1$

$0 \leq x_4 \leq 1$

And occasionally, the LP relaxation is infeasible. In this case, the IP is also infeasible.

There is no feasible solution for LP(15).

- If LP(k) is infeasible, then IP(k) is infeasible.

The LP relaxation solves the IP Cont'd

LP(15)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 1, x_2 = 1, x_3 = 1$

$0 \leq x_4 \leq 1$

And occasionally, the LP relaxation is infeasible. In this case, the IP is also infeasible.

There is no feasible solution for LP(15).

- If LP(k) is infeasible, then IP(k) is infeasible.
- In this example, the LHS of the constraint is at least 13. There is no way that the constraint can be satisfied by fractional values or integer values of x_3 and x_4.

Outline

Incumbent solution

Occasionally, the algorithm will find a feasible integer solution. We will keep track of the feasible integer solution with the best objective value so far. It is called the **incumbent**.

Incumbent solution

Occasionally, the algorithm will find a feasible integer solution. We will keep track of the feasible integer solution with the best objective value so far. It is called the **incumbent**.

The incumbent is a feasible solution for the IP. It is the best solution so far in the B&B search.

Incumbent solution

Occasionally, the algorithm will find a feasible integer solution. We will keep track of the feasible integer solution with the best objective value so far. It is called the **incumbent**.

The incumbent is a feasible solution for the IP. It is the best solution so far in the B&B search.

LP(1) bound

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Incumbent solution

Occasionally, the algorithm will find a feasible integer solution. We will keep track of the feasible integer solution with the best objective value so far. It is called the **incumbent**.

The incumbent is a feasible solution for the IP. It is the best solution so far in the B&B search.

LP(1) bound

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$0 \leq x_i \leq 1$ for i $= 1$ to 4

Incumbent solution

Occasionally, the algorithm will find a feasible integer solution. We will keep track of the feasible integer solution with the best objective value so far. It is called the **incumbent**.
The incumbent is a feasible solution for the IP. It is the best solution so far in the B&B search.

LP(1) bound

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$
$0 \leq x_i \leq 1$ for i = 1 to 4

- The optimal solution for LP(1) is $x_1 = \frac{1}{2}, x_2 = 0, x_3 = 1, x_4 = 0, z = 32;$

Incumbent solution

Occasionally, the algorithm will find a feasible integer solution. We will keep track of the feasible integer solution with the best objective value so far. It is called the **incumbent**.
The incumbent is a feasible solution for the IP. It is the best solution so far in the B&B search.

LP(1) bound

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$
$0 \leq x_i \leq 1$ for i = 1 to 4

- The optimal solution for LP(1) is $x_1 = \frac{1}{2}, x_2 = 0, x_3 = 1, x_4 = 0, z = 32$;
- **Important Observ.:** $z_{IP}(j) \leq z_{LP}(j)$ for all j, i.e., $z_{IP}(1) \leq 32$.

Incumbent solution

Occasionally, the algorithm will find a feasible integer solution. We will keep track of the feasible integer solution with the best objective value so far. It is called the **incumbent**.

The incumbent is a feasible solution for the IP. It is the best solution so far in the B&B search.

LP(1) bound

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$0 \leq x_i \leq 1$ for $\mathrm{i} = 1$ to 4

- The optimal solution for LP(1) is $x_1 = \frac{1}{2}, x_2 = 0, x_3 = 1, x_4 = 0, z = 32$;
- **Important Observ.:** $z_{IP}(j) \leq z_{LP}(j)$ for all j, i.e., $z_{IP}(1) \leq 32$.

Recall that we don't solve IP(k) directly. Instead, we solve its LP relaxation. We can use this to obtain bounds.

Pruning branches

Based on the observation

Pruning branches

Based on the observation

- We can prune the active node k IP(k) if $z_{LP(k)} \leq z_I$, where z_I is the objective value of the incumbent.

Pruning branches

Based on the observation

- We can prune the active node k IP(k) if $z_{LP(k)} \leq z_I$, where z_I is the objective value of the incumbent.
- A node is active if it has not been pruned and if LP(k) has not been solved yet.

Pruning branches

Based on the observation

- We can prune the active node k IP(k) if $z_{LP(k)} \leq z_I$, where z_I is the objective value of the incumbent.
- A node is active if it has not been pruned and if LP(k) has not been solved yet.

LP(2)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Pruning branches

Based on the observation

- We can prune the active node k IP(k) if $z_{LP(k)} \leq z_I$, where z_I is the objective value of the incumbent.
- A node is active if it has not been pruned and if LP(k) has not been solved yet.

LP(2)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 0$

$0 \leq x_i \leq 1$ for $i = 2$ to 4

Pruning branches

Based on the observation

- We can prune the active node k IP(k) if $z_{LP(k)} \leq z_I$, where z_I is the objective value of the incumbent.
- A node is active if it has not been pruned and if LP(k) has not been solved yet.

LP(2)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 0$

$0 \leq x_i \leq 1$ for $i = 2$ to 4

- The optimal solution for LP(2) is $z_{LP}(2) = 25$;

Pruning branches

Based on the observation

- We can prune the active node k IP(k) if $z_{LP(k)} \leq z_I$, where z_I is the objective value of the incumbent.
- A node is active if it has not been pruned and if LP(k) has not been solved yet.

LP(2)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 0$

$0 \leq x_i \leq 1$ for i = 2 to 4

- The optimal solution for LP(2) is $z_{LP}(2) = 25$;
- Suppose that the incumbent is $x_1 = 1, x_2 = 1, x_3 = 0, x_4 = 0, z_I = 26$;

Pruning branches

Based on the observation

- We can prune the active node k IP(k) if $z_{LP(k)} \leq z_I$, where z_I is the objective value of the incumbent.
- A node is active if it has not been pruned and if LP(k) has not been solved yet.

LP(2)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 0$

$0 \leq x_i \leq 1$ for i $= 2$ to 4

- The optimal solution for LP(2) is $z_{LP}(2) = 25$;
- Suppose that the incumbent is $x_1 = 1, x_2 = 1, x_3 = 0, x_4 = 0, z_I = 26$;

Recall that we don't solve IP(k) directly. Instead, we solve its LP relaxation. We can use this to obtain bounds.

The branch and bound algorithm

while there is some **active** nodes **do**

select an **active** node *j*

mark j as **inactive**

Solve LP(j): denote solution as x(j);

Case 1 -- **if** $z_{LP}(j) \leq z_I$ **then** ***prune*** node j;

Case 2 -- **if** $z_{LP}(j) > z_I$ and
if x(j) is feasible for IP(j)

then Incumbent := x(j), and $z_I := z_{LP}(j)$;

then ***prune*** node j;

Case 3 -- If **if** $z_{LP}(j) > z_I$ and
if x(j) is not feasible for IP(j) **then**

mark the children of node *j* as **active**

endwhile

The branch and bound algorithm

while there is some **active** nodes **do**

select an **active** node *j*

mark j as **inactive**

Solve LP(j): denote solution as x(j);

Case 1 -- **if** $z_{LP}(j) \le z_I$ **then** ***prune*** node j;

Case 2 -- **if** $z_{LP}(j) > z_I$ and
if x(j) is feasible for IP(j)

then Incumbent := x(j), and $z_I := z_{LP}(j)$;

then ***prune*** node j;

Case 3 -- If **if** $z_{LP}(j) > z_I$ and
if x(j) is not feasible for IP(j) **then**

mark the children of node *j* as **active**

endwhile

Under which condition can we not prune active node j from the B&B Tree for a maximization problem?

Example of B&B algorithm

LP(1)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

No incumbent $z_I = -\infty$ and $z_{LP(1)} = 32$.

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$0 \leq x_i \leq 1$ for i = 1 to 4

Example of B&B algorithm

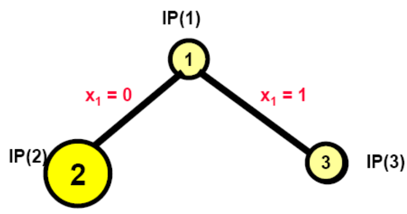
LP(1)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$0 \leq x_i \leq 1$ for i = 1 to 4

No incumbent $z_I = -\infty$ and $z_{LP(1)} = 32$.



Example of B&B algorithm

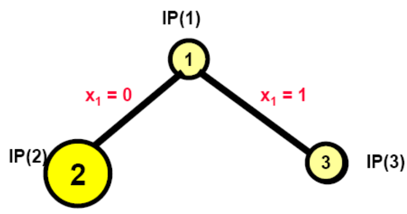
LP(1)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$0 \leq x_i \leq 1$ for i = 1 to 4

No incumbent $z_I = -\infty$ and $z_{LP(1)} = 32$.



Optimal solution for LP(2) is: $x_1 = 0, x_2 = 1, x_3 = 1, x_4 = \frac{3}{4}, z_{LP}(2) = 25;$

Example: Node 3

LP(3)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Example: Node 3

LP(3)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 1$

$0 \leq x_i \leq 1$ for i = 2 to 4

No incumbent $z_I = -\infty$ and $z_{LP(1)} = 32$.

Example: Node 3

LP(3)

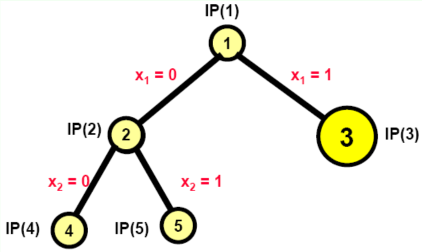
Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 1$

$0 \leq x_i \leq 1$ for i = 2 to 4

No incumbent $z_I = -\infty$ and $z_{LP(1)} = 32$.



Example: Node 3

LP(3)

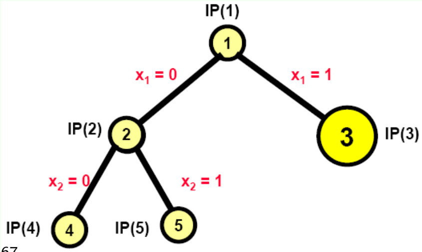
Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 1$

$0 \leq x_i \leq 1$ for i $= 2$ to 4

No incumbent $z_I = -\infty$ and $z_{LP(1)} = 32$.



Optimal solution for LP(3) is:
$x_1 = 1, x_2 = 0, x_3 = \frac{1}{4}, x_4 = 0, z_{LP}(3) = 28;$

Example: Node 4

LP(4)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 0$, $x_2 = 0$

$0 \leq x_i \leq 1$ for i = 3 to 4

No incumbent $z_I = -\infty$ and $z_{LP(1)} = 32$.

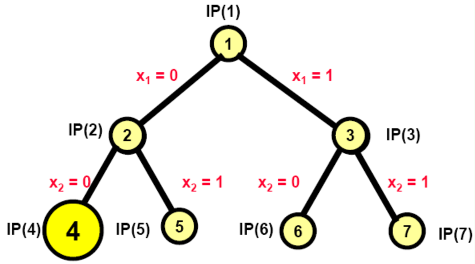
Example: Node 4

LP(4)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$
$x_1 = 0,\ x_2 = 0$
$0 \leq x_i \leq 1$ for i = 3 to 4

No incumbent $z_I = -\infty$ and $z_{LP(1)} = 32$.



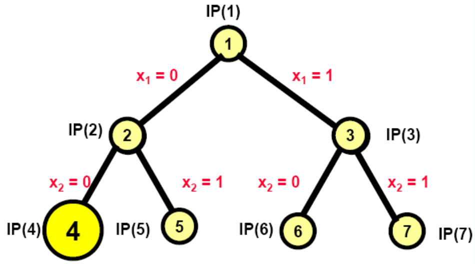
Example: Node 4

LP(4)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$
$x_1 = 0,\ x_2 = 0$
$0 \leq x_i \leq 1$ for $\text{i} = 3$ to 4

No incumbent $z_I = -\infty$ and $z_{LP(1)} = 32$.



Optimal solution for LP(4) is:
$x_1 = 0, x_2 = 0, x_3 = 1, x_4 = 1, z_{LP}(4) = 24;$

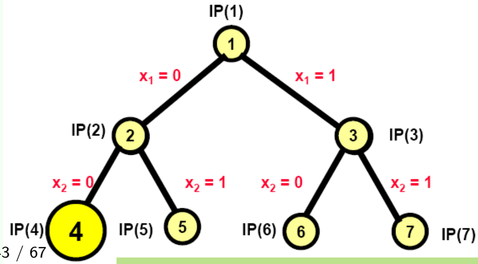
Example: Node 4

LP(4)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$
$x_1 = 0, \; x_2 = 0$
$0 \leq x_i \leq 1$ for i $= 3$ to 4

No incumbent $z_I = -\infty$ and $z_{LP(1)} = 32$.



Optimal solution for LP(4) is:
$x_1 = 0, x_2 = 0, x_3 = 1, x_4 = 1, z_{LP}(4) = 24$; Pruned.

Example: Node 5

LP(5)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$
$x_1 = 0$, $x_2 = 1$
$0 \leq x_i \leq 1$ for i = 3 to 4

Incumbent solution
$z_I = 24$.

Example: Node 5

LP(5)

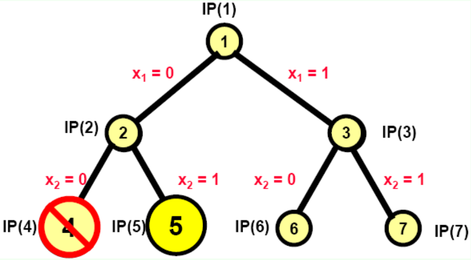
Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 0,\ x_2 = 1$

$0 \leq x_i \leq 1$ for i $= 3$ to 4

Incumbent solution $z_I = 24$.



Example: Node 5

LP(5)

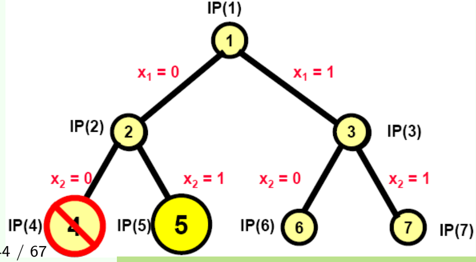
Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 0,\ x_2 = 1$

$0 \leq x_i \leq 1$ for $\ \mathrm{i} = 3\ $ to 4

Incumbent solution $z_I = 24$.



Optimal solution for LP(5) is: $x_1 = 0, x_2 = 1, x_3 = 1, x_4 = \frac{3}{4}, z_{LP}(5) = 25$;

Example: Node 6

LP(6)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$
$x_1 = 1,\ x_2 = 0$
$0 \leq x_i \leq 1$ for i = 3 to 4

Incumbent solution
$z_I = 24$.

Example: Node 6

LP(6)

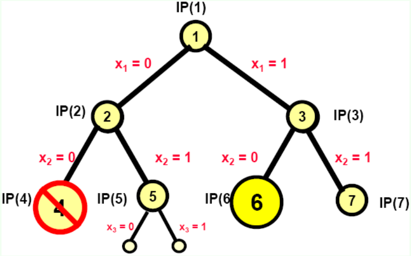
Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 1,\ x_2 = 0$

$0 \leq x_i \leq 1$ for i $= 3$ to 4

Incumbent solution $z_I = 24$.



Example: Node 6

LP(6)

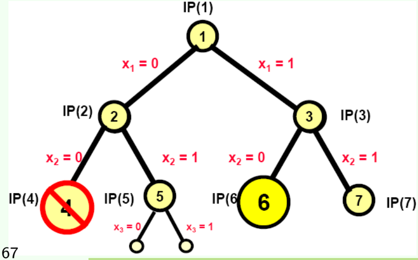
Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 1,\ x_2 = 0$

$0 \leq x_i \leq 1$ for i = 3 to 4

Incumbent solution $z_I = 24$.



Optimal solution for LP(6) is: $x_1 = 1, x_2 = 0, x_3 = \frac{1}{5}, x_4 = 0, z_{LP}(6) = 28$;

Example: Node 7

LP(7)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 1$, $x_2 = 1$

$0 \leq x_i \leq 1$ for i = 3 to 4

Incumbent solution $z_I = 24$.

Example: Node 7

LP(7)

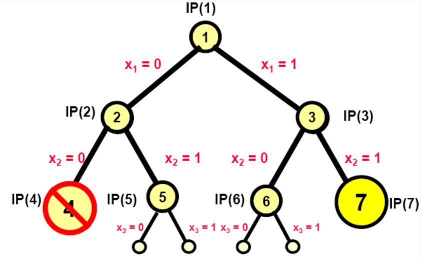
Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 1, \ x_2 = 1$

$0 \leq x_i \leq 1$ for i $= 3$ to 4

Incumbent solution $z_I = 24$.



Example: Node 7

LP(7)

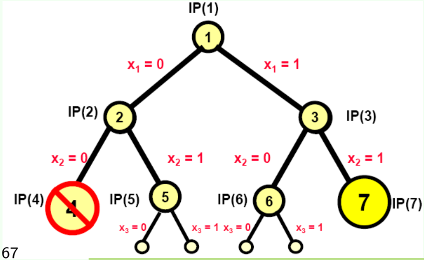
Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 1,\ x_2 = 1$

$0 \leq x_i \leq 1$ for i = 3 to 4

Incumbent solution $z_I = 24$.



Optimal solution for LP(7) is:
$x_1 = 1, x_2 = 1, x_3 = 0, x_4 = 0, z_{LP}(7) = 26$;

Example: Node 8

LP(8)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$
$x_1 = 0,\ x_2 = 1,\ x_3 = 0$
$0 \leq x_4 \leq 1$

Incumbent solution
$z_I = 26$.

Example: Node 8

LP(8)

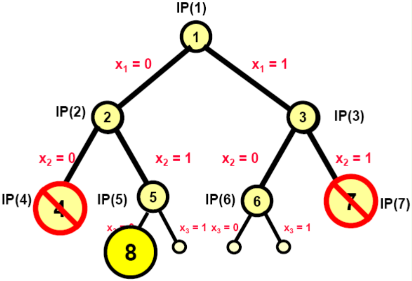
Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 0,\ x_2 = 1,\ x_3 = 0$

$0 \leq x_4 \leq 1$

Incumbent solution $z_I = 26$.



Example: Node 8

LP(8)

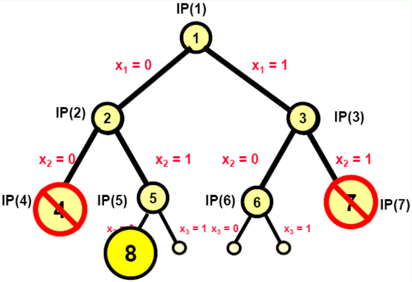
Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 0, \; x_2 = 1, \; x_3 = 0$

$0 \leq x_4 \leq 1$

Incumbent solution $z_I = 26$.



Optimal solution for LP(8) is: $x_1 = 0, x_2 = 1, x_3 = 0, x_4 = 1, z_{LP}(8) = 6$;

Example: Node 8

LP(8)

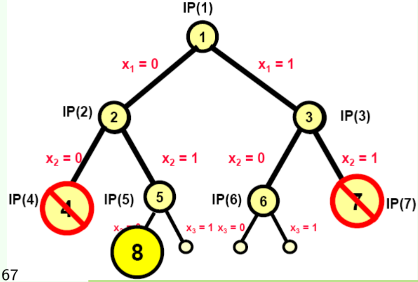
Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 0,\ x_2 = 1,\ x_3 = 0$

$0 \leq x_4 \leq 1$

Incumbent solution $z_I = 26$.



Optimal solution for LP(8) is: $x_1 = 0, x_2 = 1, x_3 = 0, x_4 = 1, z_{LP}(8) = 6$; Pruned.

Example: Node 9

LP(9)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$
$x_1 = 0,\ x_2 = 1,\ x_3 = 1$
$0 \leq x_4 \leq 1$

Incumbent solution
$z_I = 26$.

Example: Node 9

LP(9)

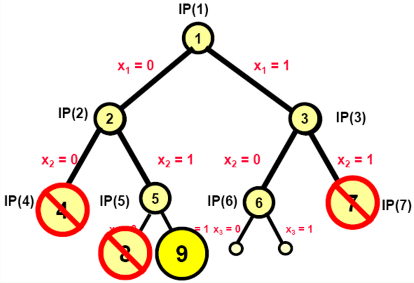
Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 0,\ x_2 = 1,\ x_3 = 1$

$0 \leq x_4 \leq 1$

Incumbent solution $z_I = 26$.



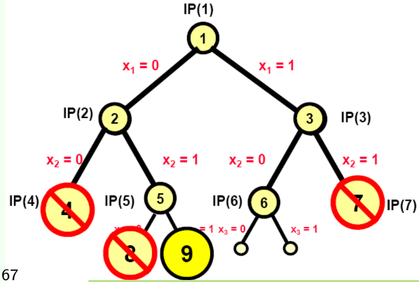
Example: Node 9

LP(9)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$
$x_1 = 0,\ x_2 = 1,\ x_3 = 1$
$0 \leq x_4 \leq 1$

Incumbent solution $z_I = 26$.



Optimal solution for LP(9) is:
$x_1 = 0, x_2 = 1, x_3 = 1, x_4 = \frac{3}{4}, z_{LP}(9) = 25;$

Example: Node 10

LP(10)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 1,\ x_2 = 0,\ x_3 = 0$

$0 \leq x_4 \leq 1$

Incumbent solution $z_I = 26$.

Example: Node 10

LP(10)

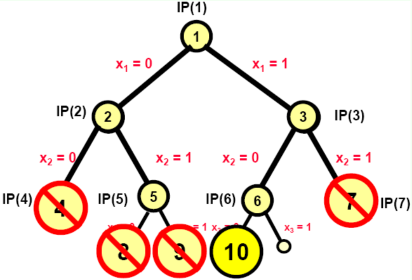
Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 1,\ x_2 = 0,\ x_3 = 0$

$0 \leq x_4 \leq 1$

Incumbent solution $z_I = 26$.



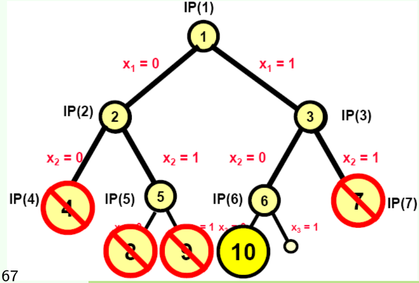
Example: Node 10

LP(10)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \le 9$
$x_1 = 1,\ x_2 = 0,\ x_3 = 0$
$0 \le x_4 \le 1$

Incumbent solution $z_I = 26$.



Optimal solution for LP(10) is:
$x_1 = 1, x_2 = 0, x_3 = 0, x_4 = \frac{1}{4}, z_{LP}(10) = 25$;

Example: Node 11

LP(11)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 1,\ x_2 = 0,\ x_3 = 1$

$0 \leq x_4 \leq 1$

Incumbent solution $z_I = 26$.

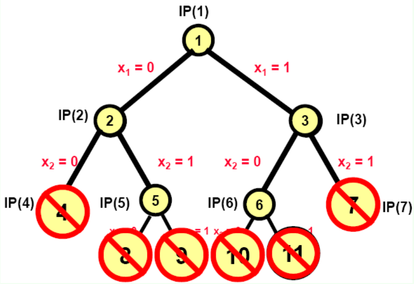
Example: Node 11

LP(11)

Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$
$x_1 = 1,\ x_2 = 0,\ x_3 = 1$
$0 \leq x_4 \leq 1$

Incumbent solution
$z_I = 26.$



Example: Node 11

LP(11)

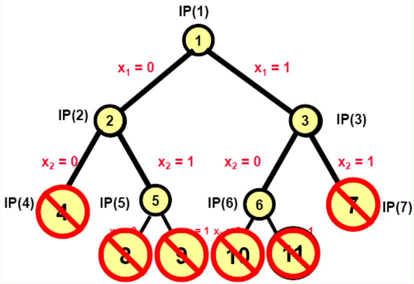
Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 1,\ x_2 = 0,\ x_3 = 1$

$0 \leq x_4 \leq 1$

Incumbent solution $z_I = 26$.



Optimal solution for LP(11): there is no feasible solution for LP(11).

Example: Node 11

LP(11)

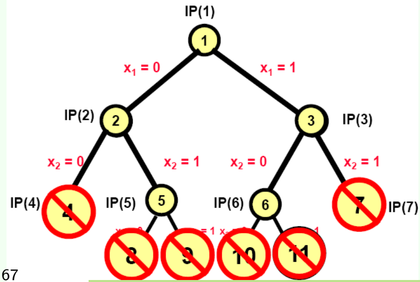
Maximize: $24x_1 + 2x_2 + 20x_3 + 4x_4$

Subject to: $8x_1 + 1x_2 + 5x_3 + 4x_4 \leq 9$

$x_1 = 1,\ x_2 = 0,\ x_3 = 1$

$0 \leq x_4 \leq 1$

Incumbent solution $z_I = 26$.



Optimal solution for LP(11): there is no feasible solution for LP(11). Pruned.

Lessons learned

- Branch and Bound can speed up the search. Only 11 nodes (LPs) out of 31 were evaluated.

Lessons learned

- Branch and Bound can speed up the search. Only 11 nodes (LPs) out of 31 were evaluated.
- Branch and Bound relies on eliminating subtrees, either because the IP at the node was solved, or else because the IP solution cannot possibly be optimum;

Lessons learned

- Branch and Bound can speed up the search. Only 11 nodes (LPs) out of 31 were evaluated.
- Branch and Bound relies on eliminating subtrees, either because the IP at the node was solved, or else because the IP solution cannot possibly be optimum;
- Complete enumerations not possible (because of the running time) if there are more than 100 variables. (Even 50 variables would take too long.)

Lessons learned

- Branch and Bound can speed up the search. Only 11 nodes (LPs) out of 31 were evaluated.
- Branch and Bound relies on eliminating subtrees, either because the IP at the node was solved, or else because the IP solution cannot possibly be optimum;
- Complete enumerations not possible (because of the running time) if there are more than 100 variables. (Even 50 variables would take too long.)
- In practice, there are lots of ways to make Branch and Bound even faster.

Lessons learned

- Branch and Bound can speed up the search. Only 11 nodes (LPs) out of 31 were evaluated.
- Branch and Bound relies on eliminating subtrees, either because the IP at the node was solved, or else because the IP solution cannot possibly be optimum;
- Complete enumerations not possible (because of the running time) if there are more than 100 variables. (Even 50 variables would take too long.)
- In practice, there are lots of ways to make Branch and Bound even faster.
  - There are several ways. One way is for the B&B algorithm to have heuristics that "intelligently" choose the best variable to branch on;

Lessons learned

- Branch and Bound can speed up the search. Only 11 nodes (LPs) out of 31 were evaluated.
- Branch and Bound relies on eliminating subtrees, either because the IP at the node was solved, or else because the IP solution cannot possibly be optimum;
- Complete enumerations not possible (because of the running time) if there are more than 100 variables. (Even 50 variables would take too long.)
- In practice, there are lots of ways to make Branch and Bound even faster.
  - There are several ways. One way is for the B&B algorithm to have heuristics that "intelligently" choose the best variable to branch on;
  - Another technique is to use "rounding", e.g., $x_1 + x_2 \leq 1.5 \rightarrow x_1 + x_2 \leq 1$, or $z_{IP} \leq Z_{LP} = 5.5 \rightarrow z_{IP} \leq 5$.

Outline

Valid inequalities

A **valid inequality** for an IP is any constraint that does not eliminate any feasible integer solutions.

Maximize: $z = 3x + 4y$

Valid inequalities

A **valid inequality** for an IP is any constraint that does not eliminate any feasible integer solutions.

Maximize: $z = 3x + 4y$

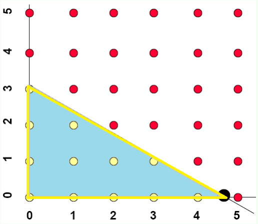
Subject to: $5x + 8y \leq 24$

Valid inequalities

A **valid inequality** for an IP is any constraint that does not eliminate any feasible integer solutions.

Maximize: $z = 3x + 4y$

Subject to: $5x + 8y \leq 24$
$0 \leq x, y \in Z$

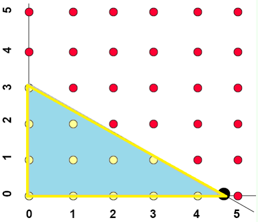


Valid inequalities

A **valid inequality** for an IP is any constraint that does not eliminate any feasible integer solutions.

Maximize: $z = 3x + 4y$

Subject to: $5x + 8y \leq 24$
$0 \leq x, y \in Z$



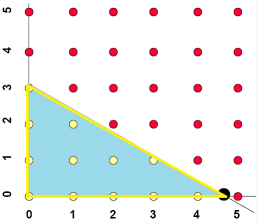
- The constraint $x \leq 5$ is a valid inequality;

Valid inequalities

A **valid inequality** for an IP is any constraint that does not eliminate any feasible integer solutions.

Maximize: $z = 3x + 4y$

Subject to: $5x + 8y \leq 24$
$0 \leq x, y \in Z$



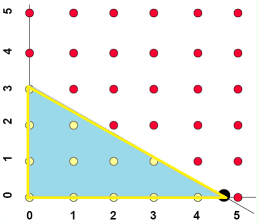
- The constraint $x \leq 5$ is a valid inequality;
- The constraint $x \leq 4$ is also a valid inequality.

Valid inequalities

A **valid inequality** for an IP is any constraint that does not eliminate any feasible integer solutions.

Maximize: $z = 3x + 4y$

Subject to: $5x + 8y \leq 24$
$0 \leq x, y \in Z$



- The constraint $x \leq 5$ is a valid inequality;
- The constraint $x \leq 4$ is also a valid inequality.

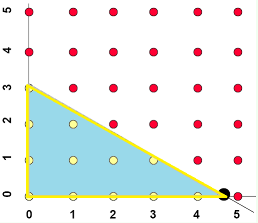
A valid inequality for an IP is any constraint that does not eliminate any feasible integer solutions. It is also called a **cutting plane**, or **cut**.

Valid inequalities

A **valid inequality** for an IP is any constraint that does not eliminate any feasible integer solutions.

Maximize: $z = 3x + 4y$

Subject to: $5x + 8y \leq 24$
$0 \leq x, y \in Z$



- The constraint $x \leq 5$ is a valid inequality;
- The constraint $x \leq 4$ is also a valid inequality.

A valid inequality for an IP is any constraint that does not eliminate any feasible integer solutions. It is also called a **cutting plane**, or **cut**. We want cuts that eliminate part of the LP feasible region.

Rounding

- A fractional bound on an integer variable can be truncated:

$$x \leq 1.5 \rightarrow x \leq 1.$$

Rounding

- A fractional bound on an integer variable can be truncated:

$$x \le 1.5 \to x \le 1.$$

- Given a constraint involving all integer variables with integer coefficients, for example

$$3x + 6y + 9z \le 11 \to x + 2y + 3z \le \lfloor \frac{11}{3} \rfloor = 3.$$

Rounding

- A fractional bound on an integer variable can be truncated:

$$x \leq 1.5 \rightarrow x \leq 1.$$

- Given a constraint involving all integer variables with integer coefficients, for example

$$3x + 6y + 9z \leq 11 \rightarrow x + 2y + 3z \leq \lfloor \frac{11}{3} \rfloor = 3.$$

- Given a constraint involving non-negative integer variables

$$\sum_i a_i x_i \leq b \rightarrow \sum_i \lfloor \frac{a_i}{c} \rfloor x_i \leq \sum_i \frac{a_i}{c} x_i \leq \frac{b}{c}$$

Rounding

- A fractional bound on an integer variable can be truncated:

$$x \leq 1.5 \rightarrow x \leq 1.$$

- Given a constraint involving all integer variables with integer coefficients, for example

$$3x + 6y + 9z \leq 11 \rightarrow x + 2y + 3z \leq \lfloor \frac{11}{3} \rfloor = 3.$$

- Given a constraint involving non-negative integer variables

$$\sum_i a_i x_i \leq b \rightarrow \sum_i \lfloor \frac{a_i}{c} \rfloor x_i \leq \sum_i \frac{a_i}{c} x_i \leq \frac{b}{c}$$

  Note that LHS is integral, so RHS can be truncated, while it does not necessarily dominate original constraint.

Gomory cuts

Gomory cuts is a general method for adding valid inequalities (also known as cuts) to all IPs.

Gomory cuts

Gomory cuts is a general method for adding valid inequalities (also known as cuts) to all IPs.

- Gomory cuts are VERY useful to improve bounds;

Gomory cuts

Gomory cuts is a general method for adding valid inequalities (also known as cuts) to all IPs.

- Gomory cuts are VERY useful to improve bounds;
- Gomory cuts are obtained from a single constraint of the optimal tableau for the LP relaxation;

Gomory cuts

Gomory cuts is a general method for adding valid inequalities (also known as cuts) to all IPs.

- Gomory cuts are VERY useful to improve bounds;
- Gomory cuts are obtained from a single constraint of the optimal tableau for the LP relaxation;
- Assume here that all variables must be integer valued.

Gomory cuts

Gomory cuts is a general method for adding valid inequalities (also known as cuts) to all IPs.

- Gomory cuts are VERY useful to improve bounds;
- Gomory cuts are obtained from a single constraint of the optimal tableau for the LP relaxation;
- Assume here that all variables must be integer valued.

Case I

All LHS coefficients are between 0 and 1:

$$0.2x_1 + 0.3x_2 + 0.3x_3 + 0.5x_4 + x_5 = 1.8$$

Gomory cuts

Gomory cuts is a general method for adding valid inequalities (also known as cuts) to all IPs.

- Gomory cuts are VERY useful to improve bounds;
- Gomory cuts are obtained from a single constraint of the optimal tableau for the LP relaxation;
- Assume here that all variables must be integer valued.

Case I

All LHS coefficients are between 0 and 1:

$$0.2x_1 + 0.3x_2 + 0.3x_3 + 0.5x_4 + x_5 = 1.8$$

Valid inequality (ignore contribution from x_5)

$$0.2x_1 + 0.3x_2 + 0.3x_3 + 0.5x_4 \geq 0.8$$

Gomory cuts Cont'd

Case II: All LHS coefficients are non-negative

$$1.2x_1 + 0.3x_2 + 2.3x_3 + 2.5x_4 + x_5 = 4.8$$

Gomory cuts Cont'd

Case II: All LHS coefficients are non-negative

$$1.2x_1 + 0.3x_2 + 2.3x_3 + 2.5x_4 + x_5 = 4.8$$

Valid inequality (focus on fractional parts)

$$0.2x_1 + 0.3x_2 + 0.3x_3 + 0.5x_4 \geq 0.8$$

Gomory cuts Cont'd

Case II: All LHS coefficients are non-negative

$$1.2x_1 + 0.3x_2 + 2.3x_3 + 2.5x_4 + x_5 = 4.8$$

Valid inequality (focus on fractional parts)

$$0.2x_1 + 0.3x_2 + 0.3x_3 + 0.5x_4 \geq 0.8$$

Case III: General case

$$1.2x_1 - 1.3x_2 - 2.4x_3 + 11.8x_4 + x_5 = 2.9$$

Gomory cuts Cont'd

Case II: All LHS coefficients are non-negative

$$1.2x_1 + 0.3x_2 + 2.3x_3 + 2.5x_4 + x_5 = 4.8$$

Valid inequality (focus on fractional parts)

$$0.2x_1 + 0.3x_2 + 0.3x_3 + 0.5x_4 \geq 0.8$$

Case III: General case

$$1.2x_1 - 1.3x_2 - 2.4x_3 + 11.8x_4 + x_5 = 2.9$$

Round down (be careful about negatives)

$$1 \cdot x_1 - 2 \cdot x_2 - 3 \cdot x_3 + 11x_4 + x_5 \leq 2$$

Gomory cuts Cont'd

Case II: All LHS coefficients are non-negative

$$1.2x_1 + 0.3x_2 + 2.3x_3 + 2.5x_4 + x_5 = 4.8$$

Valid inequality (focus on fractional parts)

$$0.2x_1 + 0.3x_2 + 0.3x_3 + 0.5x_4 \geq 0.8$$

Case III: General case

$$1.2x_1 - 1.3x_2 - 2.4x_3 + 11.8x_4 + x_5 = 2.9$$

Round down (be careful about negatives)

$$1 \cdot x_1 - 2 \cdot x_2 - 3 \cdot x_3 + 11x_4 + x_5 \leq 2$$

Valid inequality (subtract (2) from (1)):

$$0.2x_1 + 0.7x_2 + 0.6x_3 + 0.8x_4 \geq 0.9$$

Convex hull

The **convex hull** is the smallest LP feasible region that contains all of the integer solutions.

Convex hull

The **convex hull** is the smallest LP feasible region that contains all of the integer solutions.

Maximize: $z = 3x + 4y$

Subject to: $5x + 8y \leq 24$

$0 \leq x, y \in Z$

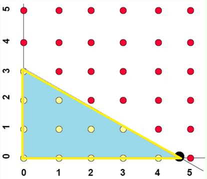
Convex hull

The **convex hull** is the smallest LP feasible region that contains all of the integer solutions.

Maximize: $z = 3x + 4y$

Subject to: $5x + 8y \leq 24$

$0 \leq x, y \in Z$

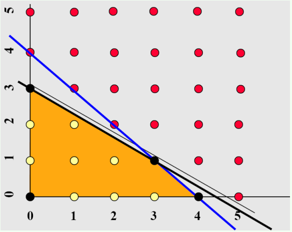


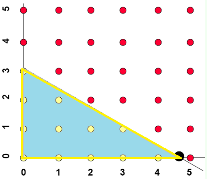
Convex hull

The **convex hull** is the smallest LP feasible region that contains all of the integer solutions.

Maximize: $z = 3x + 4y$

Subject to: $5x + 8y \leq 24$
$0 \leq x, y \in Z$





Maximize: $z = 3x + 4y$

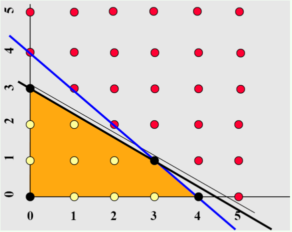
Subject to: $5x + 8y \leq 24$

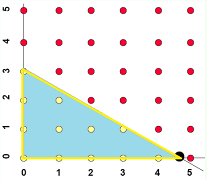
Convex hull

The **convex hull** is the smallest LP feasible region that contains all of the integer solutions.

Maximize: $z = 3x + 4y$

Subject to: $5x + 8y \leq 24$
$0 \leq x, y \in Z$





Maximize: $z = 3x + 4y$

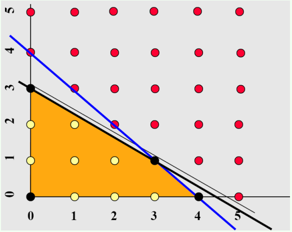
Subject to: $5x + 8y \leq 24$
$x + y \leq 4$

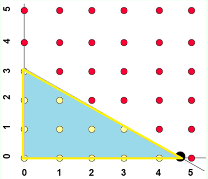
Convex hull

The **convex hull** is the smallest LP feasible region that contains all of the integer solutions.

Maximize: $z = 3x + 4y$

Subject to: $5x + 8y \leq 24$
$0 \leq x, y \in Z$





Maximize: $z = 3x + 4y$

Subject to: $5x + 8y \leq 24$
$x + y \leq 4$
$2x + 3y \leq 9$
$0 \leq x, y \in Z$

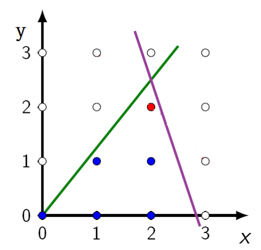
Example of convex hull

Maximize: $z = x + y$

Subject to: $-5x + 4y \leq 0$

$6x + 2y \leq 17$

$0 \leq x, y \in Z$



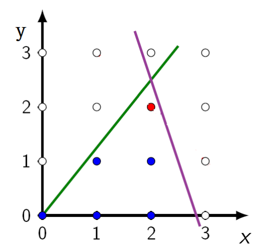
Example of convex hull

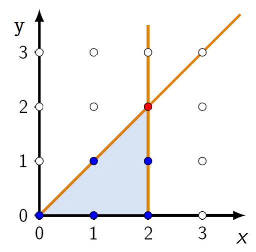
Maximize: $z = x + y$

Subject to: $-5x + 4y \leq 0$

$6x + 2y \leq 17$

$0 \leq x, y \in Z$

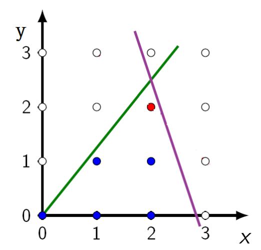


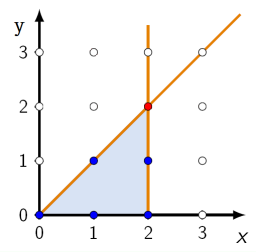


Example of convex hull

Maximize: $z = x + y$

Subject to: $-5x + 4y \leq 0$
$6x + 2y \leq 17$
$0 \leq x, y \in Z$





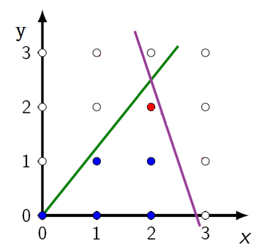
Maximize: $z = x + y$

Subject to: $-5x + 4y \leq 0$
$6x + 2y \leq 17$

Example of convex hull

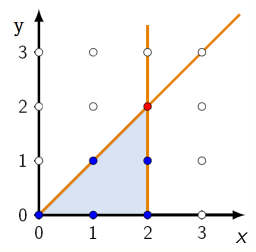
Maximize: $z = x + y$

Subject to: $-5x + 4y \leq 0$
$6x + 2y \leq 17$
$0 \leq x, y \in Z$



Maximize: $z = x + y$

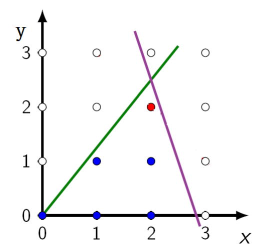
Subject to: $-5x + 4y \leq 0$
$6x + 2y \leq 17$
$x \leq 2$



Example of convex hull

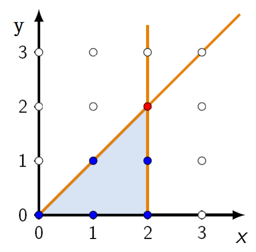
Maximize: $z = x + y$

Subject to: $-5x + 4y \leq 0$
$6x + 2y \leq 17$
$0 \leq x, y \in Z$



Maximize: $z = x + y$

Subject to: $-5x + 4y \leq 0$
$6x + 2y \leq 17$
$x \leq 2$
$y \leq x$
$0 \leq x, y \in Z$



Approaches to finding better bounds

If you solve the LP where the feasible solution is the convex hull of the integer solutions, you are guaranteed to find the optimal integer solution, because all of the corner points are integer.

Approaches to finding better bounds

If you solve the LP where the feasible solution is the convex hull of the integer solutions, you are guaranteed to find the optimal integer solution, because all of the corner points are integer.

- Try to find the convex hull **(Nearly impossible to do)**

Approaches to finding better bounds

If you solve the LP where the feasible solution is the convex hull of the integer solutions, you are guaranteed to find the optimal integer solution, because all of the corner points are integer.

- Try to find the convex hull **(Nearly impossible to do)**
  - Too many constraints;

Approaches to finding better bounds

If you solve the LP where the feasible solution is the convex hull of the integer solutions, you are guaranteed to find the optimal integer solution, because all of the corner points are integer.

- Try to find the convex hull **(Nearly impossible to do)**
  - Too many constraints;
  - Constraints are too hard to find.

Approaches to finding better bounds

If you solve the LP where the feasible solution is the convex hull of the integer solutions, you are guaranteed to find the optimal integer solution, because all of the corner points are integer.

- Try to find the convex hull **(Nearly impossible to do)**
  - Too many constraints;
  - Constraints are too hard to find.
- Find useful constraints of the convex hull **(Very hard to do)**

Approaches to finding better bounds

If you solve the LP where the feasible solution is the convex hull of the integer solutions, you are guaranteed to find the optimal integer solution, because all of the corner points are integer.

- Try to find the convex hull **(Nearly impossible to do)**
  - Too many constraints;
  - Constraints are too hard to find.
- Find useful constraints of the convex hull **(Very hard to do)**
  - Useful when it eliminates the LP optimum;

Approaches to finding better bounds

If you solve the LP where the feasible solution is the convex hull of the integer solutions, you are guaranteed to find the optimal integer solution, because all of the corner points are integer.

- Try to find the convex hull **(Nearly impossible to do)**
  - Too many constraints;
  - Constraints are too hard to find.
- Find useful constraints of the convex hull **(Very hard to do)**
  - Useful when it eliminates the LP optimum;
  - When it can be done, it's great.

Approaches to finding better bounds

If you solve the LP where the feasible solution is the convex hull of the integer solutions, you are guaranteed to find the optimal integer solution, because all of the corner points are integer.

- Try to find the convex hull **(Nearly impossible to do)**
  - Too many constraints;
  - Constraints are too hard to find.
- Find useful constraints of the convex hull **(Very hard to do)**
  - Useful when it eliminates the LP optimum;
  - When it can be done, it's great.
- Find useful valid inequalities **(Doable, but requires skill)**

Approaches to finding better bounds

If you solve the LP where the feasible solution is the convex hull of the integer solutions, you are guaranteed to find the optimal integer solution, because all of the corner points are integer.

- Try to find the convex hull **(Nearly impossible to do)**
  - Too many constraints;
  - Constraints are too hard to find.
- Find useful constraints of the convex hull **(Very hard to do)**
  - Useful when it eliminates the LP optimum;
  - When it can be done, it's great.
- Find useful valid inequalities **(Doable, but requires skill)**
  - Very widely used in practice;

Approaches to finding better bounds

If you solve the LP where the feasible solution is the convex hull of the integer solutions, you are guaranteed to find the optimal integer solution, because all of the corner points are integer.

- Try to find the convex hull **(Nearly impossible to do)**
  - Too many constraints;
  - Constraints are too hard to find.
- Find useful constraints of the convex hull **(Very hard to do)**
  - Useful when it eliminates the LP optimum;
  - When it can be done, it's great.
- Find useful valid inequalities **(Doable, but requires skill)**
  - Very widely used in practice;
  - A great approach.

Outline

Running example

Maximize: $z = x + y$

Subject to: $-5x + 4y \leq 0$
$6x + 2y \leq 17$
$0 \leq x, y \in Z$

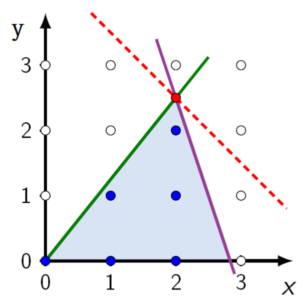
Optimal solution $= 4.5$.

Running example

Maximize: $z = x + y$

Subject to: $-5x + 4y \leq 0$
$6x + 2y \leq 17$
$0 \leq x, y \in Z$

Optimal solution = 4.5.



Running example

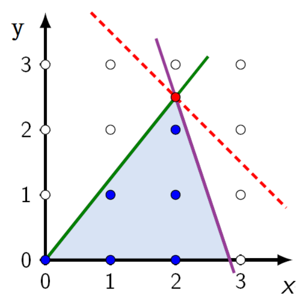
Maximize: $z = x + y$

Subject to: $-5x + 4y \leq 0$

$6x + 2y \leq 17$

$0 \leq x, y \in Z$

Optimal solution $= 4.5$.

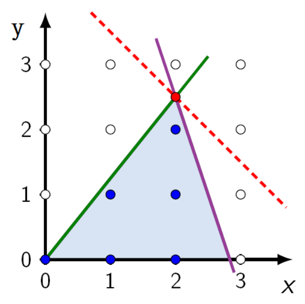


- Remove integer constraint to obtain the LP relaxation;

Running example

Maximize: $z = x + y$

Subject to: $-5x + 4y \leq 0$
$6x + 2y \leq 17$
$0 \leq x, y \in Z$



Optimal solution $= 4.5$.

- Remove integer constraint to obtain the LP relaxation;
- Optimal solution is an upper bound on the optimal cost;

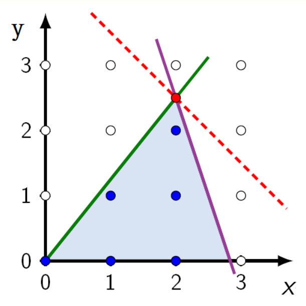
Running example

Maximize: $z = x + y$

Subject to: $-5x + 4y \leq 0$

$6x + 2y \leq 17$

$0 \leq x, y \in Z$



Optimal solution $= 4.5$.

- Remove integer constraint to obtain the LP relaxation;
- Optimal solution is an upper bound on the optimal cost;
- If solution is integral, it is optimal for the original problem.

Cutting plane method

Cutting plane algorithm:

Cutting plane method

Cutting plane algorithm:

Cutting plane method

Cutting plane algorithm:

Step 1: Solve LP relaxation

Cutting plane method

Cutting plane algorithm:

Step 1: Solve LP relaxation

Step 2: If LP solution is integral, it is optimal for the original problem. We are done!

Cutting plane method

Cutting plane algorithm:

- Step 1: Solve LP relaxation
- Step 2: If LP solution is integral, it is optimal for the original problem. We are done!
- Step 3: If LP solution is not integral, find a linear constraint that excludes the LP solution but does not exclude any integer points (always possible);

Cutting plane method

Cutting plane algorithm:

- Step 1: Solve LP relaxation
- Step 2: If LP solution is integral, it is optimal for the original problem. We are done!
- Step 3: If LP solution is not integral, find a linear constraint that excludes the LP solution but does not exclude any integer points (always possible);
- Step 4: Add the cut constraint to the problem. Return to step 1.

Cutting plane method

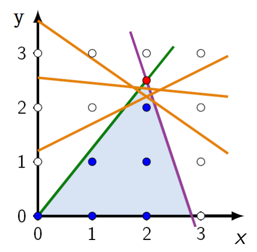
Cutting plane algorithm:

Step 1: Solve LP relaxation

Step 2: If LP solution is integral, it is optimal for the original problem. We are done!

Step 3: If LP solution is not integral, find a linear constraint that excludes the LP solution but does not exclude any integer points (always possible);

Step 4: Add the cut constraint to the problem. Return to step 1.



Cutting plane method

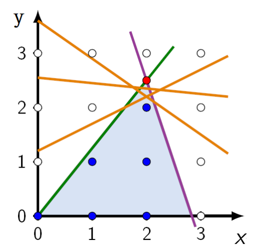
Cutting plane algorithm:

Step 1: Solve LP relaxation

Step 2: If LP solution is integral, it is optimal for the original problem. We are done!

Step 3: If LP solution is not integral, find a linear constraint that excludes the LP solution but does not exclude any integer points (always possible);

Step 4: Add the cut constraint to the problem. Return to step 1.



Maximize: $z = x + y$

Subject to: $-5x + 4y \leq 0$

$6x + 2y \leq 17$

Cutting plane method

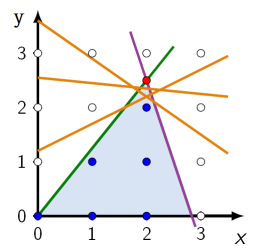
Cutting plane algorithm:

Step 1: Solve LP relaxation

Step 2: If LP solution is integral, it is optimal for the original problem. We are done!

Step 3: If LP solution is not integral, find a linear constraint that excludes the LP solution but does not exclude any integer points (always possible);

Step 4: Add the cut constraint to the problem.
Return to step 1.



Maximize: $z = x + y$

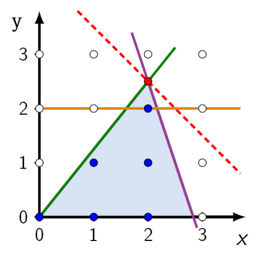
Subject to: $-5x + 4y \leq 0$

$6x + 2y \leq 17$

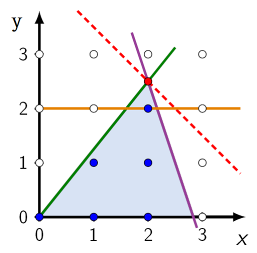
valid inequality

$0 \leq x, y \in Z$

Example of cutting plane



Example of cutting plane

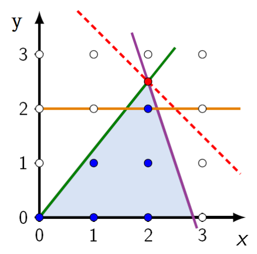


Maximize: $z = x + y$

Subject to: $-5x + 4y \leq 0$

$6x + 2y \leq 17$

Example of cutting plane



Maximize: $z = x + y$

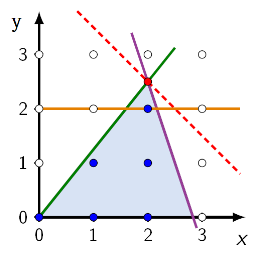
Subject to: $-5x + 4y \leq 0$

$6x + 2y \leq 17$

$y \leq 2$

$0 \leq x, y \in Z$

Example of cutting plane



Maximize: $z = x + y$

Subject to: $-5x + 4y \leq 0$

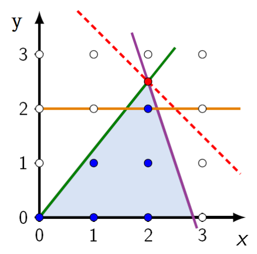
$6x + 2y \leq 17$

$y \leq 2$

$0 \leq x, y \in Z$

- The constraint $y \leq 2$ is a valid cut because it excludes the optimal LP solution but does not exclude any integer points;

Example of cutting plane

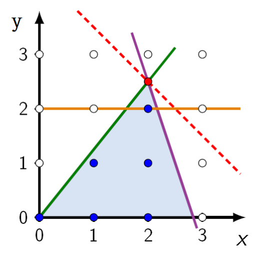


Maximize: $z = x + y$

Subject to:
$$-5x + 4y \leq 0$$
$$6x + 2y \leq 17$$
$$y \leq 2$$
$$0 \leq x, y \in Z$$

- The constraint $y \leq 2$ is a valid cut because it excludes the optimal LP solution but does not exclude any integer points;
- Now solve the LP relaxation for this new problem.

Example of cutting plane



Maximize: $z = x + y$

Subject to: $-5x + 4y \leq 0$

$6x + 2y \leq 17$

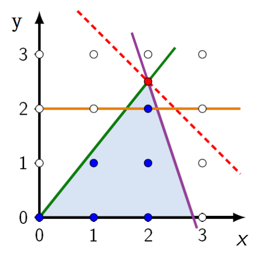
$y \leq 2$

$0 \leq x, y \in Z$

- The constraint $y \leq 2$ is a valid cut because it excludes the optimal LP solution but does not exclude any integer points;
- Now solve the LP relaxation for this new problem.

A cut must simultaneously exclude the LP solution while keeping all the feasible integer points.

Example of cutting plane



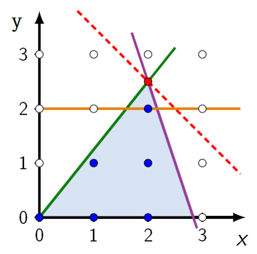
Maximize: $z = x + y$

Subject to:
$$-5x + 4y \leq 0$$
$$6x + 2y \leq 17$$
$$y \leq 2$$
$$0 \leq x, y \in Z$$

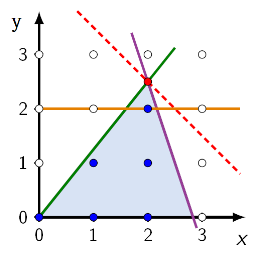
- The constraint $y \leq 2$ is a valid cut because it excludes the optimal LP solution but does not exclude any integer points;
- Now solve the LP relaxation for this new problem.

A cut must simultaneously exclude the LP solution while keeping all the feasible integer points. There always exists at least one valid cut.

Example of cutting plane Cont'd



Example of cutting plane Cont'd

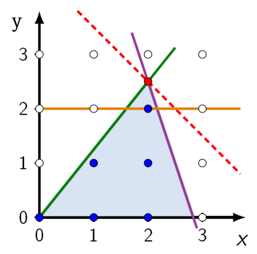


Maximize: $z = x + y$

Subject to: $-5x + 4y \leq 0$

$6x + 2y \leq 17$

Example of cutting plane Cont'd



Maximize: $z = x + y$

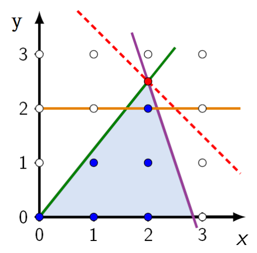
Subject to: $-5x + 4y \leq 0$

$6x + 2y \leq 17$

$y \leq 2$

$0 \leq x, y \in Z$

Example of cutting plane Cont'd

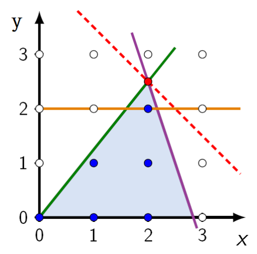


Maximize: $z = x + y$

Subject to:
$-5x + 4y \leq 0$
$6x + 2y \leq 17$
$y \leq 2$
$0 \leq x, y \in Z$

- The constraint $y \leq 2$ is a valid cut because it excludes the optimal LP solution but does not exclude any integer points;

Example of cutting plane Cont'd



Maximize: $z = x + y$

Subject to: $-5x + 4y \leq 0$

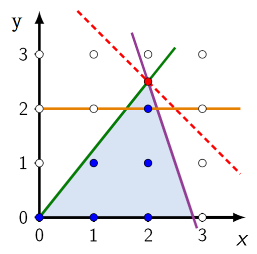
$6x + 2y \leq 17$

$y \leq 2$

$0 \leq x, y \in Z$

- The constraint $y \leq 2$ is a valid cut because it excludes the optimal LP solution but does not exclude any integer points;
- Now solve the LP relaxation for this new problem.

Example of cutting plane Cont'd



Maximize: $z = x + y$

Subject to: $-5x + 4y \leq 0$

$6x + 2y \leq 17$

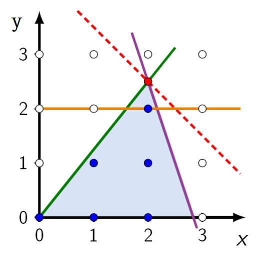
$y \leq 2$

$0 \leq x, y \in Z$

- The constraint $y \leq 2$ is a valid cut because it excludes the optimal LP solution but does not exclude any integer points;
- Now solve the LP relaxation for this new problem.

A cut must simultaneously exclude the LP solution while keeping all the feasible integer points.

Example of cutting plane Cont'd



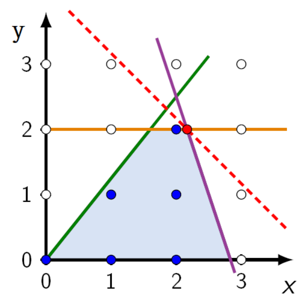
Maximize: $z = x + y$

Subject to:
$$-5x + 4y \leq 0$$
$$6x + 2y \leq 17$$
$$y \leq 2$$
$$0 \leq x, y \in Z$$

- The constraint $y \leq 2$ is a valid cut because it excludes the optimal LP solution but does not exclude any integer points;
- Now solve the LP relaxation for this new problem.

A cut must simultaneously exclude the LP solution while keeping all the feasible integer points. There always exists at least one valid cut.

Example of cutting plane Cont'd

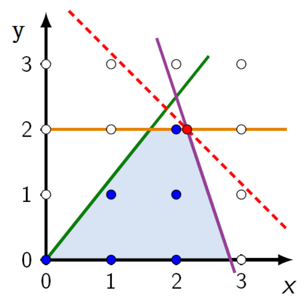


Maximize: $z = x + y$

Subject to:
$$-5x + 4y \leq 0$$
$$6x + 2y \leq 17$$
$$y \leq 2$$
$$0 \leq x, y \in Z$$

Optimal solution $= 4.1667$.

Example of cutting plane Cont'd



Maximize: $z = x + y$

Subject to: $-5x + 4y \leq 0$

$6x + 2y \leq 17$

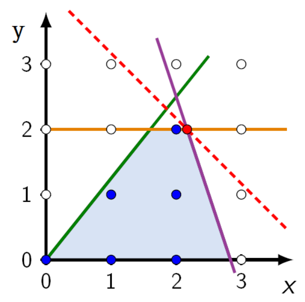
$y \leq 2$

$0 \leq x, y \in Z$

Optimal solution $= 4.1667$.

- Adding a cut reduces our upper bound because we are shrinking the feasible set (we added another constraint).

Example of cutting plane Cont'd



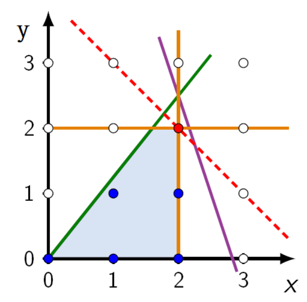
Maximize: $z = x + y$

Subject to: $-5x + 4y \leq 0$
$6x + 2y \leq 17$
$y \leq 2$
$0 \leq x, y \in Z$

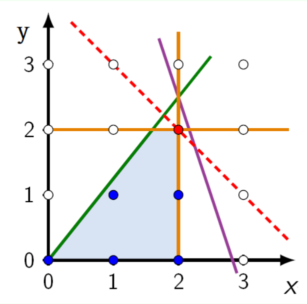
Optimal solution $= 4.1667$.

- Adding a cut reduces our upper bound because we are shrinking the feasible set (we added another constraint).
- Solution is still not an integer. Add another cut!

Example of cutting plane Cont'd



Example of cutting plane Cont'd



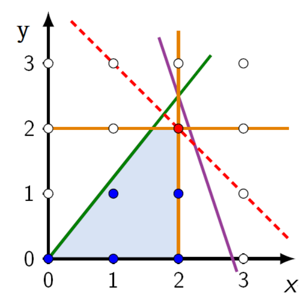
Maximize: $z = x + y$

Subject to: $-5x + 4y \leq 0$

$6x + 2y \leq 17$

$y \leq 2$

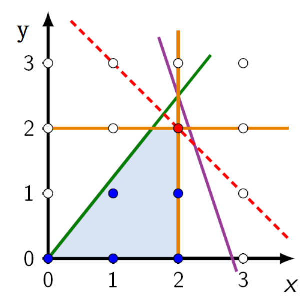
Example of cutting plane Cont'd



Maximize: $z = x + y$

Subject to: $-5x + 4y \leq 0$
$6x + 2y \leq 17$
$y \leq 2$
$x \leq 2$
$0 \leq x, y \in Z$

Example of cutting plane Cont'd

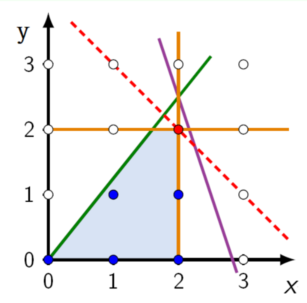


Maximize: $z = x + y$

Subject to:
$-5x + 4y \leq 0$
$6x + 2y \leq 17$
$y \leq 2$
$x \leq 2$
$0 \leq x, y \in Z$

- Optimal solution $x = 2, y = 2, z = 4$;

Example of cutting plane Cont'd



Maximize: $z = x + y$

Subject to:
$-5x + 4y \leq 0$
$6x + 2y \leq 17$
$y \leq 2$
$x \leq 2$
$0 \leq x, y \in Z$

- Optimal solution $x = 2, y = 2, z = 4$;
- LP solution is integral, so it must also be optimal for the original integer problem.

Take-home messages

- Combinatorial Optimization
  - Motivated Examples
  - Constraint
  - Piecewise Objective Function
  - Feasible Region
- Branch and Bound
  - Enumeration Tree
  - LP Relation
  - Branch and Bound
- Cutting Planes
  - Valid Inequalities
  - Cutting Planes