Integer Programming: Introduction

Integer linear programming: formulations, techniques and applications.

Santiago Valdés Ravelo
santiago.ravelo@inf.ufrgs.br

September, 2020
“Integral numbers are the fountainhead of all mathematics.”

Hermann Minkowski.

---

Previous lectures

Integer linear programming

Combinatorial optimization

NP-hardness
Previous lectures
Linear programming

Standard form:

\[
\begin{align*}
\text{max} & \quad c^t \times x \\
\text{s.t.:} & \quad A \times x \leq b \\
& \quad x \in \mathbb{R}^n_+ 
\end{align*}
\]
Cleaning the planet

The chemical company Without Straws developed a substance capable of eliminating wasted plastic. The reaction between the wasted plastic and the substance augments the plastic molecules distances which expands the occupied space by plastic seven times, however the substance molecules attract the molecules of plastic, so the volume is reduced minus three times the substance volume. Since the reaction takes place in a 20000 gal space, the total volume occupied by the reaction is bounded by such space. Besides, the reaction is only successful if the volume of plastic is not greater than 2000 gal plus the substance volume. Also, for security reasons, the difference between 16 times the plastic volume minus 10 times the substance volume should be at least 19000 gal.

The company cannot produce less than 500 gal of substance per reaction and for each 1000 gal of eliminated plastic the profit is $50.00, while the cost of each 1000 gal of substance is $20.00.

What should be the volume of plastic and substance per reaction in order to maximize the profit?
Cleaning the planet. Formulation

\(x_p\), volume (in 1000gal) of plastic to be used in the reaction.
\(x_s\), volume (in 1000gal) of substance to be used in the reaction.

\[
\begin{align*}
\text{max} & \quad 50 \times x_p - 20 \times x_s \\
\text{s.t.} & \quad 7 \times x_p - 3 \times x_s \leq 20 \\
& \quad x_p - x_s \leq 2 \\
& \quad 16 \times x_p - 10 \times x_s \geq 19 \\
& \quad x_s \geq 0.5 \\
& \quad x_s, x_p \geq 0
\end{align*}
\]
Cleaning the planet. Graphical interpretation

\[ x_p, x_s \geq 0 \]
Cleaning the planet. Graphical interpretation

\[ x_p, x_s \geq 0 \]
\[ x_s \geq 0.5 \]
Previous lectures

Cleaning the planet. Graphical interpretation

\[ x_p, x_s \geq 0 \]
\[ x_s \geq 0.5 \]
\[ x_p - x_s \leq 2 \]
Cleaning the planet. Graphical interpretation

\[ x_p, x_s \geq 0 \]
\[ x_s \geq 0.5 \]
\[ x_p - x_s \leq 2 \]
\[ 7 \times x_p - 3 \times x_s \leq 20 \]
Cleaning the planet. Graphical interpretation

\[
x_p, x_s \geq 0
\]
\[
x_s \geq 0.5
\]
\[
x_p - x_s \leq 2
\]
\[
7 \times x_p - 3 \times x_s \leq 20
\]
\[
16 \times x_p - 10 \times x_s \geq 19
\]
Cleaning the planet. Graphical interpretation

Objective function:

\[
\max 50 \times x_p - 20 \times x_s
\]

Constraints:

\[
\begin{align*}
x_p & \geq 0 \\
x_s & \geq 0.5 \\
x_p - x_s & \leq 2 \\
7x_p - 3x_s & \leq 20 \\
16x_p - 10x_s & \geq 19
\end{align*}
\]
Cleaning the planet. Graphical interpretation

Objective function:

\[ \text{max } 50 \times x_p - 20 \times x_s \]

Constraints:

\[ x_p, x_s \geq 0 \]
\[ x_s \geq 0.5 \]
\[ x_p - x_s \leq 2 \]
\[ 7 \times x_p - 3 \times x_s \leq 20 \]
\[ 16 \times x_p - 10 \times x_s \geq 19 \]
If the plastic waste is previously processed and given in blocks of 1000\textit{gal} that cannot be divided, then what happens with the feasible region and the solutions given by the linear program?
Previous lectures

Cleaning the planet. Graphical interpretation

\[ x_p \in \mathbb{Z}, x_s \in \mathbb{R} \]

\[ x_p, x_s \geq 0 \]

\[ x_s \geq 0.5 \]

\[ x_p - x_s \leq 2 \]

\[ 7 \times x_p - 3 \times x_s \leq 20 \]

\[ 16 \times x_p - 10 \times x_s \geq 19 \]

Objective function:

\[ \max 50 \times x_p - 20 \times x_s \]
If the plastic waste is previously processed and given in blocks of 1000\textit{gal} that cannot be divided, then what happens with the feasible region and the solutions given by the linear program?

What happens if the substance is also produced in containers of 1000\textit{gal} that cannot be divided?
Cleaning the planet. Graphical interpretation

Objective function:

\[
\max 50 \times x_p - 20 \times x_s
\]

Constraints:

- \( x_p, x_s \in \mathbb{Z} \)
- \( x_p, x_s \geq 0 \)
- \( x_s \geq 0.5 \)
- \( x_p - x_s \leq 2 \)
- \( 7 \times x_p - 3 \times x_s \leq 20 \)
- \( 16 \times x_p - 10 \times x_s \geq 19 \)
Integer linear programming
Standard form:

\[
\begin{align*}
\text{max} & \quad c^t \times x + d^t \times y \\
\text{s.t.:} & \quad A \times x + B \times y \leq b \\
& \quad x \in \mathbb{R}^{n_1}_{+}, y \in \mathbb{Z}^{n_2}_{+}
\end{align*}
\]
Standard form:

$$\text{max } \quad c^t \times x$$

s.t.:

$$A \times x \leq b$$

$$x \in \mathbb{Z}^n_+$$
Integer linear programming

Binary (or 0-1) linear programming

Standard form:

\[
\begin{align*}
\text{max} & \quad c^t \times x \\
\text{s.t.:} & \quad A \times x \leq b \\
& \quad x \in \{0, 1\}^n
\end{align*}
\]
Integer linear programming

Analogous equivalences as linear programming

- \( \max c^t \times x \equiv \min -c^t \times x \)

- \( A \times x \geq b \equiv -A \times x \leq -b \)

- \( A \times x = b \equiv \begin{cases} A \times x \leq b \\ A \times x \geq b \end{cases} \)

- \( x_i \in \mathbb{Z} \equiv \begin{cases} x_i', x_i'' \in \mathbb{Z}_+ \\ x_i = x_i' - x_i'' \end{cases} \)
Dramatically improves the modeling capabilities:

- Greater flexibility for modeling.
- More realistic modeling.
- Allows to model logical constraints.
- Capable of modeling non-linear functions.
Disadvantages

- More difficult to model.
- Can be much more harder to solve.
Combinatorial Optimization
Combinatorial optimization

Combinatorial optimization problems

- **INPUT.** Description of data for an instance of the problem.

- **FEASIBLE SOLUTIONS.** Set of "objects" that satisfy a collection of "rules" that depend on the input of the problem. There must be a "way" to determine if a solution is feasible based on a given input. Typically, for combinatorial problems there is a finite number of possible solutions.

- **OBJECTIVE FUNCTION.** Value associated to each feasible solution, being the task to find a feasible solution that minimizes (or maximizes) the associated objective value.
Combinatorial optimization

Maximum satisfiability (Max-SAT)

**INPUT**. A set $X$ of $n$ boolean variables and a set of $C$ of $m$ disjunctive clauses, whose literals are variables of $X$.

**FEASIBLE SOLUTION**. Any attribution $\rho : X \rightarrow \{\text{true, false}\}$.

**OBJECTIVE FUNCTION**. Maximize the number of clauses in $C$ that are satisfied after replacing each $x \in X$ by $\rho(x)$. 

\textbf{INPUT.} A set $V$ of $n$ positive values.

\textbf{FEASIBLE SOLUTION.} Any partition of $V$ in two sets $U$ and $V \setminus U$.

\textbf{OBJECTIVE FUNCTION.} Minimize $\left| \sum_{v \in U} v - \sum_{v \in V \setminus U} v \right|$. 
Combinatorial optimization

Maximum independent set

**INPUT.** A graph \((V, E)\).

**FEASIBLE SOLUTION.** Any independent set \(I \subseteq V\), i.e. a set where there are not adjacent nodes of \(G\).

**OBJECTIVE FUNCTION.** Maximize the cardinality of \(I\).
Combinatorial optimization

Minimum spanning tree

**INPUT.** A graph \((V, E)\) and a cost function over the edges 
\(\omega : E \rightarrow \mathbb{R}\).

**FEASIBLE SOLUTION.** Any spanning tree \(T \subseteq G\), i.e. a sub-tree of \(G\) that contains each node of \(G\).

**OBJECTIVE FUNCTION.** Minimize \(\sum_{e \in E_T} \omega(e)\).
Combinatorial optimization

Integer linear programming (standard form)

**INPUT.** A vector of $n$ variables $X$, a matrix $A \in \mathbb{R}^{m \times n}$, a vector $b \in \mathbb{R}^m$ (for some integer $m \geq 1$), and a cost vector $c \in \mathbb{R}^n$.

**FEASIBLE SOLUTION.** Any attribution $\rho : X \rightarrow \mathbb{Z}_+^n$, such that $A \times \rho(X) \leq b$.

**OBJECTIVE FUNCTION.** Maximize $c^t \times \rho(X)$. 
Given a combinatorial optimization problem $\Pi$ and an integer linear program $\text{PLI}$ for $\Pi$. Usually, the following properties are desirable:

- If there exists a feasible solution for $\Pi$, then there exists an associated feasible solution for $\text{PLI}$.

- If there exists a feasible solution for $\text{PLI}$, then there exists an associated feasible solution for $\Pi$.

- If $x$ is feasible for $\Pi$, then its associated solution for $\text{PLI}$ has the same objective value as $x$. 
NP-hardness
3-Satisfiability (3-SAT)

**Instance** a tuple \( \langle X, C \rangle \) where:

- \( X = \{ x_i \}_{i=1}^{n} \) is a set of \( n \) boolean variables.
- \( C = \{ c_j \}_{j=1}^{m} \) is a set of \( m \) disjunctive clauses, each one with exactly three literals. Also the literals are variables and negations of variables of \( X \).

**Output** if there exists a true-false value attribution to the variables that makes all clauses true.

**Theorem.** 3-SAT is \( NP \)-complete (Richard Karp, "Reducibility among combinatorial problems", 1972).
Variables:

- For each variable $x_i \in X$ define a variable $y_i \in \{0, 1\}$.
- For each clause $c_j \in C$ define a variable $z_j \in \{0, 1\}$.

Objective function:

- $\max \sum_{j=0}^{m} z_j$. 
NP-hardness

**Reduction. Constraints**

- For each clause $c_l = (x_i \lor x_j \lor x_k) \in C$, define the constraint $z_l - y_i - y_j - y_k \leq 0$.

- For each clause $c_l = (x_i \lor x_j \lor \neg x_k) \in C$, define the constraint $z_l - y_i - y_j + y_k \leq 1$.

- For each clause $c_l = (x_i \lor \neg x_j \lor \neg x_k) \in C$, define the constraint $z_l - y_i + y_j + y_k \leq 2$.

- For each clause $c_l = (\neg x_i \lor \neg x_j \lor \neg x_k) \in C$, define the constraint $z_l + y_i + y_j + y_k \leq 3$. 
There exists and attribution of values that makes all clauses true iff the optimal value of the integer linear program is \( m \), where:

\[
y_i = \begin{cases} 
1, & x_i = \text{true} \\
0, & x_i = \text{false}
\end{cases}
\]

If each \( c_l \) is true, then there exists a solution where each \( z_l = 1 \) and the other direction also holds:

- \( c_l = (x_i \lor x_j \lor x_k) \iff z_l - y_i - y_j - y_k \leq 0 \).
- \( c_l = (x_i \lor x_j \lor \neg x_k) \iff z_l - y_i - y_j + y_k \leq 1 \).
- \( c_l = (x_i \lor \neg x_j \lor \neg x_k) \iff z_l - y_i + y_j + y_k \leq 2 \).
- \( c_l = (\neg x_i \lor \neg x_j \lor \neg x_k) \iff z_l + y_i + y_j + y_k \leq 3 \).
Theorem. Integer linear programming is \textit{NP}-hard.
Integer Programming: Introduction

Integer linear programming: formulations, techniques and applications.

Santiago Valdés Ravelo
santiago.ravelo@inf.ufrgs.br

08 September, 2020