3 INTEGER LINEAR PROGRAMMING

PROBLEM DEFINITION

Integer linear programming problem (ILP) of the decision variables $x_1,..,x_n$:

(ILP)  \begin{align*}
\text{minimize} & \sum_{j=1}^{n} c_j x_j \\
\text{subject to} & \sum_{j=1}^{n} a_{ij} x_j \geq b_i & \text{for } i=1,\ldots,m \\
& x_j \geq 0 & \text{for } j=1,\ldots,n \\
& x_j \text{ integer} & \text{for } j \in I
\end{align*}

Matrix form:  \begin{align*}
\text{min} & \quad c^T x \\
\text{s.t.} & \quad Ax \geq b \\
& \quad x \geq 0 \\
& \quad x_j \text{ integer for } j \in I
\end{align*}
Variations:
• The objective may be maximization.
• Constraints may be of type $\leq b_i$, $\geq b_i$, or $= b_i$.
• Variables need not be nonnegative.

The problem can always be brought in the above normal form.

Without integer requirement, $I = \emptyset \rightarrow$ a linear programming problem LP.

Solution algorithms for LP:

1) Simplex algorithm (worst case exponential, experimentally average case polynomial)

2) Karmarkar's Interior Point Method (polynomial)
Subcategories:

- Pure integer programming problem: \( I = \{1, \ldots, n\} \)
- Mixed integer (linear) programming problem MIP (MILP): some of the variables are continuous, some integer.
- Binary integer programming problem (BIP): \( x_i \in \{0, 1\}, i=1, \ldots, n \).

All of these are NP-complete:

\( \text{ILP, MIP, BIP, ... } \in \text{ NPC} \).
\textit{LP-relaxation} = ILP without integer requirements

Why linear models?

- The LP-relaxation of linear IP-models can be solved efficiently and this solution can be used as a starting point for integer solutions.

- Many (most) combinatorial optimization problems can be modeled as ILP-models.

- Problems with nonlinear objective or constraint functions are uncommon in combinatorial optimization.
Applications of ILP-models:

- standard graph, routing, packing and scheduling problems of discrete optimization
  → Examples (1)-(14) in Chapter 1.3.

- LP problems in which the variables are numbers of whole pieces.

- LP problems with logical constraints (disjunction, implication etc.), fixed costs
A simple heuristic:
Solve the LP-relaxation and round the variables $x_j$, $j \in I$ to the nearest integer. Improvement: Try all rounding possibilities.

This technique is suitable only for instances with few integer variables or when their order of magnitude is large enough.

Problems:
- The rounded solution is not necessarily optimal or may be far from the optimum.
- The rounded solution may not be feasible.
- If the number of integer variables is $k$, the number of different rounding combinations is $2^k$. Exponential time, if feasibility has to be checked for all.
- Binary variables: no realistic interpretation for fractional values.
For example Knapsack problem:
$x_i = 0.5$: gives no information of the decision: to take the object or not? The rounding corresponds to a flip of a coin.
BRANCH AND BOUND METHODS

The advantage of a unique model: a general purpose Branch and Bound method.

Two basic stages of a general Branch and Bound method:
- **Branching**: splitting the problem into subproblems
- **Bounding**: calculating lower and/or upper bounds for the objective function value of the subproblem

Example of branching: separating the current subspace into two parts using the integrality requirement. Using the bounds, unpromising subproblems can be eliminated.
LP-relaxation:
- discarding the integer requirements.
- for binary variables, add bounds $0 \leq x_i \leq 1$

The LP-minimum gives a **lower bound** for the ILP-minimum:

$$\min f_{\text{LP}} \leq \min f_{\text{ILP}}.$$

**Incumbent solution** = the best IP-solution given by the solved subproblems (record holder).

The incumbent objective value is an **upper bound** for the minimum value.
A list $P$ of candidate subproblems is maintained and updated.

Subproblem is \textit{fathomed} (totally solved, examined, explored) and removed from the list, when

- it has an integer solution that is best so far and becomes the new incumbent solution, or,
- its optimum LP-solution objective is worse than the current incumbent value, or,
- the LP-problem is infeasible.

Notation:

$f^* =$ minimum value of the objective function for the current LP-subproblem

$f_{min}$ = incumbent minimum value, given by a feasible integer solution $x_{min}$

$P =$ the set of non-fathomed subproblems
BRANCH AND BOUND ALGORITHM:

0. $f_{\text{min}} := \infty$ or known $f$ value at a feasible integer solution, $P := \emptyset$
   Solve the LP-relaxation: solution $\mathbf{x}$, $f^* = f(\mathbf{x})$.
   Stop, if
   - $\mathbf{x}$ satisfies integrality constraints: it is the optimum solution.
   - the problem is infeasible or unbounded.

1. **Branching**: Select a variable $x_j$, $j \in I$ that has a fractional value:

   \[ k < x_j < k+1 \]  
   \[ (k \text{ integer}) \]

   Create two new LP-subproblems and add these to the set $P$:
   1) previous LP-subproblem + constraint $x_j \leq k$
   2) previous LP-subproblem + constraint $x_j \geq k+1$
2. Select a subproblem from set P and solve it. Remove it from set P.

3. (a) If $f^* < f_{\text{min}}$, but $x$ does not satisfy integrality constraints, go to 1 (branching).

FATHOMING:

(b) **Bounding**: If $f^* < f_{\text{min}}$ and $x$ satisfies integrality constraints, set $f_{\text{min}} := f^*$ and $x_{\text{min}} := x$. Go to 4.

(c) If $f^* \geq f_{\text{min}}$, go to 4.

(d) If the subproblem is infeasible, go to 4.
4. If there are unsolved subproblems \((P\neq \emptyset)\), go to 2. Else \((P=\emptyset)\), stop: all problems are fathomed.

If \(f_{\text{min}} = \infty\), the problem has no integer solution. Else, the solution is \(x_{\text{min}}, f_{\text{min}}\).
The LP-subproblems ↔ vertices of a binary tree

Every vertex has either
- two descendants: case 3(a), or
- none, when it is fathomed: cases 3(b),(c),(d)

Modifications for maximization:
Replace $f_{\min}, x_{\min}$ with $f_{\max}, x_{\max}$.
Replace $\infty$ with $-\infty$ in steps 1 and 4.
Reverse inequalities in step 3.
Specifications:

1) **How to select a branching variable?**

- No absolutely best branching strategy.

- Heuristic rules: Select the variable
  - with a value closest to integer
  - with the largest coefficient in the objective function
2) How to select the next LP-subproblem to be solved?

**Backtracking / Depth-first / LIFO:**

- Solve the subproblems in the reverse order of generation (Last In, First Out): take the sub-problem that was generated last (one of the two) to be solved first.
- Proceed to its descendant until the branch has been fathomed.

**Jumptracking:**

- Solve the subproblems of step 1 immediately after branching and execute step 3. In this case

\[
P = \{ \text{the subproblems waiting for branching} \}.
\]

- Select the problem with best f*-value for branching.
Example 3.1

\[ \begin{align*}
\text{min } f &= 4x_1 + 5x_2 \\
x_1 + 4x_2 &\geq 5 \\
3x_1 + 2x_2 &\geq 7 \\
x_1, x_2 &\geq 0, \text{ both integer.}
\end{align*} \]
Branch-and-Bound tree of subproblems:

LP₀

LP₁

LP₂

LP₃

LP₄

x₁ = 1.8
x₂ = 0.8
f* = 11.2

x₁ ≤ 1
x₂ = 2
f* = 14 = fₘᵢₙ

x₁ = 2
x₂ = 0.75
f* = 11.75

x₁ ≥ 2
x₂ ≥ 1

x₁ = 5
x₂ = 0
f* = 20 > fₘᵢₙ

x₁ = 2
x₂ = 1
f* = 13 = fₘᵢₙ
Comment:
The descendant LP-subproblem differ from its parent LP-problem only by one constraint

→

Use dual simplex method, starting from the basic solution of its parent LP.

The above Branch & Bound can be used for pure ILP:s, mixed ILP:s and binary problems.

For binary variables the branching constraints $x_i \leq 0$ and $x_i \geq 1$ result in values $x_i = 0$ and $x_i = 1$.

Implicit enumeration techniques like Balas' additive algorithm may be computationally more efficient for binary problems.

Specialized implementations of the B&B method exist for common problem types of Chapter 1.3 e.g. TSP or scheduling.
General goals of all Branch & Bound methods:

- to determine the bounds as tight as possible
- efficient fathoming of subproblems = pruning of nonpromising branches
- to keep the total computation time at minimum

The last goal in conflict with the first two:
tight bounds mean efficient fathoming but more computation per vertex.
CUTTING PLANE METHODS

Consider a pure integer linear programming problem, where all parameters are integers.

This can be accomplished by multiplying a constraint by a suitable constant.

→

Objective function value and all the "slack" variables have integer values (in a feasible solution).

First, solve the LP-relaxation to get a lower bound for the minimum objective value.
Final simplex tableau given: basic variables have columns with coefficient 1 in one constraint row and 0 in other rows.

Solution:
• nonbasic variables = 0
• basic variables = RHS (right hand side)

The objective function of the same form, basic variable $f$.

If the LP-solution is fractional, at least one of the RHS values is fractional.

Append to the model a constraint that cuts away a part of the feasible set, so that no integer solutions are lost.
Consider row i from the final simplex tableau, with a fractional RHS d.
\( x_{j_0} = \) the basic variable of this row
\( N = \) index set of nonbasic variables.

Row i:
\[ x_{j_0} + \sum_{j \in N} w_{ij} x_j = d \]
Define \[ \lfloor d \rfloor = \text{largest integer that is } \leq d \text{ (the whole part of } d\text{, if } d\text{ is positive),} \]
function \( \text{floor}(d) \)

\[
x_{jo} + \sum_{j \in N} w_{ij} x_j = d
\]

All variables \( \geq 0 \) \( \Rightarrow \) \( \sum_{j \in N} \lfloor w_{ij} \rfloor x_j \leq \sum_{j \in N} w_{ij} x_j \)

\[
\Rightarrow x_{jo} + \sum_{j \in N} \lfloor w_{ij} \rfloor x_j \leq d
\]

Left hand side is integer

\[
\Rightarrow x_{jo} + \sum_{j \in N} \lfloor w_{ij} \rfloor x_j \leq \lfloor d \rfloor
\]
From the first and last formula

\[ \Rightarrow \quad d - \lfloor d \rfloor \leq \sum_{j \in N} (w_{ij} - \lfloor w_{ij} \rfloor) x_j \]

Denote the fractional parts by

\[ r = d - \lfloor d \rfloor, \quad f_{ij} = w_{ij} - \lfloor w_{ij} \rfloor \]

*Cut constraint* or a *cutting plane* in the solution space:

\[ \sum_{j \in N} f_{ij} x_j \geq r \]
Equation form, using a slack variable $s_i$:

$$- \sum_{j \in N} f_{ij} x_j + s_i = -r$$

This equation is of basic form, with value of the basic variable $s_i = -r$.

Resulting simplex tableau is optimal but infeasible → we apply the dual simplex method until all variables are nonnegative.

The cut constraints do not cut out any feasible integer points and they pass through at least one integer point.
0. Solve the LP-relaxation. Stop, if all variables in the solution have integer values: then it is the optimum. Stop, if the problem is infeasible or unbounded.

1. Select a row with a fractional RHS: this is called the source row. Generate the cut constraint associated with this row:
   \[ \sum_{j \in N} f_{ij} x_j + s_i = -r \]

2. Augment the simplex tableau with a column for \( s_i \) and the new constraint row.

3. Solve the LP-problem with the dual simplex method. Stop, if all variables in the solution have integer values: then it is the optimum. Stop, if the problem is infeasible or unbounded.

   Else, go to 1
The source row can be chosen arbitrarily. Also the objective row can be used because the value of f must be integer.

If a cut constraint becomes inactive again with $s_i > 0$, then variable $s_i$ and its row can be eliminated from the tableau. This means a new cut makes the old one unnecessary. The elimination restricts the growth of the simplex tableau.

The number of iterations is difficult to estimate but being an exact method like the B&B, the computation time is not polynomially bounded.
Example 3.2 (cf. B&B solution in example 3.1)

\[
\begin{align*}
\text{min } f &= 4x_1 + 5x_2 \\
x_1 + 4x_2 &\geq 5 \\
3x_1 + 2x_2 &\geq 7 \\
x_1, x_2 &\geq 0, \text{ both integer.}
\end{align*}
\]

Basic form with slack variables \(x_3, x_4\).

\[
\begin{align*}
f - 4x_1 - 5x_2 &= 0 \\
-x_1 - 4x_2 + x_3 &= -5 \\
-3x_1 - 2x_2 + x_4 &= -7
\end{align*}
\]
Dual simplex method:

<table>
<thead>
<tr>
<th></th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$x_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f$</td>
<td>-4</td>
<td>-5</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$x_3$</td>
<td>-1</td>
<td>-4</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$x_4$</td>
<td>-3</td>
<td>-2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$f$</td>
<td>0</td>
<td>-7/3</td>
<td>0</td>
<td>-4/3</td>
</tr>
<tr>
<td>$x_3$</td>
<td>0</td>
<td>-10/3</td>
<td>1</td>
<td>-1/3</td>
</tr>
<tr>
<td>$x_1$</td>
<td>1</td>
<td>2/3</td>
<td>0</td>
<td>-1/3</td>
</tr>
<tr>
<td>$f$</td>
<td>0</td>
<td>0</td>
<td>-7/10</td>
<td>-11/10</td>
</tr>
<tr>
<td>$x_2$</td>
<td>0</td>
<td>1</td>
<td>-3/10</td>
<td>1/10</td>
</tr>
<tr>
<td>$x_1$</td>
<td>1</td>
<td>0</td>
<td>2/10</td>
<td>-4/10</td>
</tr>
</tbody>
</table>

Optimal solution of the LP-relaxation.

Last row as the source row: $x_1 + 2/10 x_3 - 4/10 x_4 = 18/10$
\[ x_1 + \frac{2}{10} x_3 - \frac{4}{10} x_4 = \frac{18}{10} \]

\[ \iff \]
\[ x_1 + (0 + \frac{2}{10}) x_3 + (-1 + \frac{6}{10}) x_4 = 1 + \frac{8}{10} \]

Cut constraint

\[ \frac{2}{10} x_3 + \frac{6}{10} x_4 \geq \frac{8}{10} \]

Basic form

\[ -\frac{2}{10} x_3 - \frac{6}{10} x_4 + s_1 = -\frac{8}{10} \]

Continue with the dual simplex method.
<table>
<thead>
<tr>
<th></th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$x_4$</th>
<th>$s_1$</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$f$</td>
<td>0</td>
<td>0</td>
<td>-7/10</td>
<td>-11/10</td>
<td>0</td>
<td>112/10</td>
</tr>
<tr>
<td>$x_2$</td>
<td>0</td>
<td>1</td>
<td>-3/10</td>
<td>1/10</td>
<td>0</td>
<td>8/10</td>
</tr>
<tr>
<td>$x_1$</td>
<td>1</td>
<td>0</td>
<td>2/10</td>
<td>-4/10</td>
<td>0</td>
<td>18/10</td>
</tr>
<tr>
<td>$s_1$</td>
<td>0</td>
<td>0</td>
<td>-2/10</td>
<td>-6/10</td>
<td>1</td>
<td>-8/10</td>
</tr>
<tr>
<td>$f$</td>
<td>0</td>
<td>0</td>
<td>-2/6</td>
<td>0</td>
<td>-11/6</td>
<td>76/6</td>
</tr>
<tr>
<td>$x_2$</td>
<td>0</td>
<td>1</td>
<td>-2/6</td>
<td>0</td>
<td>1/6</td>
<td>4/6</td>
</tr>
<tr>
<td>$x_1$</td>
<td>1</td>
<td>0</td>
<td>2/6</td>
<td>0</td>
<td>-4/6</td>
<td>14/6</td>
</tr>
<tr>
<td>$x_4$</td>
<td>0</td>
<td>0</td>
<td>2/6</td>
<td>1</td>
<td>-10/6</td>
<td>8/6</td>
</tr>
</tbody>
</table>

Second row as a source row:
\[ x_2 - \frac{2}{6} x_3 + \frac{1}{6} s_1 = \frac{4}{6} \]
\[ \iff x_2 + (-1 + \frac{4}{6}) x_3 + (0 + \frac{1}{6}) s_1 = 0 + \frac{4}{6} \]

Cut constraint:
\[ \frac{4}{6} x_3 + \frac{1}{6} s_1 \geq \frac{4}{6}. \]
Equation to the simplex tableau:
\[ -\frac{4}{6} x_3 - \frac{1}{6} s_1 + s_2 = -\frac{4}{6} \]
<table>
<thead>
<tr>
<th></th>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>$x_4$</th>
<th>$s_1$</th>
<th>$s_2$</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$f$</td>
<td>0</td>
<td>0</td>
<td>-2/6</td>
<td>0</td>
<td>-11/6</td>
<td>0</td>
<td>76/6</td>
</tr>
<tr>
<td>$x_2$</td>
<td>0</td>
<td>1</td>
<td>-2/6</td>
<td>0</td>
<td>1/6</td>
<td>0</td>
<td>4/6</td>
</tr>
<tr>
<td>$x_1$</td>
<td>1</td>
<td>0</td>
<td>2/6</td>
<td>0</td>
<td>-4/6</td>
<td>0</td>
<td>14/6</td>
</tr>
<tr>
<td>$x_4$</td>
<td>0</td>
<td>0</td>
<td>2/6</td>
<td>1</td>
<td>-10/6</td>
<td>0</td>
<td>8/6</td>
</tr>
<tr>
<td>$s_2$</td>
<td>0</td>
<td>0</td>
<td>-4/6</td>
<td>0</td>
<td>-1/6</td>
<td>1</td>
<td>-4/6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>$f$</th>
<th>$x_2$</th>
<th>$x_1$</th>
<th>$x_4$</th>
<th>$s_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-7/4</td>
</tr>
<tr>
<td>$x_2$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1/4</td>
</tr>
<tr>
<td>$x_1$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-3/4</td>
</tr>
<tr>
<td>$x_4$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>-7/4</td>
</tr>
<tr>
<td>$x_3$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1/4</td>
</tr>
</tbody>
</table>

Integer solution found: $x_1 = 2, x_2 = 1$ ja $f_{\text{min}} = 13$. 
COMBINATION: BRANCH AND CUT

In the *Branch and cut method* these approaches are combined. Cut constraints are added to the LP-relaxation before applying the Branch & Bound. By producing tighter bounds and reducing the feasible set, the fathoming becomes more efficient, which means a smaller number of subproblems have to be generated.