More LP and ILP
Alon Efrat
Integer Linear Programming (ILP)

- Linear programming problems at which values of the computed variables must be integers are called Integer Linear Programming (ILP) problems.
- If only some of the variables must be integers, we call them Mixed Integer Linear Programming problems.
- There is a huge number of problems that could be phrased as ILP. (include many NP-hard problems, where no polynomial-time algorithms exist)
- A few libraries could handle them, including CPLEX.
- Running time could varies a lot, and could be extremely slow for some instances.
- Yet extremely useful for instances when actual running time is acceptable.
- Also useful for comparing fast heuristic to global optimum.

- Big difference between LP and ILP:
  - If we could express a problem as an LP, practically we could consider it solved: LP is very fast in practice for any realistic input. Libraries are easy to use.
  - If we could express a problem as an ILP, the libraries are still easy to use, but running time varies a lot. Not always we will live long enough to see the program terminates.
Application: Bipartite Matching.

- A graph $G(V,E)$ is called bipartite if $V$ can be partitioned into two sets $V=A\cup B$, and each edge of $E$ connects a vertex of $A$ to a vertex of $B$. We sometimes denote these graphs by $G(A\cup B,E)$.

- Example: The set $A = \{a_1\ldots a_n\}$ is a set of instructors, the set $B = \{b_1\ldots b_n\}$ is the set of courses. There is an edge $(a_i, b_j) \in E$ iff instructor $a_i$ could teach course $b_j$.

- A matching is a set of edges $M$ of $E$, where each vertex of $A$ is adjacent to at most one vertex of $B$, and vice versa.

- (in the example, each instructor will teach at most one course, and vice versa)

- Maximum-cardinality matching: Find a matching with as many edges as possible.

- This problem could be solved with in $O(nm)$ time using Ford-Fulkerson algorithm. Faster algorithms exist as well. However, we will use it as an example to the ease of using ILP.
ILP for Max-Cardinality Bipartite Matching.

- For every edge $e$, define a **Boolean** variable $x_e$.
- $x_e = 1$ if $e$ participates in the matching $M$, and $x_e = 0$ otherwise.
- The goal is to maximize the number of edges in $M$, while keeping $M$ a proper matching.

Cost function: maximize $\sum_{i=1}^{m} x_i$

Subject to

1. $0 \leq x_i \leq 1$ for every $x_i$

2. for every $a \in A$ $\sum_{(a,b_j) \in E} x_{a,b} \leq 1$ (an instructor will teach at most one course)

3. for every $b \in B$ $\sum_{(a_i,b) \in E} x_{a_i,b} \leq 1$ (an course will taught by at most one instructor)

4. $x_i$ is an integer, for every $x_i$
Vertex Cover and ILP

- Given: A graph G(V,E). A subset C ⊆ V is a vertex cover if every edge (u, v) ∈ E we have either u ∈ C or v ∈ C or both.
- Finding the min-cardinality Vertex Cover is NP-Hard
- ILP for this problem: We will use boolean variables x₁…xₙ (one per vertex).
- \( v_i \in C \) iff \( x_i = 1 \) (for \( i = 1…n \))
- Formalizations of ILP:
- Variables: \( x_1…x_n \)
  
  Cost function: minimize \( \sum_{i=1}^{n} x_i \)

Subject to:
1. \( 0 \leq x_i \leq 1 \) for every \( x_i \)
2. \( x_i + x_j \geq 1 \quad \forall (v_i, v_j) \in E \)
   // either \( v_i \) or \( v_j \) are in \( C \)
3. Each \( x_i \) is an integer,
Vertex Cover and ILP

- Given: A graph $G(V,E)$. A subset $C \subseteq V$ is a vertex cover if every edge $(u,v) \in E$ we have either $u \in C$ or $v \in C$ or both.
- Finding the **min-cardinality** Vertex Cover is NP-Hard
- ILP for this problem: We will use boolean variables $x_1 \ldots x_n$ (one per vertex).
- $v_i \in C \iff x_i = 1$ (for $i = 1 \ldots n$)
- Formalizations of ILP:
- Variables: $x_1 \ldots x_n$
  - Cost function: minimize $\sum_{i=1}^{n} x_i$
  - **Subject to:**
    1. $0 \leq x_i \leq 1$ for every $x_i$
    2. $x_i + x_j \geq 1 \ \forall (v_i, v_j) \in E$
       // either $v_i$ or $v_j$ are in $C$
    3. Each $x_i$ is an integer,
Vertex Cover and ILP

- Given: A graph G(V, E). A subset C ⊆ V is a vertex cover if every edge (u, v) ∈ E we have either u ∈ C or v ∈ C or both.
- Finding the min-cardinality Vertex Cover is NP-Hard
- ILP for this problem: We will use boolean variables x₁…xₙ (one per vertex).

  
- vᵢ ∈ C iff xᵢ = 1 (for i = 1…n)
- Formalizations of ILP:
- Variables: x₁…xₙ

  Cost function: minimize \[ \sum_{i=1}^{n} x_i \]

Subject to:
1. \[ 0 \leq x_i \leq 1 \] for every xᵢ
2. \[ x_i + x_j \geq 1 \] \( \forall (v_i, v_j) \in E \) // either vᵢ or vⱼ are in C
3. Each xᵢ is an integer,
Vertex Cover and ILP

- Given: A graph $G(V,E)$. A subset $C \subseteq V$ is a vertex cover if every edge $(u, v) \in E$ we have either $u \in C$ or $v \in C$ or both.
- Finding the min-cardinality Vertex Cover is NP-Hard
- ILP for this problem: We will use boolean variables $x_1\ldots x_n$ (one per vertex).
- $v_i \in C \iff x_i = 1$ (for $i = 1\ldots n$)
- Formalizations of ILP:

Variables: $x_1\ldots x_n$

Cost function: $\text{minimize } \sum_{i=1}^{n} x_i$

Subject to:

1. $0 \leq x_i \leq 1$ for every $x_i$
2. $x_i + x_j \geq 1$ $\forall (v_i, v_j) \in E$
   // either $v_i$ or $v_j$ are in C
3. Each $x_i$ is an integer,

Toward efficient approximation, lets define

- $\vec{x}^f = (x_1^f \ldots x_n^f)$ be the solution to the LP. (drop the requirement to integer solution.
- $\vec{x}^{opt} = (x_1^{opt} \ldots x_n^{opt})$ be the solution to the ILP.
- $\vec{x}^{app} = (x_1^{app} \ldots x_n^{app})$ be an approximated integer solution, obtained as followed:
  
  if $x_i^f < 0.5$ then $x_i^{app} = 0$
  if $x_i^f \geq 0.5$ then $x_i^{app} = 1$
Vertex Cover and ILP

- Given: A graph $G(V,E)$. A subset $C \subseteq V$ is a vertex cover if every edge $(u,v) \in E$ we have either $u \in C$ or $v \in C$ or both.
- Finding the min-cardinality Vertex Cover is NP-Hard
- ILP for this problem: We will use boolean variables $x_1\ldots x_n$ (one per vertex).
- $v_i \in C$ iff $x_i = 1$ (for $i = 1\ldots n$)
- Formalizations of ILP:
  - Variables: $x_1\ldots x_n$
  - Cost function: minimize $\sum_{i=1}^{n} x_i$
  - Subject to:
    1. $0 \leq x_i \leq 1$ for every $x_i$
    2. $x_i + x_j \geq 1$ $\forall (v_i,v_j) \in E$ // either $v_i$ or $v_j$ are in $C$
    3. Each $x_i$ is an integer,

Towards efficient approximation, let's define
- $\vec{x}^f = (x_1^f \ldots x_n^f)$ be the solution to the LP. (drop the requirement to integer solution.
- $\vec{x}^{opt} = (x_1^{opt} \ldots x_n^{opt})$ be the solution to the ILP.
- $\vec{x}^{app} = (x_1^{app} \ldots x_n^{app})$ be an approximated integer solution, obtained as followed:
  - if $x_i^f < 0.5$ then $x_i^{app} = 0$
  - if $x_i^f \geq 0.5$ then $x_i^{app} = 1$

Claims
1. $\vec{x}^{app}$ is a vertex cover
2. $\text{cost}(\vec{x}^f) \leq \text{cost}(\vec{x}^{opt})$
3. $\text{cost}(\vec{x}^{app}) \leq 2\text{cost}(\vec{x}^f) \leq 2\text{cost}(x^{opt})$
Flow networks

Definition. A flow network is a directed graph $G = (V, E)$ with two distinguished vertices: a source $s$ and a sink $t$. Each edge $(u, v) \in E$ has a nonnegative capacity $c(u, v)$.

The capacity limits the number of gallons/seconds (or MB/second, or vehicle/second) that the edge could tolerate.

Goal- push as much flow as possible, from $s$ to $t$.

Example:
**Flow in Networks**

**Def:** A *flow* on $G$ is an assignment of a value $f(u,v)$, for every edge $(u, v) \in E$. These values must satisfy several constraints:

- **Capacity constraint:** $0 \leq f(u, v) \leq c(u, v)$, for every $(u, v) \in E$
- **Flow conservation:** For every vertex $u \in V$, which is not the source nor the sink, the flow arriving into $u$ must be equal to the total flow that leaves $u$. Formally

\[
\sum_{(v,u) \in E} f(v,u) = \sum_{(u,w) \in E} f(u,w)
\]

The *value* of a flow is the sum of the flows out of the source:

\[
\sum_{(s,a) \in E} f(s,a)
\]

In the example, the value of the flow equals $1+2=3$

Goal: Assign a flow to every edge, (legally), so the value of the flow is **maximize**.

Example: Flow into $b$ is $2 + 1 = 3$. Flow out of $b$ is $1 + 2 = 3$. 

![Diagram of network with flow values and flow direction](image)
The maximum-flow problem

**Maximum-flow problem:** Given a flow network $G$, find a flow of maximum value on $G$.

The value of the maximum flow is 4.
Lemma: The value of the flow equals to the sum of flows entering $t$

\[ \sum_{(s,v) \in E} f(s, v) = \sum_{(w,t) \in E} f(w, t) \]

- $\sum_{(s,v) \in E} f(s, v)$: total flow from $s$
- $\sum_{(w,t) \in E} f(w, t)$: total flow into $t$
LP could solve flow problems (but values might be non-integers)

Given: The graph, special vertices $s, t \in E$, and the capacities $c(u, v)$ for every edge $(u, v) \in E$

Define a linear programming problem (LP) that will find the maximum (legit) flow:

• **Unknown variables**: $f(u, v)$ for every edge $(u, v) \in E$.

  • If you find the usage of the parenthesis confusing, we could just name the vertices $V = \{v_1 \ldots v_n\}$ and the variables are $f_{i,j}$ for every edge $(v_i, v_j) \in E$

• **Cost Function**: Maximizes the flow from $s$.

  \[
  \max \sum_{(s,u) \in E} f(s, u)
  \]

• **Capacity constraint**: $0 \leq f(u, v)$ and $f(u, v) \leq c(c, u)$ for every edge $(u, v) \in E$

• **Flow conservation**: For every vertex $u \in V$, which is not the source nor the sink, the flow arriving into $u$ must be equal to the total flow that leaves $u$. Formally

\[
\sum_{(v,u) \in E} f(v, u) - \sum_{(u,w) \in E} f(u, w) = 0
\]

\[
\text{total flow into } u \quad \text{total flow from } u
\]
(cs445 only:) Min-Weight Vertex Cover: Exact and Approximated solutions:

- Sometimes the LP (instead of the ILP) could help us finding good approximations.
- Given: A graph $G(V,E)$. A subset $C \subseteq V$ is a vertex cover if for every edge $(u,v) \in E$, at least one of its endpoints is in $C$.
- Each vertex $v_i$ is given with a cost (or weight) $w_i > 0$.
- $w(C)$, the cost of $C$, is defined as the sum of costs of vertices in $C$.
- Finding the **min-cardinality or min-weight** Vertex Cover is NP-Hard.
- ILP for this problem: The variables are $\vec{x} = (x_1 \ldots x_n)$. All are integers and between 0 and 1.
- $v_i \in C$ iff $x_i = 1$ (for $i = 1 \ldots n$)
- Let $\vec{w} = (w_1 \ldots w_n)$

$C^{opt} = \{v_4, v_6\}$. and $w(C^{opt}) = 4 + 8 = 12$
Min-Weight Vertex Cover: Exact and Approximated solutions:

- Sometimes the LP (instead of the ILP) could help us finding good approximations.
- Given: A graph $G(V,E)$. A subset $C \subseteq V$ is a vertex cover if for every edge $(u,v) \in E$, at least one of its endpoints is in $C$.
- Each vertex $v_i$ is given with a cost (or weight) $w_i > 0$.
- $w(C)$, the cost of $C$, is defined as the sum of costs of vertices in $C$.
- Finding the min-cardinality or min-weight Vertex Cover is NP-Hard.
- ILP for this problem: The variables are $\overrightarrow{x} = (x_1 \ldots x_n)$. All are integers and between 0 and 1.
- $v_i \in C$ iff $x_i = 1$ (for $i = 1 \ldots n$)
- Let $\overrightarrow{w} = (w_1 \ldots w_n)$

**ILP:** Cost function: $\text{minimize } \overrightarrow{w} \cdot \overrightarrow{x} = \sum_{i=1}^{n} w_i x_i$

s.t. $x_i + x_j \geq 1 \quad \forall (v_i, v_j) \in E$
$0 \leq x_i \leq 1$ for every $x_i$.

In ILP, we will also required that each $x_i$ is an integer. In LP, this is not required.

- $\overrightarrow{x}^f = (x_1^f \ldots x_n^f)$ be the solution to the LP.
- $\overrightarrow{x}^{opt} = (x_1^{opt} \ldots x_n^{opt})$ - the solution to ILP. Let $C^{opt}$ be the corresponding cover.
- $\overrightarrow{x}^{app} = (x_1^{app} \ldots x_n^{app})$ be an approximate solution integer solution, obtained as followed:

  - if $x_i^f < 0.5$ then $x_i^{app} = 0$ and $v_i \not\in C^{app}$
  - if $x_i^f \geq 0.5$ then $x_i^{app} = 1$ and $v_i \in C^{app}$

$C^{opt} = \{v_4, v_6\}$. and $w(C^{opt}) = 4 + 8 = 12$. 
Min-Weight Vertex Cover: Exact and Approximated solutions:

- Sometimes the LP (instead of the ILP) could help us finding good approximations
- Given: A graph G(V,E). A subset \( C \subseteq V \) is a vertex cover if for every edge \((u, v) \in E\), at least one of its endpoints is in C.
- Each vertex \( v_i \) is given with a cost (or weight) \( w_i > 0 \).
- \( w(C) \), the cost of C, is defined as the sum of costs of vertices in C.
- Finding the min-cardinality or min-weight Vertex Cover is NP-Hard.
- ILP for this problem: The variables are \( \vec{x} = (x_1, ..., x_n) \). All are integers and between 0 and 1.
- \( v_i \in C \) iff \( x_i = 1 \) (for \( i = 1, ..., n \))
- Let \( \vec{w} = (w_1, ..., w_n) \)

ILP: Cost function: \( \text{minimize } \vec{w} \cdot \vec{x} = \sum_{i=1}^{n} w_i x_i \)

s.t.
\[ x_i + x_j \geq 1 \quad \forall (v_i, v_j) \in E \]
\[ 0 \leq x_i \leq 1 \quad \text{for every } x_i. \]

In ILP, we will also required that each \( x_i \) is an integer. In LP, this is not required.

- \( \vec{x}^f = (x_1^f, ..., x_n^f) \) be the solution to the LP.
- \( \vec{x}^{opt} = (x_1^{opt}, x_2^{opt}, ..., x_n^{opt}) \) -the solution to ILP. Let \( C^{opt} \) be the corresponding cover.
- \( \vec{x}^{app} = (x_1^{app}, x_2^{app}, ..., x_n^{app}) \) be an approximate solution integer solution, obtained as followed:
  - if \( x_i^f < 0.5 \) then \( x_i^{app} = 0 \) and \( v_i \not\in C^{app} \)
  - if \( x_i^f \geq 0.5 \) then \( x_i^{app} = 1 \) and \( v_i \in C^{app} \)

Claim 1. \( \vec{x}^{app} \) is a vertex cover

Claim 2. \( \vec{w} \cdot \vec{x}^f \leq \vec{w} \cdot \vec{x}^{opt} \)

Claim 3. \( \vec{w} \cdot \vec{x}^{app} \leq 2 \vec{w} \cdot \vec{x}^f \leq 2 \vec{w} \cdot \vec{x}^{opt} \)

In words:

The approximation costs \( \leq 2 \cdot \text{opt} \),

We say that \( C^{opt} \) is a 2-approximation of the weighted vertex cover.
Let OPT be the opt solution.
- Every chosen edge $e$ has both ends in $C$.
- But $e$ must be covered by at least one vertex of OPT. So, one end of $e$ must be in OPT.
- $|C| \leq 2 \cdot |\text{OPT}|$.
- (there are $\leq 2$ vertices of $C$ for each vertex of OPT.)
- That is, $C$ is a 2-approximation of OPT.
- Running time: $O(|E|)$

**Algorithm** \textit{VertexCoverApprox}(G)

**Input** graph $G$

**Output** a vertex cover $C$ for $G$

$C \leftarrow \text{empty set} ; \ H \leftarrow E$

\text{/* $H$ -- what is left to be covered */}
while $H$ has edges (not empty)\
    $(u,v) \leftarrow \text{An edge of } H.$
    Add both $u$ and $v$ to $C$
    for each edge $f$ of $H$ incident to $v$ or $w$
        Remove $f$ from $H$

return $C$