Combinatorial Optimization and Integer Linear Programming

Combinatorial Optimization: Introduction

Many problems arising in practical applications have a special, discrete and finite, nature:

Definition. (Linear Combinatorial Optimization Problem)

Given

- a finite set $E$ (the ground set),
- a subset $F \subseteq 2^E$ (the set of feasible solutions),
- a cost function $c : E \rightarrow \mathbb{R}$,

find a set $F^* \in F$ such that

$$c(F^*) := \sum_{e \in F^*} c(e)$$

is maximal or minimal.

Examples: Shortest Path, Traveling Salesman, and many many more . . .

Just in bioinformatics: Alignments, Threading, Clone-Probe Mapping, Probe Selection, De Novo Peptide Sequencing, Side-Chain Placement, Maximum-weight Connected Subgraph in PPI Networks, Genome Rearrangements, Cluster Editing, Finding Regulatory Modules, Finding Approximate Gene Clusters, and many more . . .

Combinatorial Optimization: Introduction (2)

Example. Optimal Microarray Probe Selection

Experimental setup (group testing):

- Goal: determine presence of targets in sample
- probes hybridize with targets $\rightarrow$ hybridization pattern

Selection phase:

- unique probes are easy to decode but difficult to find (similarities, errors, add. constraints, . . .)
- consider non-unique probes
- Task: choose few probes that still allow to infer which targets are in the sample

Combinatorial Optimization: Introduction (3)

Example hybridization matrix $(H)_{ij}$:
Assume: no errors, only one target present in sample

### Combinatorial Optimization: Introduction

Example hybridization matrix \((H)_{ij}\):

<table>
<thead>
<tr>
<th></th>
<th>(P_1)</th>
<th>(P_2)</th>
<th>(P_3)</th>
<th>(P_4)</th>
<th>(P_5)</th>
<th>(P_6)</th>
<th>(P_7)</th>
<th>(P_8)</th>
<th>(P_9)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(t_1)</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(t_2)</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>(t_3)</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>(t_4)</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Assume: no errors, only one target present in sample

### Combinatorial Optimization: Introduction

Example hybridization matrix \((H)_{ij}\):

<table>
<thead>
<tr>
<th></th>
<th>(P_1)</th>
<th>(P_2)</th>
<th>(P_3)</th>
<th>(P_4)</th>
<th>(P_5)</th>
<th>(P_6)</th>
<th>(P_7)</th>
<th>(P_8)</th>
<th>(P_9)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(t_1)</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(t_2)</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>(t_3)</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>(t_4)</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Assume: no errors, two targets present, e.g., \(t_2\) and \(t_3\)

### Combinatorial Optimization: Introduction

Example hybridization matrix \((H)_{ij}\):
We want to solve the following problem.

**Definition.** Probe Selection Problem (PSP)

- Given an incidence matrix $H$, $d \in \mathbb{N}$, and $c \in \mathbb{N}$,
- find the smallest subset $D \subseteq \mathbb{N}$, such that
  - all targets are covered by at least $d$ probes
  - all different subsets of targets $S$ and $T$ up to cardinality $c$ are $d$-separable with respect to $D$

**Observation.** PSP is a combinatorial optimization problem, because

- ground set = candidate probes, i.e., $E := \{1, 2, ..., n\}$.
- feasible solutions = feasible designs, i.e.,
  $$\mathcal{F} := \{D \in 2^E \mid D \text{ satisfies coverage and separation constraints}\}$$
- all costs $c(e) := 1$.

**Combinatorial Optimization: Introduction**

*More examples.* What about

$$\min \{3x^2 + 2 \mid x \in \mathbb{R}\} ?$$

Or

$$\max 2x_1 + 3x_2$$

s.t.  

$$x_1 + 2x_2 \leq 3$$

$$3x_1 - x_2 \leq 5$$

$$x_1, x_2 \in \mathbb{N} ?$$

Interesting combinatorial problems have an exponential number of feasible solutions. [Otherwise, a straightforward polynomial-time algorithm finds optimal solutions.]

Combinatorial optimization: find solutions faster than by complete enumeration.

**Combinatorial Optimization**

Now, given a combinatorial optimization problem $C = (E, \mathcal{F}, c)$, we define, for each feasible solution $F \in \mathcal{F}$, its characteristic vector $\chi^F \in \{0, 1\}^E$ as

$$\chi^F_e := \begin{cases} 1 & e \in F \\ 0 & \text{otherwise} \end{cases}.$$
Then, assuming the objective is to maximize, \( C \) can be seen as maximizing over a polytope, i.e.,

\[
\max \{ c^T x \mid x \in \text{conv} \{ \chi_F \in \{0, 1\}^E \mid F \in \mathcal{F} \} \}.
\]

Why polytope?

**Theorem.** (Minkowski 1896, Weyl 1935)

Each polytope \( P = \{ x \in \mathbb{R}^n \mid Ax \leq b, l \leq x \leq u \} \) can be written as

\[
P = \text{conv}(V)
\]

where \( V \) is a finite subset of \( \mathbb{R}^n \) and vice versa.

**Combinatorial Optimization**

It is possible to switch between these descriptions as \( \mathcal{H} \)-polytope (halfspaces) and \( \mathcal{V} \)-polytope (vertices) with the Fourier-Motzkin elimination method.

**Example.**

Consider the \( \mathcal{V} \)-polytope defined by

\[
P = \text{conv}\left\{ \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \end{pmatrix} \right\}
\]

**End (Example).**

So, we can just compute the \( \mathcal{H} \)-polytope \( \{ x \in \mathbb{R}^n \mid Ax \leq b \} \) for \( C \) and optimize over it using, e.g., the Simplex method?

Unfortunately, it is not so easy:

- In general, we cannot find \( A \) and \( b \) in polynomial time.
- The size of \( A \) and \( b \) might be exponential.
- The coefficients in \( A \) and \( b \) can be exponentially large.

**Combinatorial Optimization**

A little bit of light... often, finding an integer linear programming (ILP) formulation is easier:

\[
\max \{ c'^T x' \mid A' x' \leq b, x' \in \mathbb{Z} \}.
\]

But: solving LPs is easy, solving ILPs is not!