Optimization

randomized rounding of LPs

Example: Diet Problem

Athlete wants to minimize protein consumption while consuming 5 units of fat/day

\[ \text{subject to } \begin{align*}
-2x_1 + x_2 & \leq 5 \\
x_1 + 2x_2 & \leq 5 \\
x_1, x_2 & > 0
\end{align*} \]

\[ \begin{array}{c|cc}
\text{protein/lb} & 2 & 1 \\
\text{fat/lb} & 4 & 2 \\
\text{panda/berry/lb} & 1 & 2
\end{array} \]

\[ x_1 \text{ # lbs of steak/day} \]
\[ x_2 \text{ # lbs of PB/day} \]

\[ \text{max } 2x_1 + x_2 \]
\[ \text{subject to } \begin{align*}
-2x_1 + x_2 & \leq 5 \\
x_1 + 2x_2 & \leq 5 \\
x_1, x_2 & > 0
\end{align*} \]

Linear cost fns define family of parallel hyperplanes.

Optimal feasible pt must occur at corner, a.k.a. vertex.

Feasible set is "polyhedron": intersection of halfspaces if also bounded & nonempty. "Polytope" = bounded polyhedron.

Feasible set convex.

Unfortunately, too many vertices to enumerate. \( \approx \frac{m^2}{n} \)

Ex: \( x_1 \leq 1 \) is in \( n \)-dimensional hypercube.
Input to LP problem

\[ \text{max } c^T x \]
\[ \text{subject to } Ax \leq b \]
\[ x \geq 0 \]

There are efficient algs for LP:
- Simplex
- Ellipsoid
- Interior pt methods

Bottomline: If you can formulate your problem as LP, it can be solved in time \( \text{poly}(m, n, \text{size of LP}) \)

MAXSAT
Given a Boolean formula in CNF (and sat)
- \( n \) Boolean vars \( x_1, \ldots, x_n \)
- \( m \) clauses \( C_1, \ldots, C_m \)
  - Each clause is disjunction of \( k \) literals
  - Each clause is some \( \bar{x}_i \) vars & \( \bar{x}_j \) negations
  - e.g. \( x_3 \lor \bar{x}_5 \lor x_7 \)
- weight \( q \) on \( j \)th clause

MAXSAT: find assignment that maximizes \( \sum_{j} q_j \)-satisfied clauses

MAX K SAT: all clauses have \( \leq k \) literals
MAX EK SAT: all clauses have exactly \( k \) literals
Random assignment

We will consider \( \text{MAX 2-SAT} \) all clauses have 1 or 2 literals.

\[
\begin{align*}
\text{Step 1: } & \quad \text{Formulate MAX-2-SAT as integer linear program} \\
& \quad \text{where we require all vars to be } 0, 1, 13 \\
\text{Suppose input formula has } n \text{ vars, } m \text{ clauses.} \\
\text{Define vars } & \quad x_1, \ldots, x_n \text{ vars in formula} \\
& \quad z_1, \ldots, z_m \text{ vars in formula} \\
\text{where } & \quad x_i = \begin{cases} 0 & \text{if clause is not satisfied} \\ 1 & \text{else} \end{cases} \\
& \quad z_j = \begin{cases} 0 & \text{if clause satisfied} \\ 1 & \text{else} \end{cases} \\
\text{Constraints:} & \quad \sum_{i=1}^{n} x_i = k \\
& \quad \sum_{j=1}^{m} z_j = \frac{m}{2} \\
& \quad x_i \leq z_i \\
& \quad x_i \geq 0, z_i \geq 0 \quad \forall i, j
\end{align*}
\]
**Step 2:** Relax to a LP and sum \( \Rightarrow \frac{z^*}{, z_j, x_i, x^n} \) fractional

\[
\max \sum_{j} w_j z_j
\]

\[
\sum_{i \in C_j} y_{ij} \geq z_j \quad j = 1, \ldots, m
\]

where \( y_{ij} = \begin{cases} x_i & \text{if } v_i \text{ pos in } C_j \\ 1-x_i & \text{neg in } C_j \end{cases} \)

\( x_i \in \{0,1\} \quad i = 1 \ldots n \)

\( z_j \in \{0,1\} \quad j = 1 \ldots m \)

**Key observation:**

**Step 3:** Round the solution \( \Rightarrow \) integer solution

integer linear programming is NP-hard though!
Corollary:

- Similar analysis shows randomized rounding gives $1 - \frac{1}{2}$ approx alg for MAXSAT.
- Choosing better $f$ - randomized rounding soln
  - unbiased randomized alg
  $\Rightarrow$ $\frac{34}{35}$ approx alg
- Solving LP & then setting $x_i = \begin{cases} 1 & \text{True w.p. } f(x_i) \\ 0 & \text{False w.p. } 1 - f(x_i) \end{cases}$

$\Rightarrow$ $\frac{34}{35}$ approx alg.

Best possible if we compare our solution against $OPT_{LP}$.

Ex.: $(x_1 \lor x_2) \land (x_3 \lor \overline{x_2}) \land (x_4 \lor x_5) \land (x_6 \lor \overline{x_3})$
Randomized rounding summary

An important technique for obtaining approx algs for NP-hard problems

Recipe

1. Set problem up as integer linear program. vars \( x_i \)
2. Relax ILP \( \rightarrow \) LP \( \quad \forall x_i \in [0,1] \)
3. Solve LP to optimality \( \rightarrow x_i^* \)
4. Construct solution by randomly rounding \( x_i \rightarrow [0,1] \)
   \( \text{treat } x_i^* \text{ as probability} \)
5. bound quality of soln by comparing to LP orig

Another example: Congestion minimization

Input: directed graph \( G=(V,E) \)
set of pairs \((s_i,t_i) \) \( \forall i=1:k \)
Output: path \( P_i \) from \( s_i \rightarrow t_i \) \( \forall i=1:k \)
\( s, t \) \( \text{Congestion is minimized} \)
\( \text{route } P_i \text{ that intersect any edge} \)

NP-hard

Approx alg via randomized rounding

1. Set up ILP [multi commodity flow]
   \[ \begin{align*}
   \text{vars } & f_i(e) \in [0,1], \text{ flow from } s_i \rightarrow t_i \text{ on edge } e \\
   \text{C } & \text{ congestion}
   \end{align*} \]
   \[ \begin{align*}
   \min C & \\
   \text{subject to } & \sum_{e \in \text{flow } s \rightarrow t} f_i(e) = \sum_{e \in \text{flow } s \rightarrow t} f_i(e) \quad \forall e \in E, \forall i \\
   & \sum_{e \in \text{flow } s \rightarrow t} f_i(e) = 1 \quad \forall i \\
   & \sum_{e \in \text{flow } s \rightarrow t} f_i(e) \leq C \quad \forall i, e
   \end{align*} \]

2. Relax to LP

3. Solve LP
Round sdn

5. Bound result.

Theorem. If \( C^2 < \text{clean} \) for some cost \( c \), then with the max congestion on any edge \( e \in C^2 + \sqrt{\text{max}} \) approx ratio \( \leq 2 \).

Proof. Fix \( e = (u \to v) \), let \( x(e) = \left\{ \begin{array}{ll} 1 & \text{if } u \to v \text{ is } \text{opt} \text{.} \\ 0 & \text{otherwise} \end{array} \right. \),

\[
E(x,\pi) = \sum_{P \in \mathcal{P}} f_P \cdot x(e).
\]
If only knew $C^k \geq 1$

**Theorem** \[ \Pr(\text{any edge has congestion } \geq \frac{6 \ln n}{\ln \ln n}, \text{ badness } C^k) \leq \frac{1}{n} \]

**Proof**

$\Pr(\text{any edge has congestion } \geq \frac{6 \ln n}{\ln \ln n}, \text{ badness } C^k) \leq \frac{1}{n}$

Let $X(e)=\sum_{i=1}^{\infty} e_i e_i^c$

$\Pr(X(e) \geq (1+\delta) C^k) \leq e^{-\frac{1}{C^k(1+\delta) - 1}}$ [Chernoff bound]

$\Pr(X(e) > \alpha C^k) \leq e^{-\frac{1}{\alpha C^k - 1}}$

$\leq e^{-\frac{3 \ln n}{\alpha C^k - 1}} \leq \frac{1}{n}$

$\Rightarrow \Pr(\text{any edge has congestion } \geq \alpha C^k) \leq \sum_e \Pr(X(e) > \alpha C^k)$

$\leq n \cdot \frac{1}{n} = \frac{1}{n}$

Optimal:

For graphs with $\frac{\text{opt}}{\text{cbr}} = \mathcal{O}\left(\frac{\ln \ln n}{\ln \ln n}\right)$ it is unlikely

So cannot do better with such an approach

**Hardness (directed graphs)**

Every poly time alg has $\Omega\left(\frac{\ln n}{\ln \ln n}\right)$ approx ratio

unless $\text{NP} \subseteq \text{BPTIME}(n^{o(n)})$
Inputs $G=(V,E)$ $w_{ij} \forall (i,j) \in E$

Goal: partition vertex set so as to max weight of edges crossing cut.

**IP formulation of MAXCUT**

$$x_i \in \{0,1\} \quad \text{on side 1 of partition}$$

$$z_{ij} = \begin{cases} 1 & \text{edge $(i,j)$ cut} \\ 0 & \text{o.w.} \end{cases}$$

$$\max \sum_{(i,j) \in E} w_{ij} z_{ij}$$

$$z_{ij} \leq x_i - x_j \quad \forall (i,j) \in E$$

$$z_{ij} \leq 2(1 - x_i - x_j) \quad \forall (i,j) \in E$$

$$x_i \in [0,1] \quad \forall i \in V$$

$$z_{ij} \in [0,1] \quad \forall (i,j) \in E$$
Another approach

First, notation change:
\[ \forall i \quad x_i \in \{-1, +1\} \]
\[ \text{define } y_{ij} = x_i x_j \quad \forall i, j \in V \]

Want \( E(x; V) \)
\[ \text{s.t. } y_{ij} = x_i x_j \forall i, j \]

\[ \max \sum_{(i,j) \in E} \frac{1}{2} (1 - y_{ij}) \]
\[ y_{ij} = y_{ji} \quad \forall i, j \in V \]
\[ y_{ii} = 1 \quad \forall i \in V \]

purple + brown is exact.
Intro to semi-definite programming

Linear programming where vars are entries in a PSD matrix

**Definition**
If $A$ is a symmetric $n \times n$ matrix, then $A$ is a positive semi-definite (PSD) matrix if $A \succeq 0$

Any of the following equivalent conditions hold:

1. $V \in \mathbb{R}^n$, $V^T V \succeq 0$
2. $A$ has nonnegative eigenvalues
3. $A = V^T V$ for some real matrix $V$, $m 
4. $A = \sum_{i=1}^n \lambda_i x_i x_i^T$ for some $\lambda_i \geq 0$ and orthonormal vectors $x_i \in \mathbb{R}^n$

**Semidefinite program (SDP)**

\[
\begin{align*}
\text{max } & \sum_{i,j} c_{i,j} x_{i,j} \\
\text{subject to } & \sum_{j} a_{i,jk} x_{j} = b_{k} \\
& x_{i,j} \geq 0, \quad V_{i,j} \\
& x = (x_{i,j}) \geq 0
\end{align*}
\]

**Vector program**

\[
\begin{align*}
\text{max } & \sum_{i,j} c_{i,j} (v_i \cdot v_j) \\
\text{subject to } & \sum_{j} a_{i,jk} (v_i \cdot v_j) = b_{k} \\
& v_i \in \mathbb{R}^n, \quad i=1,n
\end{align*}
\]

given $X \Rightarrow X = V^TV_j$, 
set $v_i$ to be in $\operatorname{conv} \{V \}$

**Key facts**

SDPs can be solved to within additive error $\varepsilon$ in time $\text{poly}(\text{size input}, \log(1/\varepsilon))$

in our discussions, we ignore additive error $\varepsilon$