Linear and conic programming relaxations: Graph structure and message-passing

Martin Wainwright

UC Berkeley
Departments of EECS and Statistics

Banff Workshop

Partially supported by grants from: National Science Foundation
Alfred P. Sloan Foundation

Outline

1. Conic programming relaxations based on moments
   - From integer program to linear program
   - Codeword and marginal polytopes
   - First-order relaxation and tightness
   - Sherali-Adams and Lasserre sequences
2. Analysis of LP relaxations in coding
   - geometry and pseudocodeword
   - worst-case guarantees for expanders
   - some probabilistic analysis
   - primal-dual witnesses in LP decoding

Parity check matrices and factor graphs

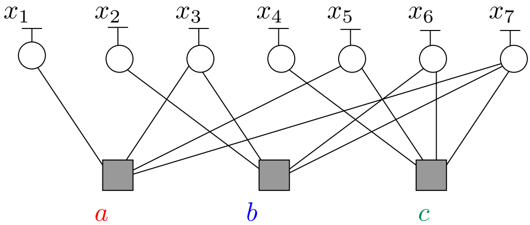
Binary linear code as null space:

$$\mathbb{C} = \big\{\mathbf{x} \in \{0,1\}^n \mid H\mathbf{x} = 0\big\},$$

for some parity check matrix $H \in \mathbb{R}^{m \times n}$.

Example: $m = 3$ constraints over $n = 7$ bits

$$H = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$



Optimal (maximum likelihood) decoding

Given: Likelihood vector $\theta = (\theta_1, \theta_2, \ldots, \theta_n)$ (typically from stochastic communication channel)

Goal: Determine most likely codeword:

$$\widehat{\mathbf{x}}_{\text{MAP}} \quad = \quad \arg\max_{\mathbf{x} \in C} \sum_{i=1}^{n} \theta_i x_i.$$

- known to be difficult in general (NP-complete)
- certain sub-classes of codes are polynomial-time decodable:
  - trellis codes
  - tree-structured codes
  - cut-set codes on planar graphs
  - more generally: codes with *sum-of-circuits* property (Seymour, 1981)
- meta-“theorem” in information theory: codes exactly decodable in polynomial-time are not “good”

From integer program to linear program

Any integer program (IP) can be converted to a linear program.

- re-write IP as maximization over convex hull:

$$\max_{\mathbf{x}\in\mathbb{C}} \sum_{i=1}^{n} \theta_i x_i = \max_{\substack{p(\mathbf{x})\geq 0 \\ \sum_{\mathbf{x}\in\mathbb{C}} p(\mathbf{x})=1}} \sum_{\mathbf{x}\in\mathbb{C}} p(\mathbf{x}) \left\{ \sum_{i=1}^{n} \theta_i x_i \right\}.$$

From integer program to linear program

Any integer program (IP) can be converted to a linear program.

- re-write IP as maximization over convex hull:

$$\max_{\mathbf{x}\in\mathbb{C}} \sum_{i=1}^{n} \theta_i x_i = \max_{\substack{p(\mathbf{x})\geq 0 \\ \sum_{\mathbf{x}\in\mathbb{C}} p(\mathbf{x})=1}} \sum_{\mathbf{x}\in\mathbb{C}} p(\mathbf{x}) \left\{ \sum_{i=1}^{n} \theta_i x_i \right\}.$$

- use linearity of expectation:

$$\begin{aligned} \max_{\substack{p(\mathbf{x})\geq 0 \\ \sum_{\mathbf{x}\in\mathbb{C}} p(\mathbf{x})=1}} \sum_{\mathbf{x}\in\mathbb{C}} p(\mathbf{x}) \sum_{i=1}^{n} x_i \theta_i &= \max_{\substack{p(\mathbf{x})\geq 0 \\ \sum_{\mathbf{x}\in\mathbb{C}} p(\mathbf{x})=1}} \sum_{i=1}^{n} \underbrace{\left\{ \sum_{\mathbf{x}\in\mathbb{C}} p(\mathbf{x}) x_i \right\}} \theta_i \\ &= \max_{\mu\in\mathcal{M}(\mathbb{C})} \sum_{i=1}^{n} \mu_i \theta_i \end{aligned}$$

From integer program to linear program

Any integer program (IP) can be converted to a linear program.

- re-write IP as maximization over convex hull:

$$\max_{\mathbf{x}\in\mathbb{C}} \sum_{i=1}^{n} \theta_i x_i = \max_{\substack{p(\mathbf{x})\geq 0 \\ \sum_{\mathbf{x}\in\mathbb{C}} p(\mathbf{x})=1}} \sum_{\mathbf{x}\in\mathbb{C}} p(\mathbf{x}) \left\{ \sum_{i=1}^{n} \theta_i x_i \right\}.$$

- use linearity of expectation:

$$\max_{\substack{p(\mathbf{x})\geq 0 \\ \sum_{\mathbf{x}\in\mathbb{C}} p(\mathbf{x})=1}} \sum_{\mathbf{x}\in\mathbb{C}} p(\mathbf{x}) \sum_{i=1}^{n} x_i \theta_i = \max_{\substack{p(\mathbf{x})\geq 0 \\ \sum_{\mathbf{x}\in\mathbb{C}} p(\mathbf{x})=1}} \sum_{i=1}^{n} \underbrace{\left\{ \sum_{\mathbf{x}\in\mathbb{C}} p(\mathbf{x}) x_i \right\}} \theta_i$$

$$= \max_{\mu\in\mathcal{M}(\mathbb{C})} \sum_{i=1}^{n} \mu_i \theta_i$$

Key question:

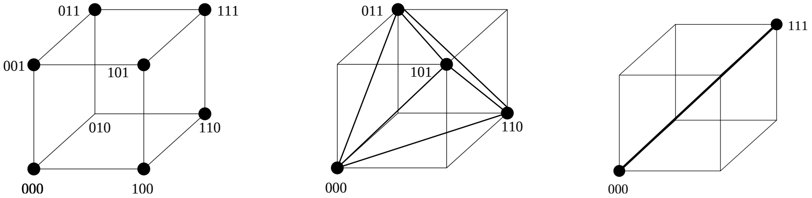
What is the set $\mathcal{M}(\mathbb{C})$ of $(\mu_1, \mu_2, \ldots, \mu_n)$ that are realizable in this way?

Codeword polytope ($\equiv$ cycle polytope)

Definition:

The *codeword polytope* $\mathcal{M}(\mathbb{C}) \subseteq [0,1]^n$ is the convex hull of all codewords

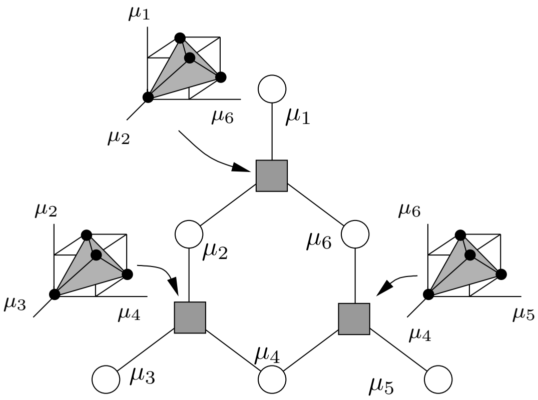
$$\mathcal{M}(\mathbb{C}) = \left\{ \begin{array}{c} \mu \in [0,1]^n \;\; | \;\; \text{there exists } p(\mathbf{x}) \geq 0 \text{ with } \sum_{\mathbf{x} \in \mathbb{C}} p(\mathbf{x}) = 1, \\ \text{such that} \quad \mu_s = \sum_{\mathbf{x} \in \mathbb{C}} p(\mathbf{x})\, x_s \quad \text{for all } s = 1, 2, \ldots, n \end{array} \right\}$$



(a) Uncoded (b) One check (c) Two checks

- $\mathcal{M}(\mathbb{C}) \subseteq [0,1]^n$, with vertices corresponding to codewords
- useful to think of $\{p(\mathbf{x}),\ \mathbf{x} \in \mathbb{C}\}$ as a probability distribution over codewords

First-order linear programming relaxation



- each parity check $a \in C$ defines a *local codeword polytope* $\mathcal{L}_1(a) \equiv \mathcal{M}(a)$
- first-order relaxation obtained by imposing all local constraints:

$$\mathcal{L}_1(\mathbb{C}) \;:=\; \cap_{a \in C} \mathcal{L}_1(a).$$

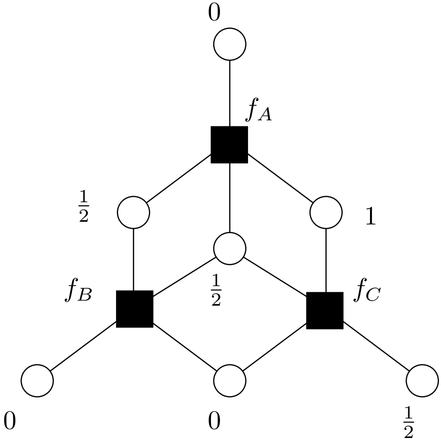
Illustration: A fractional vertex (pseudocodeword)

Check A:

$$\begin{bmatrix} 0 \\ \frac{1}{2} \\ \frac{1}{2} \\ 1 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} + \frac{1}{2} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

Check A:

$$\begin{bmatrix} \frac{1}{2} \\ \frac{1}{2} \\ 0 \\ 0 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} + \frac{1}{2} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$



Exactness for trees

Proposition:

On any tree, first-order LP relaxation is exact, and max-product algorithm solves the dual LP. (WaiJaaWil02, WaiJor03)

Proof sketch:

- given $(\mu_1, \ldots, \mu_n) \in \mathcal{L}_1(\mathbb{C})$, need to construct a global distribution $p(\cdot)$ such that

$$\sum_{\mathbf{x} \in \mathbb{C}} p(\mathbf{x}) x_i = \mu_i \qquad \text{for all } i = 1, \ldots, n.$$

Exactness for trees

Proposition:

On any tree, first-order LP relaxation is exact, and max-product algorithm solves the dual LP. (WaiJaaWil02, WaiJor03)

Proof sketch:

- given $(\mu_1, \ldots, \mu_n) \in \mathcal{L}_1(\mathbb{C})$, need to construct a global distribution $p(\cdot)$ such that

$$\sum_{\mathbf{x} \in \mathbb{C}} p(\mathbf{x}) x_i = \mu_i \qquad \text{for all } i = 1, \ldots, n.$$

- consider local code $\mathbb{C}(a)$ defined over each parity check: e.g., if $a = \{4, 7, 9\}$, and $x_a = (x_4, x_7, x_9)$:

$$\mathbb{C}(a) = \{(x_4, x_7, x_9) \mid x_4 \oplus x_7 \oplus x_9 = 0\}$$

Exactness for trees

Proposition:

On any tree, first-order LP relaxation is exact, and max-product algorithm solves the dual LP. (WaiJaaWil02, WaiJor03)

Proof sketch:

- given $(\mu_1, \ldots, \mu_n) \in \mathcal{L}_1(\mathbb{C})$, need to construct a global distribution $p(\cdot)$ such that

$$\sum_{\mathbf{x} \in \mathbb{C}} p(\mathbf{x}) x_i = \mu_i \qquad \text{for all } i = 1, \ldots, n.$$

- consider local code $\mathbb{C}(a)$ defined over each parity check: e.g., if $a = \{4, 7, 9\}$, and $x_a = (x_4, x_7, x_9)$:

$$\mathbb{C}(a) = \{(x_4, x_7, x_9) \mid x_4 \oplus x_7 \oplus x_9 = 0\}$$

- by definition of $\mathcal{L}_1(\mathbb{C})$, there exist marginal distributions $\{\mu_a(x_a) \mid x_a \in \mathbb{C}(a)\}$ for each parity check such that:

$$\sum_{x'_a \in \mathbb{C}(a),\ x'_i = x_i} \mu_a(x'_a) = \mu_i(x_i) \qquad \text{for all } i \in a.$$

From local to global consistency

Proof sketch (continued):

- we now have the following objects:

  Bit marginals $\quad \mu_i(x_i) = \begin{cases} 1 - \mu_i \\ \mu_i \end{cases}$

  Check-based marginals $\quad \mu_a(x_a)$ over local codes $\mathbb{C}(a)$.

From local to global consistency

Proof sketch (continued):

- we now have the following objects:

  Bit marginals $\qquad \mu_i(x_i) = \begin{cases} 1 - \mu_i \\ \mu_i \end{cases}$

  Check-based marginals $\qquad \mu_a(x_a)$ over local codes $\mathbb{C}(a)$.

- consider candidate distribution $p_\mu(\cdot)$ given by

$$p_\mu(x_1, x_2, \ldots, x_n) = \frac{1}{Z(\mu)} \prod_{i=1}^{n} \mu_i(x_i) \prod_{a \in C} \frac{\mu_a(x_a)}{\prod_{i \in a} \mu_i(x_i)}$$

From local to global consistency

Proof sketch (continued):

- we now have the following objects:

  Bit marginals $\qquad \mu_i(x_i) = \begin{cases} 1 - \mu_i \\ \mu_i \end{cases}$

  Check-based marginals $\qquad \mu_a(x_a)$ over local codes $\mathbb{C}(a)$.

- consider candidate distribution $p_\mu(\cdot)$ given by

$$p_\mu(x_1, x_2, \ldots, x_n) = \frac{1}{Z(\mu)} \prod_{i=1}^{n} \mu_i(x_i) \prod_{a \in C} \frac{\mu_a(x_a)}{\prod_{i \in a} \mu_i(x_i)}$$

- Key property of tree-structured graphs:
  - distribution is already normalized: $Z(\mu) = 1$
  - Bitwise consistency: $\sum_{\mathbf{x} \in \mathbb{C}} p(\mathbf{x}) x_i = \mu_i$ for all $i = 1, 2, \ldots, n$.

From local to global consistency

Proof sketch (continued):

- we now have the following objects:

  Bit marginals $\quad \mu_i(x_i) = \begin{cases} 1-\mu_i \\ \mu_i \end{cases}$

  Check-based marginals $\quad \mu_a(x_a)$ over local codes $\mathbb{C}(a)$.

- consider candidate distribution $p_\mu(\cdot)$ given by

$$p_\mu(x_1, x_2, \ldots, x_n) = \frac{1}{Z(\mu)} \prod_{i=1}^{n} \mu_i(x_i) \prod_{a \in C} \frac{\mu_a(x_a)}{\prod_{i \in a} \mu_i(x_i)}$$

- Key property of tree-structured graphs:
  - distribution is already normalized: $Z(\mu) = 1$
  - Bitwise consistency: $\sum_{\mathbf{x} \in \mathbb{C}} p(\mathbf{x}) x_i = \mu_i$ for all $i = 1, 2, \ldots, n$.

- proof via induction:
  - orient tree: specify some arbitrary vertex as the root
  - perform leaf-stripping operation

Hierarchies of relaxations

Moment-based perspective leads naturally to hierarchies via *lifting operations*.

Example:

- say given binary quadratic program over ordinary graph $G = (V, E)$:

$$\max_{\mathbf{x} \in \{0,1\}^n} \Big\{ \sum_{i=1}^{n} \theta_i x_i + \sum_{(i,j) \in E} \theta_{ij} x_i x_j \Big\}.$$

- relevant moments after converting to linear program:

  Vertex-based moment: $\mu_i = \mathbb{P}[x_i = 1]$ for all $i = 1, \ldots, n$

  Edge-based moment: $\mu_{ij} = \mathbb{P}[x_i = 1, x_j = 1]$ for all $(i,j) \in E$

Hierarchies of relaxations

Moment-based perspective leads naturally to hierarchies via *lifting operations*.

Example:

- say given binary quadratic program over ordinary graph $G = (V, E)$:

$$\max_{\mathbf{x} \in \{0,1\}^n} \Big\{ \sum_{i=1}^{n} \theta_i x_i + \sum_{(i,j) \in E} \theta_{ij} x_i x_j \Big\}.$$

- relevant moments after converting to linear program:
  Vertex-based moment: $\mu_i = \mathbb{P}[x_i = 1]$ for all $i = 1, \ldots, n$
  Edge-based moment: $\mu_{ij} = \mathbb{P}[x_i = 1, x_j = 1]$ for all $(i, j) \in E$

- moment polytope: cut or correlation polytope (Deza & Laurent, 1997)

- first-order LP relaxation involves four constraints per edge:

$$\begin{aligned}
\mathbb{P}[x_i = 1, x_j = 1] &= \mu_{ij} \geq 0 \\
\mathbb{P}[x_i = 1, x_j = 0] &= \mu_i - \mu_{ij} \geq 0 \\
\mathbb{P}[x_i = 0, x_j = 1] &= \mu_j - \mu_{ij} \geq 0 \\
\mathbb{P}[x_i = 0, x_j = 0] &= 1 + \mu_{ij} - \mu_i - \mu_j \geq 0.
\end{aligned}$$

Example: Sherali-Adams relaxations for $n = 3$

First-order: Imposes positive semidefinite constraints on three 4×4 sub-matrices.

$$\begin{bmatrix} 1 & \mu_1 & \mu_2 & \mu_3 & \mu_{12} & \mu_{23} & \mu_{13} & \mu_{123} \\ \mu_1 & \mu_1 & \mu_{12} & \mu_{13} & \mu_{12} & \mu_{123} & \mu_{13} & \mu_{123} \\ \mu_2 & \mu_{12} & \mu_2 & \mu_{23} & \mu_{12} & \mu_{23} & \mu_{123} & \mu_{123} \\ \mu_3 & \mu_{13} & \mu_{23} & \mu_3 & \mu_{123} & \mu_{23} & \mu_{13} & \mu_{123} \\ \mu_{12} & \mu_{12} & \mu_{12} & \mu_{123} & \mu_{12} & \mu_{123} & \mu_{123} & \mu_{123} \\ \mu_{23} & \mu_{123} & \mu_{23} & \mu_{23} & \mu_{123} & \mu_{23} & \mu_{123} & \mu_{123} \\ \mu_{13} & \mu_{13} & \mu_{123} & \mu_{13} & \mu_{123} & \mu_{123} & \mu_{13} & \mu_{123} \\ \mu_{123} & \mu_{123} & \mu_{123} & \mu_{123} & \mu_{123} & \mu_{123} & \mu_{123} & \mu_{123} \end{bmatrix}$$

(Sherali & Adams, 1990)

Example: Sherali-Adams relaxations for $n = 3$

First-order: Imposes positive semidefinite constraints on three 4×4 sub-matrices.
Another matrix controlled by the first-order relaxation.

$$\begin{bmatrix} 1 & \mu_1 & \mu_2 & \mu_3 & \mu_{12} & \mu_{23} & \mu_{13} & \mu_{123} \\ \mu_1 & \mu_1 & \mu_{12} & \mu_{13} & \mu_{12} & \mu_{123} & \mu_{13} & \mu_{123} \\ \mu_2 & \mu_{12} & \mu_2 & \mu_{23} & \mu_{12} & \mu_{23} & \mu_{123} & \mu_{123} \\ \mu_3 & \mu_{13} & \mu_{23} & \mu_3 & \mu_{123} & \mu_{23} & \mu_{13} & \mu_{123} \\ \mu_{12} & \mu_{12} & \mu_{12} & \mu_{123} & \mu_{12} & \mu_{123} & \mu_{123} & \mu_{123} \\ \mu_{23} & \mu_{123} & \mu_{23} & \mu_{23} & \mu_{123} & \mu_{23} & \mu_{123} & \mu_{123} \\ \mu_{13} & \mu_{13} & \mu_{123} & \mu_{13} & \mu_{123} & \mu_{123} & \mu_{13} & \mu_{123} \\ \mu_{123} & \mu_{123} & \mu_{123} & \mu_{123} & \mu_{123} & \mu_{123} & \mu_{123} & \mu_{123} \end{bmatrix}$$

(Sherali & Adams, 1990)

Example: Lasserre relaxations for $n = 3$

First-order: Imposes positive semidefinite constraint on 4×4 matrix.

$$
\begin{bmatrix}
1 & \mu_1 & \mu_2 & \mu_3 & \mu_{12} & \mu_{23} & \mu_{13} & \mu_{123} \\
\mu_1 & \mu_1 & \mu_{12} & \mu_{13} & \mu_{12} & \mu_{123} & \mu_{13} & \mu_{123} \\
\mu_2 & \mu_{12} & \mu_2 & \mu_{23} & \mu_{12} & \mu_{23} & \mu_{123} & \mu_{123} \\
\mu_3 & \mu_{13} & \mu_{23} & \mu_3 & \mu_{123} & \mu_{23} & \mu_{13} & \mu_{123} \\
\mu_{12} & \mu_{12} & \mu_{12} & \mu_{123} & \mu_{12} & \mu_{123} & \mu_{123} & \mu_{123} \\
\mu_{23} & \mu_{123} & \mu_{23} & \mu_{23} & \mu_{123} & \mu_{23} & \mu_{123} & \mu_{123} \\
\mu_{13} & \mu_{13} & \mu_{123} & \mu_{13} & \mu_{123} & \mu_{123} & \mu_{13} & \mu_{123} \\
\mu_{123} & \mu_{123} & \mu_{123} & \mu_{123} & \mu_{123} & \mu_{123} & \mu_{123} & \mu_{123}
\end{bmatrix}
$$

(Lasserre, 2001)

Example: Lasserre relaxations for $n = 3$

Second-order: Imposes positive semidefinite constraint on 7×7 matrix.

$$
\begin{bmatrix}
1 & \mu_1 & \mu_2 & \mu_3 & \mu_{12} & \mu_{23} & \mu_{13} & \mu_{123} \\
\mu_1 & \mu_1 & \mu_{12} & \mu_{13} & \mu_{12} & \mu_{123} & \mu_{13} & \mu_{123} \\
\mu_2 & \mu_{12} & \mu_2 & \mu_{23} & \mu_{12} & \mu_{23} & \mu_{123} & \mu_{123} \\
\mu_3 & \mu_{13} & \mu_{23} & \mu_3 & \mu_{123} & \mu_{23} & \mu_{13} & \mu_{123} \\
\mu_{12} & \mu_{12} & \mu_{12} & \mu_{123} & \mu_{12} & \mu_{123} & \mu_{123} & \mu_{123} \\
\mu_{23} & \mu_{123} & \mu_{23} & \mu_{23} & \mu_{123} & \mu_{23} & \mu_{123} & \mu_{123} \\
\mu_{13} & \mu_{13} & \mu_{123} & \mu_{13} & \mu_{123} & \mu_{123} & \mu_{13} & \mu_{123} \\
\mu_{123} & \mu_{123} & \mu_{123} & \mu_{123} & \mu_{123} & \mu_{123} & \mu_{123} & \mu_{123}
\end{bmatrix}
$$

(Lasserre, 2001)

Tightness and hypergraph structure

Question: When are these relaxations tight?

- always tight after n stages of lifting (constraining all 2^n moments)
- exist (binary) problems that require n steps
- in a worst-case sense: tightness determined by treewidth

Tightness and hypergraph structure

Question: When are these relaxations tight?

- always tight after n stages of lifting (constraining all 2^n moments)
- exist (binary) problems that require n steps
- in a worst-case sense: tightness determined by treewidth

Consider family of $\{0,1\}$-polynomial programs:

$$\max \sum_{i=1}^{n} \theta_i x_i \qquad \text{subject to polynomial constraints}$$

$$g_\ell(x_1, \ldots, x_n) \leq 0, \quad \ell = 1, \ldots, M$$

Tightness and hypergraph structure

Question: When are these relaxations tight?

- always tight after n stages of lifting (constraining all 2^n moments)
- exist (binary) problems that require n steps
- in a worst-case sense: tightness determined by treewidth

Consider family of $\{0,1\}$-polynomial programs:

$$\max \sum_{i=1}^{n} \theta_i x_i \qquad \text{subject to polynomial constraints}$$

$$g_\ell(x_1, \ldots, x_n) \; \leq 0, \quad \ell = 1, \ldots, M$$

Theorem

Form the hypergraph G with vertex $V = \{1, 2, \ldots, n\}$ and hyperedge set $E = \{V(g_\ell), \; \ell = 1, \ldots, M\}$, and let t be its treewidth.

(a) *The Sherali-Adams relaxation is tight at order t.*

(b) *The Lasserre relaxation is tight at order $t+1$.*

(WaiJor03)

Linear programming (LP) decoding

Based on first-order LP relaxation of ML integer program:

$$\max_{\mathbf{x} \in \mathbb{C}} \sum_{i=1}^{n} \theta_i x_i \leq \max_{\mu \in \mathcal{L}_1(\mathbb{C})} \sum_{i=1}^{n} \theta_i \mu_i$$

where the vectors $\mu = (\mu_1, \ldots, \mu_n)$ belong to the relaxed constraint set:

$$\mathcal{L}_1(\mathbb{C}) = \left\{ \begin{array}{ll} \mu \in [0,1]^n & \mid \sum_{i \in N(a)} |\mu_i - z_i| \geq 1 \quad \forall \text{ odd parity } z_a \in \{0,1\}^{|N(a)|} \\ & \qquad \text{and for all checks } a \in C \end{array} \right\}$$

Relaxed set $\mathcal{L}_1(\mathbb{C})$ defined by $T = \sum_{a \in C} 2^{d_a - 1}$ constraints in total, where $d_a = |N(a)|$.

Example: For check $a = \{1, 2, 3\}$, require $2^{3-1} = 4$ constraints:

$$\begin{aligned} (1 - \mu_1) + \mu_2 + \mu_3 &\geq 0 \\ \mu_1 + (1 - \mu_2) + \mu_3 &\geq 0 \\ \mu_1 + \mu_2 + (1 - \mu_3) &\geq 0 \\ (1 - \mu_1) + (1 - \mu_2) + (1 - \mu_3) &\geq 0 \end{aligned}$$

Different types of channels

- communication channel modeled as a conditional distribution

$$\mathbb{P}[\mathbf{y} \mid \mathbf{x}] \; = \quad \text{prob. of observing } \mathbf{y} \text{ given that } \mathbf{x} \text{ transmitted}$$

- channels are often modeled as memoryless: $\mathbb{P}[\mathbf{y} \mid \mathbf{x}] = \prod_{i=1}^{n} \mathbb{P}[y_i \mid x_i]$

Different types of channels

- communication channel modeled as a conditional distribution

$$\mathbb{P}[\mathbf{y} \mid \mathbf{x}] \;=\; \quad \text{prob. of observing } \mathbf{y} \text{ given that } \mathbf{x} \text{ transmitted}$$

- channels are often modeled as memoryless: $\mathbb{P}[\mathbf{y} \mid \mathbf{x}] = \prod_{i=1}^{n} \mathbb{P}[y_i \mid x_i]$
- some examples:
  - binary erasure channel (BEC) with erasure prob. $\alpha \in [0, 1]$:

$$y_i = \begin{cases} x_i & \text{with prob. } 1 - \alpha \\ * & \text{with prob. } \alpha. \end{cases}$$

Different types of channels

- communication channel modeled as a conditional distribution

$$\mathbb{P}[\mathbf{y} \mid \mathbf{x}] \;=\; \text{prob. of observing } \mathbf{y} \text{ given that } \mathbf{x} \text{ transmitted}$$

- channels are often modeled as memoryless: $\mathbb{P}[\mathbf{y} \mid \mathbf{x}] = \prod_{i=1}^{n} \mathbb{P}[y_i \mid x_i]$
- some examples:
  - binary erasure channel (BEC) with erasure prob. $\alpha \in [0, 1]$:

$$y_i = \begin{cases} x_i & \text{with prob. } 1 - \alpha \\ * & \text{with prob. } \alpha. \end{cases}$$

  - binary symmetric channel (BSC) with flip prob. $p \in [0, 1]$:

$$y_i = \begin{cases} x_i & \text{with prob. } 1 - \\ 1 - x_i & \text{with prob. } p. \end{cases}$$

Different types of channels

- communication channel modeled as a conditional distribution

$$\mathbb{P}[\mathbf{y} \mid \mathbf{x}] \;=\; \text{prob. of observing } \mathbf{y} \text{ given that } \mathbf{x} \text{ transmitted}$$

- channels are often modeled as memoryless: $\mathbb{P}[\mathbf{y} \mid \mathbf{x}] = \prod_{i=1}^{n} \mathbb{P}[y_i \mid x_i]$
- some examples:
  - binary erasure channel (BEC) with erasure prob. $\alpha \in [0, 1]$:

$$y_i = \begin{cases} x_i & \text{with prob. } 1-\alpha \\ * & \text{with prob. } \alpha. \end{cases}$$

  - binary symmetric channel (BSC) with flip prob. $p \in [0, 1]$:

$$y_i = \begin{cases} x_i & \text{with prob. } 1- \\ 1-x_i & \text{with prob. } p. \end{cases}$$

  - additive white Gaussian noise channel (AWGN):

$$y_i = (2x_i - 1) + \sigma w_i \quad \text{where } w_i \sim N(0, 1)$$

Different types of channels

- communication channel modeled as a conditional distribution

$$\mathbb{P}[\mathbf{y} \mid \mathbf{x}] \;=\; \text{prob. of observing } \mathbf{y} \text{ given that } \mathbf{x} \text{ transmitted}$$

- channels are often modeled as memoryless: $\mathbb{P}[\mathbf{y} \mid \mathbf{x}] = \prod_{i=1}^{n} \mathbb{P}[y_i \mid x_i]$
- some examples:
  - binary erasure channel (BEC) with erasure prob. $\alpha \in [0,1]$:

$$y_i = \begin{cases} x_i & \text{with prob. } 1-\alpha \\ * & \text{with prob. } \alpha. \end{cases}$$

  - binary symmetric channel (BSC) with flip prob. $p \in [0,1]$:

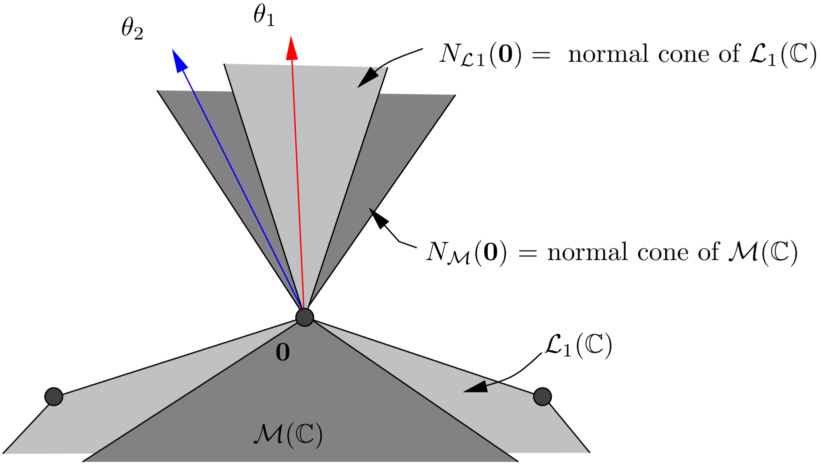
$$y_i = \begin{cases} x_i & \text{with prob. } 1- \\ 1-x_i & \text{with prob. } p. \end{cases}$$

  - additive white Gaussian noise channel (AWGN):

$$y_i = (2x_i - 1) + \sigma w_i \quad \text{where } w_i \sim N(0,1)$$

- input to LP decoding algorithm: likelihoods $\theta_i = \log \frac{\mathbb{P}[y_i | x_i = 1]}{\mathbb{P}[y_i | x_i = 0]}$

Geometry of LP decoding



$$\begin{aligned} \text{Prob. of successful ML decoding} &= \mathbb{P}\big[\theta \in N_{\mathcal{M}}(\mathbf{0})\big] \\ \text{Prob. of successful LP decoding} &= \mathbb{P}\big[\theta \in N_{\mathcal{L}_1}(\mathbf{0})\big] \end{aligned}$$

Some known results

- LP decoding equivalent to message-passing for binary erasure channel (stopping sets $\Longleftrightarrow$ pseudocodewords)

Some known results

- LP decoding equivalent to message-passing for binary erasure channel (stopping sets $\Longleftrightarrow$ pseudocodewords)

- positive results:
  - linear LP pseudoweight for expander codes and BSC (Feldman et al., 2004)
  - linear pseudoweight scaling for truncated Gaussian (Feldman et al., 2005)

Some known results

- LP decoding equivalent to message-passing for binary erasure channel (stopping sets $\Longleftrightarrow$ pseudocodewords)

- positive results:
  - linear LP pseudoweight for expander codes and BSC (Feldman et al., 2004)
  - linear pseudoweight scaling for truncated Gaussian (Feldman et al., 2005)

- negative results:
  - sublinear LP pseudoweight for AWGN (Koetter & Vontobel, 2003, 2005)
  - bounds on BSC pseudodistance (Vontobel & Koetter, 2006)

Some known results

- LP decoding equivalent to message-passing for binary erasure channel (stopping sets $\Longleftrightarrow$ pseudocodewords)

- positive results:
  - linear LP pseudoweight for expander codes and BSC (Feldman et al., 2004)
  - linear pseudoweight scaling for truncated Gaussian (Feldman et al., 2005)

- negative results:
  - sublinear LP pseudoweight for AWGN (Koetter & Vontobel, 2003, 2005)
  - bounds on BSC pseudodistance (Vontobel & Koetter, 2006)

- various extensions to basic LP algorithm:
  - stopping set redundancy for BEC (Vardy et al., 2006)
  - facet guessing (Dimakis et al., 2006, 2009)
  - loop corrections for LP decoding (Chertkov et al., 2006)
  - higher-order relaxations (Feldman et al., 2005, others...)

- various iterative "message-passing" algorithms for solving LP:
  - tree-reweighted (TRW) max-product (WaiJaaWil03, Kolmogorov, 2005)
  - zero-temperature limits of convex BP (Weiss et al., 2006, Johnson et al., 2008)
  - adaptive LP-solver (Taghavi & Siegel, 2006)
  - interior-point methods (Vontobel, 2008)
  - proximal methods (Agarwal et al., 2009)

Performance for the BEC

- standard iterative decoding (sum-product; belief propagation) takes a very simple form in the BEC: (e.g., Luby et al., 2001)

```
While there exists at least one erased (∗) bit:
1. Find check node with exactly one erased bit nbr.
2. Set erased bit neighbor to the XOR of other bit neighbors.
3. Repeat.
```

- success/failure is determined by presence/absence of stopping sets in the erased bits (Di et al., 2002)

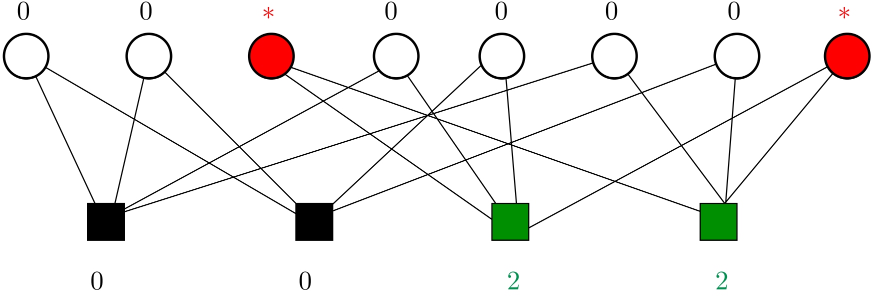
- for LP decoding, cost vector takes form $\theta_s = \begin{cases} -1 & \text{if } y_s = 1 \\ 1 & \text{if } y_s = 0 \\ 0 & \text{if } y_s \text{ erased} \end{cases}$.

- stopping sets correspond to cost vectors that lie outside the relaxed normal cone $N_{\mathcal{L}_1}(\mathbf{0})$

Stopping sets for the BEC

Definition: A *stopping set* S is a set of bits such that:

- every bit in S is erased
- every check that is adjacent to S has degree at least two (with respect to S)



LP decoding in the BEC

The performance of the LP decoder in the BEC is completely characterized by stopping sets:

Theorem

(a) *LP decoding succeeds in the BEC if and only the set of erasures does not contain a stopping set.*

(b) *Therefore, the performance of (first-order) LP decoding is equivalent to sum-product/belief propagation decoding in the BEC.*

(Feldman et al., 2003)

LP decoding in the BEC

The performance of the LP decoder in the BEC is completely characterized by stopping sets:

Theorem

(a) *LP decoding succeeds in the BEC if and only the set of erasures does not contain a stopping set.*

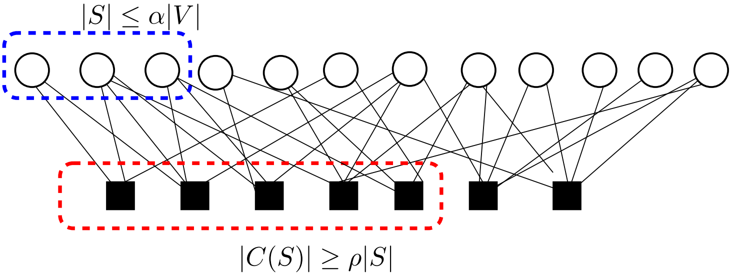
(b) *Therefore, the performance of (first-order) LP decoding is equivalent to sum-product/belief propagation decoding in the BEC.*

(Feldman et al., 2003)

- Shannon capacity: a code of rate $R = 1 - m/n$ should be able to correct a fraction m/n of erasures
- **Corollary:** With appropriate choices of low-density parity check codes, LP decoding can achieve capacity in the BEC.

Codes based on expander graphs

- previous work on expander codes (e.g., SipSpi02; BurMil02; BarZem02)
- graph expansion: yields stronger results beyond girth-based analysis



Definition: Let $\alpha \in (0,1)$. A factor graph $G = (V, C, E)$ is a (α, ρ)-*expander* if for all subsets $S \subset V$ with $|S| \leq \alpha|V|$, at least $\rho|S|$ check nodes are incident to S.

Worst-case constant fraction for expanders

Theorem (Linear fraction guarantee)

Let $\mathbb{C}$ be an LDPC described by a factor graph G with regular variable (bit) degree d_v. Suppose that G is an $(\alpha, \delta d_v)$-expander, where $\delta > 2/3 + 1/(3d_v)$ and δd_v is an integer.
Then the LP decoder can correct any pattern of $\frac{3\delta-2}{2\delta-1}(\alpha n)$ bit flips.

Worst-case constant fraction for expanders

Theorem (Linear fraction guarantee)

Let $\mathbb{C}$ be an LDPC described by a factor graph G with regular variable (bit) degree d_v. Suppose that G is an $(\alpha, \delta d_v)$-expander, where $\delta > 2/3 + 1/(3d_v)$ and δd_v is an integer.
Then the LP decoder can correct any pattern of $\frac{3\delta-2}{2\delta-1}(\alpha n)$ bit flips.

- key technical device: use of dual witness
  - by code/polytope symmetry: assume WLOG that 0^n sent
  - LP succeeds when 0^n sent $\iff$ primal optimum $p^* = 0$
  - suffices to construct dual optimal solution with $q^* = 0$

Worst-case constant fraction for expanders

Theorem (Linear fraction guarantee)

Let $\mathbb{C}$ be an LDPC described by a factor graph G with regular variable (bit) degree d_v. Suppose that G is an $(\alpha, \delta d_v)$-expander, where $\delta > 2/3 + 1/(3d_v)$ and δd_v is an integer.
Then the LP decoder can correct any pattern of $\frac{3\delta-2}{2\delta-1}(\alpha n)$ bit flips.

- key technical device: use of dual witness
  - by code/polytope symmetry: assume WLOG that 0^n sent
  - LP succeeds when 0^n sent $\iff$ primal optimum $p^* = 0$
  - suffices to construct dual optimal solution with $q^* = 0$
- caveat: constant fraction very low (e.g., $c = 0.00017$ for $R = 0.5$)
- potential gaps in the analysis
  - analysis adversarial in nature
  - dual witness relatively weak

Proof technique: Construction of dual witness

Primal LP: Vars. $\{\mu_i,\ i \in V\}$, $\quad \{\mu_{a,J},\ a \in F, J \subseteq N(a), \quad |J| \text{ even}\}$

$$\min. \quad \sum_{i \in V} \theta_i \mu_i \quad \text{s.t.} \begin{cases} \mu_{a,J} \geq 0 \\ \sum_{J \in \mathbb{C}(a)} \mu_{a,J} = 1 \\ \sum_{J \in \mathbb{C}(a), J_v = 1} \mu_{a,J} \quad = \mu_v \end{cases}$$

Proof technique: Construction of dual witness

Primal LP: Vars. $\{\mu_i,\ i \in V\}$, $\{\mu_{a,J},\ a \in F, J \subseteq N(a), \quad |J| \text{ even}\}$

$$\text{min.} \quad \sum_{i \in V} \theta_i \mu_i \quad \text{s.t.} \begin{cases} \mu_{a,J} \geq 0 \\ \sum_{J \in \mathbb{C}(a)} \mu_{a,J} = 1 \\ \sum_{J \in \mathbb{C}(a), J_v = 1} \mu_{a,J} = \mu_v \end{cases}$$

Dual LP: Vars. $\{v_a,\ a \in F\}$ $\{\tau_{ia},\ (i,a) \in E\}$ unconstrained

$$\text{max.} \quad \sum_{a \in F} v_a \quad \text{s.t.} \begin{cases} \sum_{i \in S} \tau_{ia} \geq v_a \text{ for all} \quad a \in C, J \subseteq C(a), |J| \text{ even} \\ \sum_{a \in N(i)} \tau_{ia} \leq \theta_i \qquad \text{for all } i \in V \end{cases}$$

Dual witness to zero-valued primal solution

- assume WLOG that 0^n is sent: suffices to construct a dual solution with value $q^* = 0$
- dual LP simplifies substantially as follows:

Dual feasibility: Find real numbers $\{\tau_{ia},\ (i,a) \in E\}$ such that

$$\begin{aligned} \tau_{ia} + \tau_{ja} &\geq 0 \qquad \forall\, a \in C, \text{ and } i,j \in N(a) \\ \sum_{a \in N(i)} \tau_{ia} &< \theta_i \qquad \text{for all } i \in V \end{aligned}$$

- random weights $\theta_i \in \mathbb{R}$ defined by channel; e.g., for binary symmetric channel

$$\theta_i = \begin{cases} 1 & \text{with prob. } 1-p \\ -1 & \text{with prob. } p \end{cases}$$

Probabilistic analysis of LP decoding over BSC

Consider an ensemble of LDPC codes with rate R, regular vertex degree d_v, and blocklength n. Suppose that the code is a $(\nu, \left(\frac{p}{d_v}\right) d_v)$ expander.

Theorem

For each (R, d_v, n), there is a fraction $\alpha > 0$ and error exponent $c > 0$ such that the LP decoder succeeds with probability $1 - \exp(-cn)$ over the space of bit flips $\leq \lfloor \alpha n \rfloor$. *(DasDimKarWai07)*

Remarks:

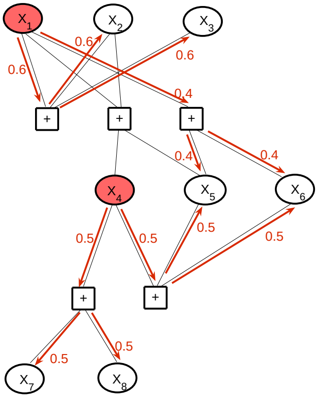
- the correctable fraction α is always larger than the worst case guarantee $\frac{3\frac{p}{d_v}-2}{2\frac{p}{d_v}-1}\nu$.
- concrete example: rate $R = 0.5$, degree $d_v = 8$ and $p = 6$ yields a correctable fraction $\alpha = 0.002$.

Hyperflow-based dual witness

A *hyperflow* is a collection of weights $\{\tau_{ia}, (i,a) \in E\}$ such that:
(a) for each check $a \in F$, exists some $\gamma_a \geq 0$ and privileged neighbor $i^* \in N(a)$ such that

$$\tau_{ia} = \begin{cases} -\gamma_a & \text{for } i = i^* \\ +\gamma_a & \text{for } i \neq i^*. \end{cases}.$$

(b) $\sum_{a \in N(i)} \tau_{ia} < \theta_i$ for all $i \in V$.



Proposition:

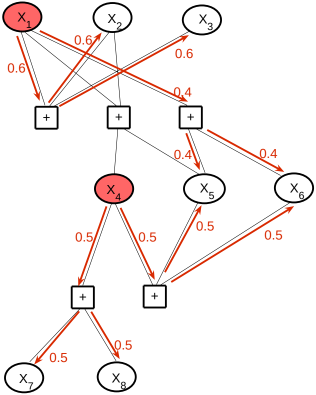
A hyperflow exists $\Longleftrightarrow$
$\exists$ a dual feasible point with zero value.

Hyperflow-based dual witness

A *hyperflow* is a collection of weights $\{\tau_{ia}, (i,a) \in E\}$ such that:
(a) for each check $a \in F$, exists some $\gamma_a \geq 0$ and privileged neighbor $i^* \in N(a)$ such that

$$\tau_{ia} = \begin{cases} -\gamma_a & \text{for } i = i^* \\ +\gamma_a & \text{for } i \neq i^*. \end{cases}.$$

(b) $\sum_{a \in N(i)} \tau_{ia} < \theta_i$ for all $i \in V$.



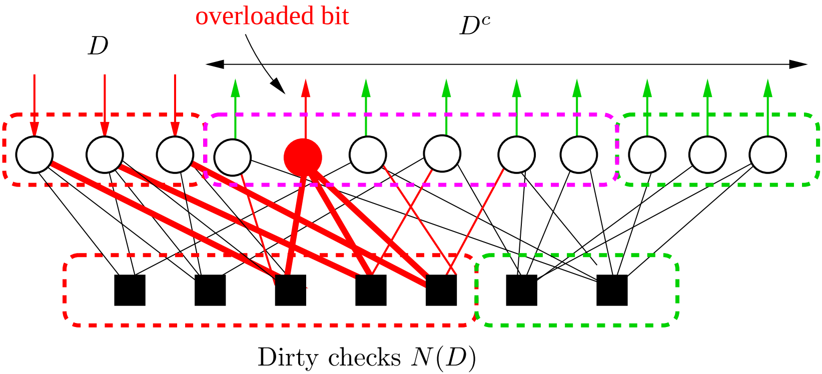
Proposition:

A hyperflow exists $\iff$
$\exists$ a dual feasible point with zero value.

Hyperflow (epidemic) interpretation:

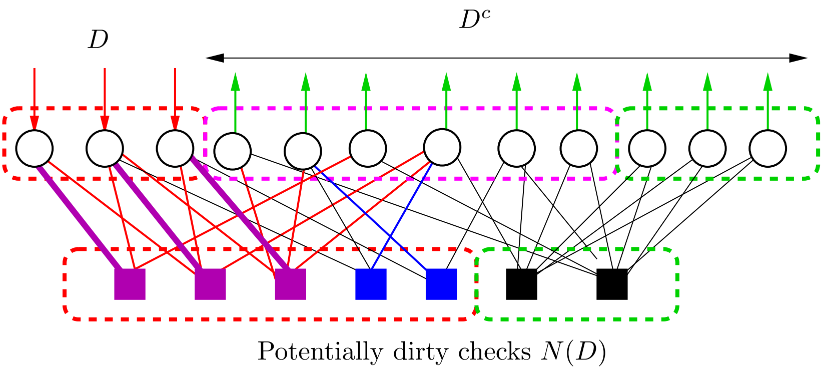
- each flipped bit adds 1 unit of "poison"; each clean bit absorbs at most 1 unit
- each infected check relays poison to all of its neighbors

Naive routing of poison may fail



- need to route 1 unit of poison away from each flipped bit
- each unflipped bit $j \in D^c$ can neutralize at most one unit
- **Consequence:** naive routing of poison can lead to overload

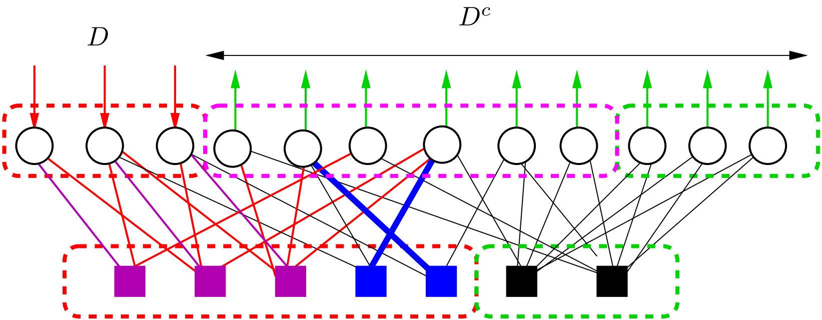
Routing poison via generalized matching



Definition: For positive integers p, q, a (p, q)-matching is defined by the conditions:

(i) **every flipped bit $i \in D$ is matched with p distinct checks.**

Routing poison via generalized matching



Potentially dirty checks $N(D)$

Definition: For positive integers p, q, a (p, q)-matching is defined by the conditions:

(i) every flipped bit $i \in D$ is matched with p distinct checks.

(ii) **every unflipped bit $j \in D^c$ matched with $\max\{Z_j - (d_v - q),\ 0\}$ checks from $N(D)$, where $Z_j = |N(j) \cap N(D)|$.**

Generalized matching implies hyperflow

Lemma

Any (p, q) matching with $2p + q > 2d_v$ can be used to construct a valid hyperflow.

Generalized matching implies hyperflow

Lemma

Any (p, q) matching with $2p + q > 2d_v$ can be used to construct a valid hyperflow.

Proof sketch:

- construct hyperflow with each flipped bit routing $\gamma \geq 0$ units to each of p checks

Generalized matching implies hyperflow

Lemma

Any (p, q) matching with $2p + q > 2d_v$ can be used to construct a valid hyperflow.

Proof sketch:

- construct hyperflow with each flipped bit routing $\gamma \geq 0$ units to each of p checks
- each flipped bit can receive at most $(d_v - p)\gamma$ units from other dirty checks (to which it is not matched)

Generalized matching implies hyperflow

Lemma

Any (p, q) matching with $2p + q > 2d_v$ can be used to construct a valid hyperflow.

Proof sketch:

- construct hyperflow with each flipped bit routing $\gamma \geq 0$ units to each of p checks
- each flipped bit can receive at most $(d_v - p)\gamma$ units from other dirty checks (to which it is not matched)
- hence we require that $-p\gamma + (d_v - p)\gamma < -1$, or $\gamma > 1/(2p - d_v)$

Generalized matching implies hyperflow

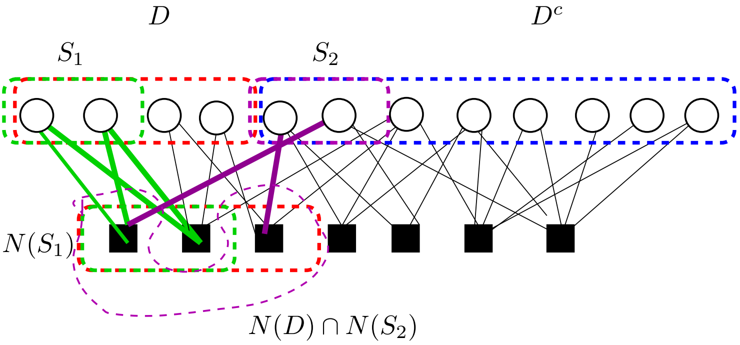
Lemma

Any (p, q) matching with $2p + q > 2d_v$ can be used to construct a valid hyperflow.

Proof sketch:

- construct hyperflow with each flipped bit routing $\gamma \geq 0$ units to each of p checks
- each flipped bit can receive at most $(d_v - p)\gamma$ units from other dirty checks (to which it is not matched)
- hence we require that $-p\gamma + (d_v - p)\gamma < -1$, or $\gamma > 1/(2p - d_v)$
- each unflipped bit receives at most $(d_v - q)\gamma$ units so that we need $\gamma < 1/(d_v - q)$

Generalized matching and Hall's theorem



- by generalized Hall's theorem, (p,q)-matching fails to exist if only if there exist subsets $S_1 \subseteq D$ and $S_2 \subseteq D^c$ that *contract*:

$$\underbrace{|N(S_1) \cup [N(S_2) \cap N(D)]|}_{\text{available matches}} \leq \underbrace{p|S_1| + \sum_{j \in S_2} \max\{0, q - (d_v - Z_j)\}}_{\text{total requests}}.$$

High-level summary of key steps

1. Randomly constructed LDPC is "almost-always" expander with high probability (w.h.p.)
   - weaker notion than classical expansion: holds for larger sizes
   - proof: union bounds plus martingale concentration

High-level summary of key steps

1. Randomly constructed LDPC is "almost-always" expander with high probability (w.h.p.)
   - weaker notion than classical expansion: holds for larger sizes
   - proof: union bounds plus martingale concentration

2. Prove that an "almost-always" expander will have a generalized matching w.h.p.:
   - requires concentration statements
   - generalized Hall's theorem

High-level summary of key steps

1. Randomly constructed LDPC is "almost-always" expander with high probability (w.h.p.)
   - weaker notion than classical expansion: holds for larger sizes
   - proof: union bounds plus martingale concentration

2. Prove that an "almost-always" expander will have a generalized matching w.h.p.:
   - requires concentration statements
   - generalized Hall's theorem

3. Generalized matching guarantees existence of hyperflow.

High-level summary of key steps

1. Randomly constructed LDPC is "almost-always" expander with high probability (w.h.p.)
   - weaker notion than classical expansion: holds for larger sizes
   - proof: union bounds plus martingale concentration

2. Prove that an "almost-always" expander will have a generalized matching w.h.p.:
   - requires concentration statements
   - generalized Hall's theorem

3. Generalized matching guarantees existence of hyperflow.

4. Valid hyperflow is a dual witness for LP decoding succcess.

Summary and some papers

- broad families of conic programming (LP, SOCP, SDP) based on moments
- worst-case tightness intimately related to (hyper)graph structure
- known average-case results also exploit graph structure:
  - girth and “locally treelike” properties
  - graph expansion
- many open questions remain....

Summary and some papers

- broad families of conic programming (LP, SOCP, SDP) based on moments
- worst-case tightness intimately related to (hyper)graph structure
- known average-case results also exploit graph structure:
  - girth and "locally treelike" properties
  - graph expansion
- many open questions remain....

Some papers:

1. Wainwright, M. J. and Jordan, M. I. (2008) Graphical models, exponential families, and variational methods. *Foundations and Trends in Machine Learning*, Volume 1, Issues 1–2, pages 1–305. December 2008.
2. Daskalakis, C., Dimakis, A. D., Karp, R. and Wainwright, M. J. (2008). Probabilistic analysis of linear programming decoding. *IEEE Transactions on Information Theory*, Vol. 54(8), pp. 3565 - 3578, August 2008
3. Feldman, J., Malkin, T., Servedio, R.A., Stein, C. and Wainwright, M. J., (2007). LP Decoding Corrects a Constant Fraction of Errors. *IEEE Transactions on Information Theory*, 53(1):82–89, January 2007.