Linear Programming (LP) Decoding Corrects a Constant Fraction of Errors

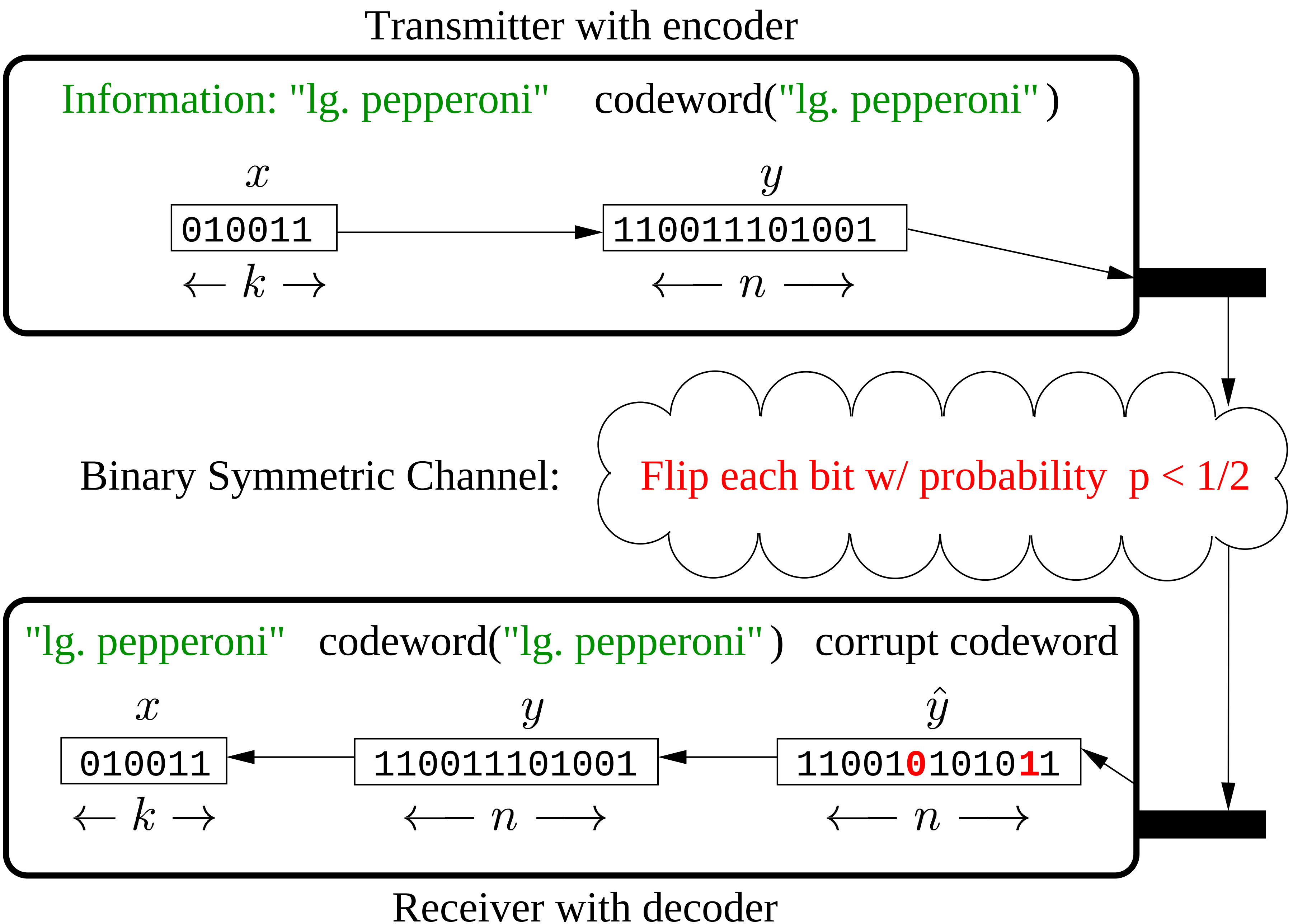
Jon Feldman

Columbia University

Joint work with Tal Malkin, Cliff Stein, Rocco Servedio (Columbia);

Martin Wainwright (UC Berkeley)

Binary error-correcting codes



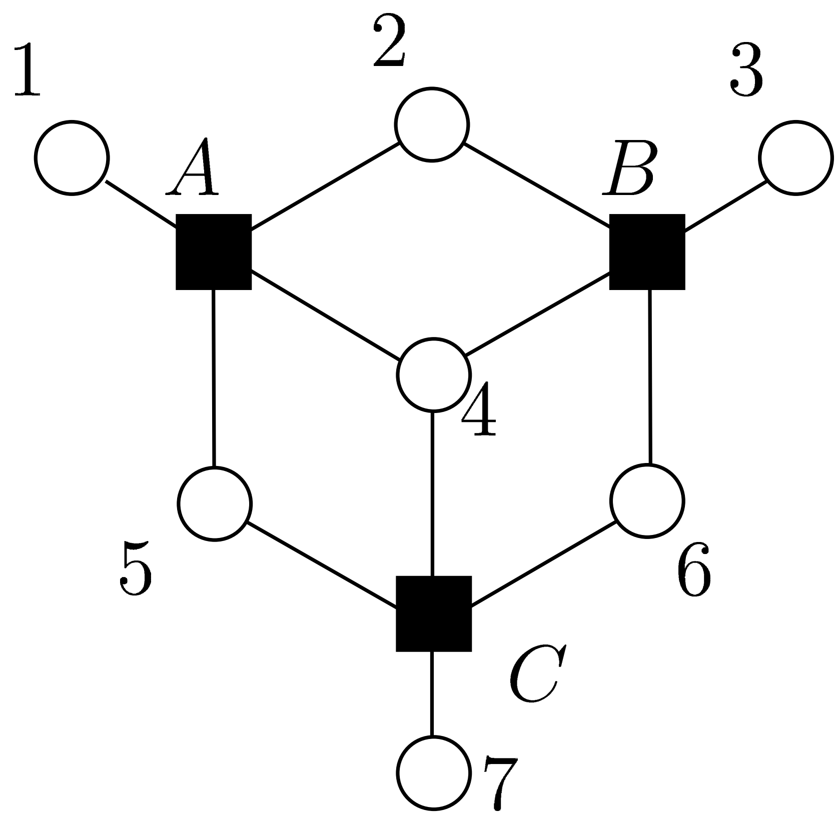
Basic Coding Terminology

- A code is a subset $C \subseteq \{0,1\}^n$, where $|C| = 2^k$. If $y \in C$, then y is a codeword.
- Dimension = k = info bits in each codeword.
- Length = n = size of a codeword.
- Rate = k/n = info per transmitted code bit.
- (Minimum) distance Δ = $\min_{y,y' \in C} \Delta(y, y')$.
  Relative (minimum) distance $\delta = \Delta/n$.
- Word error rate (WER) = probability of decoding failure = $\Pr_{noise}[\text{ transmitted } y \neq \text{ decoded } y]$.
  Practical measure of performance.
- Goals: high rate, large distance, low WER, low (construction, encoding, decoding) complexity.

Correcting a constant fraction of error

- A code *family* is an infinite set of codes $C_1, C_2, \ldots$ of increasing length $n_1 < n_2 < \ldots$.
- One major goal of coding theory: construct a family of codes and a decoder, where:
  - The codes have constant rate r.
  - The decoder runs in time poly(n).
  - The decoder succeeds if $\leq \alpha n$ bits flipped, where α constant. (Note: $\implies$ WER $\leq 2^{-\Omega(n)}$.)
- Achieved by GMD [F], iterative bit-flipping [G, SS, BZ], list decoding [GI].
- This talk: LP decoding [FK '02] corrects a constant fraction of errors, using expanding LDPC codes.

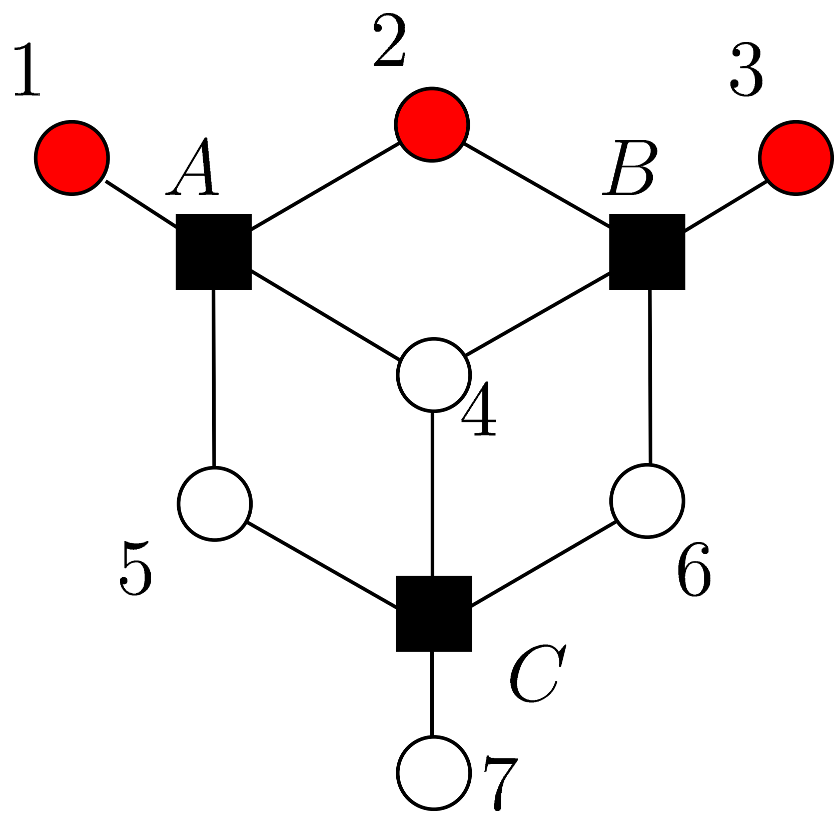
Low-density parity-check codes, factor graph



- Codebit nodes $1 \dots n$.
- Check nodes $1 \dots m$.
- Codewords: $y \in \{0,1\}^n$ where all check neighborhoods have even parity w.r.t. y.
- Rate $\geq 1 - m/n$.

- Low density: constant degree.
- Codeword examples:
  - 0000000

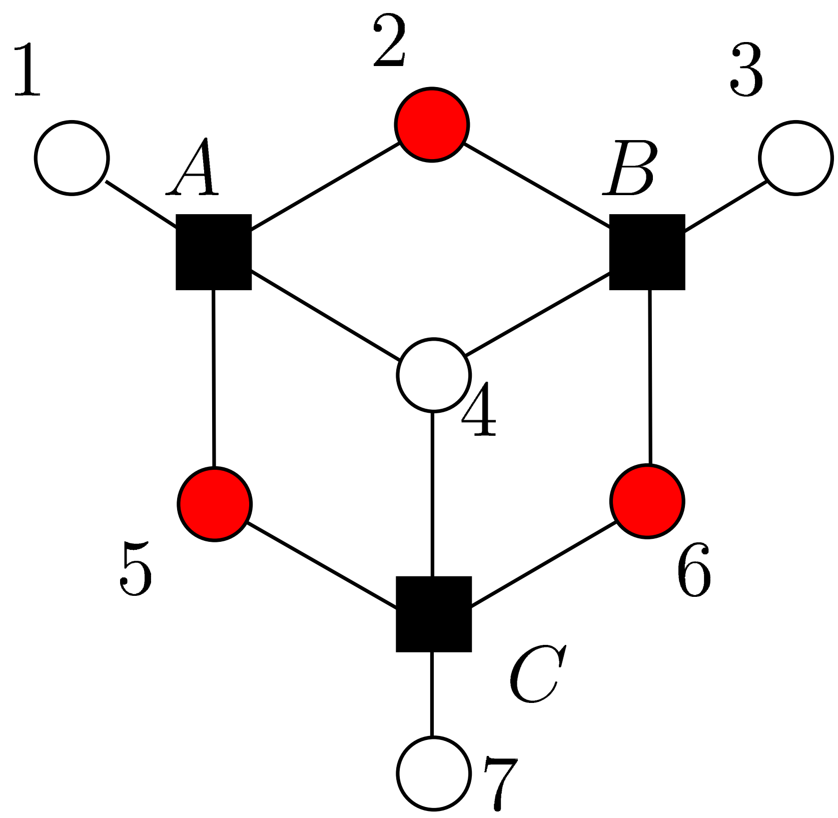
Low-density parity-check codes, factor graph



- Codebit nodes $1 \ldots n$.
- Check nodes $1 \ldots m$.
- Codewords: $y \in \{0,1\}^n$ where all check neighborhoods have even parity w.r.t. y.
- Rate $\geq 1 - m/n$.

- Low density: constant degree.
- Codeword examples:
  - 0000000, 1110000

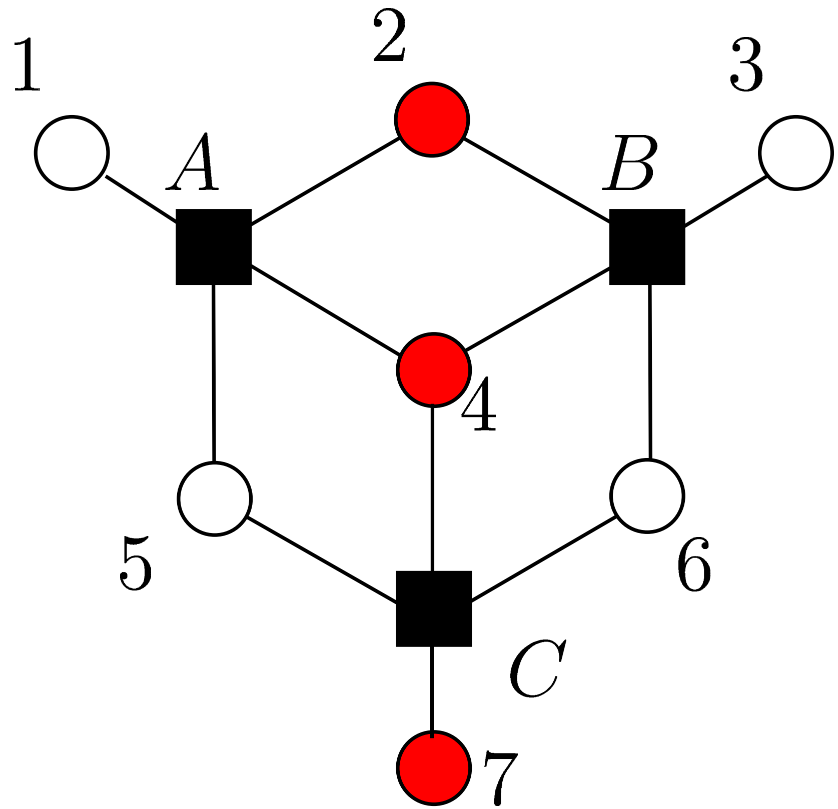
Low-density parity-check codes, factor graph



- Codebit nodes $1 \ldots n$.
- Check nodes $1 \ldots m$.
- Codewords: $y \in \{0,1\}^n$ where all check neighborhoods have even parity w.r.t. y.
- Rate $\geq 1 - m/n$.

- Low density: constant degree.
- Codeword examples:
  - 0000000, 1110000, 0100110

Low-density parity-check codes, factor graph



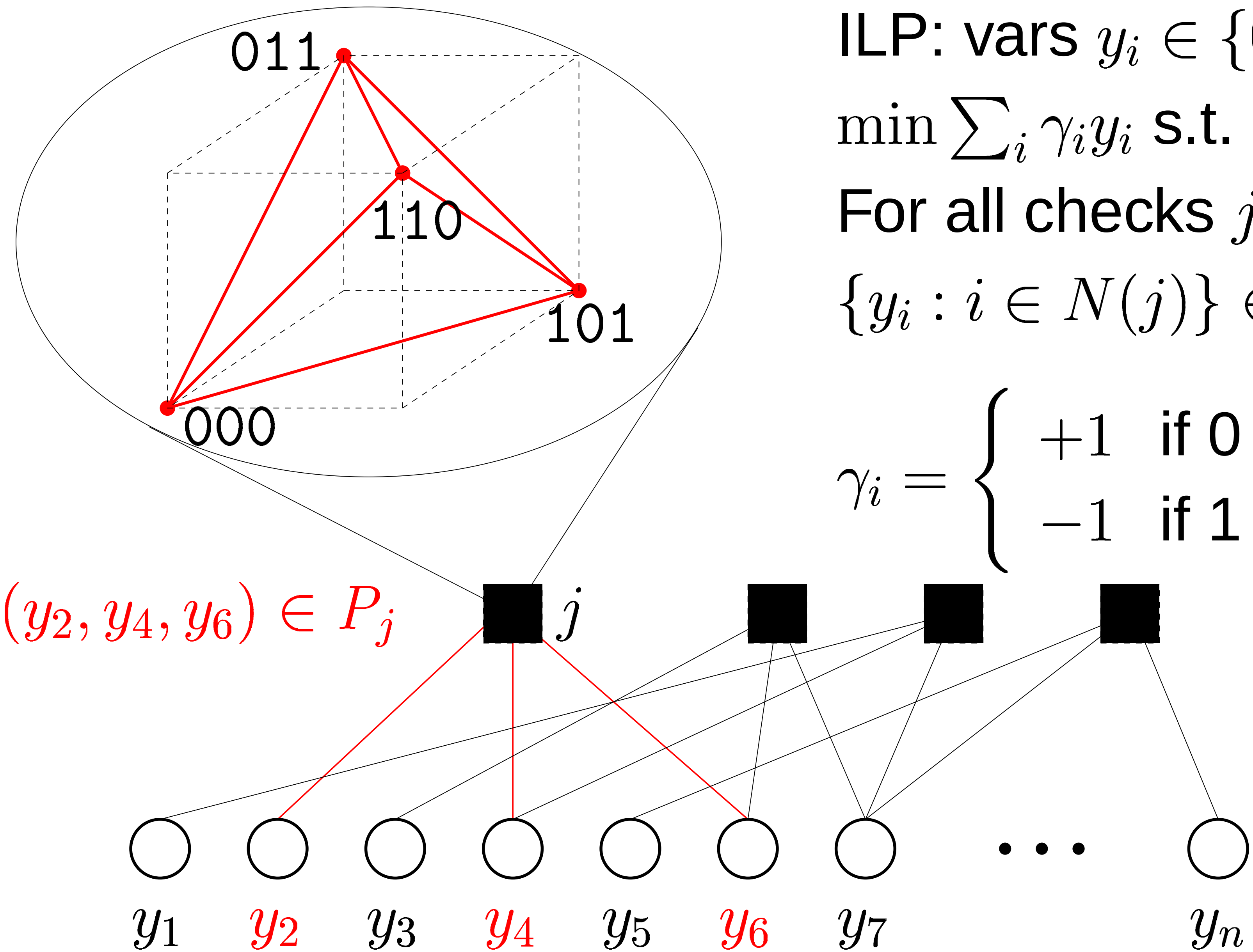
- Codebit nodes $1 \ldots n$.
- Check nodes $1 \ldots m$.
- Codewords: $y \in \{0,1\}^n$ where all check neighborhoods have even parity w.r.t. y.
- Rate $\geq 1 - m/n$.

- Low density: constant degree.
- Codeword examples:
  - 0000000, 1110000, 0100110, 0101001

Turbo codes and low-density parity-check (LDPC) codes

- Turbo codes [BGT ’93], LDPC codes [Gal ’62], with message-passing algs: lowest WER (in practice).
- Most successful theory: density evolution [RU, LMSS, RSU, BRU, CFDRU, ..., ’99...present].
  - Non-constructive, assumes local tree structure.
- “Finite-Length” analysis:
  - ML decoding finds most likely codeword; sub-optimal decoding finds most likely *pseudocodeword*.
  - Combinatorially understood pseudocodewords:
    - Deviation sets [Wib ’96, FKV ’01],
    - Tail-biting trellises [FKKR ’01],
    - Stopping sets (erasure channel) [DPRTU ’02].

LP relaxation on the factor graph [FKW '03]



ILP: vars $y_i \in \{0, 1\}$.

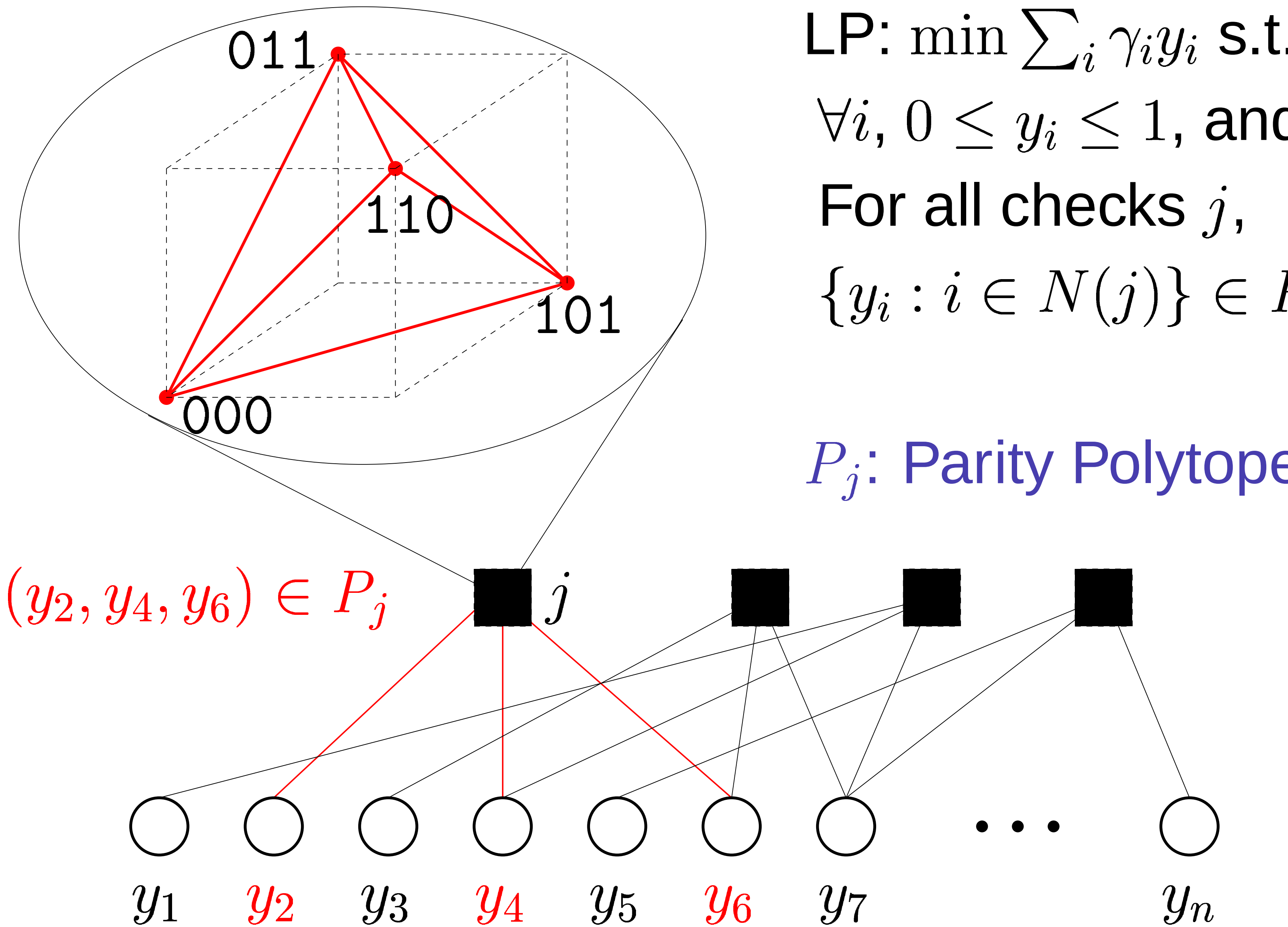
$\min \sum_i \gamma_i y_i$ s.t.

For all checks j,

$\{y_i : i \in N(j)\} \in P_j$.

$$\gamma_i = \begin{cases} +1 & \text{if 0 rec.} \\ -1 & \text{if 1 rec.} \end{cases}$$

LP relaxation on the factor graph [FKW '03]



LP: $\min \sum_i \gamma_i y_i$ s.t.

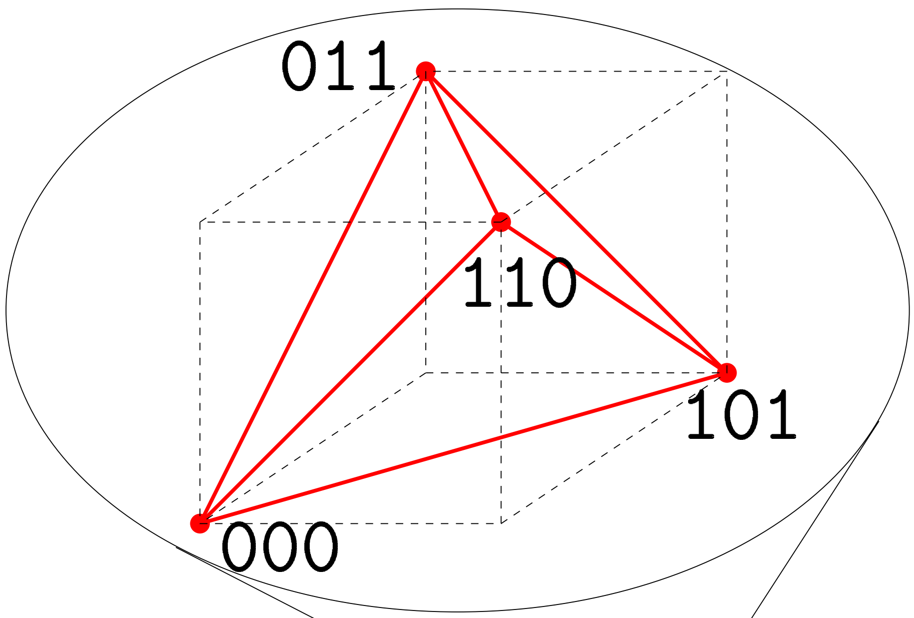
$\forall i$, $0 \le y_i \le 1$, and

For all checks j,

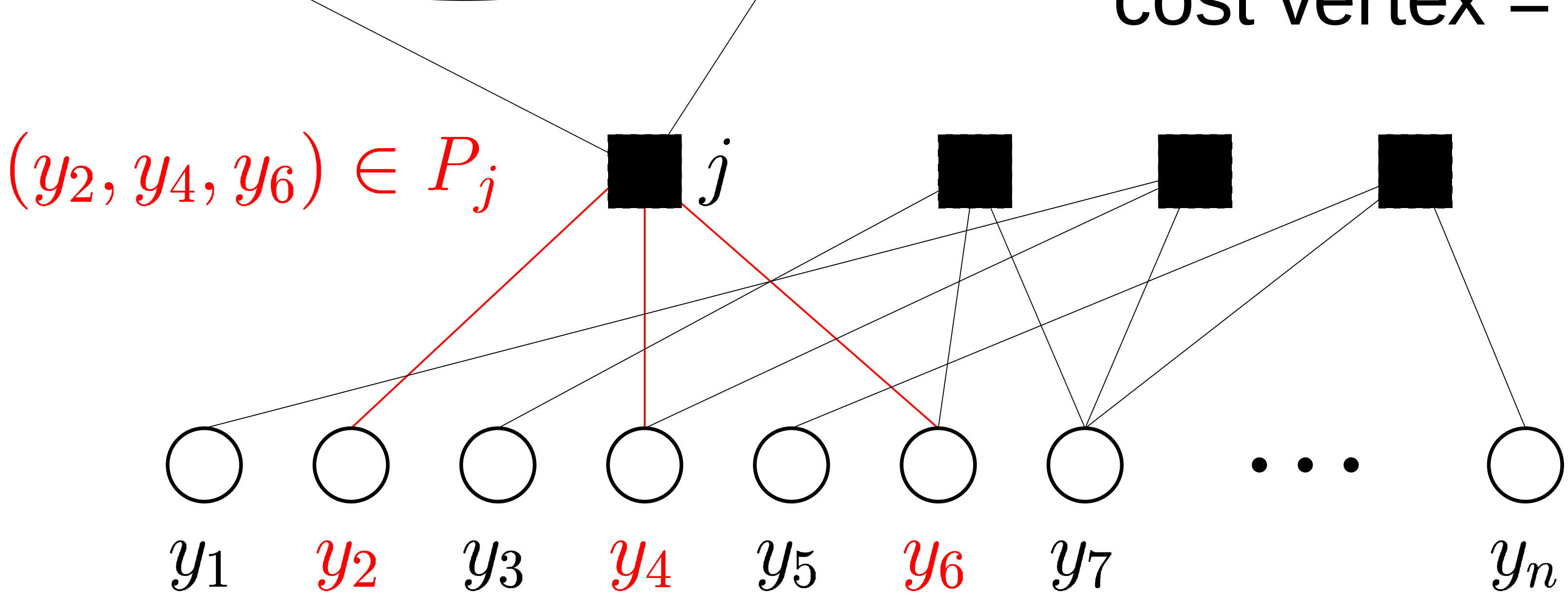
$\{y_i : i \in N(j)\} \in P_j$.

P_j: Parity Polytope [Y,J]

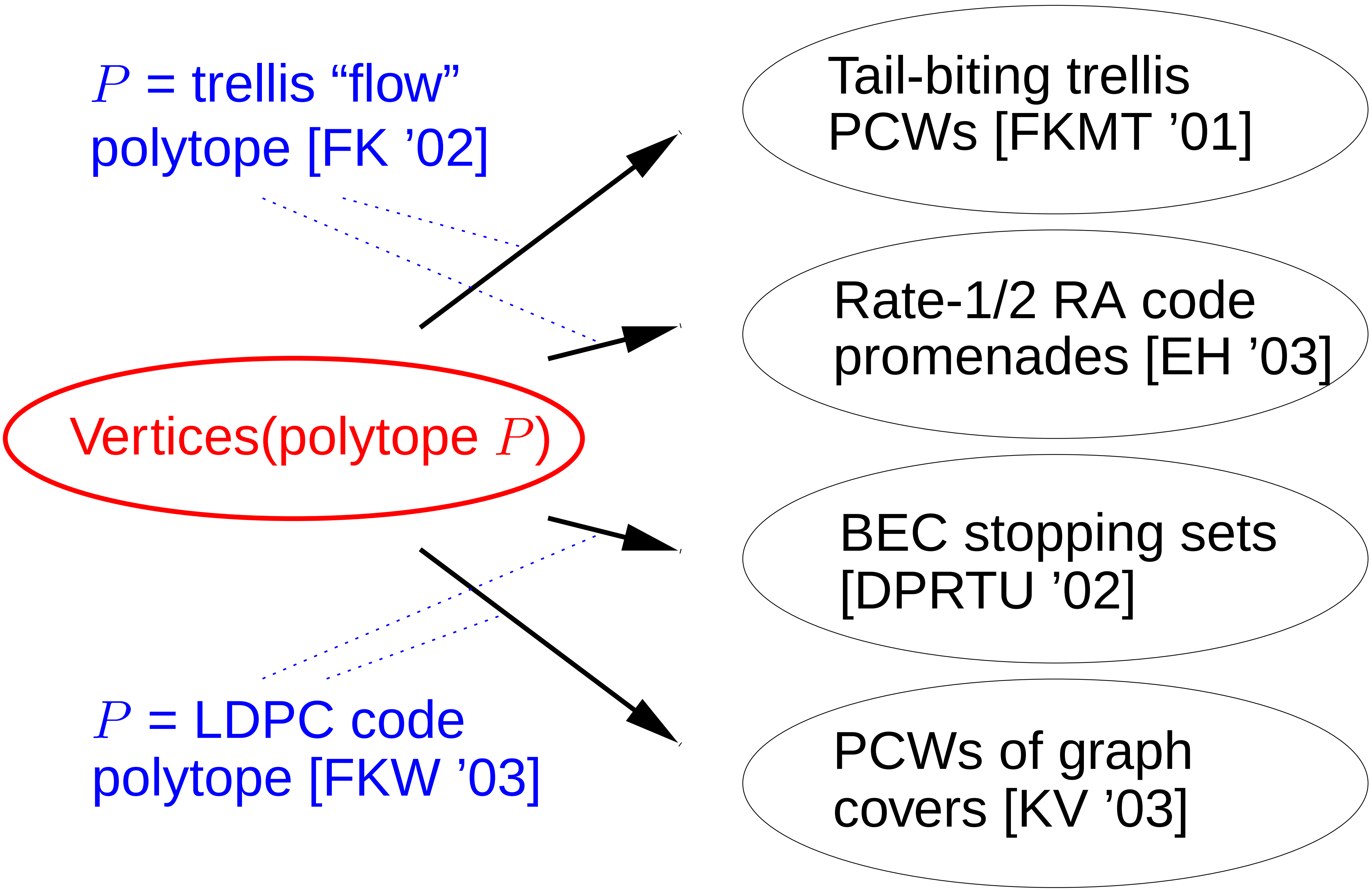
LP relaxation on the factor graph [FKW '03]



- Algorithm:
  1) Solve LP
  2) Output y if integral
- ML certificate
- Success $\iff$ lowest cost vertex = trans. y



Unifying other understood pseudocodewords



Success conditions: find zero-valued dual point

- Assume 0^n is transmitted (polytope symmetry); assume unique LP optimum (no problem).

  success $\iff$ Point 0^n is LP optimum

  $\iff$ $\exists$ dual feasible point w/ value 0

- Take LP dual, set dual objective = 0: polytope $\hat{P}$.

  success $\iff$ $\hat{P}$ non-empty

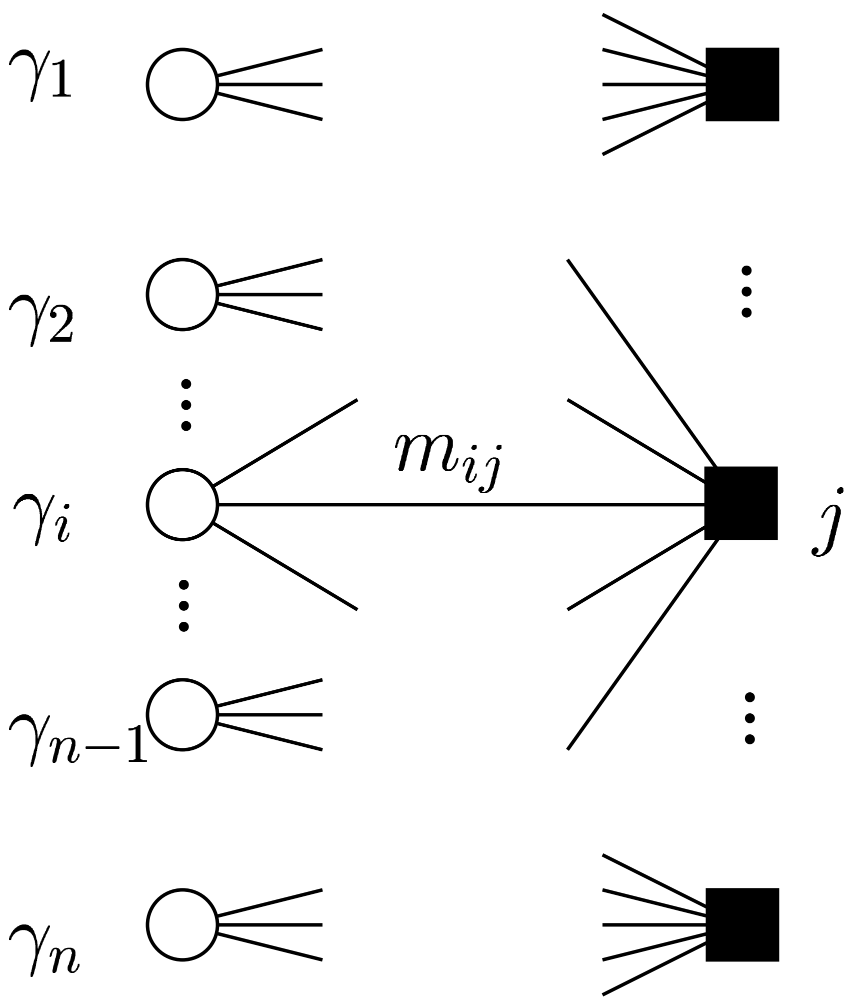
- Main result:

  Theorem: Suppose G (regular left-degree c) is an $(\alpha n, \delta c)$-expander, where $\delta > 2/3 + 1/(3c)$. Then the LP decoder succeeds if $< \frac{3\delta-2}{2\delta-1}\alpha n$ bits are flipped by the channel.

- New generalization to expander codes.

Edge weights

- Polytope $\hat{P}$ for LDPC code relaxation:



- Edge weights m_{ij} (free).
- For all code bits (left nodes) i,

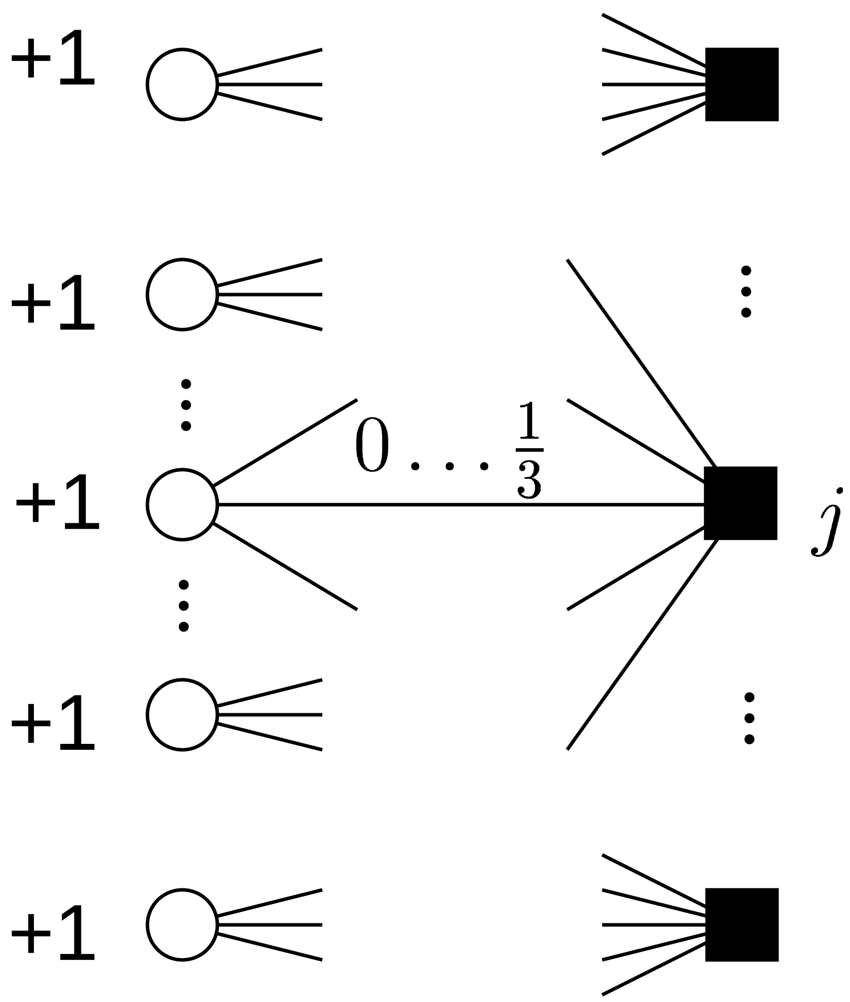
$$\sum_{j \in N(i)} m_{ij} \leq \gamma_i.$$

- For all checks j, pairs $i, i' \in N(j)$,

$$m_{ij} + m_{i'j} \geq 0.$$

Edge weights

- Polytope $\hat{P}$ for LDPC code relaxation:



- Edge weights m_{ij} (free).
- For all code bits (left nodes) i,

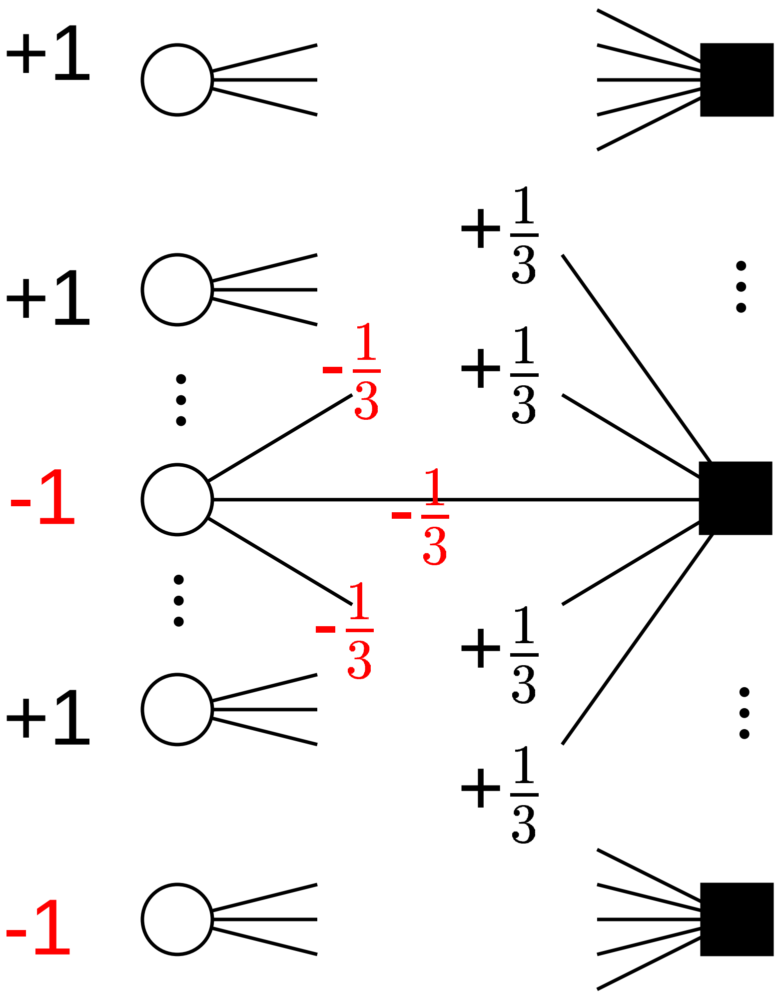
$$\sum_{j \in N(i)} m_{ij} \leq \gamma_i.$$

- For all checks j, pairs $i, i' \in N(j)$,

$$m_{ij} + m_{i'j} \geq 0.$$

Edge weights

- Polytope $\hat{P}$ for LDPC code relaxation:



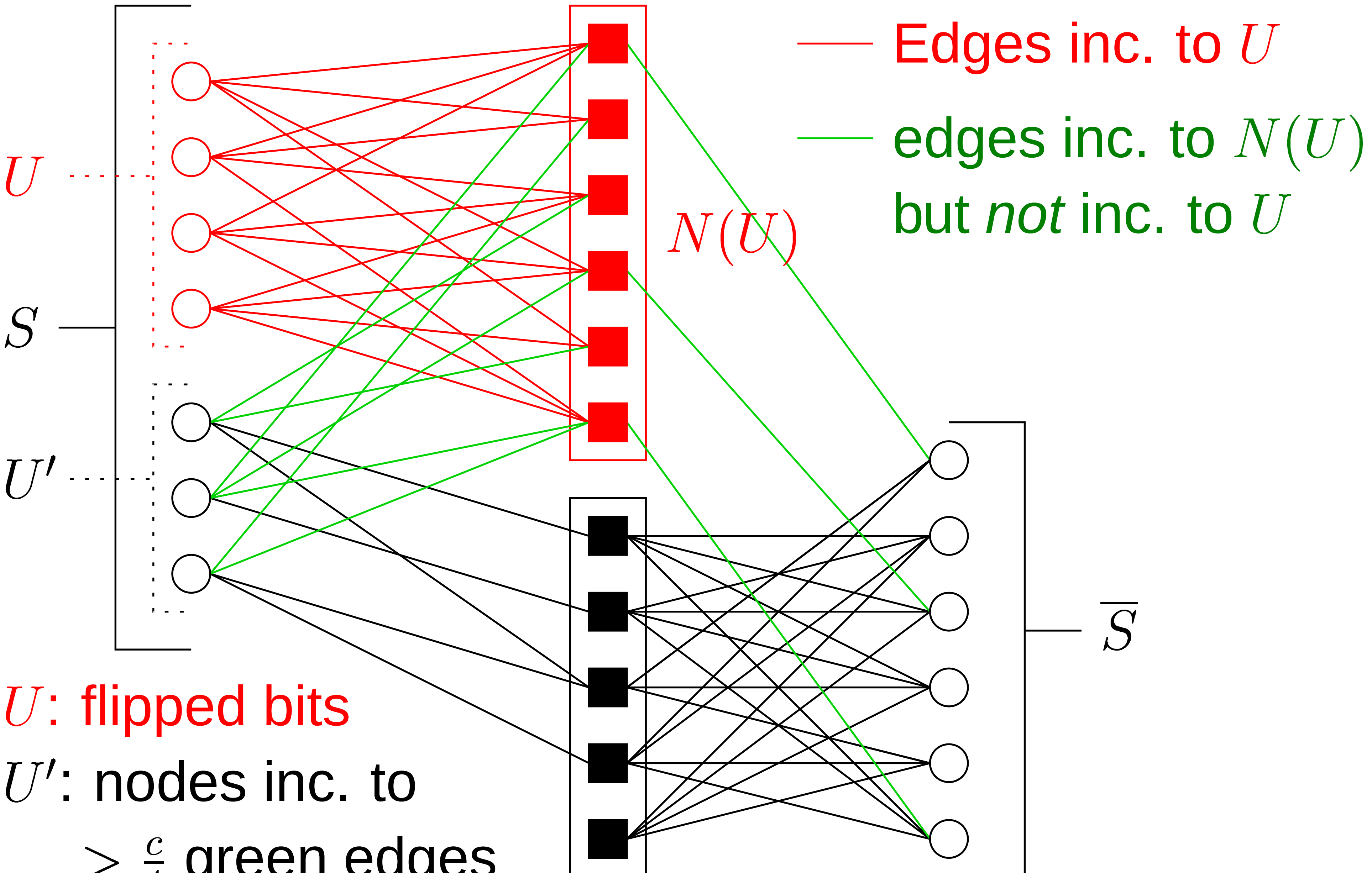
- Edge weights m_{ij} (free).
- For all code bits (left nodes) i,

$$\sum_{j \in N(i)} m_{ij} \leq \gamma_i.$$

- For all checks j, pairs $i, i' \in N(j)$,

$$m_{ij} + m_{i'j} \geq 0.$$

Weighting scheme: node sets S, U, U'

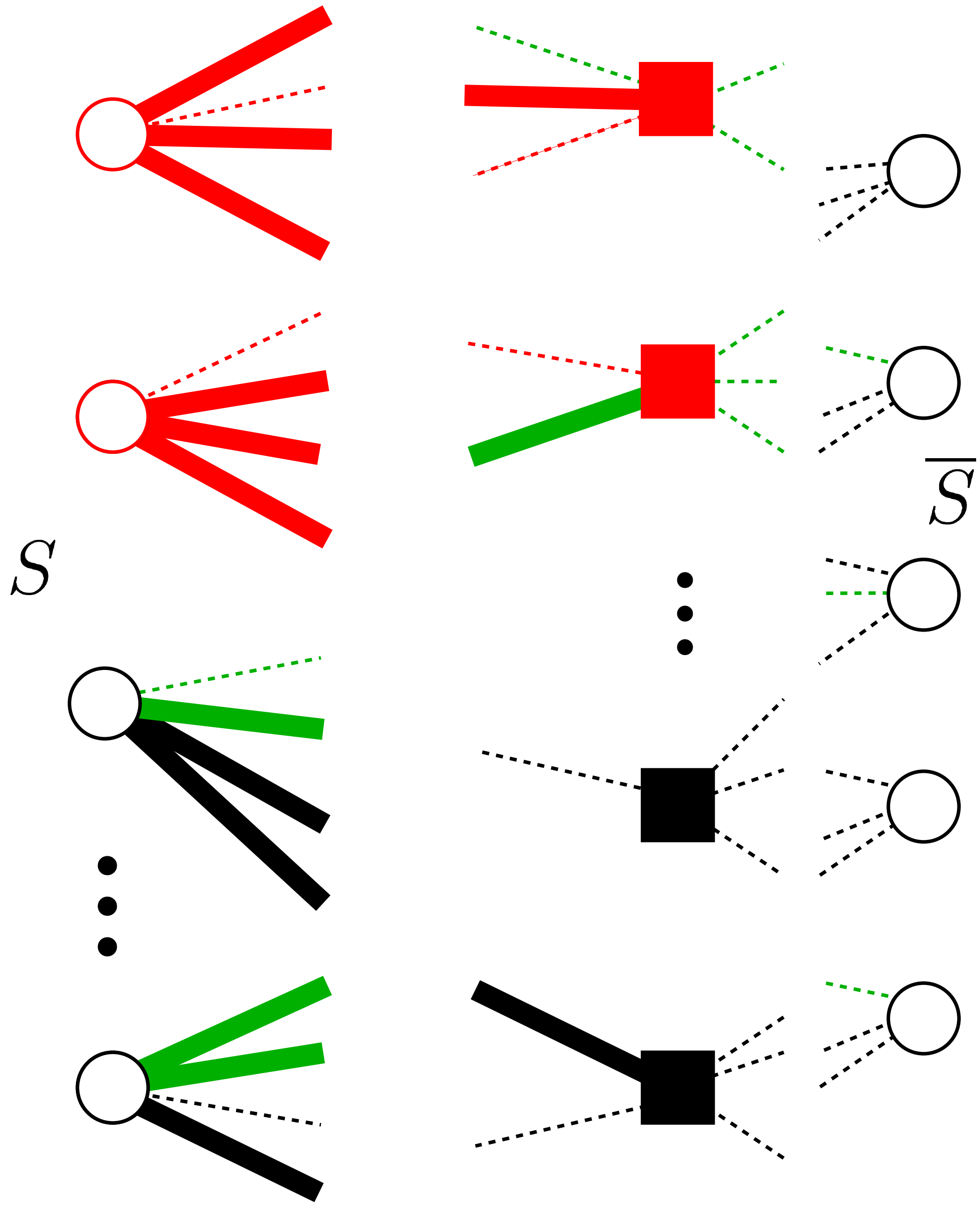


U: flipped bits

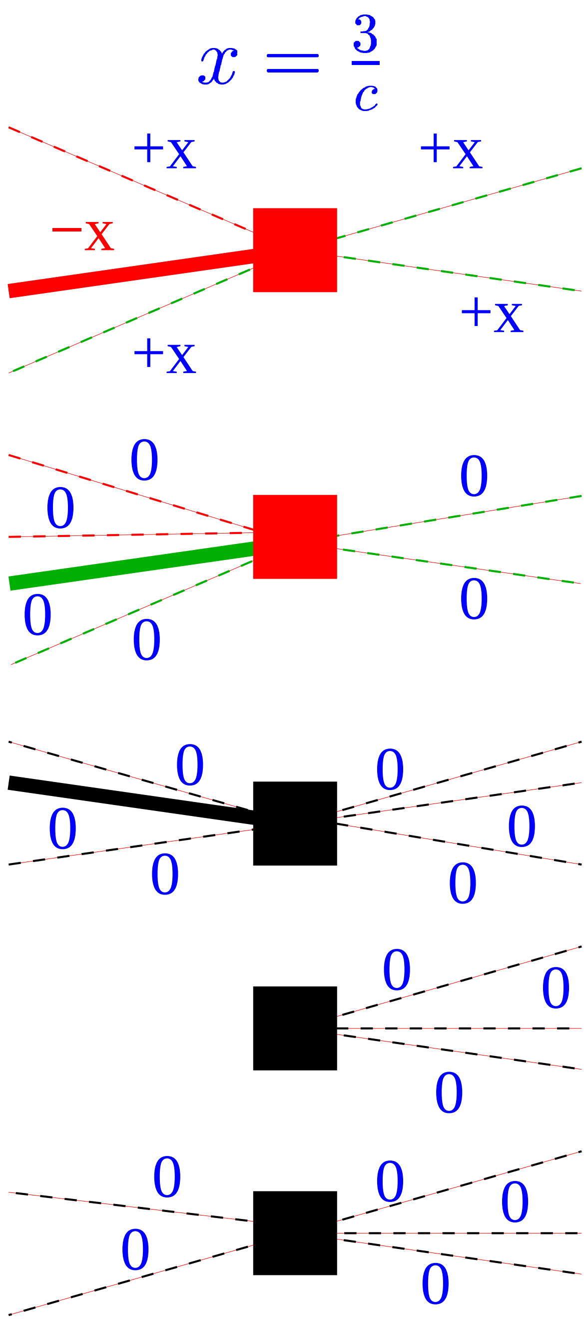
U': nodes inc. to $> \frac{c}{4}$ green edges (G is c-left-regular)

Weighting scheme: “The matching” M

- Find edge set M:
  - Nodes in S inc. to $\frac{3c}{4}$ M**-edges.**
  - Checks inc. to ≤ 1 M**-edge.**
- $(\alpha n, \delta)$-expander: every set of size $\leq \alpha n$ expands by a factor of $\geq \delta$.
- $(\alpha n, \frac{3c}{4})$-expander $\Longleftrightarrow$ $\exists$ matching M for all S, $|S| \leq \alpha n$.



Weighting scheme: weight values

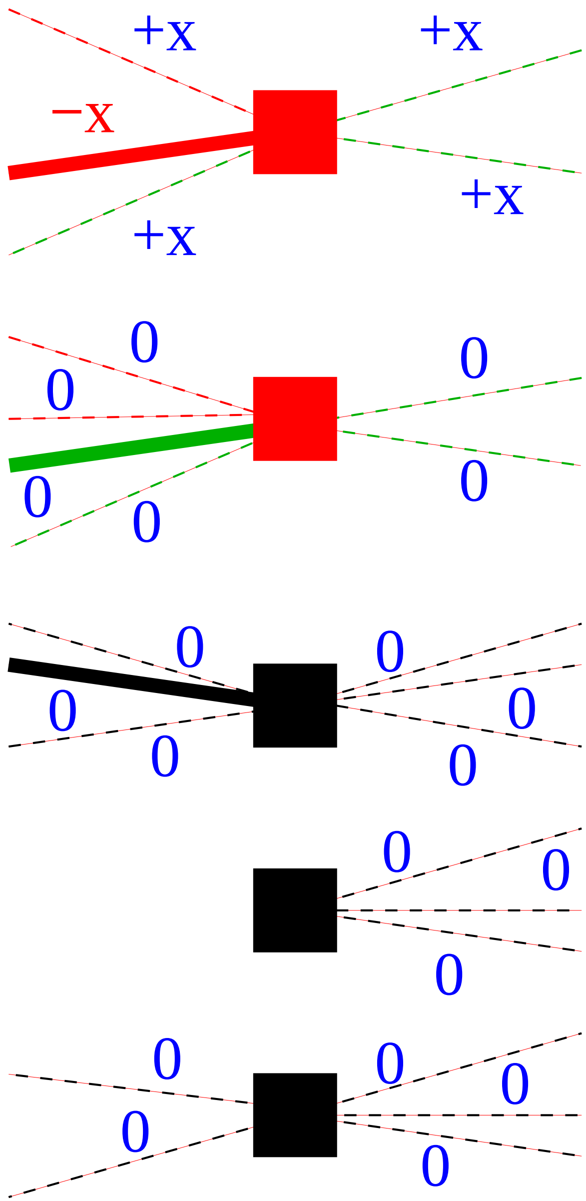


- For all checks j with incident red M-edge (i, j):
  - Set $m_{ij} = -x$;
  - Set all other incident edges $m_{i'j} = +x$.

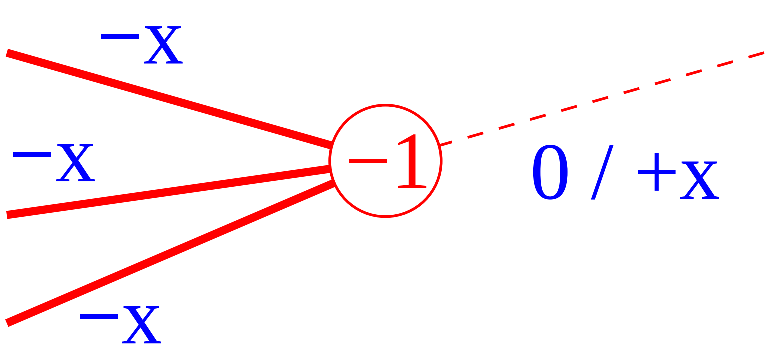
- Set all other $m_{ij} = 0$.

Weighting scheme: weight values

$x = \frac{3}{c}$



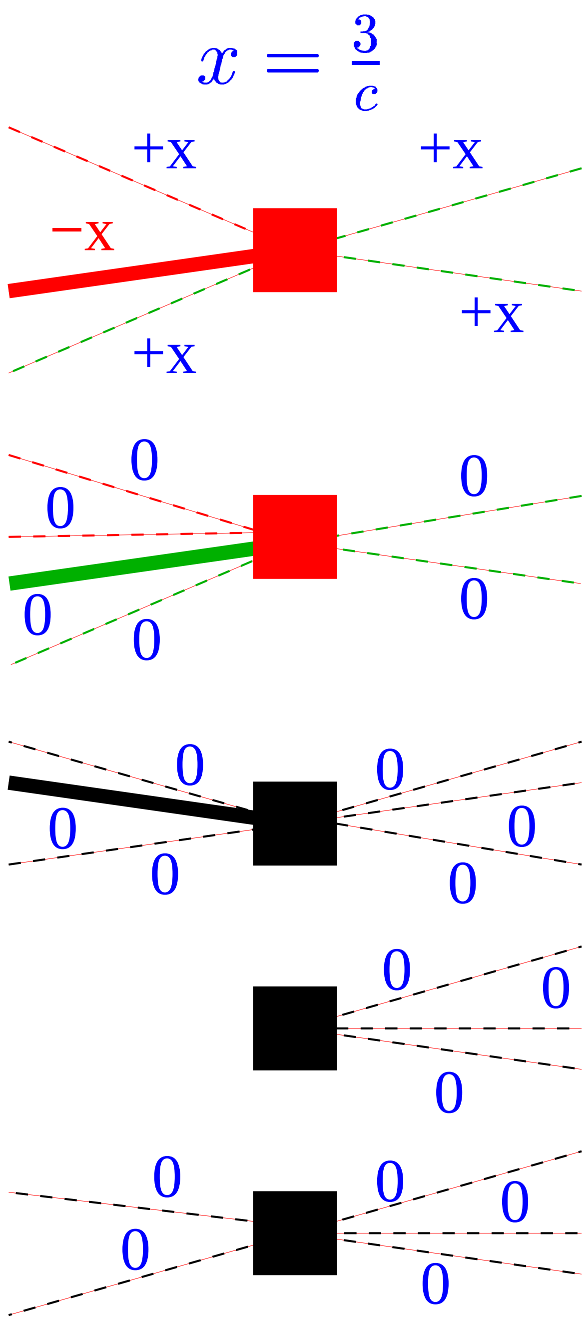
- Case 1: Node in U.



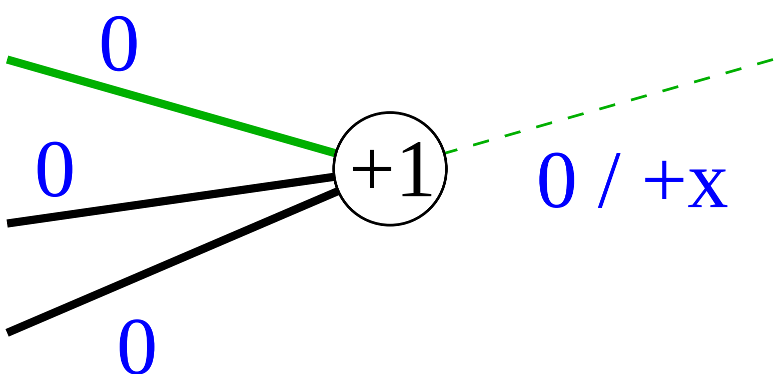
Node has $\frac{3}{4}c$ M-edges, each with weight $-x$, so

$$\begin{aligned} \sum m_{ij} &\leq \frac{1}{4}cx - \frac{3}{4}cx \\ &= -3/2 \\ &< -1. \end{aligned}$$

Weighting scheme: weight values



- Case 2: Node in U'.

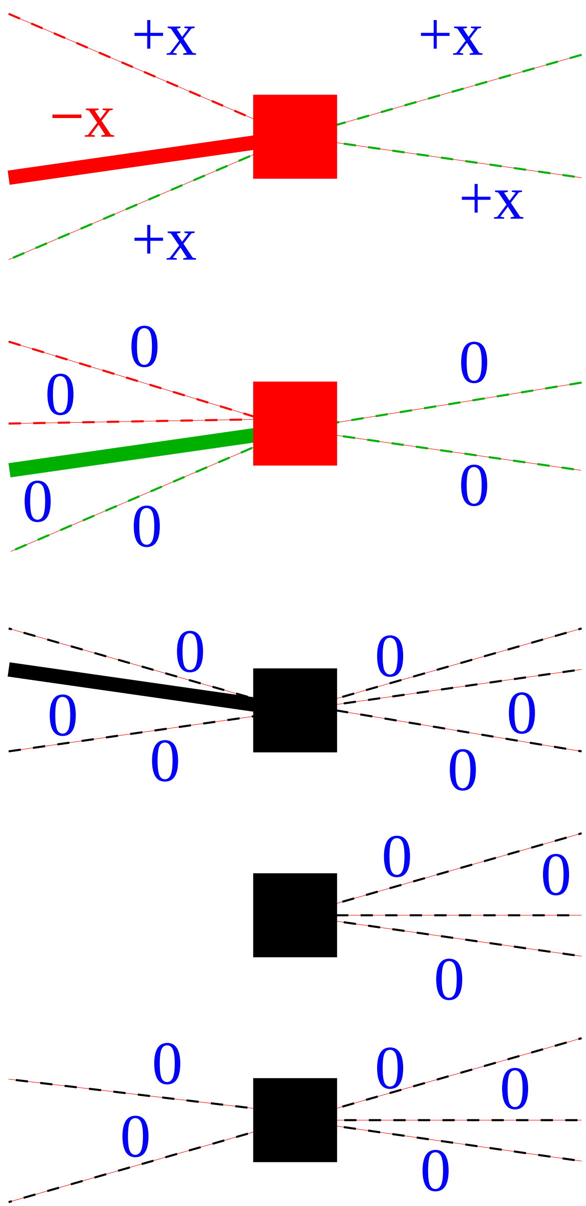


Node has $\frac{3}{4}c$ M-edges, each with weight 0, so

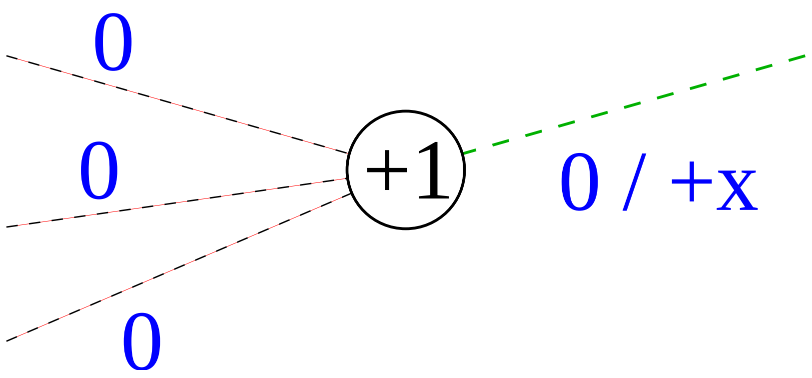
$$\begin{aligned} \sum m_{ij} &\leq \frac{1}{4}cx \\ &= 3/4 \\ &< +1. \end{aligned}$$

Weighting scheme: weight values

$x = \frac{3}{c}$



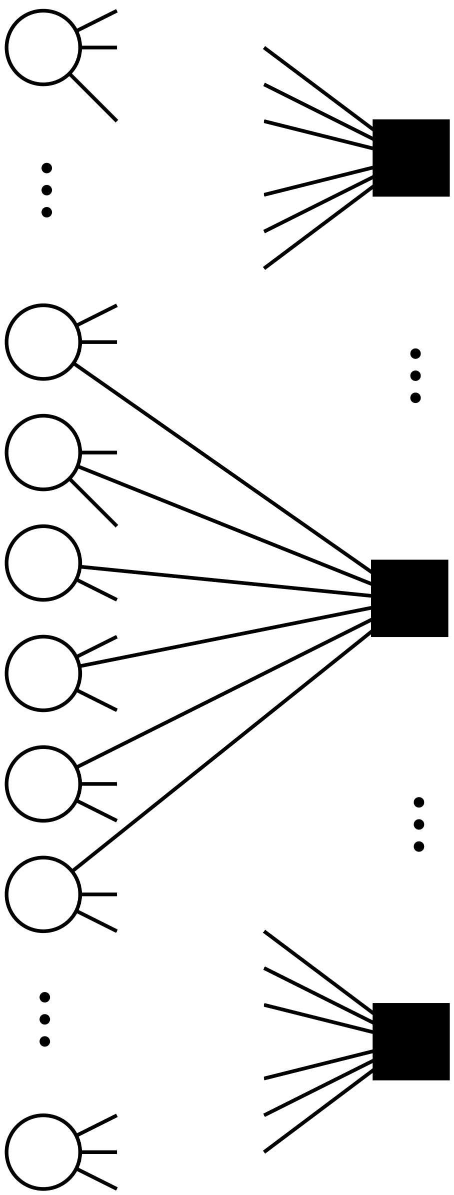
- Case 3: Node in $\overline{S}$.



Node has $\frac{3}{4}c$ edges *not* incident to $N(U)$. Each such edge has weight 0, so

$$\begin{aligned} \sum m_{ij} &\leq \frac{1}{4}cx \\ &= 3/4 \\ &< +1. \qquad \square \end{aligned}$$

Expander codes



- General version of expander codes [SS, BZ]:
  - Each "check" node j has subcode C_j.
  - Overall codeword: setting of bits to left nodes s.t. each check nbhd $N(j)$ is a codeword of C_j.
  - LDPC codes: special case where C_j = single parity check code.
- Ex: G is (3,6)-regular, C_j = $\{000000, 111000, 000111, 111111\}$.

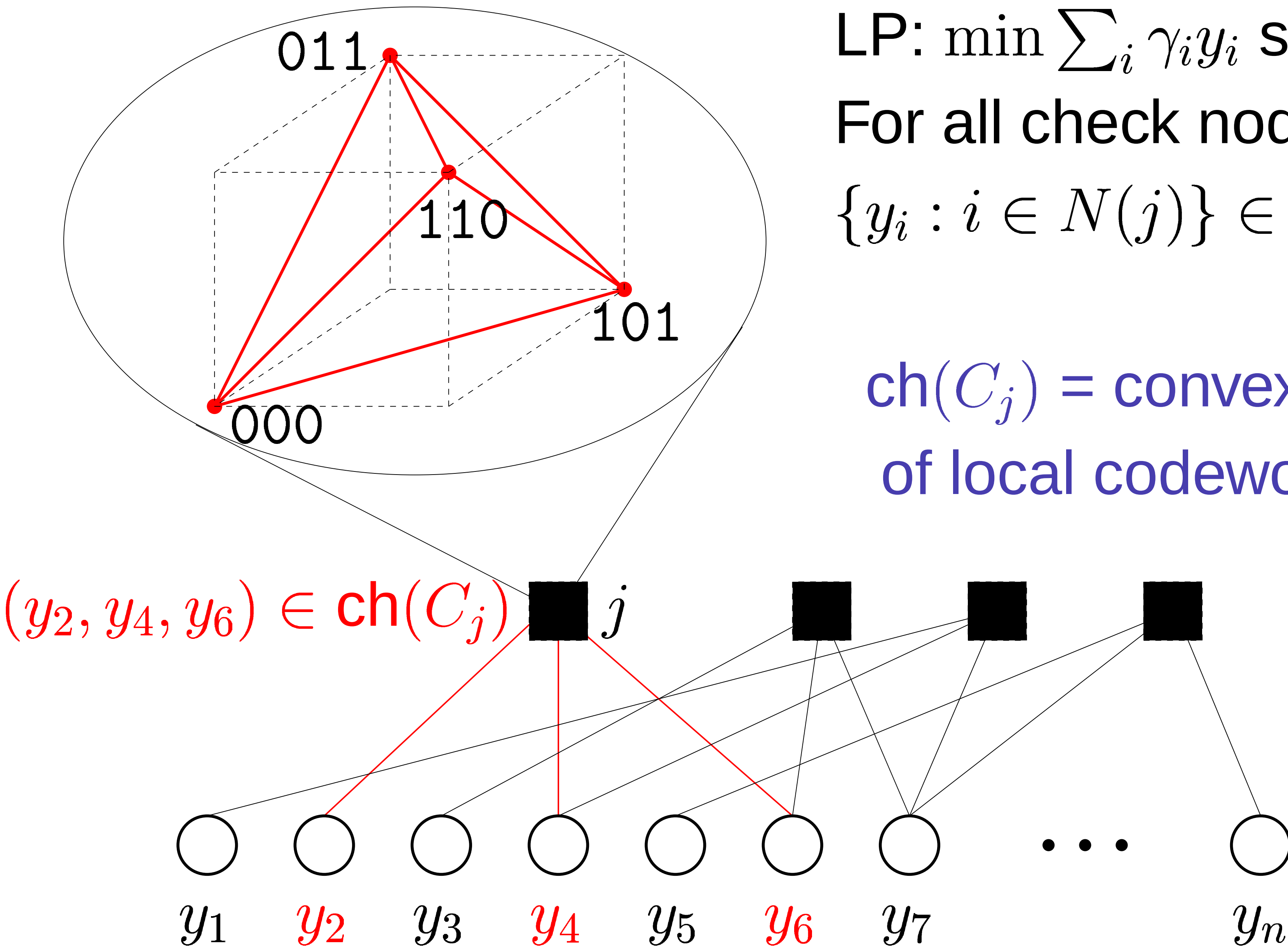
LP Relaxation for general expander codes

LP: $\min \sum_i \gamma_i y_i$ s.t.

For all check nodes j,

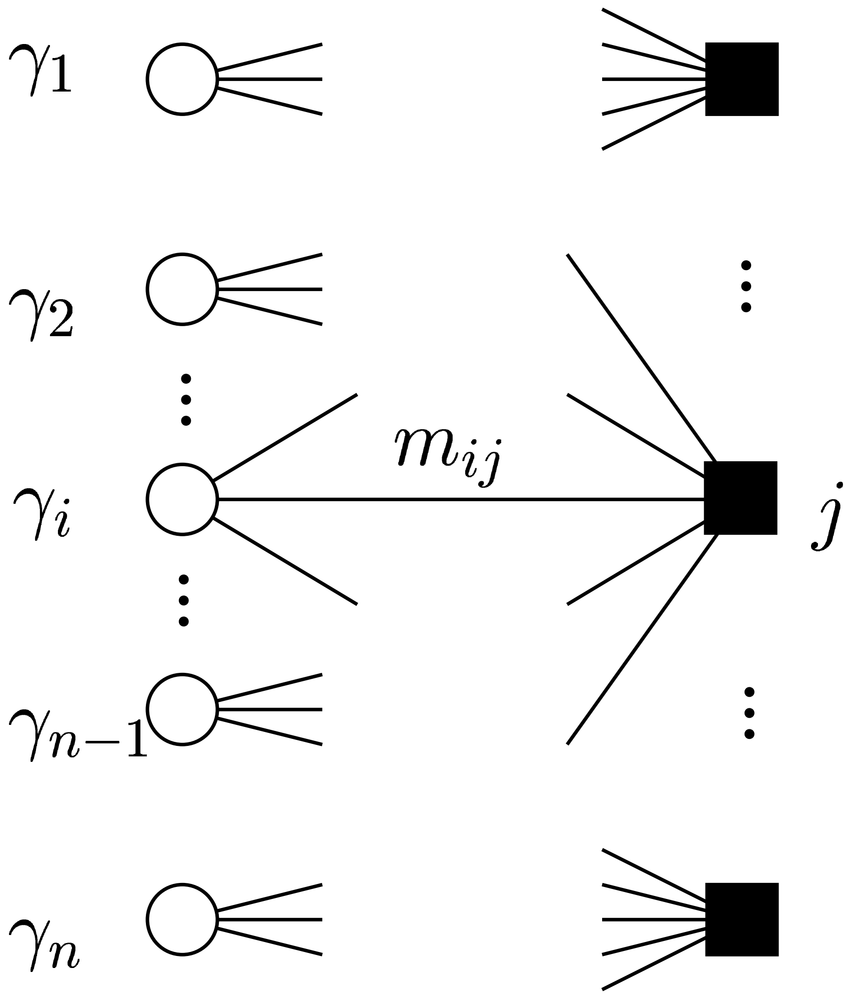
$\{y_i : i \in N(j)\} \in$ ch(C_j).

ch(C_j) = convex hull
of local codewords.



Edge weights for general expander codes

- Polytope $\hat{P}$ for general expander codes:



- Edge weights m_{ij}.
- For all code bits (left nodes) i,

$$\sum_{j \in N(i)} m_{ij} \leq \gamma_i.$$

- For all checks j, codewords $c \in C_j$,

$$\sum_{i \in \text{sup}(c)} m_{ij} \geq 0$$

Code construction

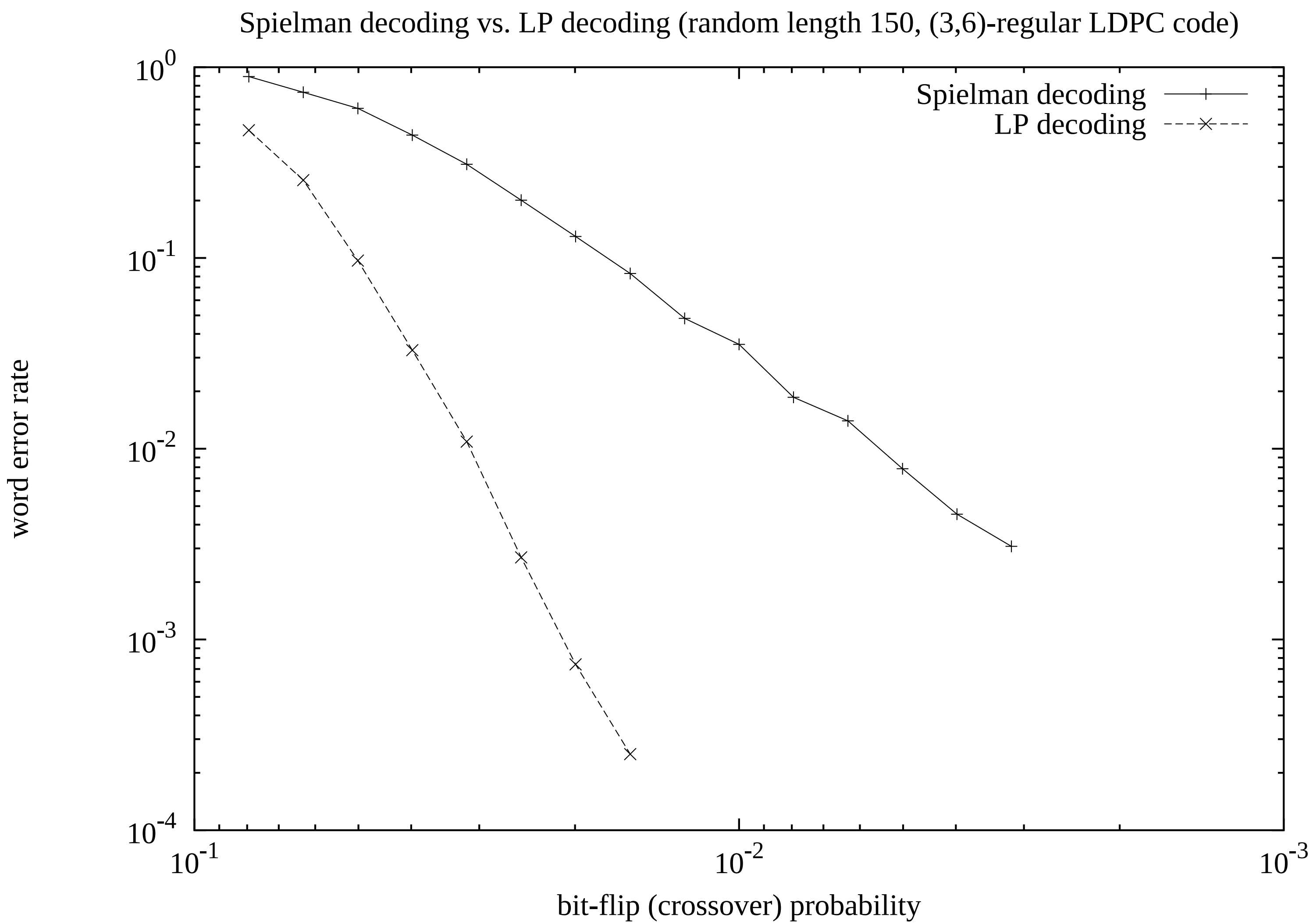
- Let G be $(2, d)$-regular (edge-incidence graph of d-regular expander). Fix some $0 < \epsilon < 1$.
- Set d sufficiently large s.t. C_j lies on GV-bound
  - Code C_j has distance ϵ, rate $1 - H(\epsilon)$.
  - Rate of overall code $\geq 1 - 2H(\epsilon)$.
- Weighting scheme: also benefits from expansion.
- Using Ramanujan graphs, Alon/Chung:

> **Theorem:** The LP decoder succeeds if $< \frac{\epsilon^2}{4} n$ bits are flipped by the channel.

- Sipser/Spielman: it. decoding corrects $\epsilon^2/48$ errors.
- Barg/Zemor: diff. algorithm, corrects $\epsilon^2/4$ errors.

Future Work #1

Improve results for LDPC codes, explain difference in performance.



Future Work #2

Explain weird situation using LDPCCs on AWGN:

- AWGN channel: $y_i \in \{-1, +1\}$ transmitted, $y_i + \mathcal{N}(0, \sigma^2)$ received.
- Log-likelihood ratio: set $\gamma_i =$ received value.
- Koetter/Vontobel [03]: Using LLRs γ_i, LP decoding has WER $= 2^{-O(n^{1-\epsilon})}$ for some $\epsilon > 0$.
- But, if you *quantize* first (set $\gamma_i' =$ sign(γ_i)), you get BSC, and using our result, get WER = $2^{-\Omega(n)}$.
- In other words, it is sometimes *good* to throw out information.
- Optimal decoders do not have this property; somehow this sub-optimal decoder does.

Future Work #3-#8

- Using more general codes, compete with best known results on rate vs. fraction corrected (Forney, Barg/Zemor, Guruswami/Indyk).
- Find more general weighting scheme $\rightarrow$ use more general graph-theoretic properties than expansion.
- Prove something better for turbo codes.
- Deepen connection to iterative algorithms (sum-product).
- Use non-linear optimization.
- Consider non-binary codes.