Decoding Turbo-Like Codes via Linear Programming

Jon Feldman David Karger

MIT Laboratory for Computer Science

Cambridge, MA, USA

{jonfeld,karger}@theory.lcs.mit.edu

Turbo Codes

- Introduced in 1993 [Berrou, Glavieux, Thitimajshima].
- Unprecedented error-correcting performance.
- Simple encoder, “belief-propagation” decoder.
- Theoretical understanding limited:
  - Distance properties bad [KU ’98, BMMS ’02];
  - Analysis for random codes [LMSS ’01, DPTRU ’02];
  - Decoder unpredictable (may not even converge!).
- Related: low-density parity-check codes, expander codes, expander-based codes, tornado codes, etc.

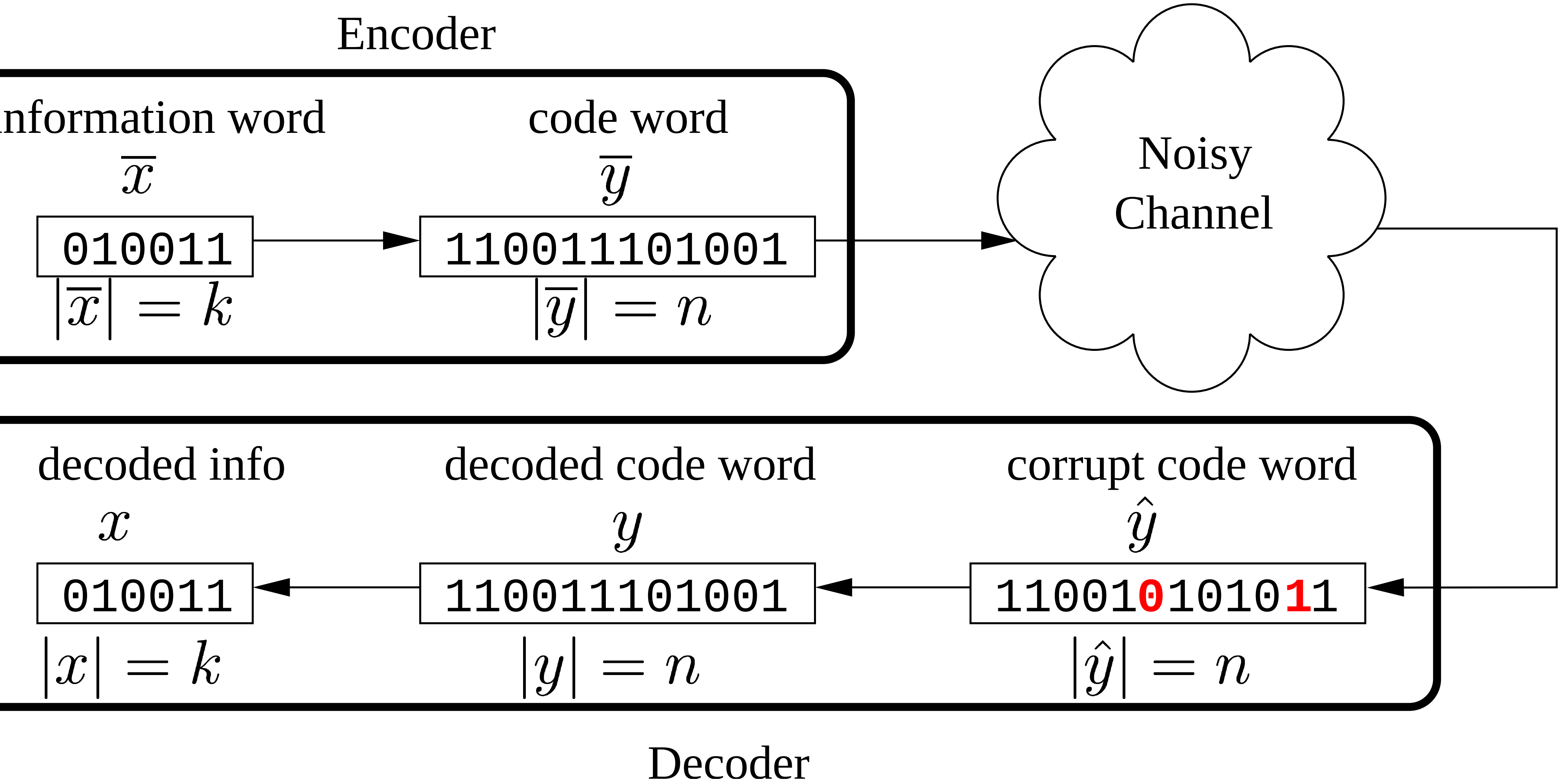
Our contributions

- Polynomial-time decoder using linear programming.
- Decodes any turbo code, other related codes (LDPC).
- Exact characterization of error patterns that cause decoding failure (not known for BP).
- Code construction with inverse-poly error bound (also not known for BP).

Outline

- Error correcting codes.
- Using LP relaxation for decoding.
- Turbo Codes (Repeat-Accumulate codes).
- Code construction, error rate bounds.

Binary Error-Correcting Code



- Each bit flipped indep. w/ prob. p (small constant).

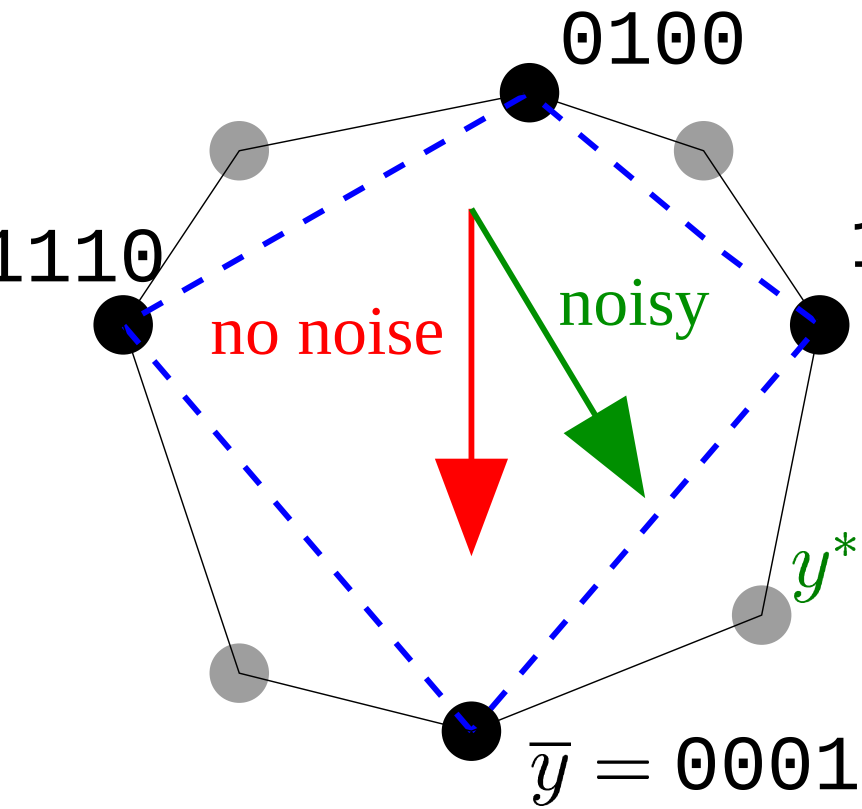
Maximum-Likelihood Decoding

Given: Corrupt code word $\hat{y}$.
Find: Code word y such that Hamming distance $\Delta(\hat{y}, y)$ is minimized.

- Integer/Linear Programming formulation:
  - Code $C \subset \{0,1\}^n$.
  - Variables $y_t \in \{0,1\}$ for each code bit.
  - Polytope $P \subset \mathbb{R}^n$ s.t. $P \cap \{0,1\}^n = C$.
  - Integer Program: $\text{Minimize } \Delta_\ell(\hat{y}, y) \text{ s.t. } y \in P$.
  - Relaxation: $0 \le y_t \le 1$.

Linear Programming Relaxation

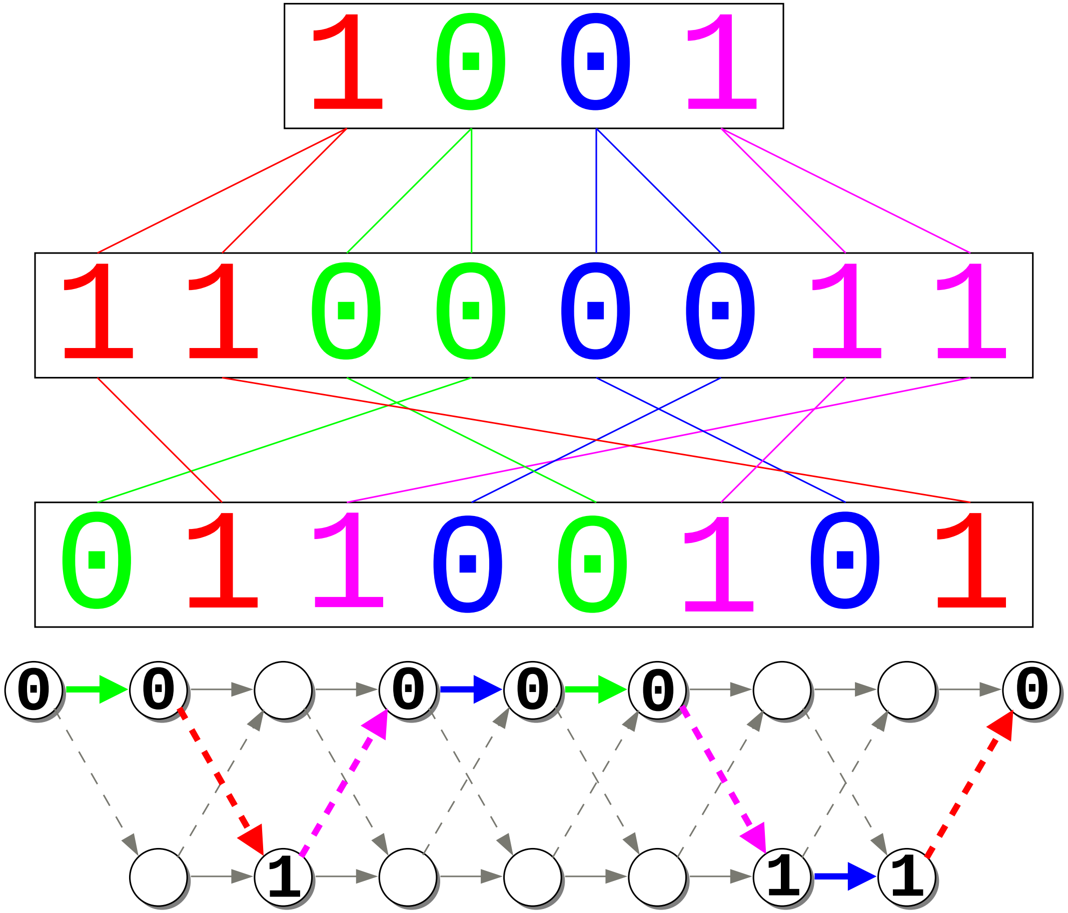
- Algorithm: Solve LP. If y^* integral, output y^*, else "error."
- *ML certificate* property: all outputs are ML code words.
- How do we measure the quality of a relaxation?
  - Want low word error rate (WER) := $\Pr_{\text{noise}}[\, y \neq \overline{y}\,]$.



- LP: Min $\Delta_\ell(\hat{y}, y)$: $y \in P$.
- No noise: $\overline{y}$ optimal.
- Noise: perturbation of objective function.
- Design relaxation where only large perturbations cause word error.

Repeat-Accumulate Codes

[Divsalar, Jin, McEliece, 1998]



“Trellis”

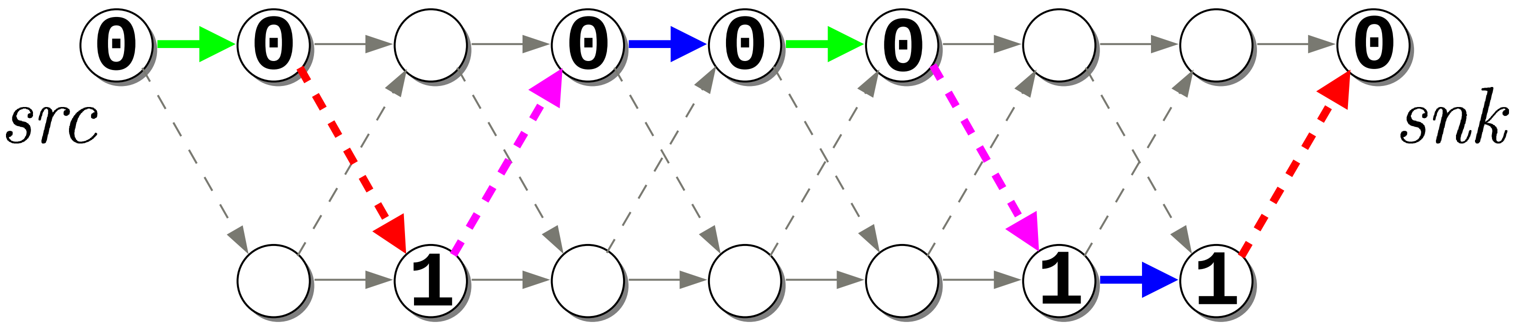
Information (word $\overline{x}$)

Repeat

Permute

Accumulate (node labels make code word $\overline{y}$)

Repeat-Accumulate Linear Program



- Code words $\iff$ *agreeable* paths.
- RALP: "flow-like" LP to find the min-cost agreeable path.
  - Flow $\bar{f}$: integral unit flow along path taken by encoder.
  - If $\bar{f}$ is the min-cost agreeable flow $\implies$ decoding success.
- *Tanner graph*: Model of the code. Edge costs: -1 for each bit flipped in the channel, $+1$ for each bit not flipped.
- *Promenade*: Closed circuit of the Tanner graph G.

Using Promenades for Error Bounds

Theorem 1: RALP decodes correctly iff there is no negative-cost promenade in G.

- Analogous theorem holds for any "turbo-like" code or LDPC code, with a generalization of "promenade."
- For rate-1/2 RA codes: If G has large girth $\implies$ promenades large $\implies$ negative cost promenades rare.
- Erdös (or [BMMS '02]): Hamiltonian 3-regular graph with girth $\log n$.

Theorem 2: For any $\epsilon > 0$, as long as $p < 2^{-4(\epsilon+(\log 24)/2)}$, WER $\leq n^{-\epsilon}$.

Extensions

- Connections to iterative methods [FKW, (*Allerton '02)*]:
  - Iterative "tree-reweighted max-product" tries to solve dual of our LP.
  - Subgradient method for solving LP very similar to standard belief propagation.
- Generic LP for any low-density parity-check code (incl. all turbo-like codes).
  - Connections to "min-sum" belief-propagation algorithm.
  - Lifting procedure to approach ML decoding.
- Tighter analysis of promenade distribution.
- Other "memoryless" channels (e.g. AWGN).

Future Work

- New constructions and WER bounds:
  - Lower rate turbo codes (rate-1/3 RA).
  - Conjecture: $\exists$ rate-$1/k$ RA code s.t. WER $\leq 2^{-n^{\epsilon}}$.
  - Other LDPC codes (expander codes, irregular LDPC codes, etc.)?
- Faster algorithm for solving agreeable flow / decoding LPs?
- Deeper connections to belief-propagation?
- LP decoding of other code families, channel models?