Energy efficient bit-flipping LDPC decoders

Chris Winstead

Emmanuel Boutillon, Gopal Sundar, Tasnuva Tithi

LE/FT Lab
Department of Electrical and Computer Engineering
Utah State University

June 8, 2016

Outline

1. Background: message-passing vs bit-flipping LDPC decoders
2. Recent advances:
   - Noisy and Probabilistic bit-flipping[1]
   - Differential Decoding[2]
   - Rewinding[3]
3. Architecture comparisons
4. Hardware results for 10GBase-T ethernet
5. Conclusions

[1] Sundararajan, Winstead, and Boutillon 2014; Rasheed et al. 2014.
[2] Cushon et al. 2014.
[3] Tithi, Winstead, and Sundararajan 2015.

Preview

Bit-flipping decoders used to be “toy” algorithms.

Preview

Bit-flipping decoders used to be "toy" algorithms.

In the past few years, they became competitive with standard algorithms.

Preview

Bit-flipping decoders used to be "toy" algorithms.

In the past few years, they became competitive with standard algorithms.

Noise enhancement and message memory have seemingly magical properties, but are hard to analyze

Preview

Bit-flipping decoders used to be "toy" algorithms.

In the past few years, they became competitive with standard algorithms.

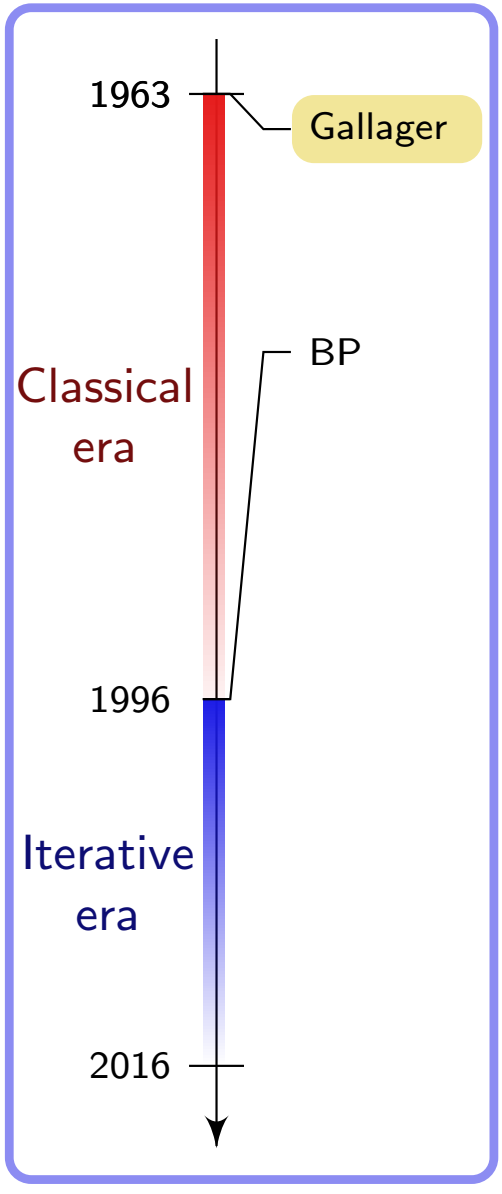
Noise enhancement and message memory have seemingly magical properties, but are hard to analyze

Results for a commercial 10 Gigabit ethernet standard:

- Best energy efficiency reported (up to 34$\times$ lower than benchmark)
- Lowest gate area reported (up to 6.6$\times$ lower than benchmark)
- Little or no loss in performance compared to benchmark
- Good throughput and average latency
- Some tradeoff in worst-case latency

Background on LDPC Algorithms

Short Introduction to Coding Theory



Gallager's LDPC Codes

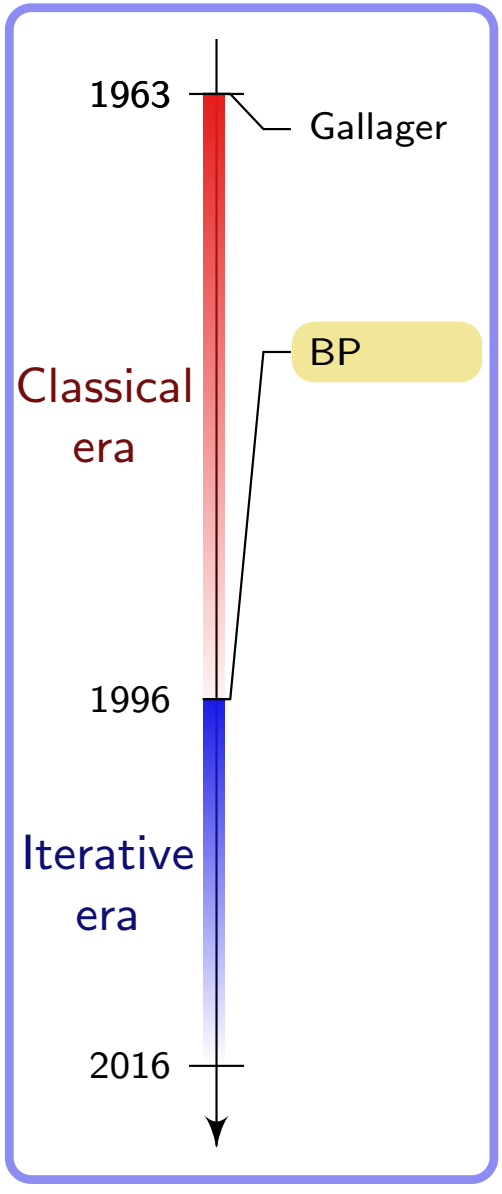
Gallager introduced the main ideas:

- Decoding with probability calculations.
- Bit-flipping algorithms for binary symmetric channels.

After 1996, a general theory of iterative message passing emerged based on belief propataion.

After 2001, bit-flipping algorithms were extended to AWGN and other channels, but got comparatively little attention.

Short Introduction to Coding Theory



Gallager's LDPC Codes

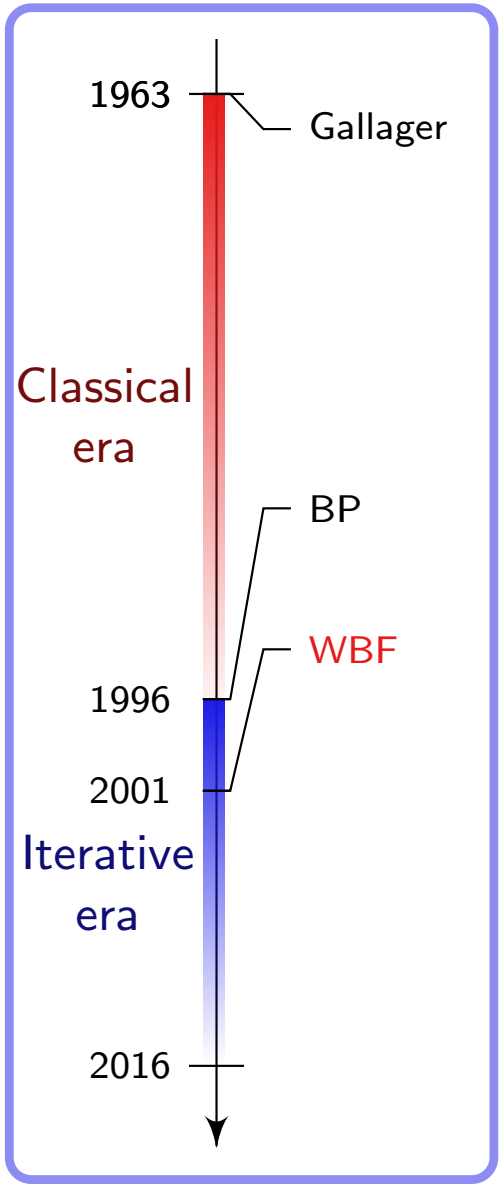
Gallager introduced the main ideas:

- Decoding with probability calculations.
- Bit-flipping algorithms for binary symmetric channels.

After 1996, a general theory of iterative message passing emerged based on belief propataion.

After 2001, bit-flipping algorithms were extended to AWGN and other channels, but got comparatively little attention.

Short Introduction to Coding Theory



Gallager's LDPC Codes

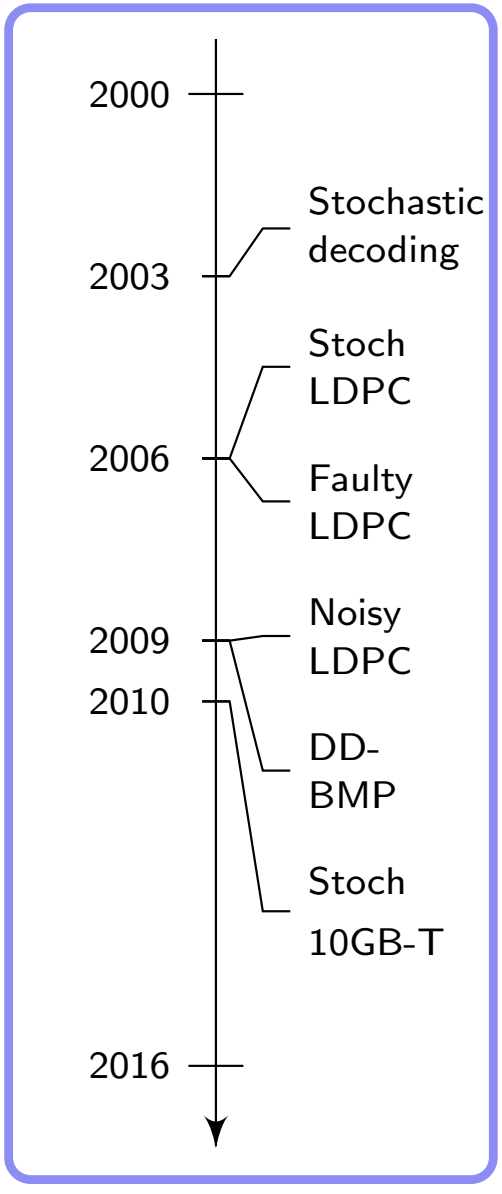
Gallager introduced the main ideas:

- Decoding with probability calculations.
- Bit-flipping algorithms for binary symmetric channels.

After 1996, a general theory of iterative message passing emerged based on belief propataion.

After 2001, bit-flipping algorithms were extended to AWGN and other channels, but got comparatively little attention.

Short Introduction to Coding Theory



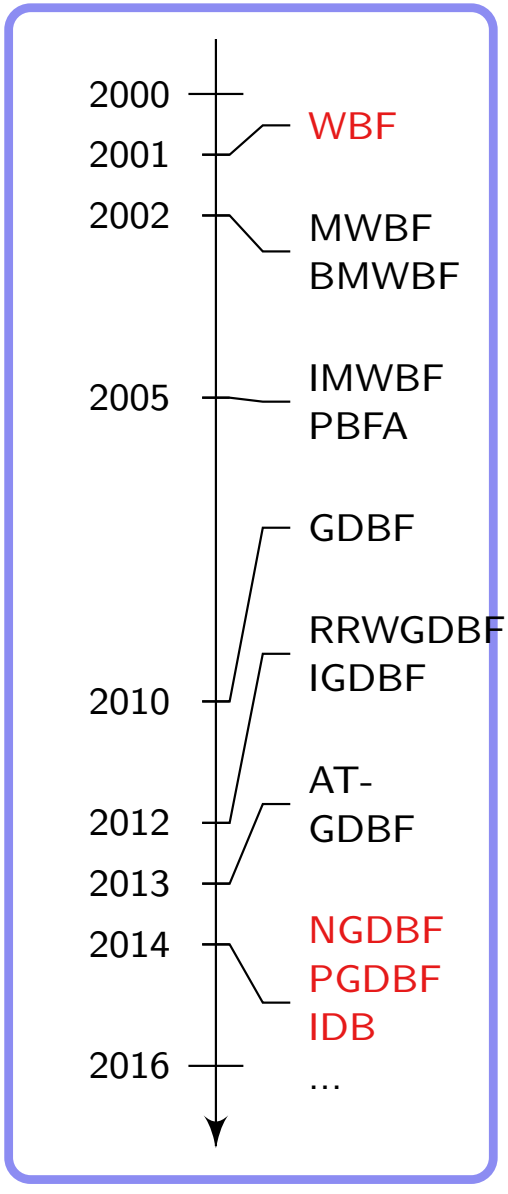
Single-Bit Message Passing (SBM)

Reduced-complexity decoding was introduced with stochastic decoding, and later differential decoding with binary message passing (DD-BMP).

Message-passing is close to BP, but only exchange one bit per message instead of computing probabilities.

SBM is distinct from bit-flipping.

Short Introduction to Coding Theory



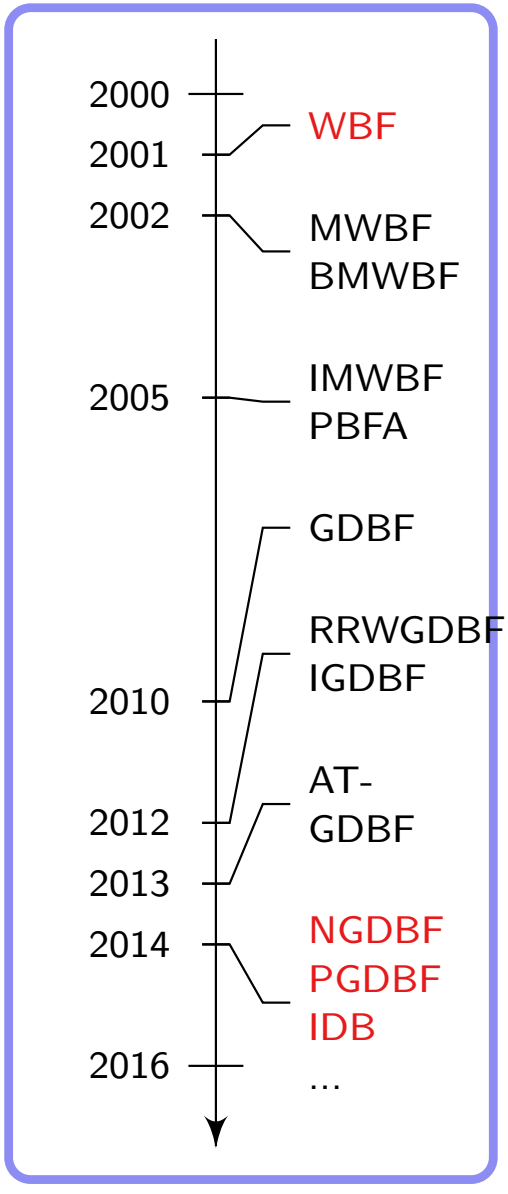
Bit-Flipping Algorithms

Bit-flipping algorithms diversified into many heuristic approaches with partial theoretical basis.

Acronyms grew longer.

After 2012, big improvements appeared with "Noisy", "Probabilistic" and "Differential" methods (NGDBF, PGDBF, IDB, resp.).

Short Introduction to Coding Theory



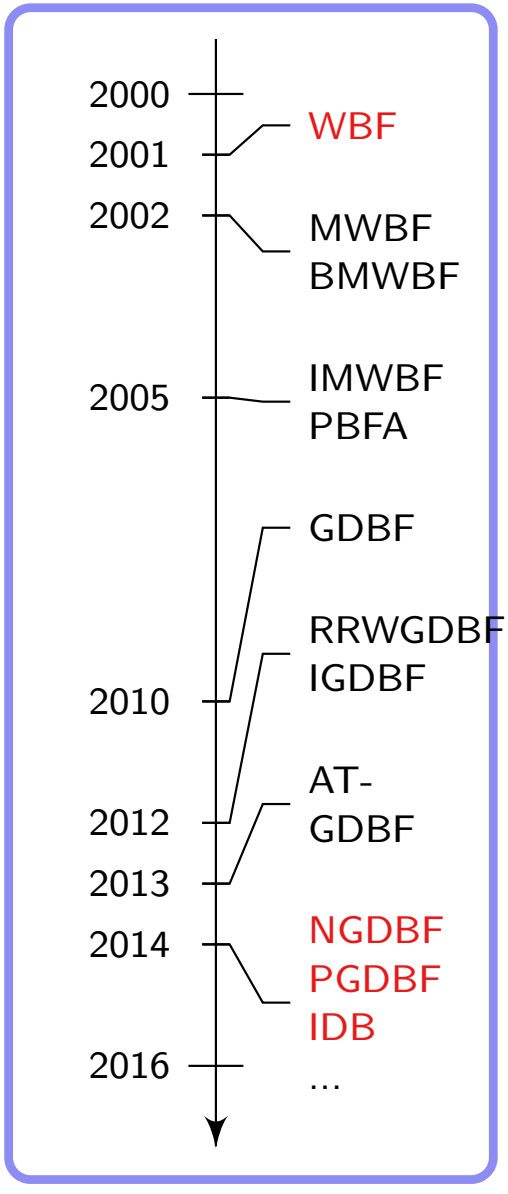
Bit-Flipping Algorithms

Bit-flipping algorithms diversified into many heuristic approaches with partial theoretical basis.

Acronyms grew longer.

After 2012, big improvements appeared with "Noisy", "Probabilistic" and "Differential" methods (NGDBF, PGDBF, IDB, resp.).

Short Introduction to Coding Theory



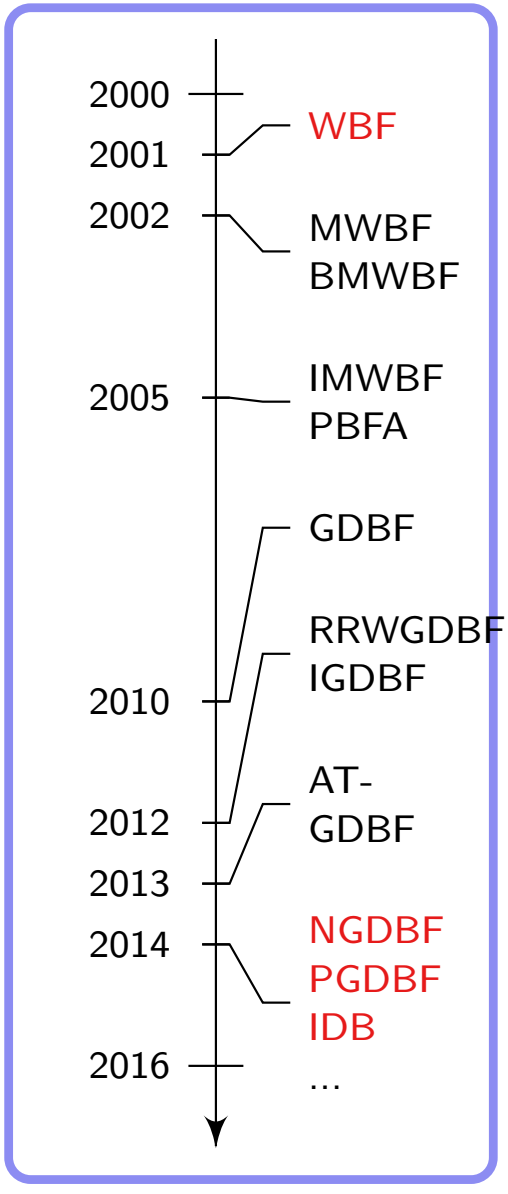
Bit-Flipping Algorithms

Bit-flipping algorithms diversified into many heuristic approaches with partial theoretical basis.

Acronyms grew longer.

After 2012, big improvements appeared with "Noisy", "Probabilistic" and "Differential" methods (NGDBF, PGDBF, IDB, resp.).

Short Introduction to Coding Theory



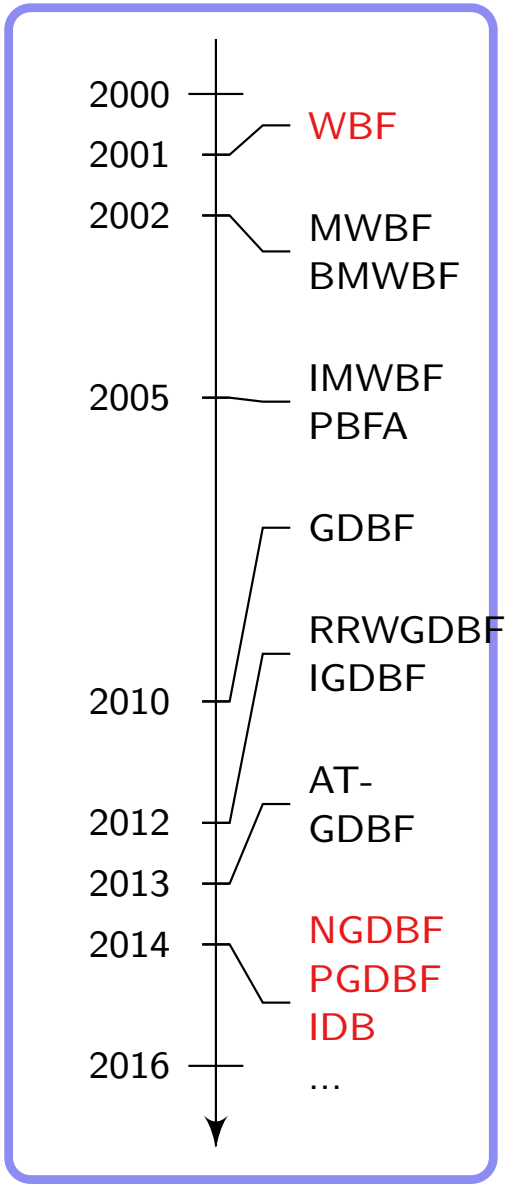
Bit-Flipping Algorithms

Are these just new exhibits in the "zoo" of decoding algorithms?

Answer: No! The latest algorithms are highly competitive with message-passing and can beat BP in some cases.

But... the research is still mostly empirical and heuristic.

Short Introduction to Coding Theory



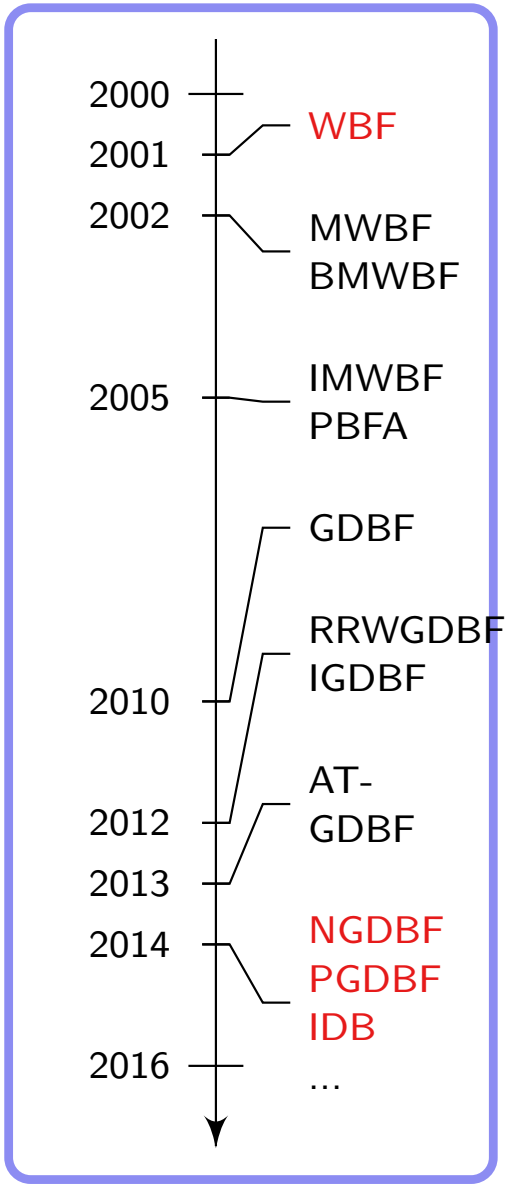
Bit-Flipping Algorithms

Are these just new exhibits in the "zoo" of decoding algorithms?

Answer: No! The latest algorithms are highly competitive with message-passing and can beat BP in some cases.

But... the research is still mostly empirical and heuristic.

Short Introduction to Coding Theory

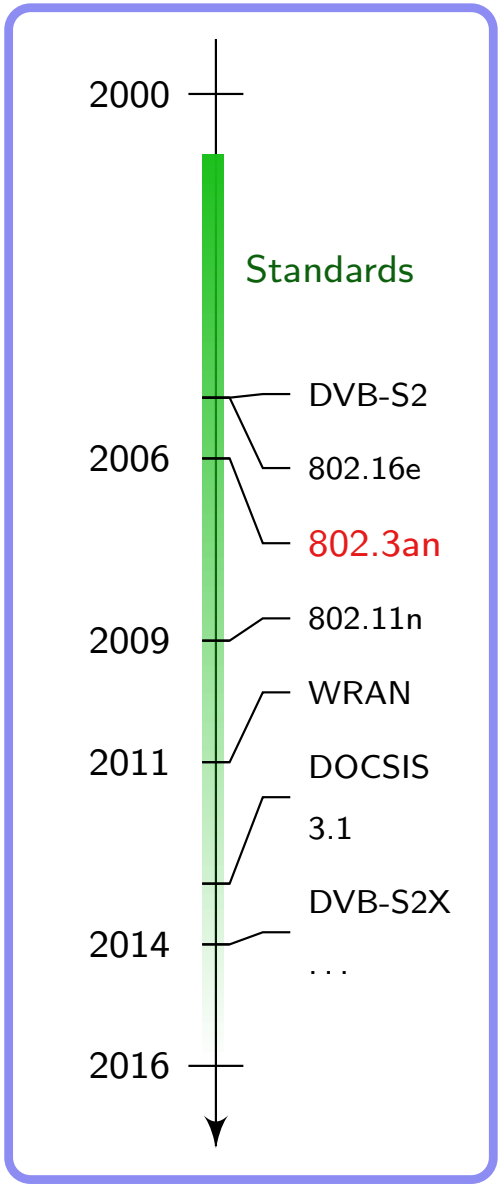


Bit-Flipping Algorithms

Are these just new exhibits in the "zoo" of decoding algorithms?

Answer: No! The latest algorithms are highly competitive with message-passing and can beat BP in some cases.

But... the research is still mostly empirical and heuristic.



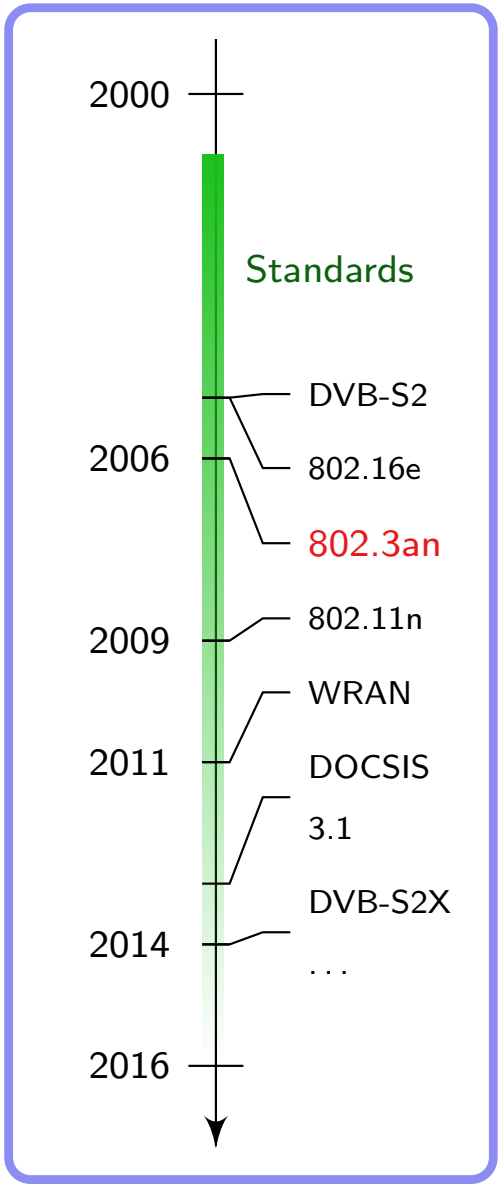
Standards

To assess applicability, it helps to look at performance on commercial standard codes.

LDPC codes are now adopted in numerous standards:

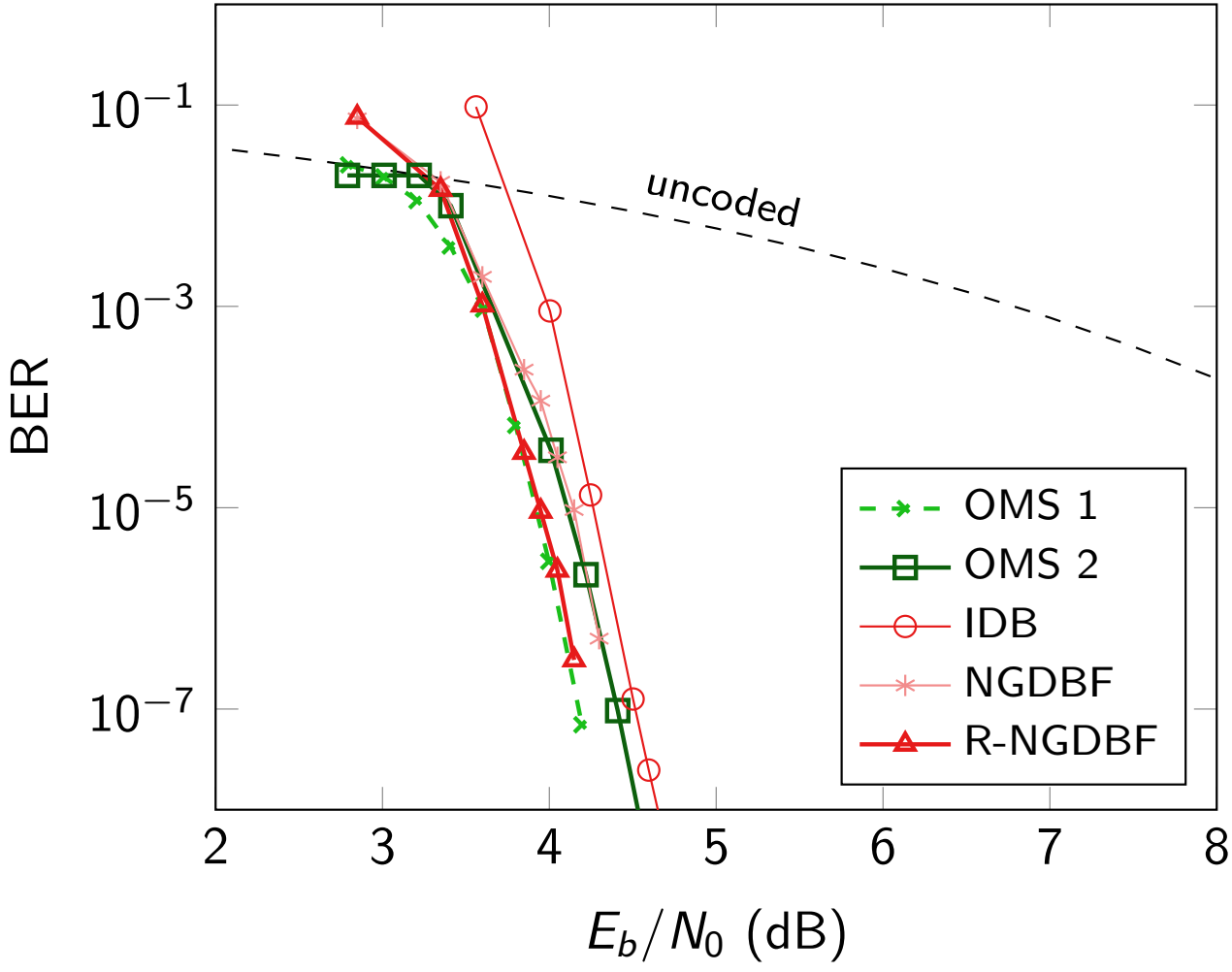
- Digital Video Broadcast (DVB-S2)
- WiMAX (802.16e)
- Wifi (802.11n and 802.11ac)
- 10GBase-T ethernet (802.3an)
- Home network (G.hn)
- Mobile connectivity (3GPP2-UMB)

Short Introduction to Coding Theory

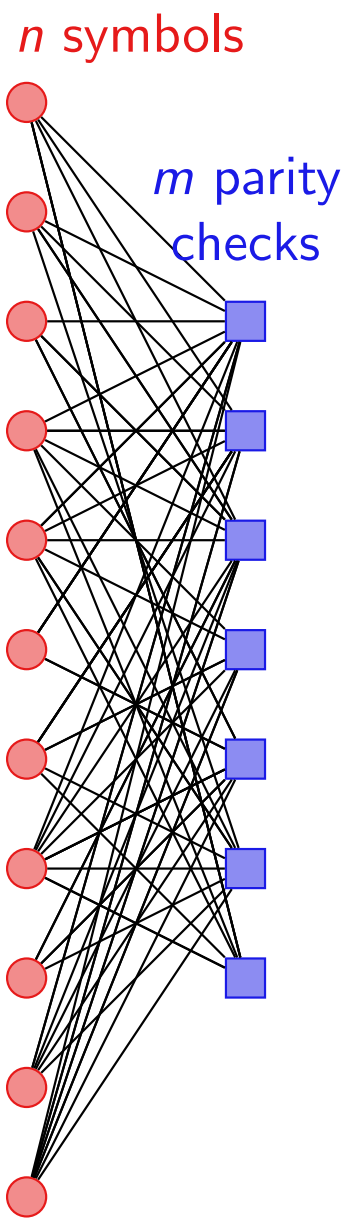


Standards

Today we'll look at 10GBase-T. Red curves: bit-flipping. Green curves: benchmark OMS.



Classifications: Message Passing



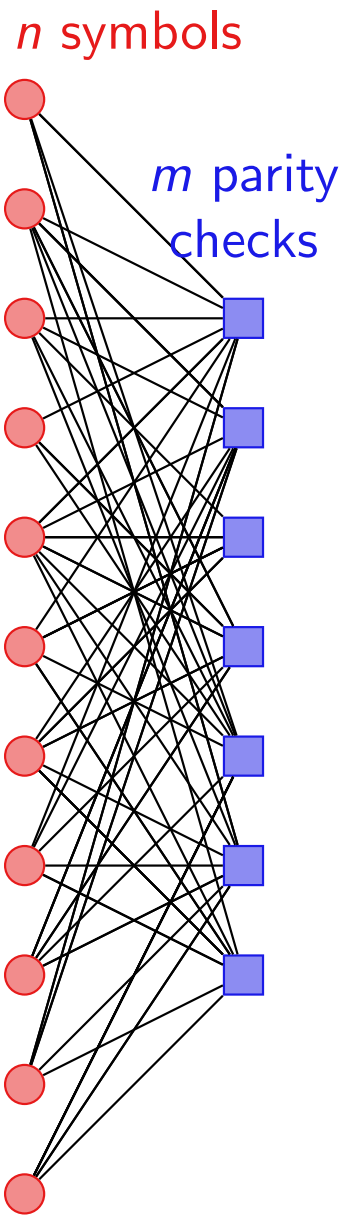
LDPC codes are usually modeled by a Tanner graph.

- Symbol node (code bits)
- Check nodes (parity constraints): adjacent bits must have even parity

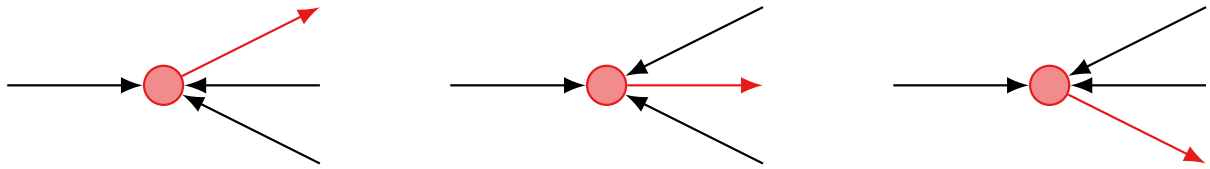
If there are a few errors, then at least one parity constraint is violated.

Objective: make minimum corrections to satisfy all parity checks.

Classifications: Message Passing



Message passing decoders compute extrinsic messages for each edge in the graph.

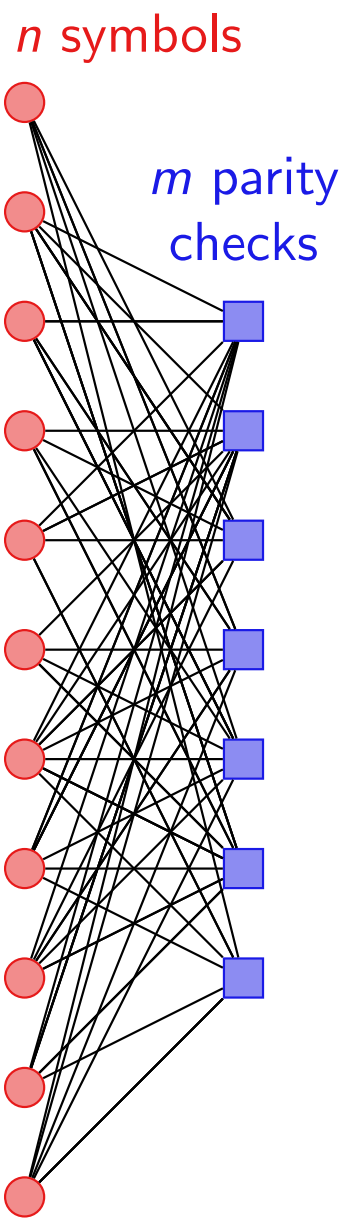


A extrinsic message on edge E is a function of received messages on adjacent edges excluding E.

Memoryless messages depend solely on the most recent received message values.

Memoryless extrinsic message-passing is most amenable to analysis.

Classifications: Memory



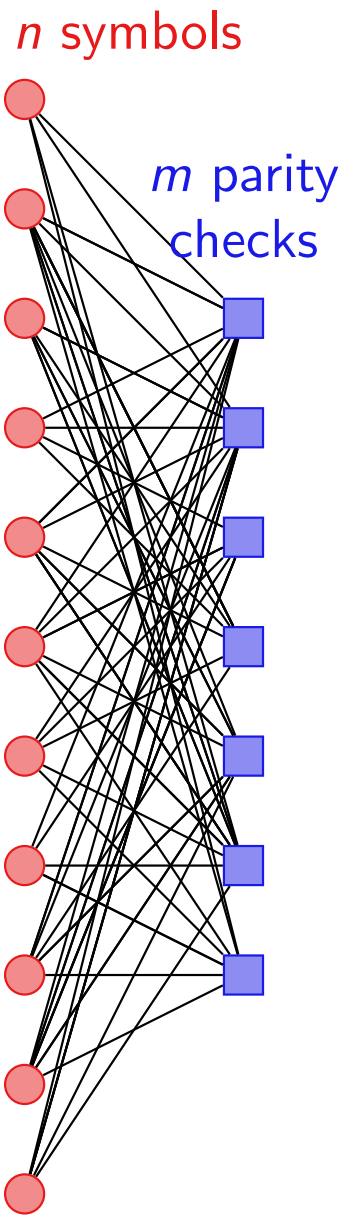
Memoryless messages depend solely on the most recent received message values.

Messages with memory depend also on some state variable in the local node, and that state variable is a function of previously received messages.

Memoryless extrinsic message-passing is most amenable to analysis.

Memory is very hard to analyze.

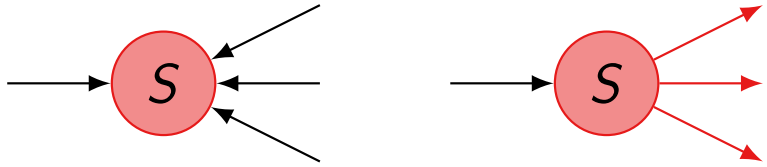
Classifications: Bit-Flipping



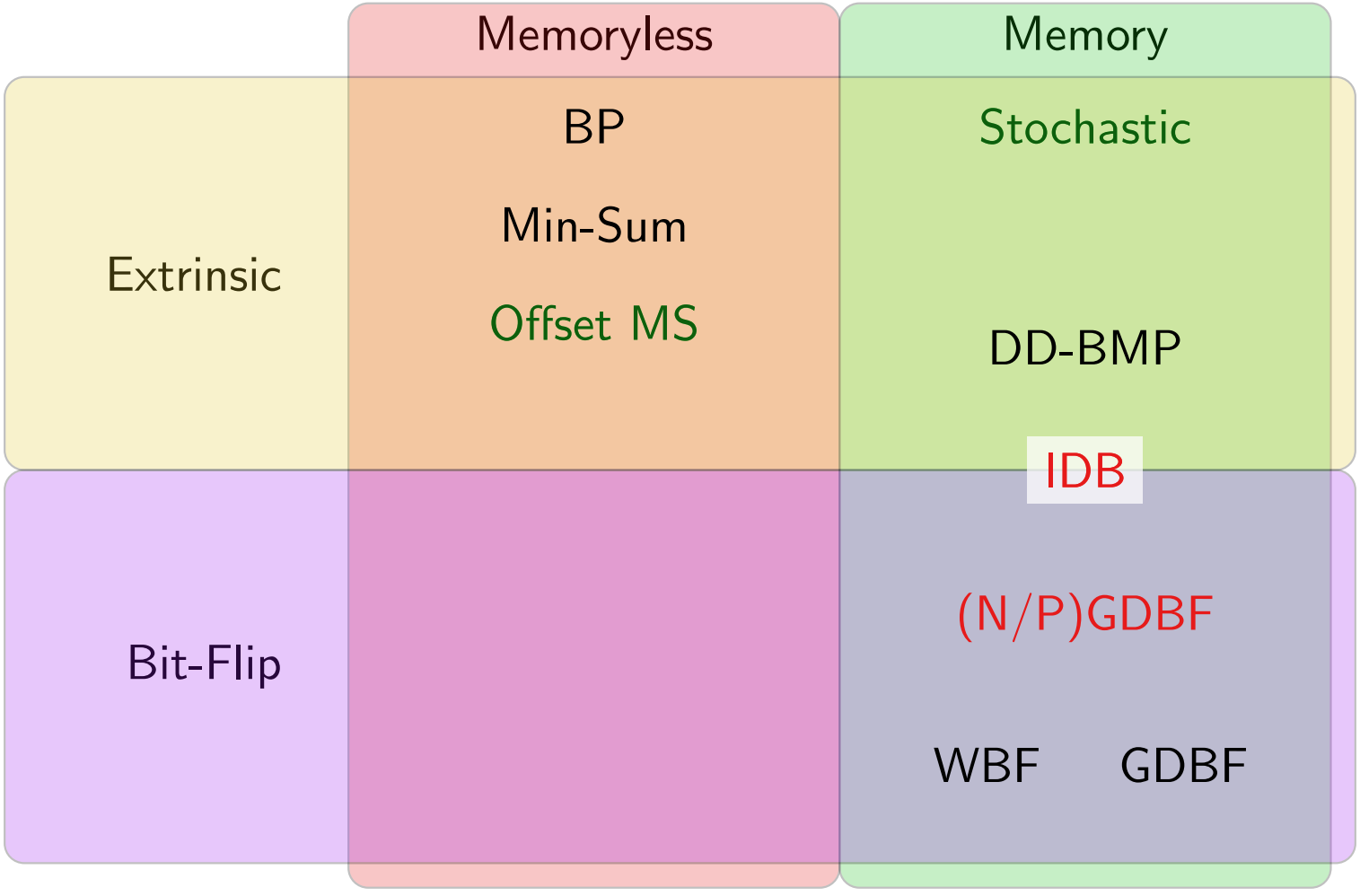
Bit-flipping algorithms are not extrinsic and have memory.

The node's state S is a function of all received messages.

The same message is transmitted on all edges.



Classifications

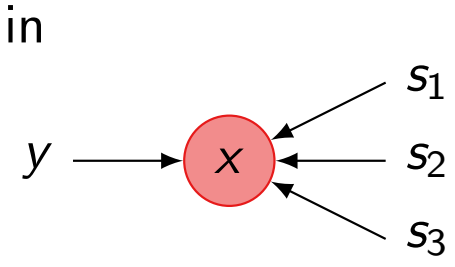


GDBF Algorithm (parallel flipping)

Parameter: global threshold θ.

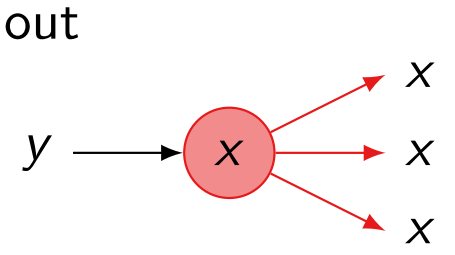
Inputs (AWGN channel)

channel sample	y_i.
hypothesis (memory state)	$x_i \in \{-1, +1\}$.
parity checks	$s_j \in \{-1, +1\}$



Operations

Flip function	$\Delta_i = x_i\, y_i + \sum_{j \in \mathcal{M}_i} s_j$
Threshold update:	flip x_i if $\Delta_i < \theta$



Then transmit x_i to adjacent parity nodes.

Noisy GDBF

Modifies the flip function:

$$\Delta_i = x_i\, y_i + w \sum_{j \in \mathcal{M}_i} s_j + q_i$$

Key changes:

- w weight parameter to optimize parity contribution
- q_i Gaussian noise perturbation, variance proportional to channel noise

Other heuristics have been studied, but are not used for 10GBase-T.

IDB

"Improved Differential Binary" is based on DD-BMP.

Inputs (AWGN channel)

Channel sample (y_i) and hypothesis (x_i) are same as GDBF.

parity checks $s_{i,j} \in \{-1, +1\}$

Parity messages are extrinsic in IDB.

Operations

State Memory $M_i^{(t+1)} = M_i^{(t)} + w \sum_{j \in \mathcal{M}_i \setminus j} s_{i,j} - d\, x_i^{(t)}$

Sign update: $x_i^{(t+1)} = \mathrm{sgn}_r \left(M_i^{(t)} \right)$

IDB Dynamics

IDB uses a degeneration parameter d to push the memory toward zero.

$$M_i^{(t+1)} = M_i^{(t)} + w \sum_{j \in \mathcal{M}_i \backslash j} s_{i,j} - d\, x_i^{(t)}$$

This causes oscillation when $\sum s_{i,j}$ is close to zero.

The oscillation is desirable; believed to help escape from trapping sets.

Noisy GDBF perturbations have the same interpretation: noise disrupts the stability of trapping sets.

Rewinding

Repeated decoding of failed frames was used in stochastic decoders and other algorithms.

The decoding trajectory is non-deterministic, so by starting over you could get a better answer.

Works especially well with NGDBF.

IDB Relaunching

IDB uses a deterministic rewinding scheme.

If a frame fails, it is restarted with modified channel samples:

$$M_i^{(0)} = \operatorname{sgn}_r(y_i) \cdot \max\left(\frac{1 - \operatorname{sgn}_r(y_i)}{2}, |y_i| - F(p,\, i)\right)$$

where p is the number of repetition attempts,
$F(p, i)$ is an empirically determined adjustment function

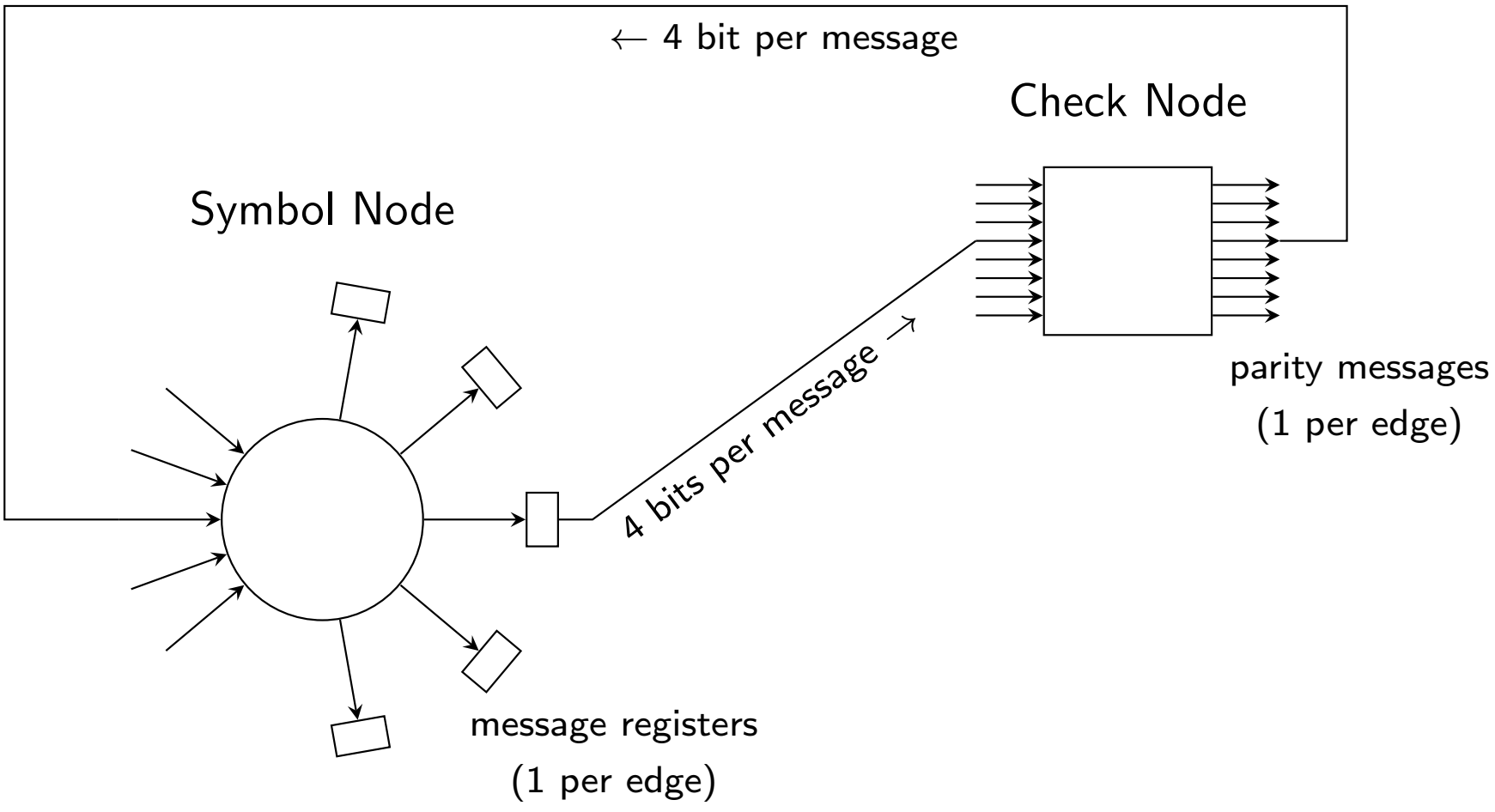
$$F(p,\, i) = \begin{cases} 1, & p < 2 \\ (p + i - 1) \mod 5 + 1, & \text{otherwise} \end{cases}$$

The adjustment introduces a periodic perturbation, intended to disrupt stronger trapping sets.

Architectures

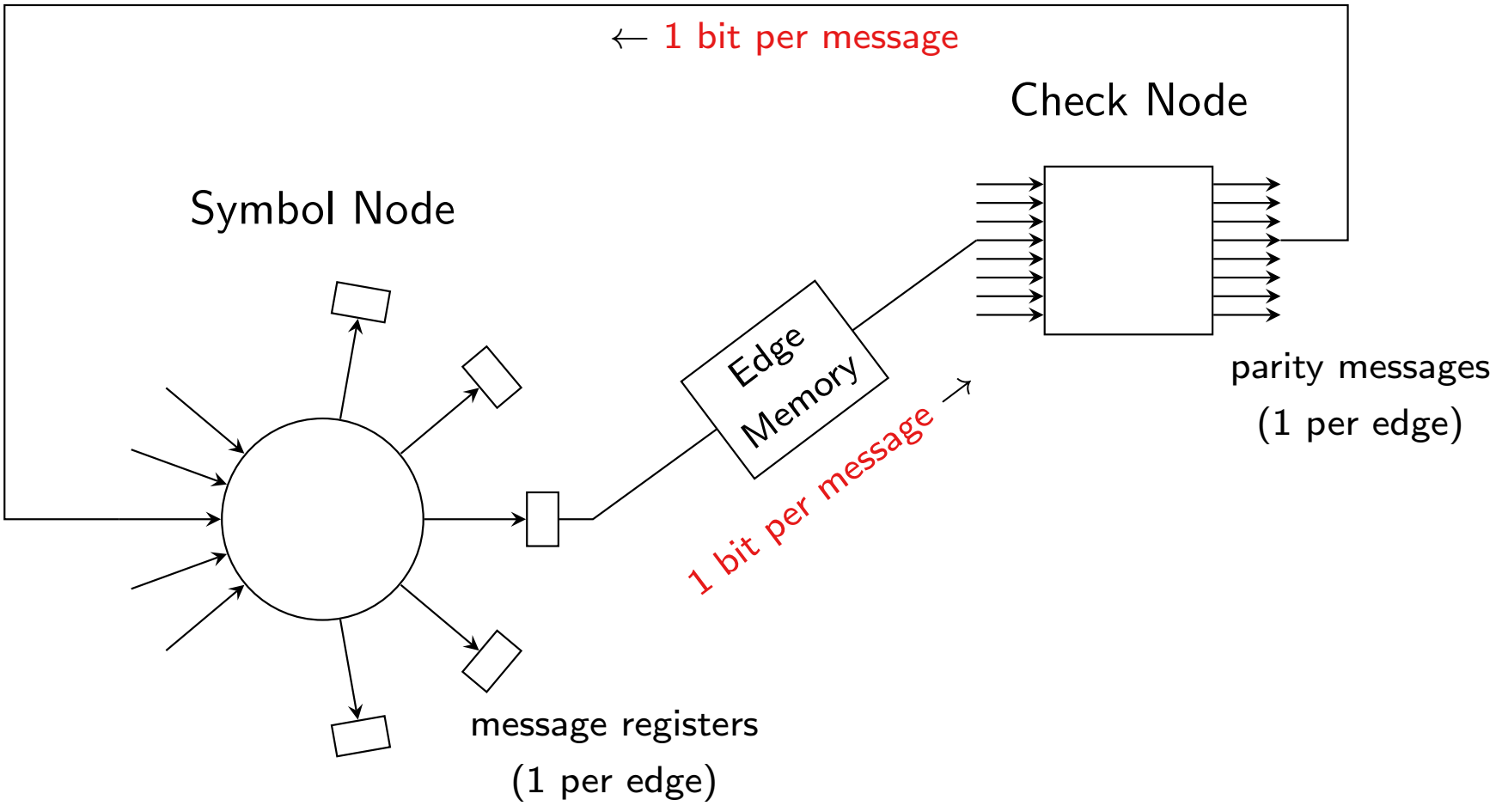
Datapath comparison

Standard message-passing algorithm (belief propagation):



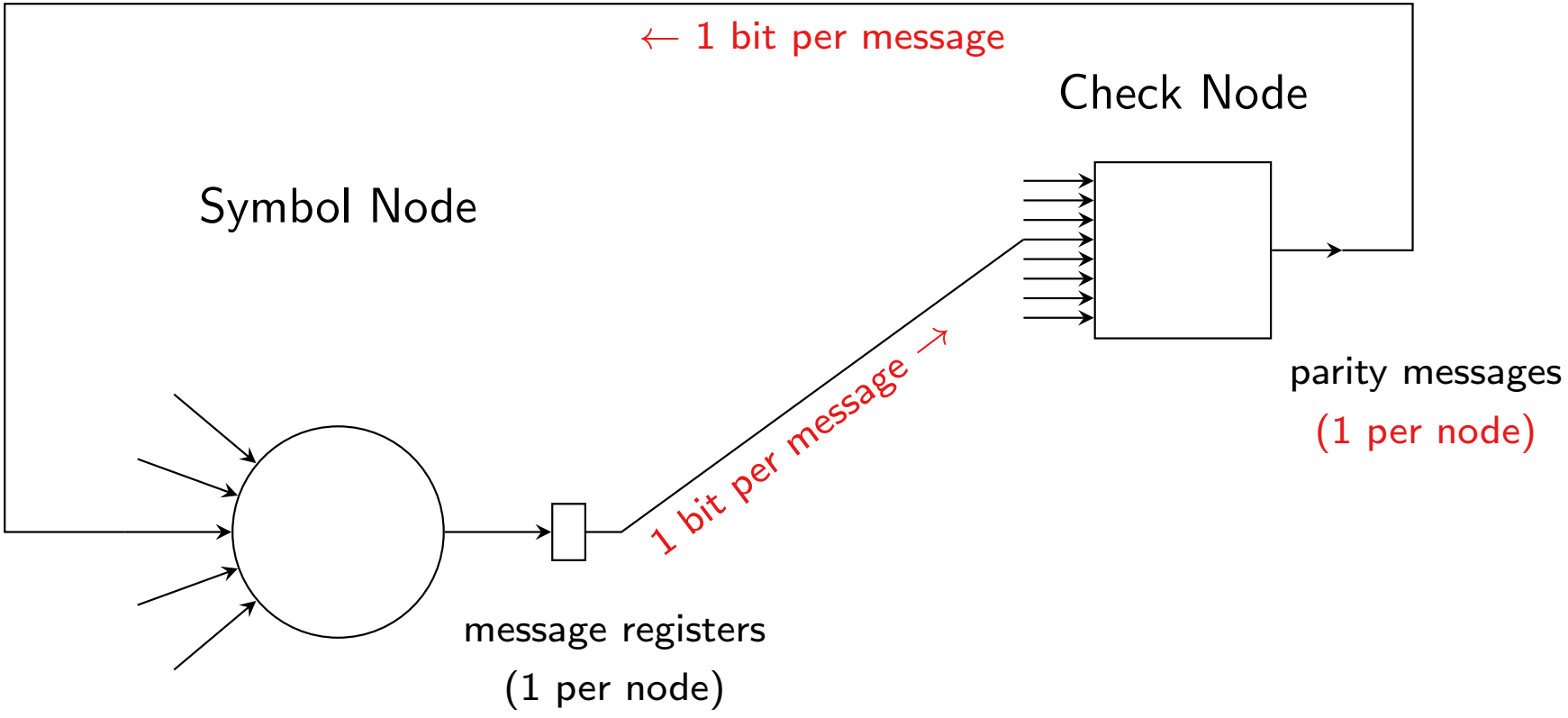
Datapath comparison

Stochastic algorithm (successive relaxation):

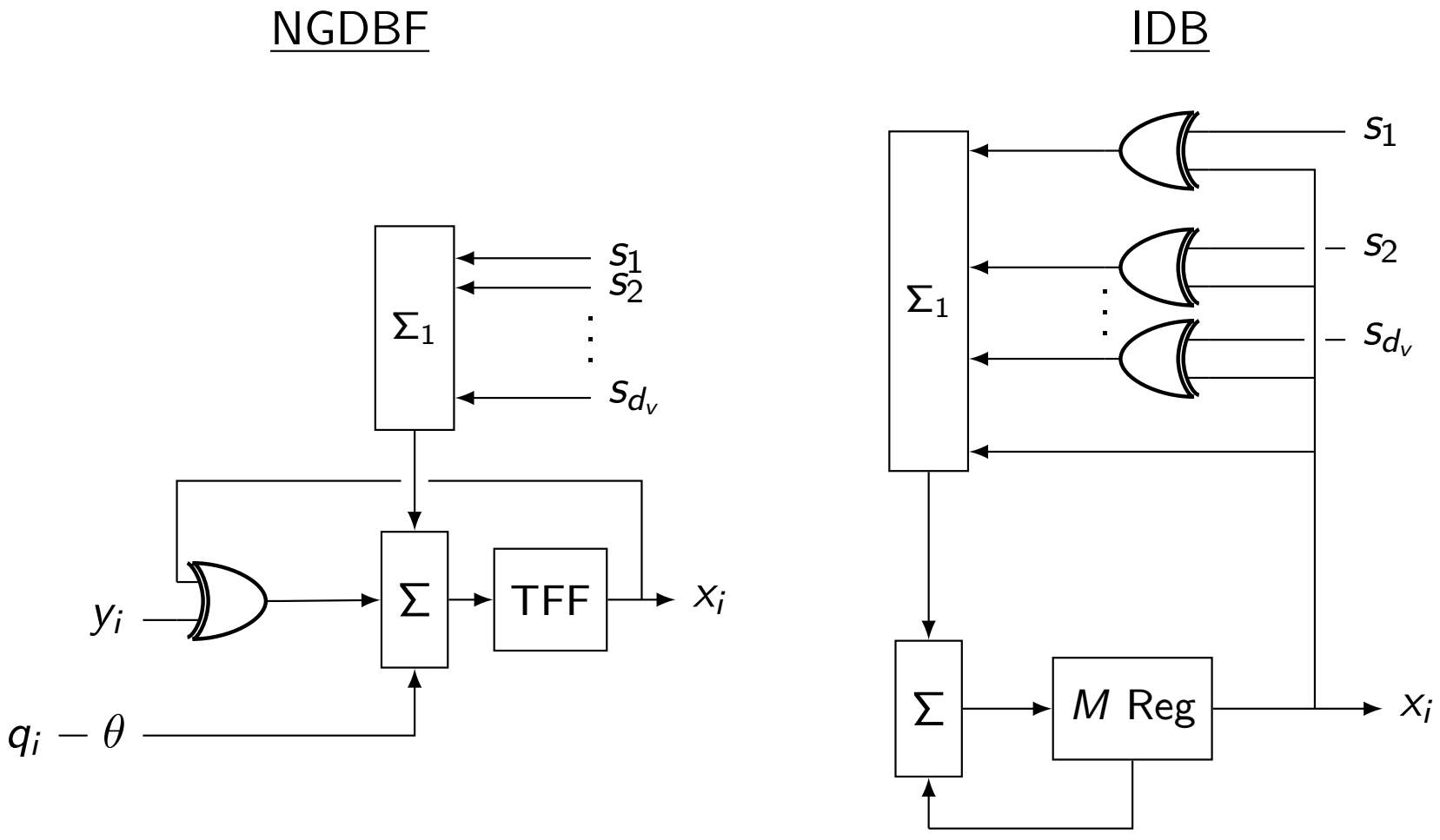


Datapath comparison

Bit-flipping algorithm:



Symbol Node Designs



Very similar structures; IDB has a larger register and more XOR gates.

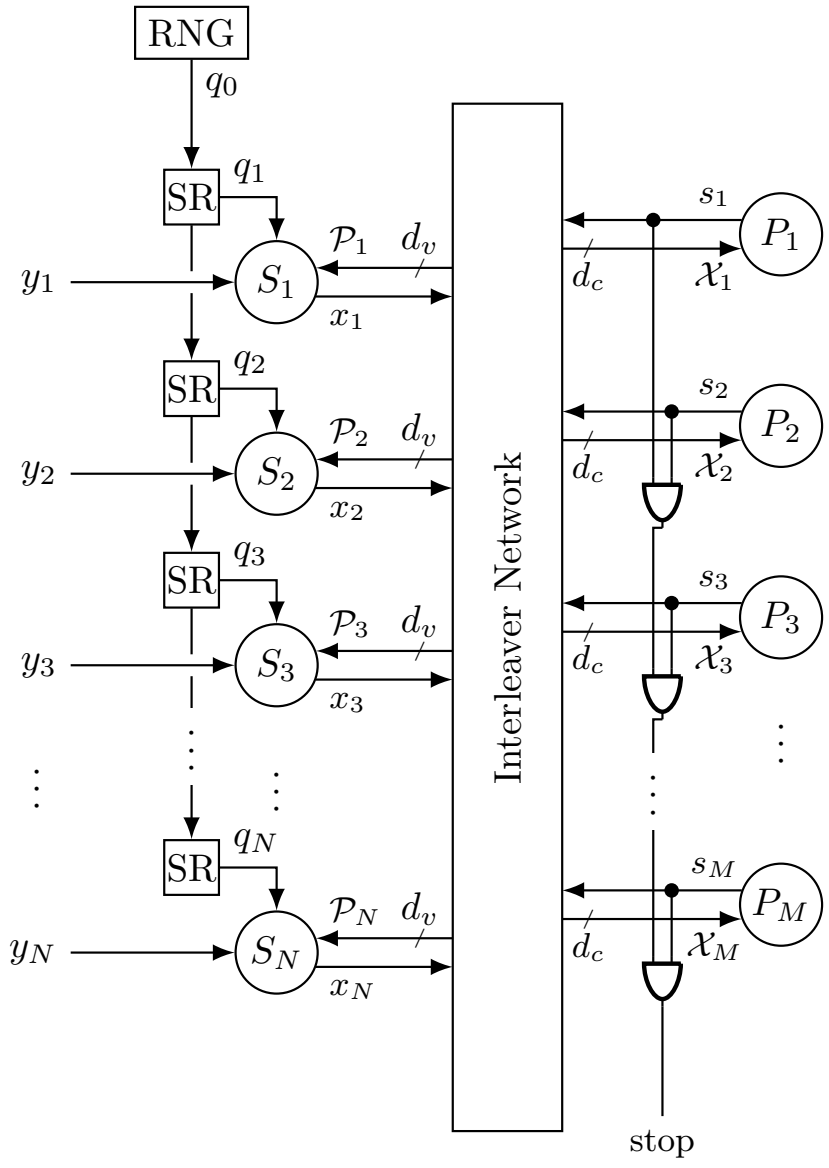
Top Architecture

IDB and NGDBF are nearly the same, except NGDBF needs noise generation.

The final design eliminates the RNG.

Noise samples are hard-coded, circulated in a shift-register.

Frame completion times contribute a randomized phase in the shift buffer.



Hardware Comparisons

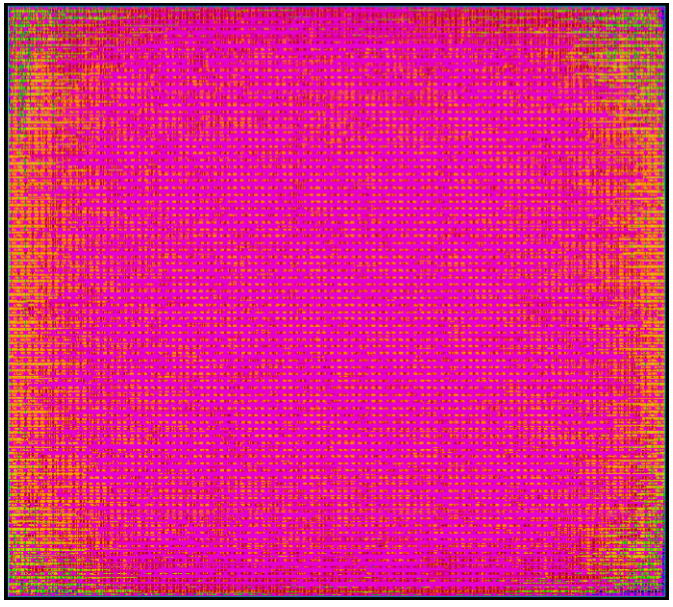
Application to the 802.3an 10GBaseT standard

This standard uses a rate 0.8143 $(2048, 1723)$ code with regular $(6, 32)$ degree distribution.

We implemented a demonstration NGDBF decoder in an ST Micro 65 nm technology.

Close comparisons are available in the literature for 65 nm decoders, which include

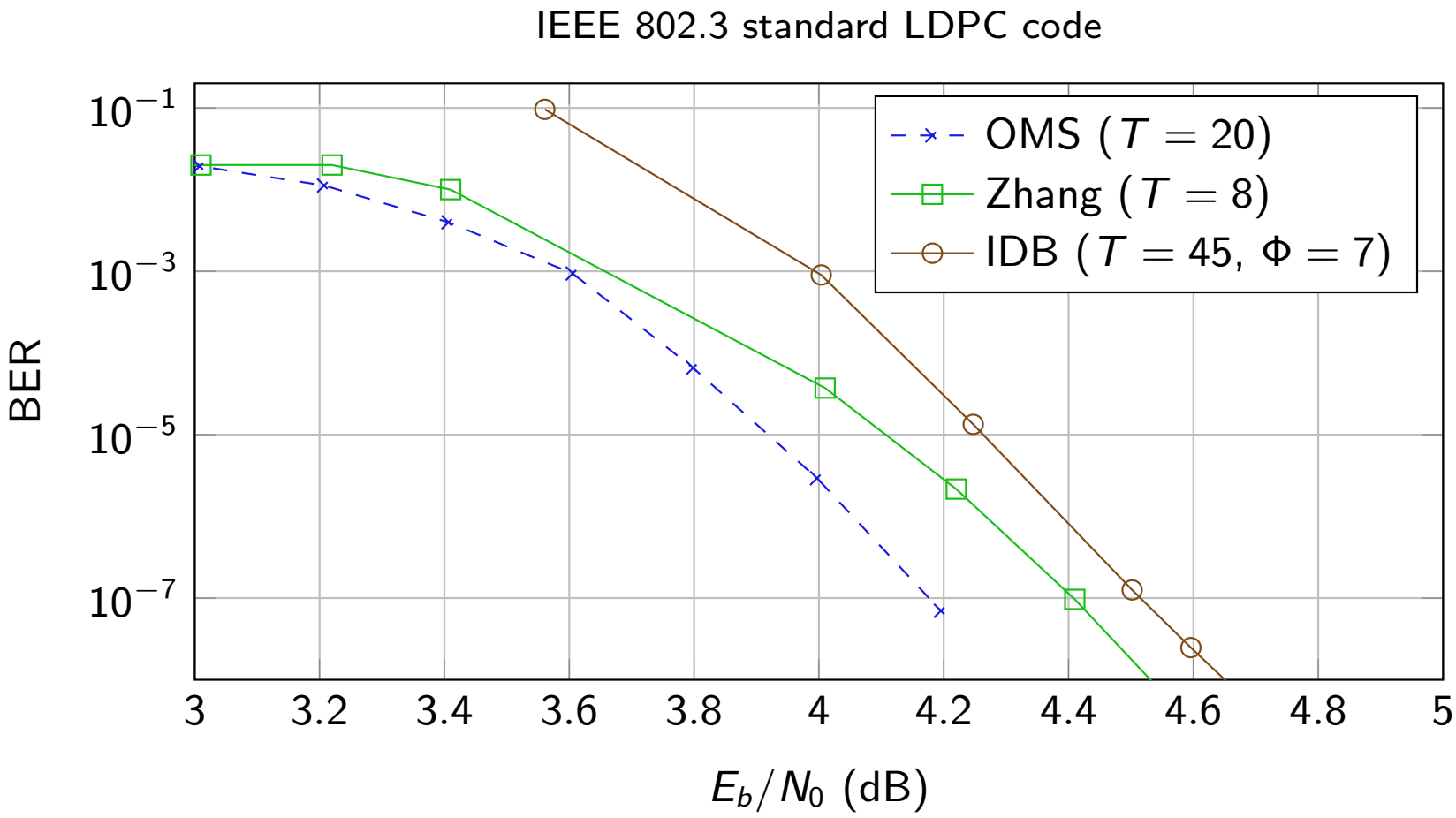
- Offset Min-Sum (OMS)[a]
- OMS Split-Row architecture[b]
- IDB[c]



Chip layout from Encounter

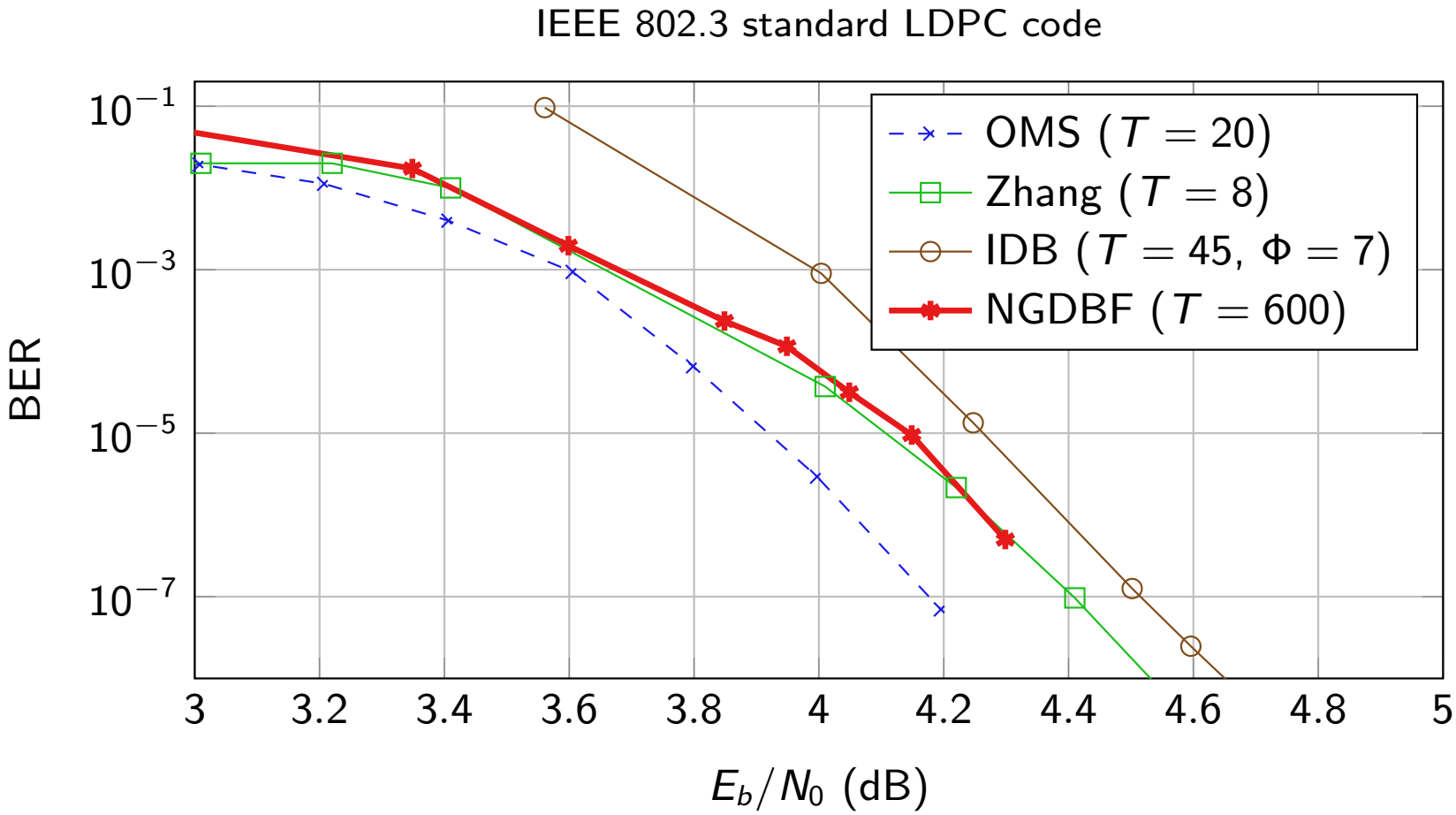
[a] Zhang et al. 2010.
[b] Mohsenin et al. 2010.
[c] Cushon et al. 2014.

Performance Comparison on 802.3



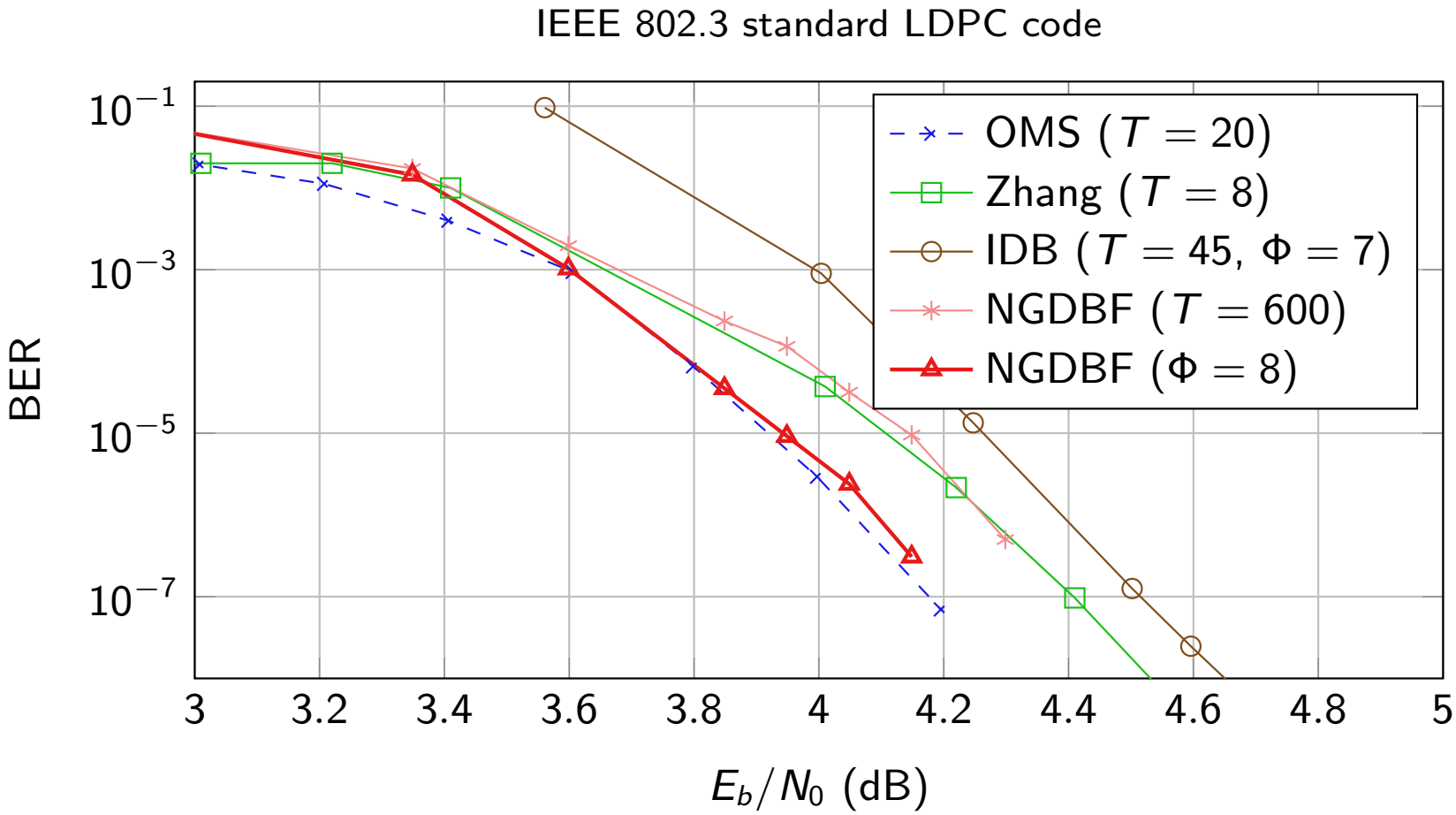
OMS ($T = 20$) represents the limit of performance.
The Zhang design uses $T = 8$ iterations to meet the throughput spec.
IDB comes relatively close to the OMS performance.

Performance Comparison on 802.3



NGDBF comes quite close to the Zhang benchmark.

Performance Comparison on 802.3



Repeated NGDBF with up to 8 attempts equals the limiting OMS performance.

ASIC Comparison for the 802.3 code

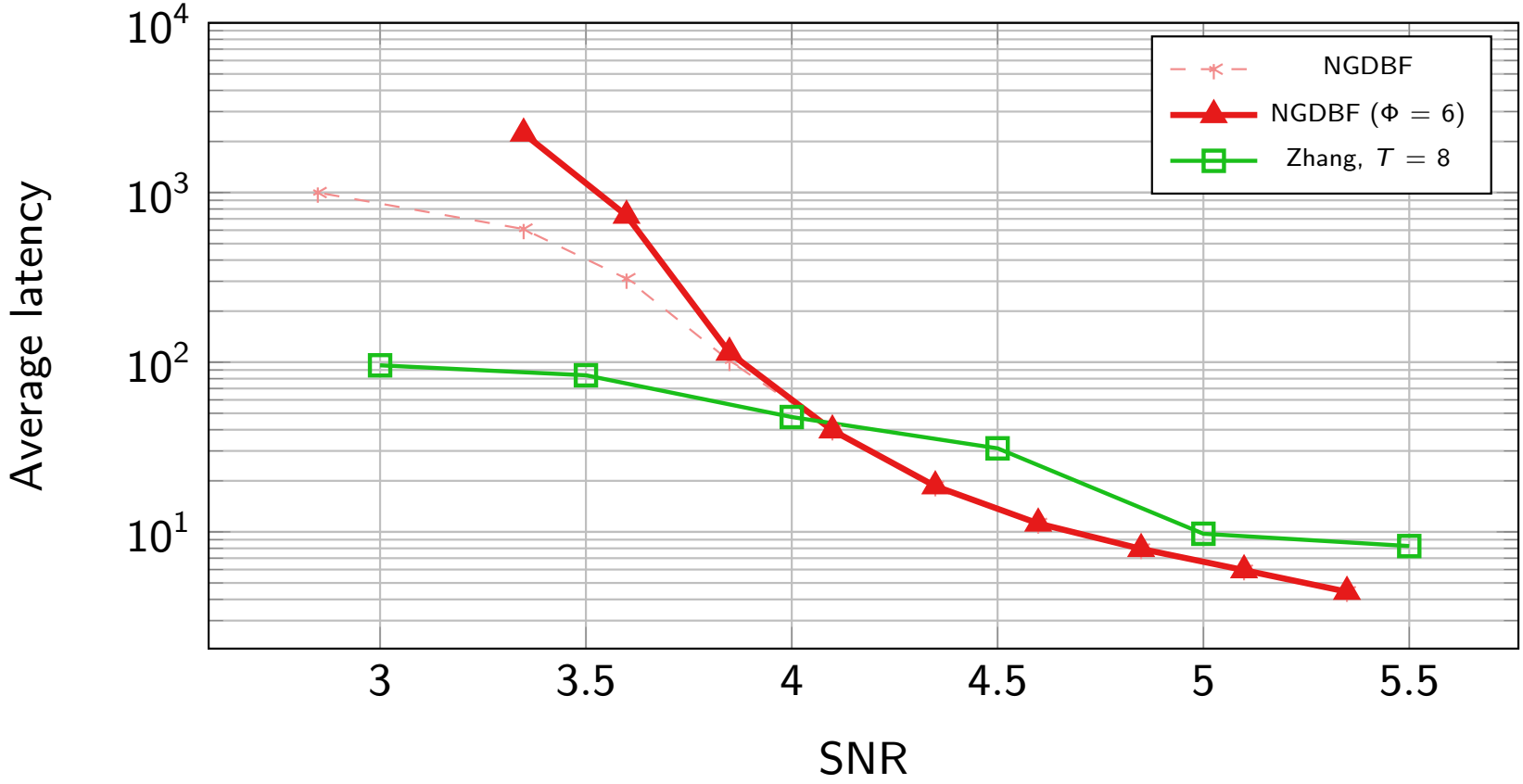
All designs are in 65 nm CMOS. These are **post-P&R** results:

Design:	NGDBF	IDB [4]	Split-Row MS[5]	OMS [6]
Quantization (bits)	7	6	5	4
Area (mm^2)	0.81	1.44	4.84	5.35
Clock (MHz)	188.67	520	195	700
E_b/N_0 at $\mathrm{BER} = 10^{-7}$	4.45	4.5	4.55 4.25	
At $\mathrm{SNR} = 4.55$ dB:				
Power (mW)	61.6	462	1359	-
Throughput (Gbps)	14.6	126.3	92.8	-
EpB (pJ/bit)	4.21	3.65	14.6	-
At $\mathrm{SNR} = 5.5$ dB:				
Power (mW)	63	478	-	2800
Throughput (Gbps)	36.4	171.8	-	47.7
EpB (pJ/bit)	1.73	2.78	-	58.7
Bit-flipping disadvantage:				
Worst-case Throughput (Gbps)	0.645	3.38	36.3	14.9

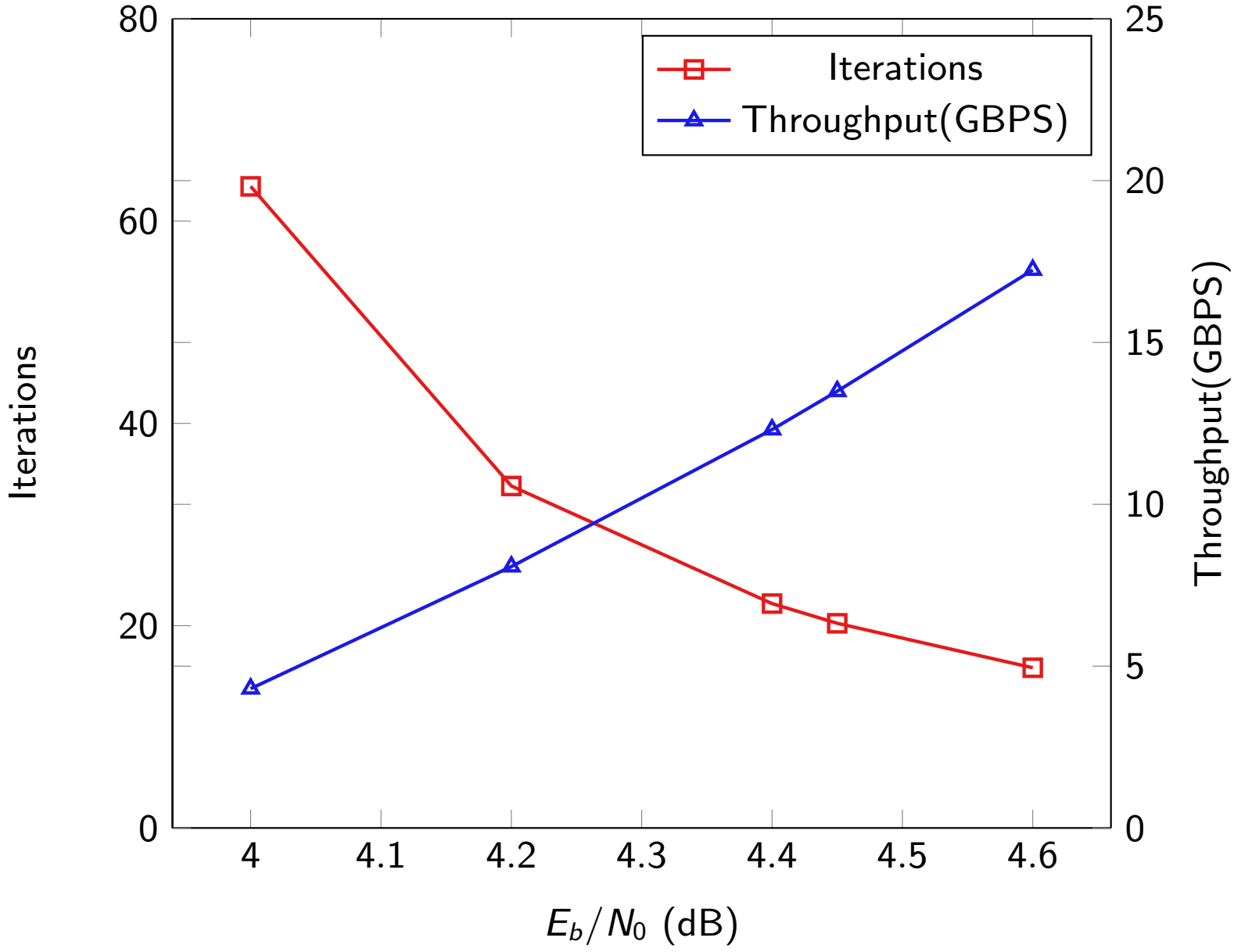
[4] Cushon et al. 2014
[5] Mohsenin et al. 2010
[6] Zhang et al. 2010

Average Latency

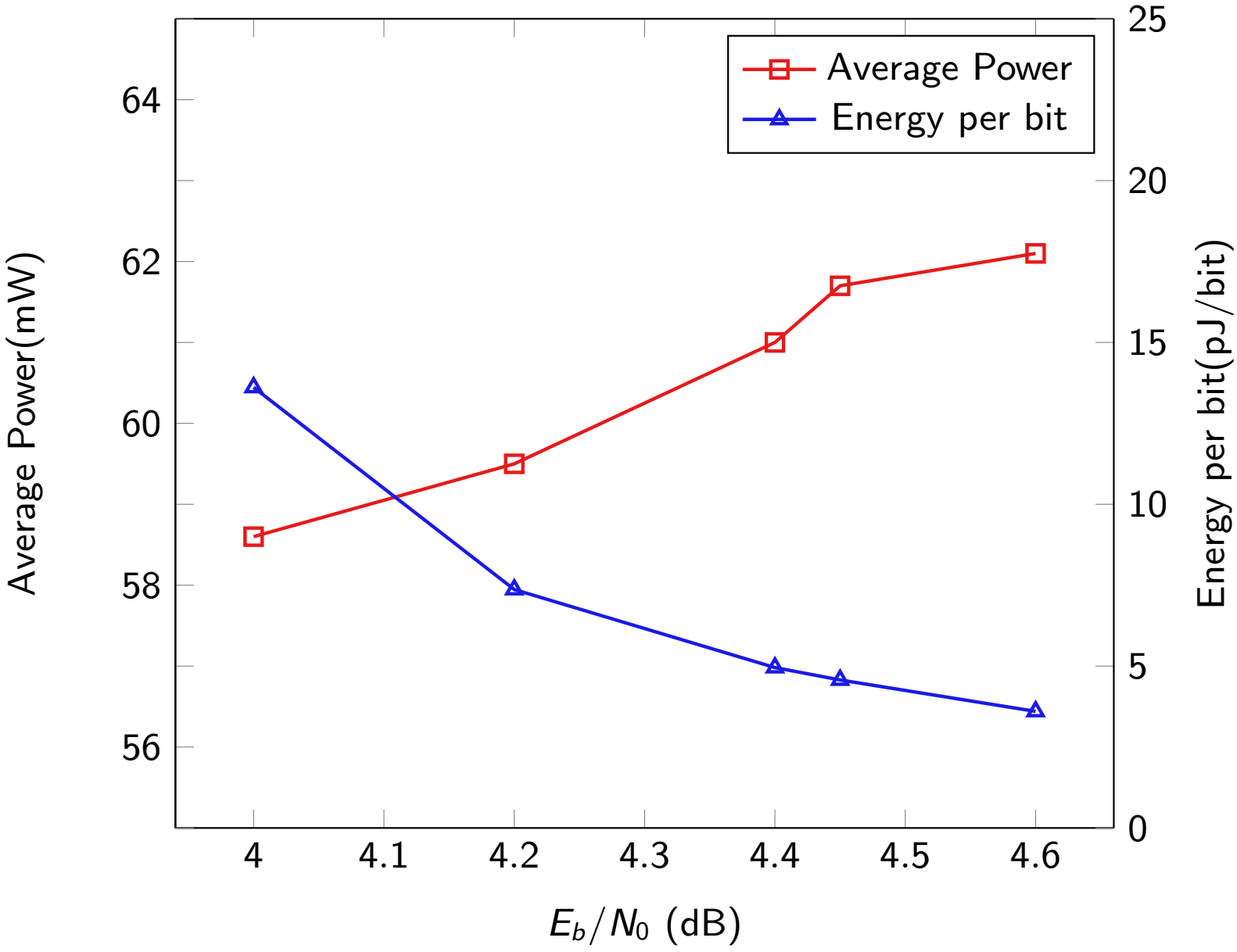
NGDBF uses a maximum of 600 clock cycles per decoding phase, with up to 8 phases. This is a large worst case latency, however the average latency is less than the Zhang benchmark at high SNR.



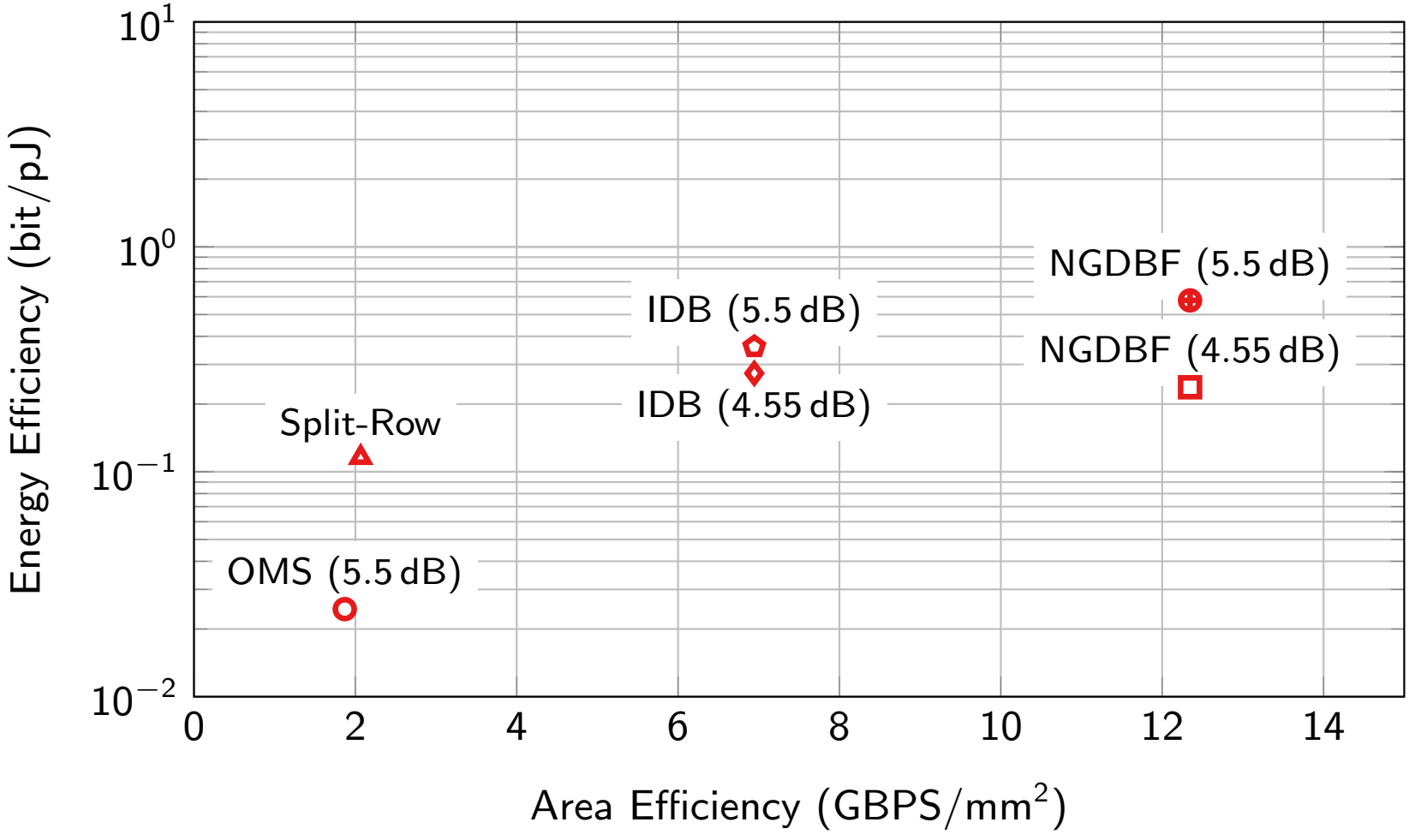
Throughput vs SNR



Energy Efficiency vs SNR

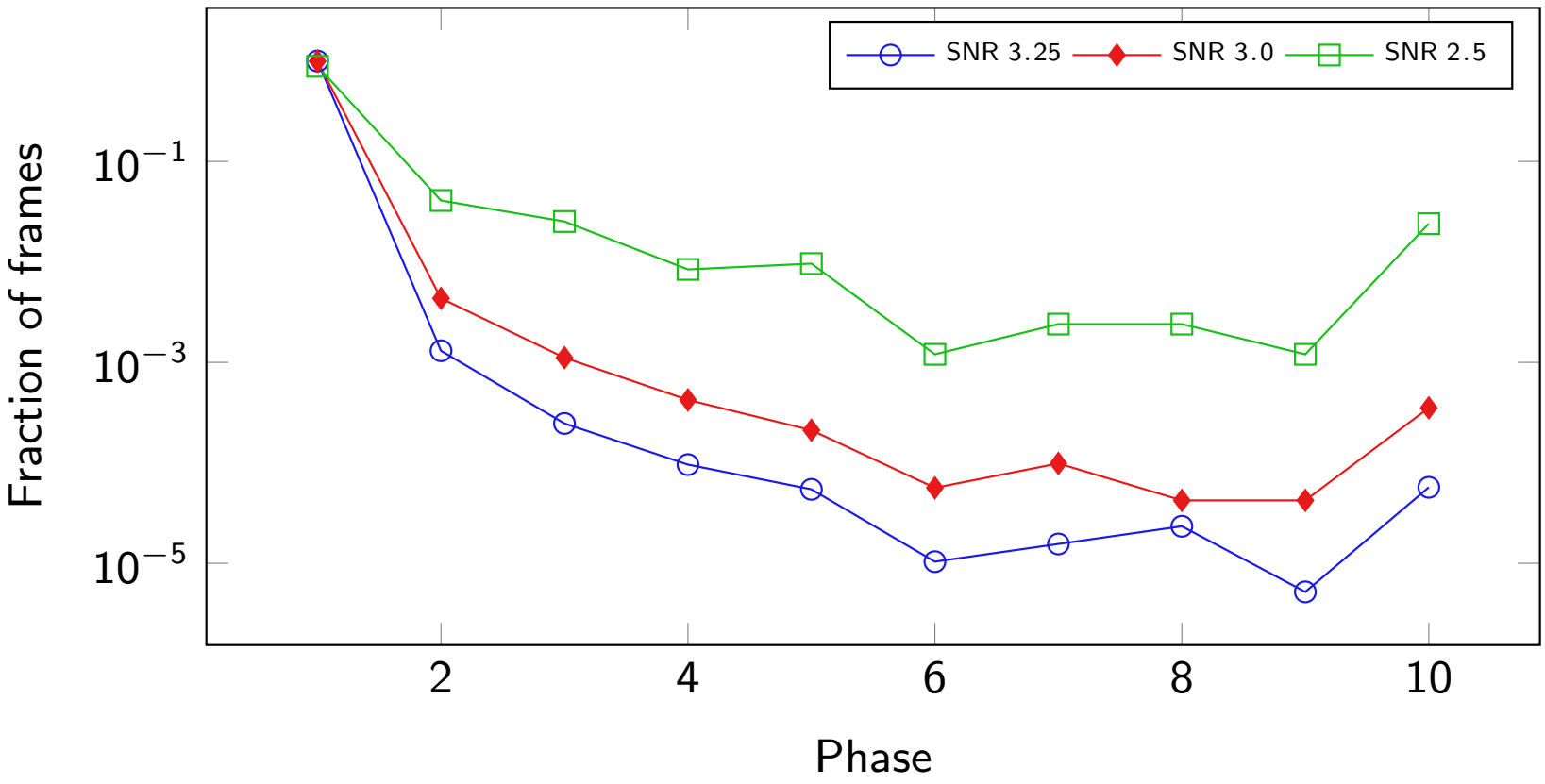


Energy Efficiency vs Area Efficiency



Re-decoding statistics

Re-decoding is needed for rare frames, but significantly improves performance.



Conclusions

IDB and NGDBF both rely on dynamic perturbations to disrupt local attractors (i.e. trapping sets).

These could be called "fiddle factors", but the benefits are hard to ignore.

For now, heuristic progress is more rapid than theoretical insight, but we're working to close that gap.

Acknowledgements

This research was supported by the US National Science Foundation under award ECCS-0954747, and by the Franco-American Fulbright Commission for the international exchange of scholars.

Thank you for listening!

Questions?

References I

- Sundararajan, Gopalakrishnan, Chris Winstead, and Emmanuel Boutillon (2014). "Noisy Gradient Descent Bit-Flip Decoding for LDPC Codes". In: *IEEE Transactions on Communications*.
- Rasheed, O.-A. et al. (2014). "Fault-Tolerant Probabilistic Gradient-Descent Bit Flipping Decoders". In: *IEEE Commun. Letters* 18.9, pp. 1487 –1490.
- Cushon, K. et al. (2014). "High-Throughput Energy-Efficient LDPC Decoders Using Differential Binary Message Passing". In: *Signal Processing, IEEE Transactions on* 62.3, pp. 619–631. ISSN: 1053-587X. DOI: 10.1109/TSP.2013.2293116.
- Tithi, Tasnuva, Chris Winstead, and Gopalakrishnan Sundararajan (2015). *Decoding LDPC codes via Noisy Gradient Descent Bit-Flipping with Re-Decoding*. arXiv:1503.08913. URL: http://arxiv.org/abs/1503.08913.
- Zhang, Zhengya et al. (2010). "An Efficient 10GBASE-T Ethernet LDPC Decoder Design With Low Error Floors". In: *IEEE J. Solid-State Circ.* 45, pp. 843–855. ISSN: 0018-9200. DOI: 10.1109/JSSC.2010.2042255.

References II

Mohsenin, T. et al. (2010). "A Low-Complexity Message-Passing Algorithm for Reduced Routing Congestion in LDPC Decoders". In: *IEEE Trans. Circ. Syst. I, Reg. Papers* 57, pp. 1048–1061. ISSN: 1549-8328. DOI: 10.1109/TCSI.2010.2046957.