Analysis and design of Min-Sum-based decoders running on noisy hardware

Christiane Kameni-Ngassa[*], Valentin Savin[*], David Declercq[#]

[*] CEA-LETI, MINATEC campus, Grenoble, France
[#] ETIS ENSEA, Université de Cergy-Pontoise, France

GdR-ISIS Workshop, Telecom ParisTech, July 2, 2013.
Utilisation de codes détecteurs et/ou correcteurs d'erreurs pour fiabiliser les traitements numériques au sein de circuits non fiables

Research reported in this presentation was supported by the Seventh Framework Programme of the European Union, under Grant Agreement number 309129 (**i-RISC** project).

Context & Objective

- Context
  - Next-generation electronic circuit design
    - increase in density integration
    - process variations, post CMOS technologies
    - lower power supply (reduction by 20% per technology node)
  - Low energy consumption (sustainability concerns)
    - aggressive voltage scaling

Next-generation chips will have to be built out from **unreliable components**

Reliability is among the ITRS Overall Design Technology Top-5 Challenges (2010)

- Objective
  - Design fault tolerant solutions for LDPC decoders operating on circuits built out from unreliable (faulty) components
  - Can **MP decoders** provide reliable error protection when they operate on faulty devices?

Min-Sum decoder on faulty devices

- Noisy components: new source of errors
  - Such errors may propagate through decoding iterations…
  - How does this impact on the error-correction capability of the decoder?
    - how to make sure that such an error propagation is not catastrophic?

- Theoretical analysis of “noisy” Min-Sum
  - Develop “noisy versions” of density-evolution
    - evaluate the theoretical performance loss due to noisy components
    - serve as guidelines for practical fault-tolerant implementations

- Practical fault-tolerant Min-Sum-based decoders
  - Evaluate the impact of faulty components on the performance of practical “finite-length” Min-Sum-based decoders

Min-Sum decoder

Initialization: $\forall n = 1, \dots, N;\ \forall m \in H(n)$

$$\gamma_n = \log(\Pr(x_n = 0 \mid y_n)/\Pr(x_n = 1 \mid y_n))$$
$$\alpha_{m,n} = \gamma_n$$

Iterations

- **CNU**: $\forall m = 1, \dots, M;\ \forall n \in H(m)$

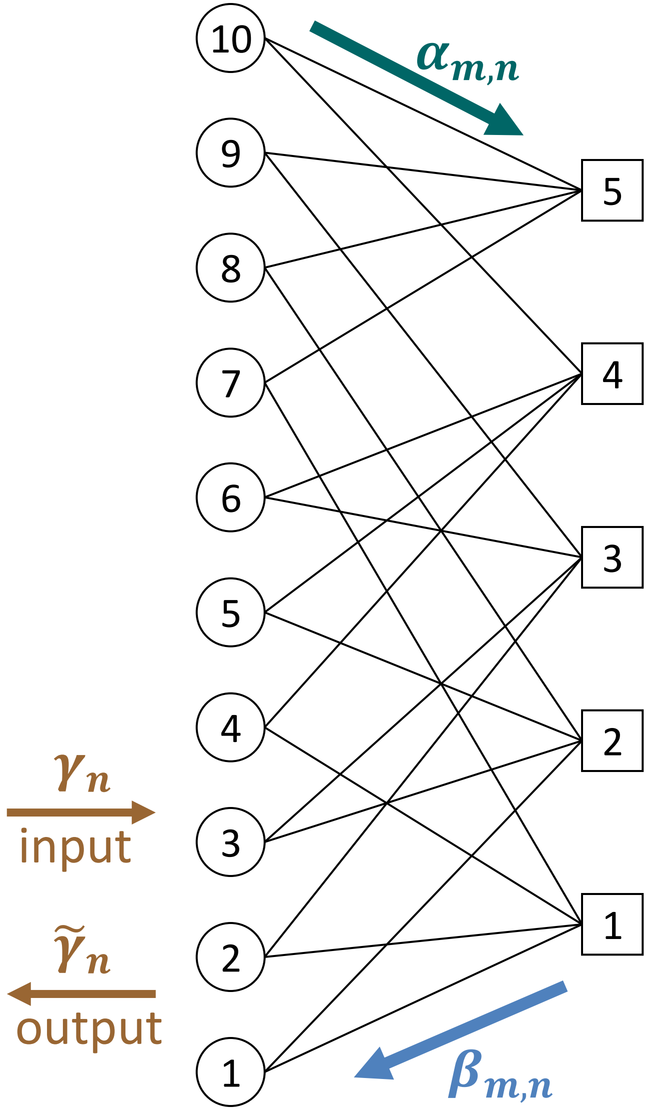
$$\beta_{m,n} = \left(\prod_{n' \in H(m) \setminus n} \mathrm{sgn}(\alpha_{m,n'})\right) \min_{n' \in H(m) \setminus n} (|\alpha_{m,n'}|)$$

- **VNU**: $\forall n = 1, \dots, N;\ \forall m \in H(n)$

$$\alpha_{m,n} = \gamma_n + \sum_{m' \in H(n) \setminus m} \beta_{m',n}$$

- **AP-LLR**: $\forall n = 1, \dots, N$

$$\tilde{\gamma}_n = \gamma_n + \sum_{m \in H(n)} \beta_{m,n}$$



Min-Sum decoder on faulty devices

Initialization: $\forall n = 1, \dots, N;\ \forall m \in H(n)$

$$\gamma_n = \log(\Pr(x_n = 0 \mid y_n)/\Pr(x_n = 1 \mid y_n))$$
$$\alpha_{m,n} = \gamma_n$$

Iterations

- **CNU**: $\forall m = 1, \dots, M;\ \forall n \in H(m)$

$$\beta_{m,n} = \left(\prod_{n' \in H(m) \setminus n} \operatorname{sgn}(\alpha_{m,n'}) \right) \min_{n' \in H(m) \setminus n} (|\alpha_{m,n'}|)$$

comparators

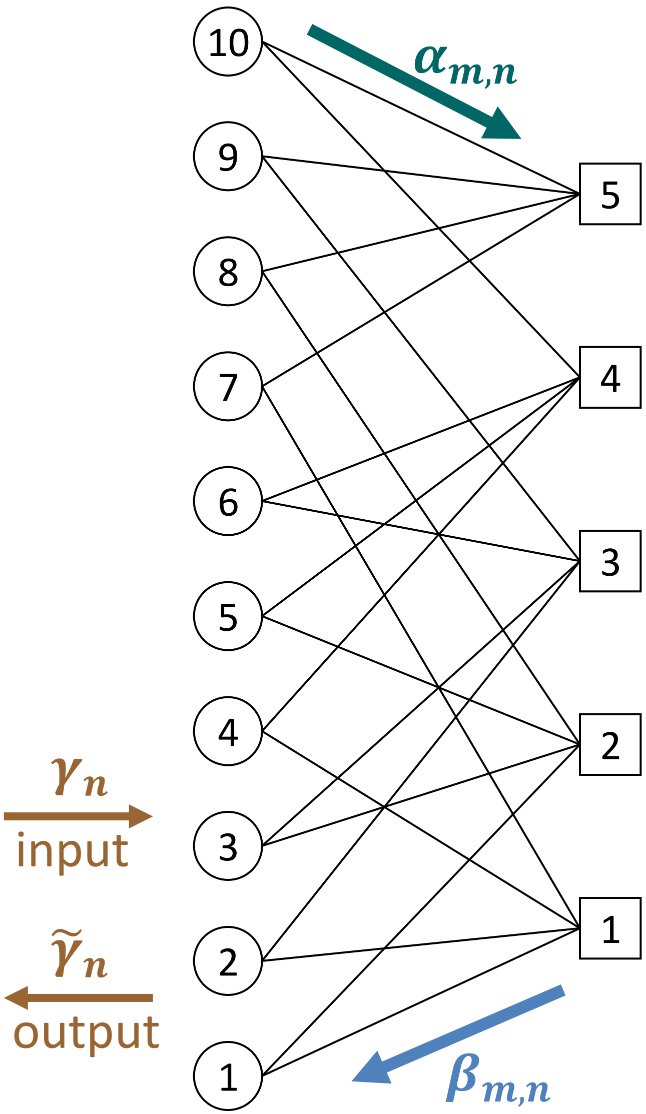
- **VNU**: $\forall n = 1, \dots, N;\ \forall m \in H(n)$

$$\alpha_{m,n} = \gamma_n + \sum_{m' \in H(n) \setminus m} \beta_{m',n}$$

- **AP-LLR**: $\forall n = 1, \dots, N$

$$\tilde{\gamma}_n = \gamma_n + \sum_{m \in H(n)} \beta_{m,n}$$

adders



Error models for faulty arithmetic units

- Probabilistic adder (Q bits)
  - Two parameters: the **depth D** and the **error probability P_a**
  - P_a is the probability that an error occurs on at least one of the D LSBs

	Q			D		2	1	
correct output	1	0	0	1	1	0	1	two's complement
error pattern				0	1	1	0	rand integer in $[1, 2^D-1]$
erroneous output	1	0	0	1	0	1	1	

- Probabilistic comparator
  - P_c is the probability that the output is in error

Part I:

Theoretical analysis of “noisy” Min-Sum decoder

Noisy density evolution

- Previous works
  - **Varshney-2011**
    - **concentration and convergence properties** were proved for the asymptotic performance of noisy message-passing decoders
    - density evolution equations were derived for the noisy **Gallager-A** decoder
  - **Tabatabaei-2013**
    - derived DE for noisy **Gallager-B** decoder defined over binary and non-binary alphabets
  - deal with very simple error models
    - emulate the noisy implementation of the decoder, by passing each of the exchanged messages through a binary (or non-binary) symmetric channel

Noisy density evolution

- We derived DE for fixed-point Min-Sum decoder
  - integrates above error models for arithmetic units (adder/comparator)

- Exchanged messages are random variables
  - Fixed-point implementation $\Rightarrow$ finite alphabet
  - C the PMF of input LLR values γ_n (depends only on the channel model)
  - $A^{(\ell)}, B^{(\ell)}$, and $\widetilde{C}^{(\ell)}$ the PMFs of $\alpha_{m,n}$, $\beta_{m,n}$, and $\widetilde{\gamma}_n$ at iteration ℓ

- **DE equations** (asymptotic performance)
  - Recursive formula (by tracking the update rules of exchanged messages):

$$\left(A^{(\ell+1)}, B^{(\ell+1)}, \widetilde{C}^{(\ell+1)}\right) = f\left(A^{(\ell)}, B^{(\ell)}, \widetilde{C}^{(\ell)}\right)$$

  - Under the assumption that incoming messages to any VNU and CNU are independent
  - In particular, the graph must be cycle-free

Noisy density evolution

- $P_\ell = \Pr(\tilde{\gamma}_n < 0)$ is the **error probability** at iteration ℓ
- $P_\infty = \lim_{\ell \to \infty} P_\ell$ – **output error probability** (does not always exist!)

- **Useful decoder:** P_∞ exits and $P_\infty < P_0$
- **η-threshold:** $P_{\mathrm{th}}(\eta) = \sup\{P_0 \mid P_\infty \text{ exists and } P_\infty < \eta\}$

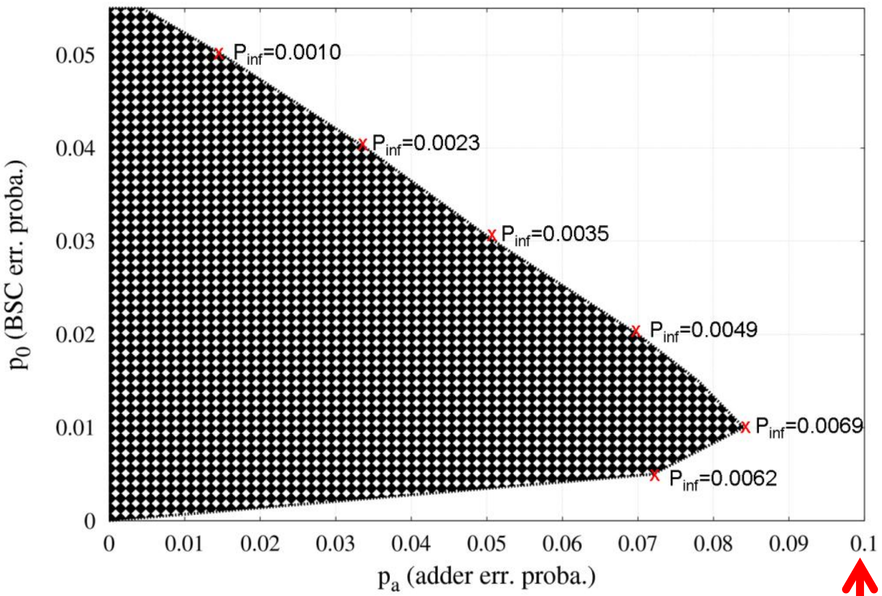
- **DE equations** (asymptotic performance)
  - Recursive formula (by tracking the update rules of exchanged messages):

$$\left(A^{(\ell+1)}, B^{(\ell+1)}, \tilde{C}^{(\ell+1)}\right) = f\left(A^{(\ell)}, B^{(\ell)}, \tilde{C}^{(\ell)}\right)$$

  - Under the assumption that incoming messages to any VNU and CNU are independent
  - In particular, the graph must be cycle-free

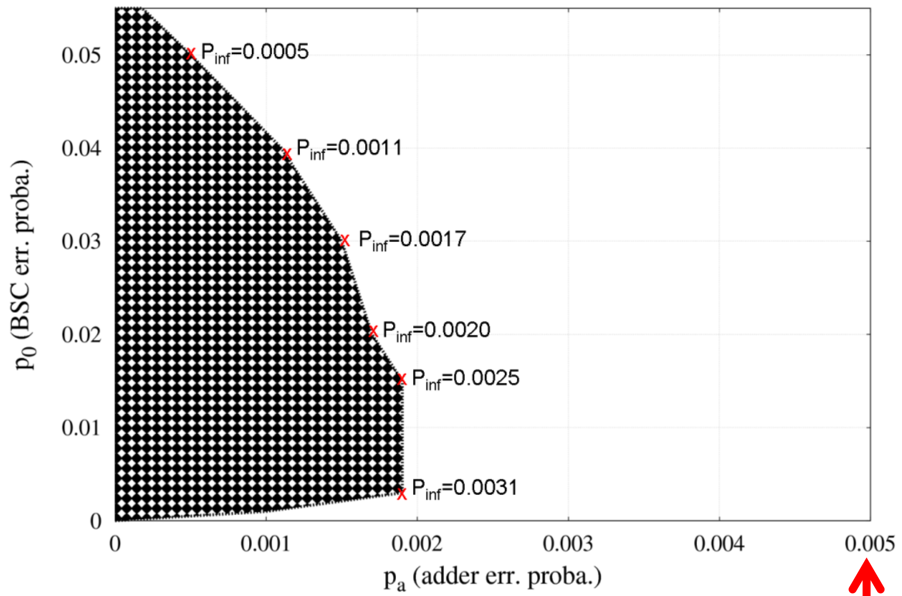
Useful regions for Min-Sum decoder / BSC

- (3, 6)-regular LDPC codes, fixed-point MS
  - Q = 5 bits (number of bits of the adder)
  - P_c = 0.001 (error probability of the comparator)



Depth D = 4

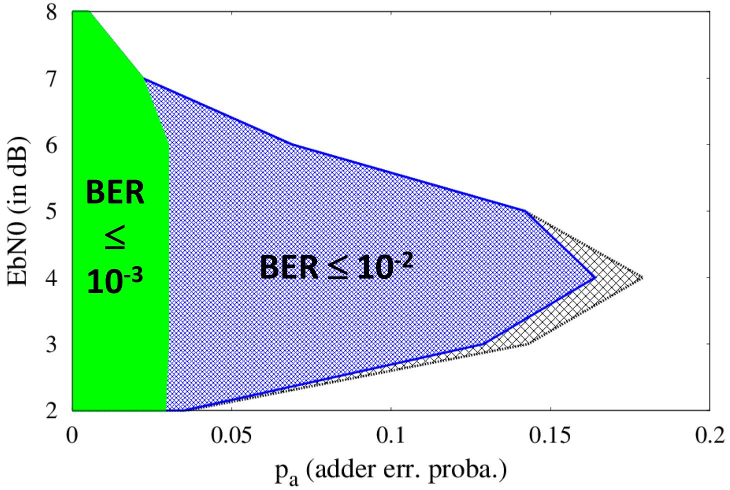
10 %



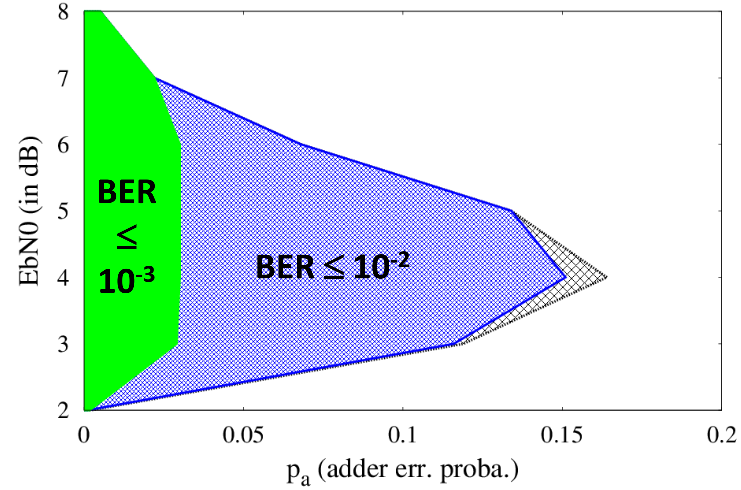
Depth D = 5

0.5 %

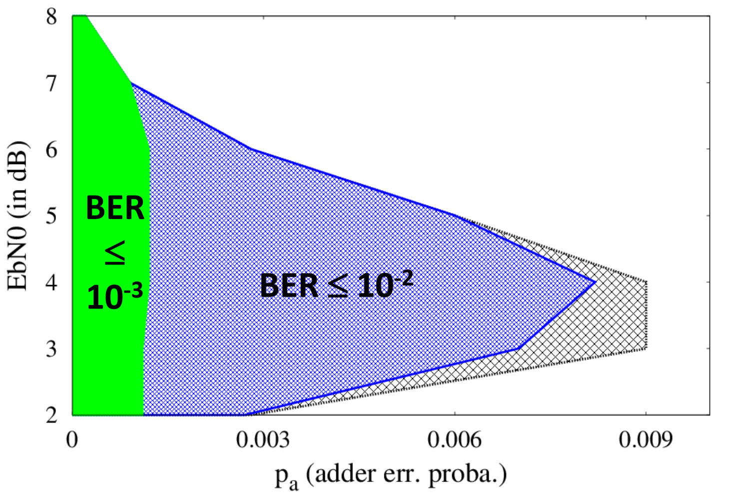
Useful regions for Min-Sum decoder / BI-AWGN



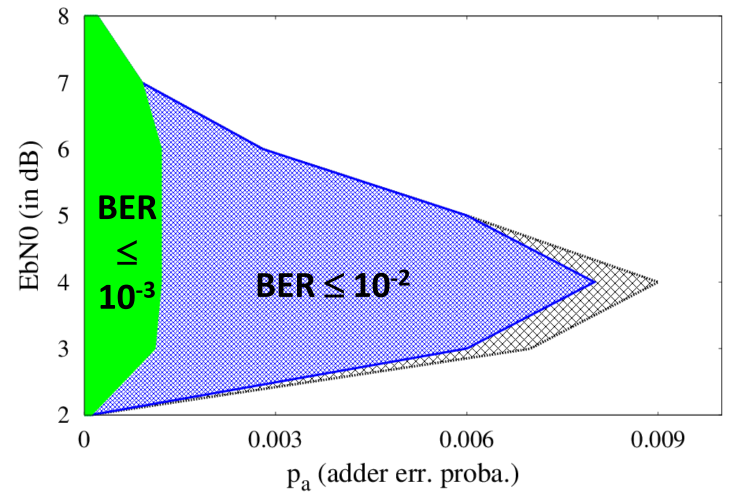
***D* = 4, P_c = 0.001**



***D* = 4, P_c = 0.01**



***D* = 5, P_c = 0.001**



***D* = 5, P_c = 0.001**

First conclusion…

- Errors caused by noisy components do not necessarily propagate catastrophically through decoding iterations
  - Min-Sum decoder can still provide error protection with a given level of reliability, assuming that decoder's components are reasonably noisy…

- Some characteristics of the Min-Sum decoder
  - Less sensitive to **errors in comparators**
  - Less sensitive to **errors in the LSBs** of the adder
  - Highly sensitive to **errors in the sign bit** of the adder

Part II:

Practical fault-tolerant Min-Sum-based decoders

Practical implementation of Min-Sum decoder

(1)

Initialization: $\forall n = 1, \dots, N;\ \forall m \in H(n)$

$$\gamma_n = \log(\Pr(x_n = 0 \mid y_n)/\Pr(x_n = 1 \mid y_n))$$
$$\alpha_{m,n} = \gamma_n$$

Iterations

- **CNU**: $\forall m = 1, \dots, M;\ \forall n \in H(m)$

$$\beta_{m,n} = \left(\prod_{n' \in H(m)\setminus n} \operatorname{sgn}(\alpha_{m,n'})\right) \min_{n' \in H(m)\setminus n}(|\alpha_{m,n'}|)$$

- **VNU**: $\forall n = 1, \dots, N;\ \forall m \in H(n)$

$$\alpha_{m,n} = \gamma_n + \sum_{m' \in H(n)\setminus m} \beta_{m',n}$$

- **AP-LLR**: $\forall n = 1, \dots, N$

$$\tilde{\gamma}_n = \gamma_n + \sum_{m \in H(n)} \beta_{m,n}$$

(2)

Initialization: $\forall n = 1, \dots, N;\ \forall m \in H(n)$

$$\gamma_n = \log(\Pr(x_n = 0 \mid y_n)/\Pr(x_n = 1 \mid y_n))$$
$$\alpha_{m,n} = \gamma_n$$

Iterations

- **CNU**: $\forall m = 1, \dots, M;\ \forall n \in H(m)$

$$\beta_{m,n} = \left(\prod_{n' \in H(m)\setminus n} \operatorname{sgn}(\alpha_{m,n'})\right) \min_{n' \in H(m)\setminus n}(|\alpha_{m,n'}|)$$

- **AP-LLR**: $\forall n = 1, \dots, N$

$$\tilde{\gamma}_n = \gamma_n + \sum_{m \in H(n)} \beta_{m,n}$$

- **VNU**: $\forall n = 1, \dots, N;\ \forall m \in H(n)$

$$\alpha_{m,n} = \tilde{\gamma}_n - \beta_{m,n}$$

Remark: MS(1) and MS(2) are equivalent if exact (noiseless) arithmetic

MS(1) and MS(2) are **NOT equivalent** if probabilistic (noisy) arithmetic

Practical implementation of Min-Sum decoder

Iterations

- **CNU**: $\forall m = 1, \dots, M;\ \forall n \in H(m)$

$$\beta_{m,n} = \left(\prod_{n' \in H(m) \backslash n} \operatorname{sgn}(\alpha_{m,n'}) \right) \min_{n' \in H(m) \backslash n} (|\alpha_{m,n'}|)$$

- **VNU**: $\forall n = 1, \dots, N;\ \forall m \in H(n)$

$$\alpha_{m,n} = \gamma_n + \sum_{m' \in H(n) \backslash m} \beta_{m',n}$$

- **AP-LLR**: $\forall n = 1, \dots, N$

$$\tilde{\gamma}_n = \gamma_n + \sum_{m \in H(n)} \beta_{m,n}$$

(1)

Iterations

- **CNU**: $\forall m = 1, \dots, M;\ \forall n \in H(m)$

$$\beta_{m,n} = \left(\prod_{n' \in H(m) \backslash n} \operatorname{sgn}(\alpha_{m,n'}) \right) \min_{n' \in H(m) \backslash n} (|\alpha_{m,n'}|)$$

- **AP-LLR**: $\forall n = 1, \dots, N$

$$\tilde{\gamma}_n = \gamma_n + \sum_{m \in H(n)} \beta_{m,n}$$

- **VNU**: $\forall n = 1, \dots, N;\ \forall m \in H(n)$

$$\alpha_{m,n} = \tilde{\gamma}_n - \beta_{m,n}$$

(2)

The computation of $\alpha_{m,n}$ takes $d_n - 1$ additions

(d_n denotes the degree of variable-node n)

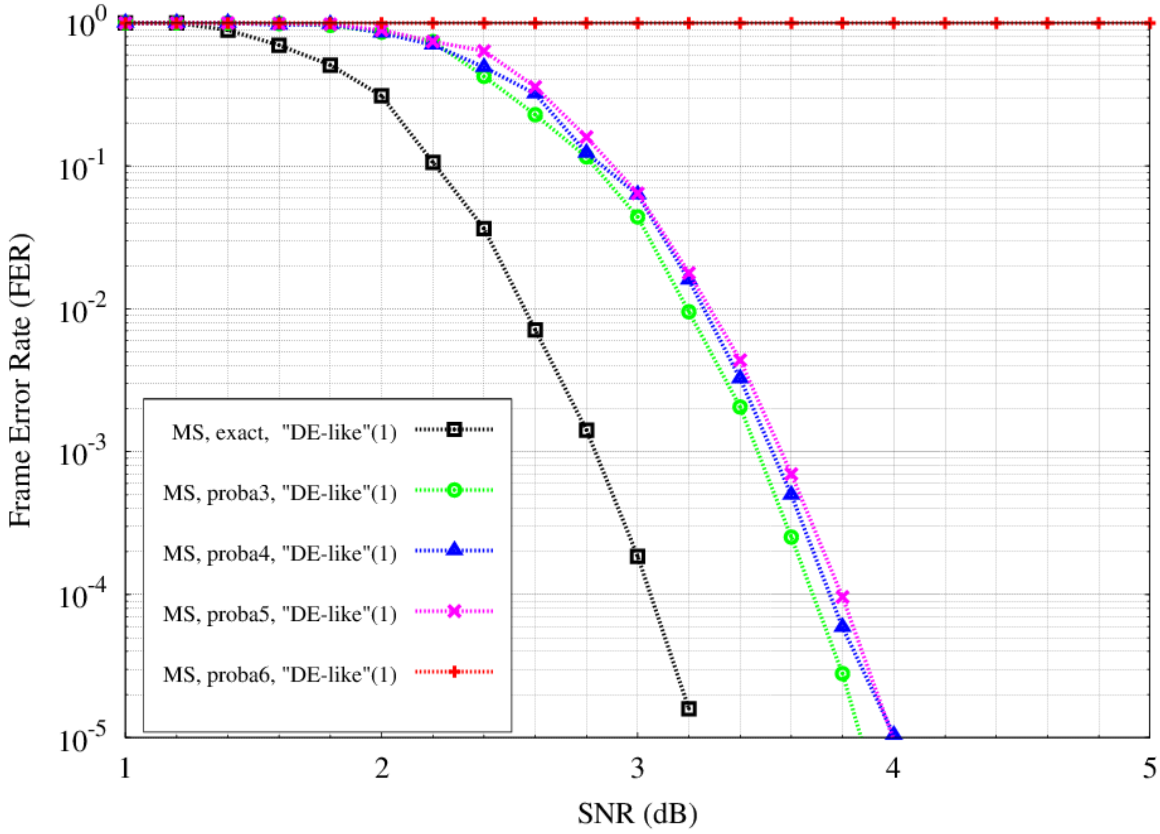
The computation of $\alpha_{m,n}$ takes $d_n + 1$ additions

$\Rightarrow$ d_n additions to compute $\tilde{\gamma}_n$, and 1 more addition to derive the $\alpha_{m,n}$ value

$\Rightarrow$ **An increased number of additions results in an increased error probability of $\alpha_{m,n}$**

Practical implementation of Min-Sum decoder

- Mackay's regular (3,6)-LDPC code, [K = 504, N = 1008]
- Fixed-point MS decoder: 5 / 6 bits (exchanged messages / AP-LLR)



Comp.err. prob: $P_c = 0.01$
Adder err. prrob: $P_a = 0.01$

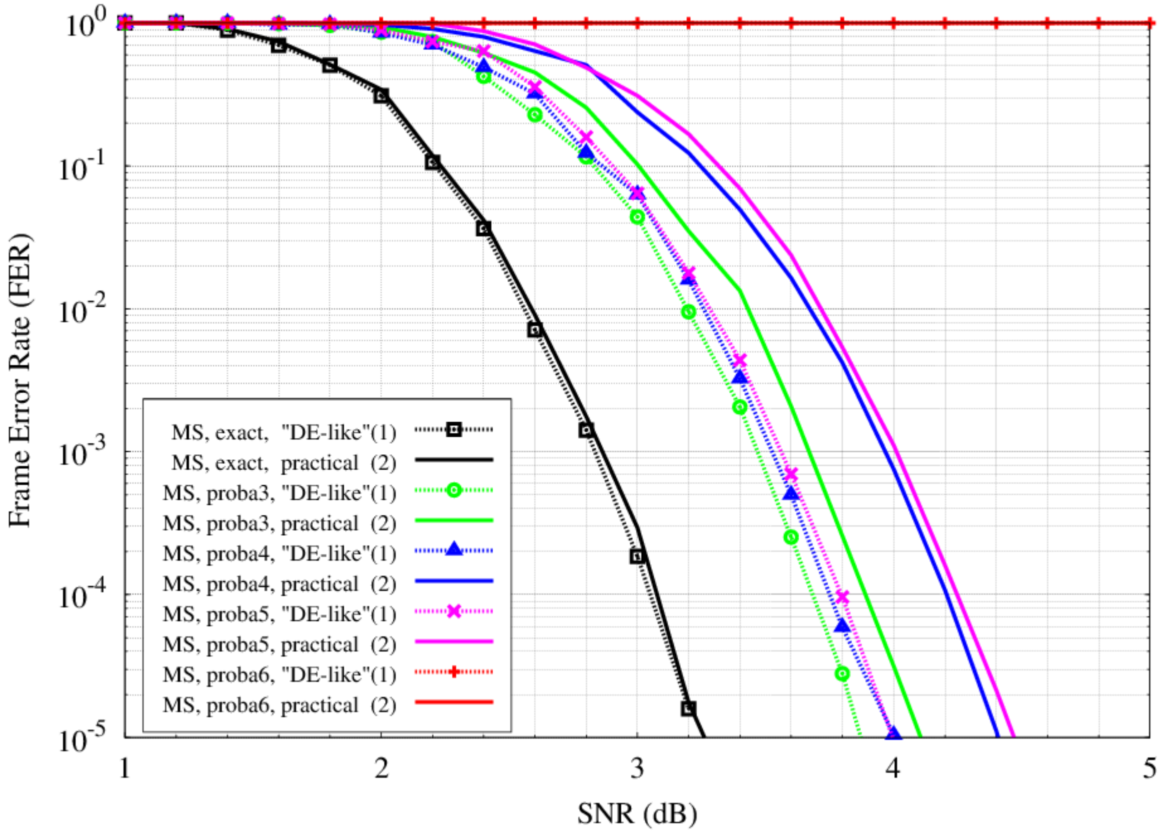
Color code:

Noiseless
Depth = 3
Depth = 4
Depth = 5
Depth = 6

Dashed curve: "DE-like" (1)

Practical implementation of Min-Sum decoder

- Mackay's regular (3,6)-LDPC code, [K = 504, N = 1008]
- Fixed-point MS decoder: 5 / 6 bits (exchanged messages / AP-LLR)



Comp.err. prob: $P_c = 0.01$
Adder err. prrob: $P_a = 0.01$

Color code:

Noiseless
Depth = 3
Depth = 4
Depth = 5
Depth = 6

Dashed curve: "DE-like" (1)
Solid curve: Practical (2)

Performance of Min-Sum-based decoder

- Min-Sum-based decoders
  - improved versions of the MS algorithm, with only a very limited (usually negligible) increase in complexity
  - Offset-Min-Sum (OMS)
  - Self-Corrected Min-Sum (SCMS)
    - intrinsic ability to detect and discard unreliable messages during the iterative decoding process.
  - **Only “practical” implementations are considered**

Performance of Min-Sum-based decoder

- Min-Sum-based decoders
  - Self-Corrected Min-Sum (SCMS)
    - intrinsic ability to detect and discard unreliable messages during the iterative decoding process.
    - a variable-to-check message $\alpha_{m,n}$ is erased (set to zero) if its sign changed with respect to the previous iteration

Initialization: $\forall n = 1, \dots, N;\ \forall m \in H(n)$

$$\gamma_n = \log(\Pr(x_n = 0 \mid y_n)/\Pr(x_n = 1 \mid y_n))$$
$$\alpha_{m,n} = \gamma_n$$

Iterations

- **CNU**: $\forall m = 1, \dots, M;\ \forall n \in H(m)$

$$\beta_{m,n} = \left(\prod_{n' \in H(m)\setminus n} \operatorname{sgn}(\alpha_{m,n'})\right) \min_{n' \in H(m)\setminus n}(|\alpha_{m,n'}|)$$

- **AP-LLR**: $\forall n = 1, \dots, N$

$$\tilde{\gamma}_n = \gamma_n + \sum_{m \in H(n)} \beta_{m,n}$$

- **VNU**: $\forall n = 1, \dots, N;\ \forall m \in H(n)$

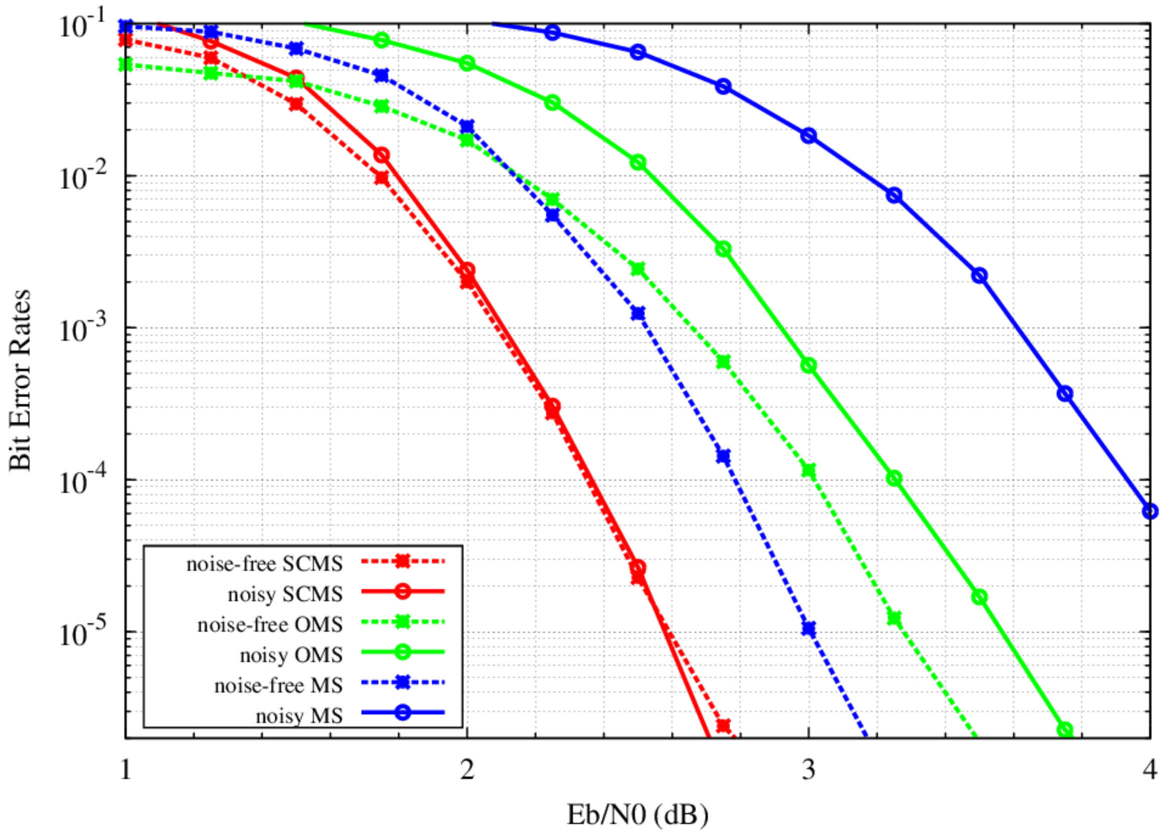
$$\alpha_{m,n}^{\text{tmp}} = \tilde{\gamma}_n - \beta_{m,n}$$

if $\operatorname{sgn}(\alpha_{m,n}^{\text{tmp}}) = \operatorname{sgn}(\alpha_{m,n})$ then $\alpha_{m,n} = \alpha_{m,n}^{\text{tmp}}$

else $\alpha_{m,n} = 0$

Performance of Min-Sum-based decoder

- Mackay's regular (3,6)-LDPC code, [K = 504, N = 1008]
- Fixed-point decoders: 4 / 5 bits (exchanged messages / AP-LLR)



Comp.err. prob: $P_c = 0.01$
Adder err. prob: $P_a = 0.01$
depth = 4

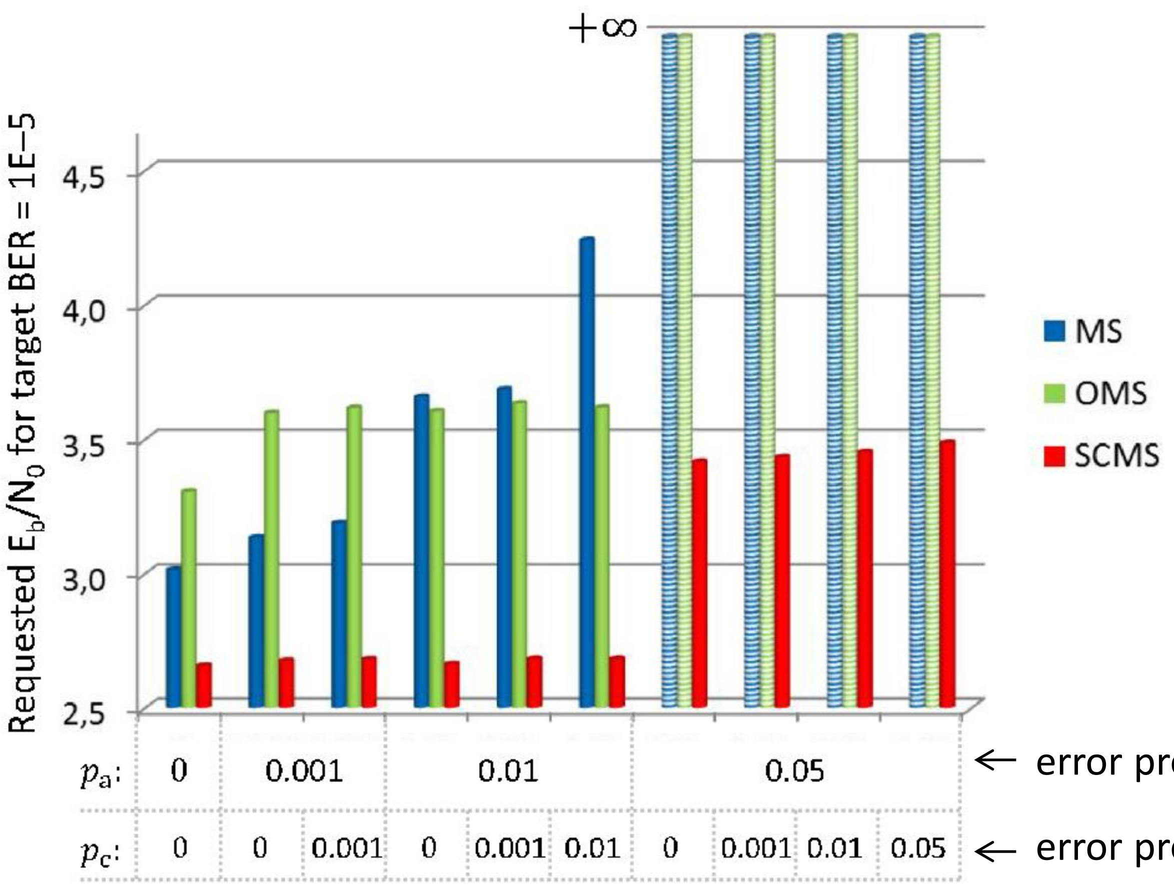
Color code:
SCMS
MS
OMS

Dashed curves: noiseless
Solid curves: noisy

Remark: noiseless SCMS achieves ≈ Belief Propagation performance

Performance of Min-Sum-based decoder

- Mackay's regular (3,6)-LDPC code, [K = 504, N = 1008]
- Fixed-point decoders: 4 / 5 bits (exchanged messages / AP-LLR)



← error prob. of the adder (**depth = 4**)

← error prob. of the comparator

Requested channel E_b/N_0 for target decoded BER = 10^{-5}

($E_b/N_0 = \propto$ means target BER cannot be achieved)

Conclusion

- "Adjustable" error-models for noisy Min-Sum-based decoders

- Density evolution analysis of the noisy Min-Sum decoder
  - proved that error protection (with a certain level of reliability) is still possible
  - characterized the sensitivity of the decoder to variations of the parameters of the error model, in terms of useful regions

- Finite-length performance of Min-Sum-based decoders
  - highlighted the limitations of the **theoretical analysis** with respect to **practical implementations**
  - evaluate finite-length performance for various parameters of the hardware noise model
  - **SCMS**: intrinsic ability to detect and discard unreliable messages, which proves to be particularly useful for noisy implementations

leti

LABORATOIRE D'ÉLECTRONIQUE
ET DE TECHNOLOGIES
DE L'INFORMATION

CEA-Leti
MINATEC Campus, 17 rue des Martyrs
38054 GRENOBLE Cedex 9
Tel. +33 4 38 78 36 25

www.leti.fr

Merci de votre attention

Min-Sum decoder / flooding vs. serial implementation

MS – flooding implementation

Initialization: $\forall n = 1, \dots, N;\ \forall m \in H(n)$

$$\gamma_n = \log(\Pr(x_n = 0 \mid y_n)/\Pr(x_n = 1 \mid y_n))$$
$$\alpha_{m,n} = \gamma_n$$

Iterations

- **CNU**: $\forall m = 1, \dots, M;\ \forall n \in H(m)$

$$\beta_{m,n} = \left(\prod_{n' \in H(m) \setminus n} \operatorname{sgn}(\alpha_{m,n'})\right) \min_{n' \in H(m) \setminus n} (|\alpha_{m,n'}|)$$

- **AP-LLR**: $\forall n = 1, \dots, N$

$$\tilde{\gamma}_n = \gamma_n + \sum_{m \in H(n)} \beta_{m,n}$$

- **VNU**: $\forall n = 1, \dots, N;\ \forall m \in H(n)$

$$\alpha_{m,n} = \tilde{\gamma}_n - \beta_{m,n}$$

MS – serial implementation

Initialization: $\forall n = 1, \dots, N;\ \forall m \in H(n)$

$$\tilde{\gamma}_n = \gamma_n = \log(\Pr(x_n = 0 \mid y_n)/\Pr(x_n = 1 \mid y_n))$$
$$\beta_{m,n} = 0$$

Iterations

- **Check-Nodes Loop**: $\forall m = 1, \dots, M$
  - **VNU**: $\forall n \in H(m)$

    $$\alpha_{m,n} = \tilde{\gamma}_n - \beta_{m,n}$$

  - **CNU:** $\forall n \in H(m)$

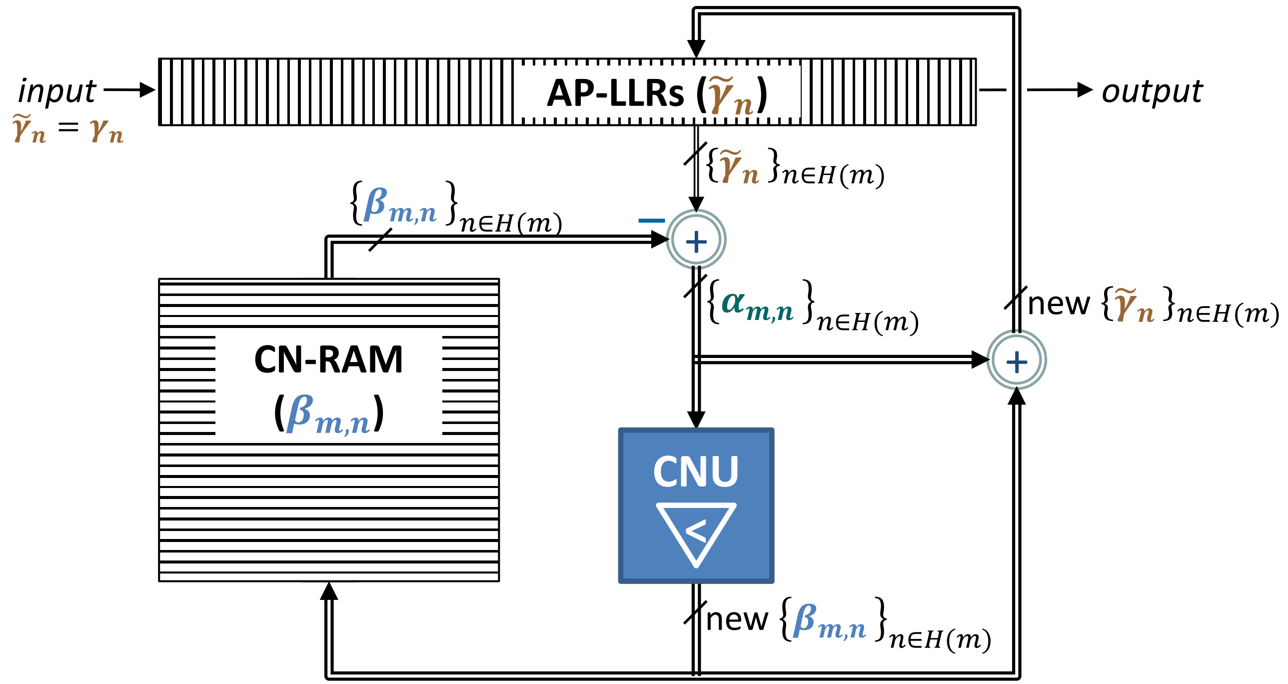
    $$\beta_{m,n} = \left(\prod_{n' \in H(m) \setminus n} \operatorname{sgn}(\alpha_{m,n'})\right) \min_{n' \in H(m) \setminus n} (|\alpha_{m,n'}|)$$

  - **AP-LLR**: $\forall n \in H(m)$

    $$\tilde{\gamma}_n = \alpha_{m,n} + \beta_{m,n}$$

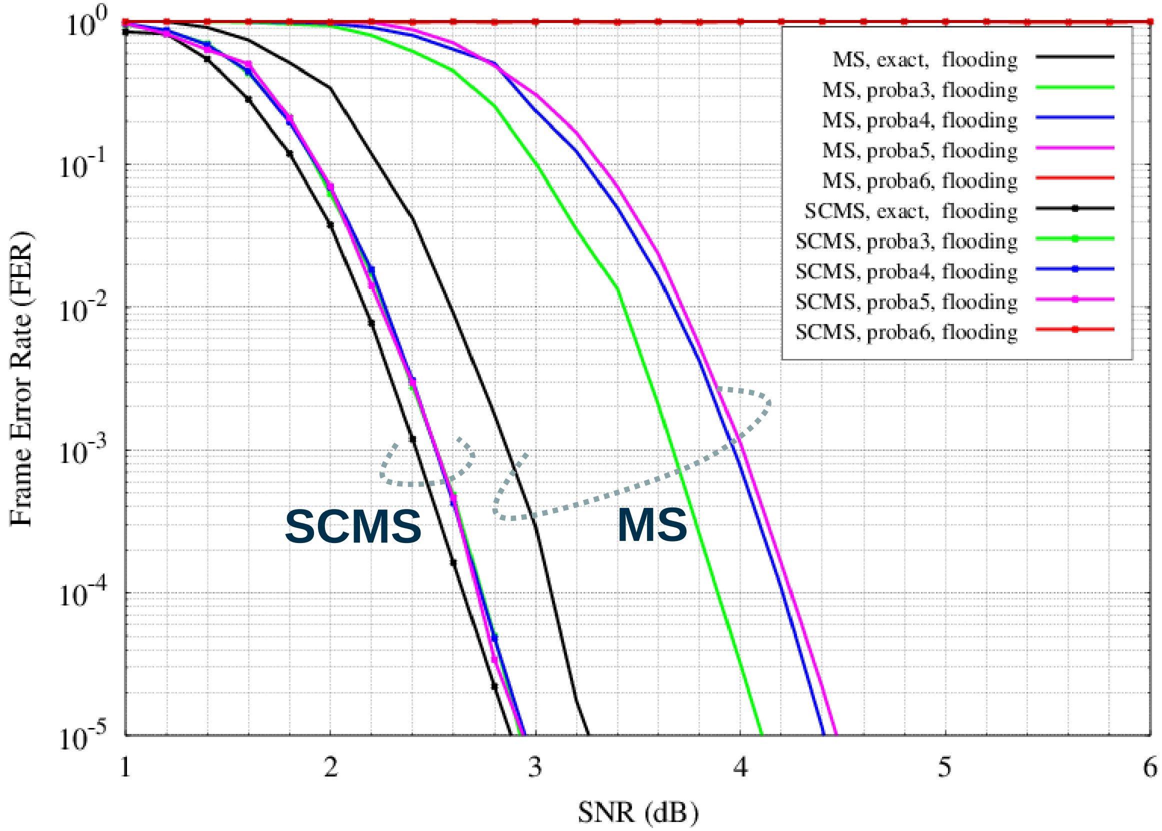
Min-Sum decoder / serial implementation

MS – serial implementation



Flooding implementation / SCMS vs. MS

- Mackay's regular (3,6)-LDPC code, [K = 504, N = 1008]
- Fixed-point decoders: 4 / 6 bits (exchanged messages / AP-LLR)

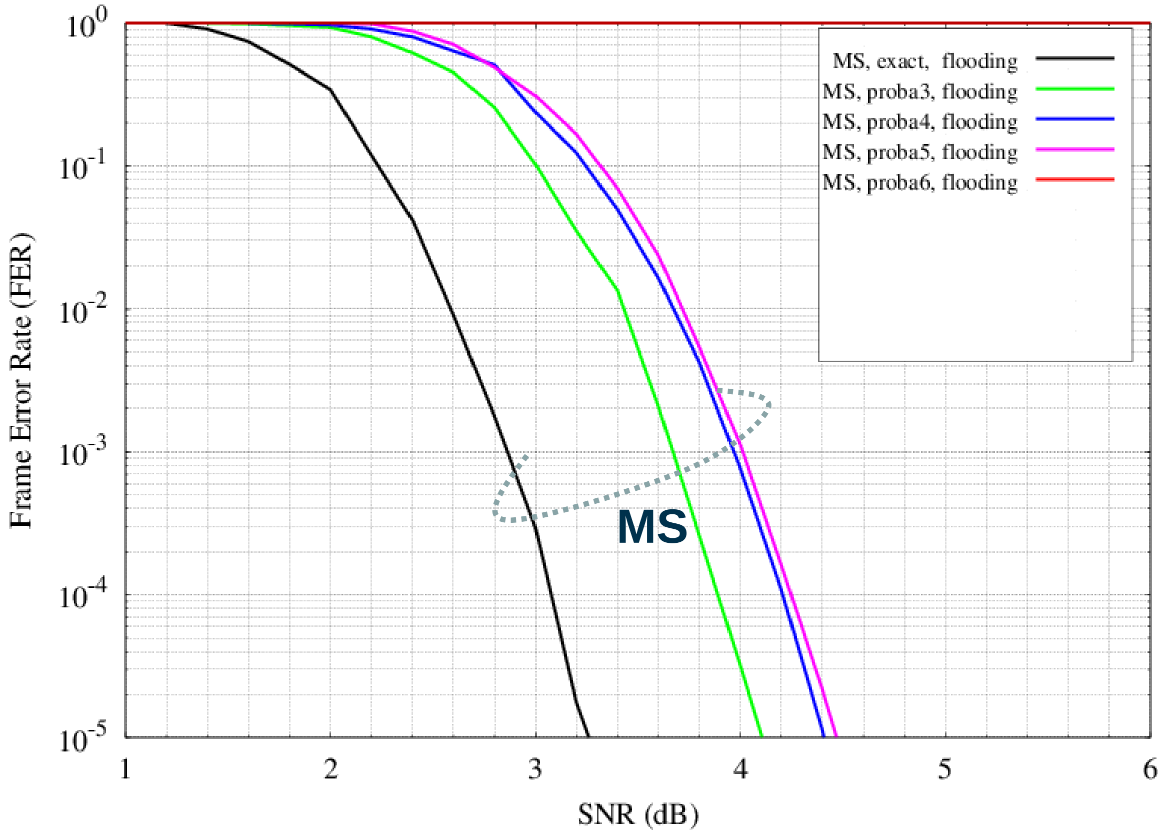


Comp.err. prob: $P_c = 0.01$
Adder err. prrob: $P_a = 0.01$

Color code:
Noiseless
Depth = 3
Depth = 4
Depth = 5
Depth = 6

MS decoder / flooding implementation

- Mackay's regular (3,6)-LDPC code, [K = 504, N = 1008]
- Fixed-point decoders: 4 / 6 bits (exchanged messages / AP-LLR)



Comp.err. prob: $P_c = 0.01$
Adder err. prrob: $P_a = 0.01$

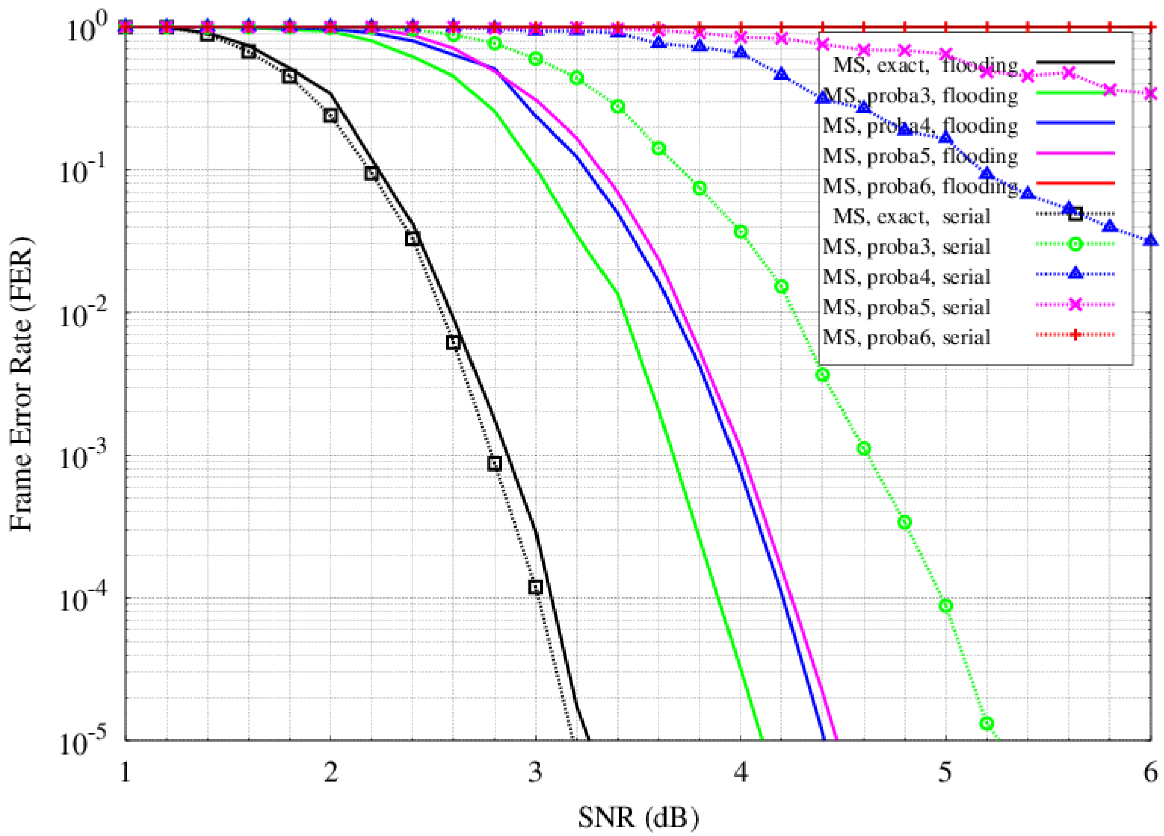
Color code:

Noiseless
Depth = 3
Depth = 4
Depth = 5
Depth = 6

Solid curves: **flooding**

MS decoder / flooding vs. serial implementation

- Mackay's regular (3,6)-LDPC code, [K = 504, N = 1008]
- Fixed-point decoders: 4 / 6 bits (exchanged messages / AP-LLR)



Comp.err. prob: $P_c = 0.01$
Adder err. prrob: $P_a = 0.01$

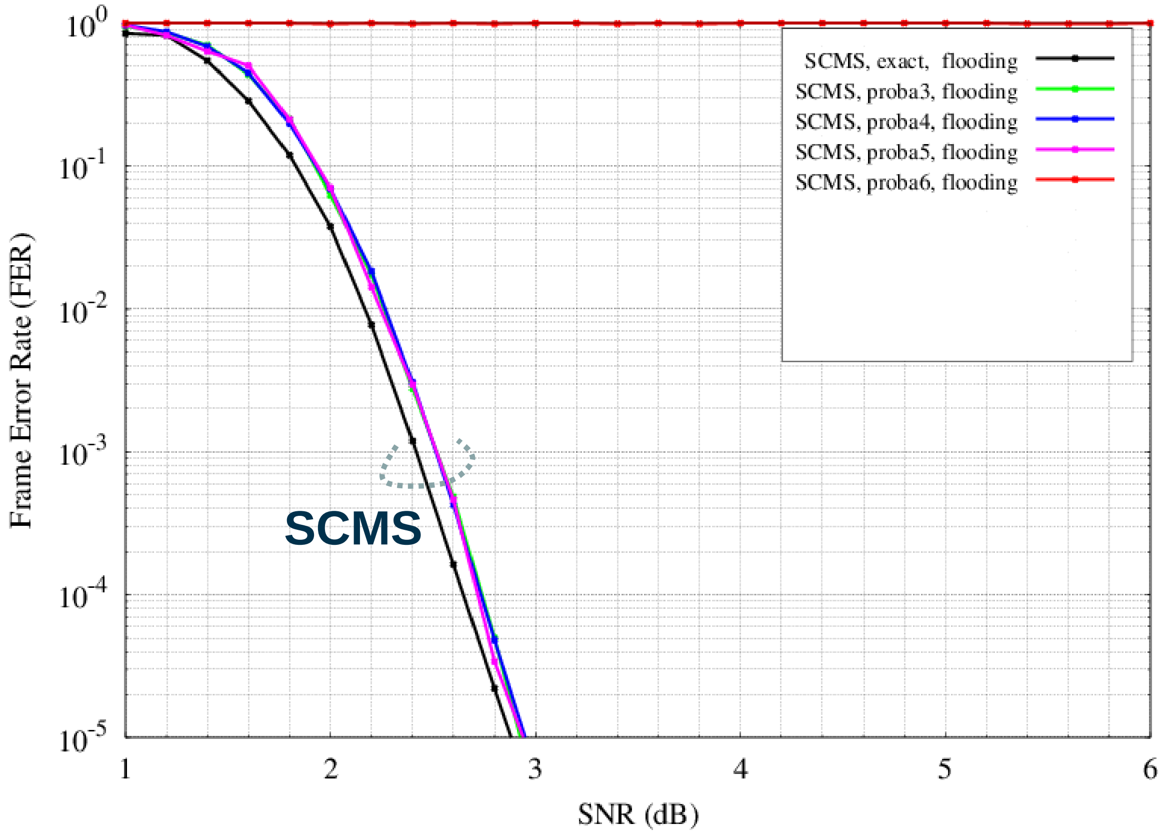
Color code:

Noiseless
Depth = 3
Depth = 4
Depth = 5
Depth = 6

Solid curves: **flooding**
Dashed curves: **serial**

SCMS decoder / flooding implementation

- Mackay's regular (3,6)-LDPC code, [K = 504, N = 1008]
- Fixed-point decoders: 4 / 6 bits (exchanged messages / AP-LLR)



Comp.err. prob: $P_c = 0.01$
Adder err. prrob: $P_a = 0.01$

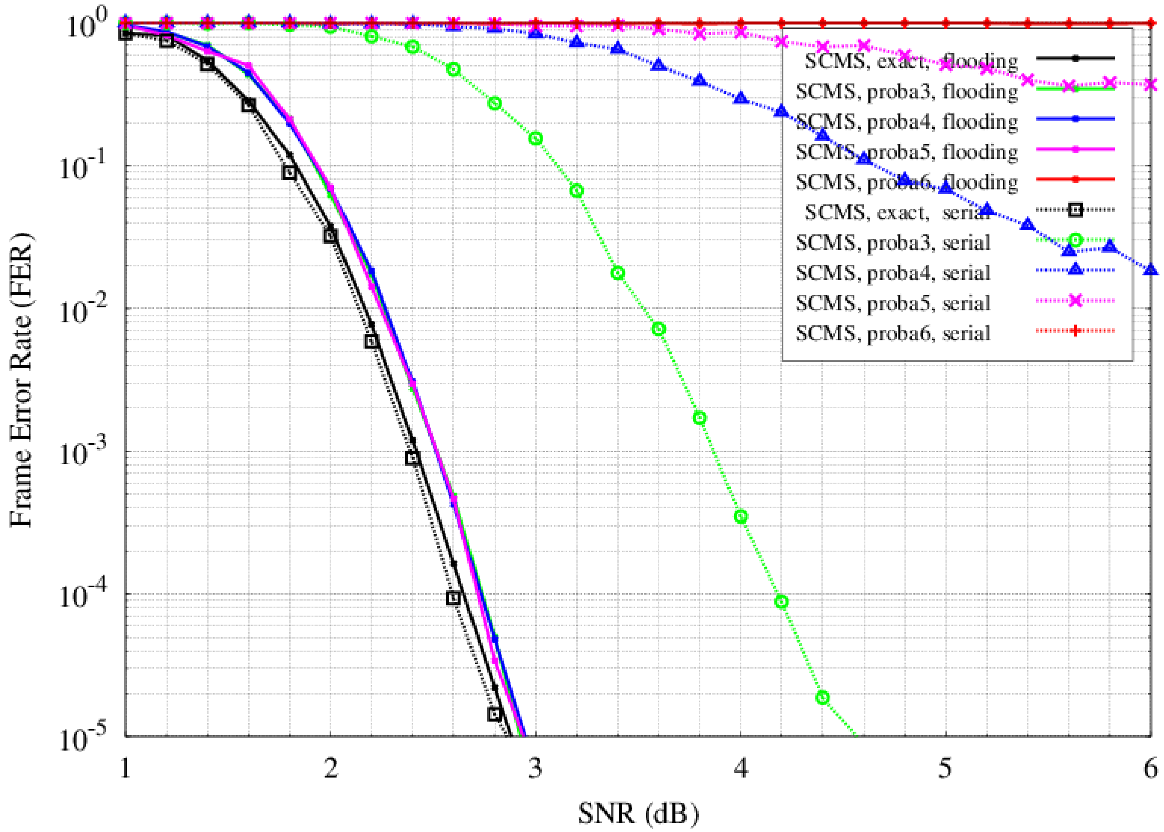
Color code:

Noiseless
Depth = 3
Depth = 4
Depth = 5
Depth = 6

Solid curves: **flooding**

SCMS decoder / flooding vs. serial implementation

- Mackay's regular (3,6)-LDPC code, [K = 504, N = 1008]
- Fixed-point decoders: 4 / 6 bits (exchanged messages / AP-LLR)



Comp.err. prob: $P_c = 0.01$
Adder err. prrob: $P_a = 0.01$

Color code:

Noiseless
Depth = 3
Depth = 4
Depth = 5
Depth = 6

Solid curves: **flooding**
Dashed curves: **serial**