# SSY 196
# Stochastic Decoding

Pavel Gueorguiev

# Presentation Overview

- Recap LDPC codes, decoding, SPA, implementation difficulties and hardware
- Intro to stochastic computing
- Application to LDPC decoding
  - Derivation of stochastic parity and check nodes
- The Markov based stochastic decoder (MbSD)
- Mathematical bounds and average behaviour of MbSD
- Results
- Drawbacks and future
- Questions

# Quick Recap

- LDPC codes are used in modern communication standards: DVB, wireless broadband IEEE 802.16e (WiMAX), Ethernet IEEE 802.3an (2048,1723) LDPC code (year 2006).
- LDPC codes are capacity approaching, for an appropriate block length
- Suboptimal decoding using sum-product algorithm (SPA)
- In general, message passing form:

CN to VN Node:

$$\alpha_{a\to i}^{t} = \frac{1}{2} - \frac{1}{2}\prod_{j\in\mathcal{N}(a)\setminus\{i\}}(1-2\,\alpha_{j\to a}^{t}),$$

VN to CN Node:

$$\alpha_{i\to a}^{t+1} = \frac{\alpha_i \prod_{b\in\mathcal{N}(i)\setminus\{a\}} \alpha_{b\to i}^{t}}{\alpha_i \prod_{b\in\mathcal{N}(i)\setminus\{a\}} \alpha_{b\to i}^{t} + (1-\alpha_i)\prod_{b\in\mathcal{N}(i)\setminus\{a\}}(1-\alpha_{b\to i}^{t})}$$

Messages: can be LLRs (as in class) or different probability distributions

# SPA Problems

- An example of a small (tiny) LDPC code parity check matrix: (16,6,6).
- The seemingly random H matrix gives rise to a large number of interconnected wires, making chip design a challenge
- Wiring complexity has an impact on the dye area of the chip and power consumption
- Stochastic decoding addresses this problem, in addition it reduces the CN and VN node complexities (previous slide)
- Parasitic capacitance in SPA decoders causes problems are high frequencies

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

# Introduction to Stochastic Computing

- First introduced in the 1953 by von Neumann, further developed in 1960
- Competitor to traditional digital logic, it lost momentum in the 1980s
- Best shown by example. Let's say we want to multiply two numbers

Using a multiplier circuit: A = 0.25, B = 0.778.  A*B = 0.1945

Now instead of multiplication, we can do something else.

1. Turn the original numbers that we wish to multiply (A and B) into 'Bernoulli' random sequences, with their respective probability distribution:

# Example Continued...

p = P(Xi = 1) = 0.25, i = 1,2,...,N  q = P(Yi = 1) = 0.778, i = 1,2,...,N

E.g. p = [0001000100010001...], [1100000010100000...], etc. Note: Sequences are non-unique!

2. Notice the table on the right, bottom right entry is the probability we are interested in!! (it is infact the multiplication, and the only value we want)

3. The operation outlined in 2. is accomplished by an AND gate (which possesses the correct truth table for our problem).

| | P(Xi = 0) | P(Xi = 1) |
|---|---|---|
| P(Yi = 0) | **(1-p)(1-q)** | **p(1-q)** |
| P(Yi = 1) | **q(1-p)** | **pq** |

# Example Continued...

4. Passing the corresponding Bernoulli sequences through The AND gate, counting the number of 1s and dividing by N we get the desired result.

- Numerical example:
  N = 500, 0.1961, Average Error: ~1.4%
  N = 5000, 0.1944, Average Error: ~0.3%
  N = 50000, 0.1946, Average Error: ~0.15%

AND Gate Truth Table

| | |
|---|---|
| 0 | 0 |
| 0 | 1 |



Output of an AND gate with the respective Bernoulli Sequence passed through it

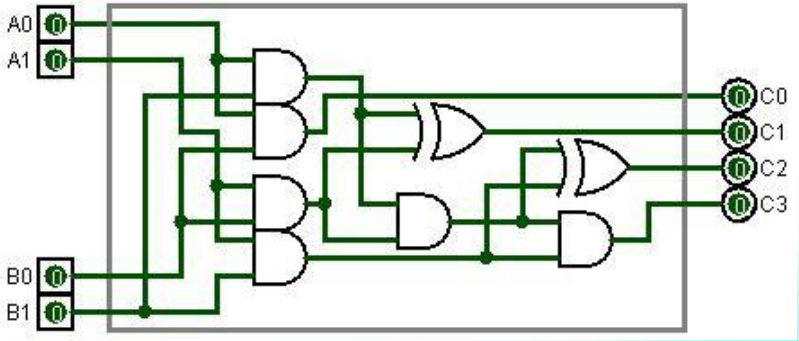
| | |
|---|---|
| 0 | 0 |
| 0 | pq |

# But why are we doing this?!

- The point here is we've converted multiplication (using large circuits) to a simple AND gate, that's quite remarkable!

Drawbacks:

- 2n bits of accuracy of result scales with O(2^(4n)),.
- Require a pseudo-random number gen.
- Latching (more on this later)



# Stochastic Decoding

- Messages are all encoded using Bernoulli sequences, with appropriate distribution. E.g.: For channel input:

  $\alpha_i := \mathbb{P}(x_i = 1 \mid y_i)$ , $\mathbb{P}(Z_i = 1) = \alpha_i,$ Zi - Bernoulli sequence, length 2k
- Algorithm has decoding cycles (iterating through the Bernoulli sequences, for reference many cycles equals one SPA iteration)
- Check, and Variable nodes are recalculated using the stochastic approach, usually all stochastic decoders have similar (but not exactly the same) CN and VN update equations
- Particular algorithm presented is the Markov based Stochastic Decoder (MbSD)
  - A convenient algorithm for mathematical design and approximation

# Stochastic VN and CN equations and intuition

From SPA equation:

$$\alpha^t_{a->i} = \frac{1}{2} - \frac{1}{2} \prod_{j \epsilon N(a)/i} (1 - 2\alpha^t_{j->a})$$

Gallager's Lemma:

now since messages are encoded using Bernoulli RVs,

$$\mathbb{P}(Z^t_{j->a} = 1) = \alpha^t_{j->a}$$

$$\mathbb{P}(Z^t_{a->i} = 1) = \frac{1}{2} - \frac{1}{2} \prod_{j \epsilon N(a)/i} (1 - 2\mathbb{P}(Z^t_{j->a} = 1))$$

$= \mathbb{P}$(odd number of 1s in $jN(a)/i$ digits with $\mathbb{P}(j = 1) = \alpha^t_{a->i}$), By extention to Gallager's proof for even number of 1s. It is therefore statistically consistent to use a modulo-2 sum operator overall the extrinsic incoming nodes leading to:

$$Z^t_{a->i} = \sum\nolimits_{jN(a)/i} Z^t_{j->a}$$

The summation is in mod 2, and finally recovering the distribution can be done with a counter and dividing by total sequence length, N:

$$\alpha^t_{a->i} = \mathbb{P}(Z^t_{a->i} = 1)$$

# VN to CN Update. Why ‘Markov’!?

- Derivation of the CN update isn’t trivial
- The “Equality” operator generates Markov chains with desirable properties
  - Markov naming convention. $Y = \circledcirc_{i=1}^{d} X_i$ “Forms a time-reversible Markov chain”

Let $Z_i$ and $Z_{b->i}$ (for $b \epsilon N(i)/a$), be independent Bernoulli sequences with

$$\mathbb{P}(Z_i = 1) = \alpha_i, and, \mathbb{P}(Z_{b->i} = 1) = \alpha_{b->i}^{t+1}$$

The SPA equation now becomes, the probability:

$$\{Z_i = 1, Z_{b->i} = 1, \forall b \epsilon N(i)/\{a\}\}$$

conditioned on the event

$$\{Z_i = Z_{b->i}, \forall b \epsilon N(i)/\{a\}\}$$

# Inuition of the update equations

- CN to VN update
  - Mod-2 summation
  - Statistically consistent with the SPA algorithm (on average, in the SD case).
- VN to CN update
  - Equality check, you sent along the common value on the edges. If it's 0 is common you send 0, if 1 you send 1, and if there's disagreement you send the previously sent bit, (Y(0) = {0,1} with 50% probability)
  - These nodes are asymptotically consistent as k-> $\infty$ . Remember k is (a half) of the length of the Bernoulli sequence.

# MbSD Algorithm

1. Initialize messages from variable nodes to check nodes at time $t = 0$ by $Z^0_{i\to a} = Z^0_i$.

2. VN Update:

$$Z^t_{a\to i} = \bigoplus_{j\in\mathcal{N}(a)\setminus\{i\}} Z^t_{j\to a}.$$

$$W^{t+1}_{i\to a} = \bigodot_{b\in\mathcal{N}(i)\setminus\{a\}} Z^t_{b\to i} \odot Z^{t+1}_i.$$

3. CN Update: $\{W^{t+1}_{i\to a}(k+1), W^{t+1}_{i\to a}(k+2), \ldots, W^{t+1}_{i\to a}(2k)\}$.

Draw IID samples of the above distribution to generate $Z^{t+1}_{i\to a}(\ell)$

4. Finally the marginal distributions are estimated by: $U^{t+1}_i = \bigodot_{a\in\mathcal{N}(i)} Z^t_{a\to i} \odot Z^{t+1}_i$

$$\hat{\eta}^{t+1}_i = \frac{1}{k} \sum_{\ell=k+1}^{2k} U^{t+1}_i(\ell)$$

# Questions about MbSD to answer

- Due to the random nature of SD some questions are to be answered:
  - Behaviour on average
  - Is typical performance around the average or a measure around the average
- Assumption: SP message updates are consistent (there exists a sequence such that the estimated marginals tent to the true marginals as iterations tend to infinity)
- Theorem 1: stochastic marginals approach SP marginals:

$$\max_{1 \leqslant i \leqslant n} \left| \mathbb{E}\left[\hat{\eta}_i^t\right] - \mu_i^* \right| \leqslant \delta,$$

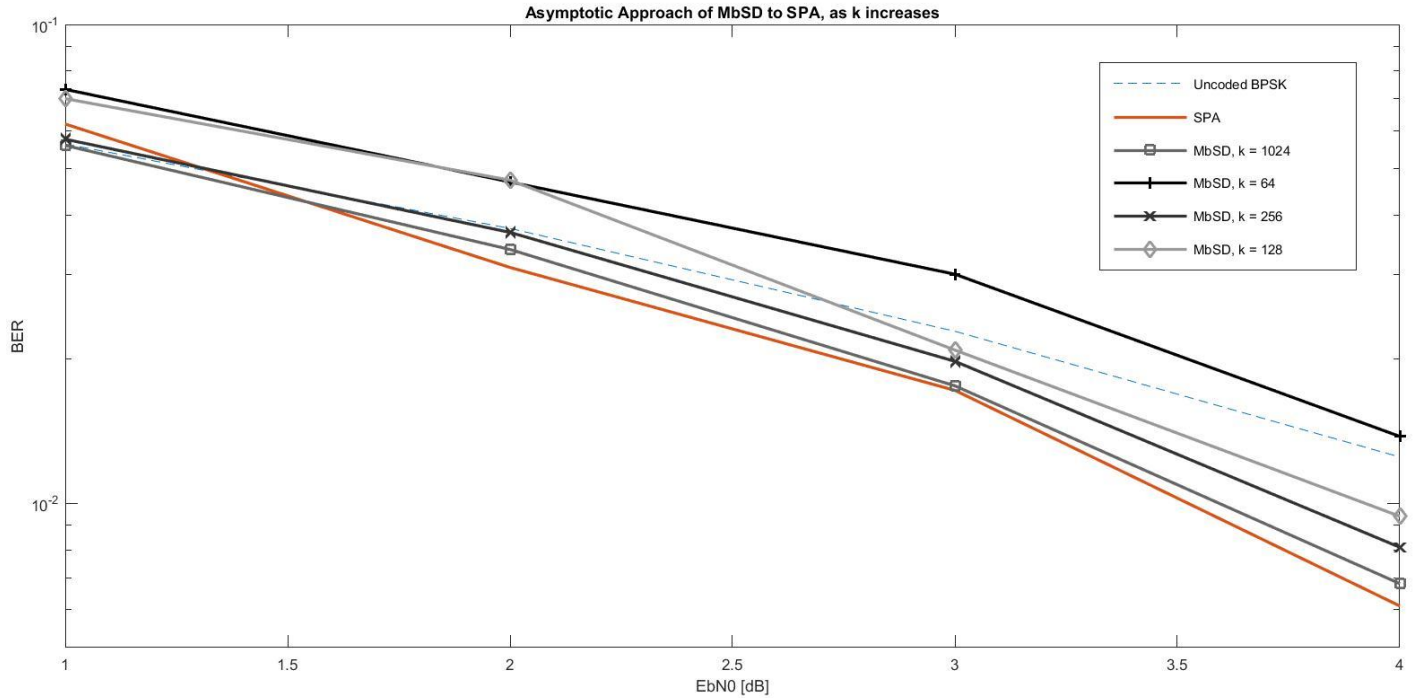
- Theorem 2: Variance decreases in linear time as k increases:

$$\max_{1 \leqslant i \leqslant n} \max_{t \geqslant 0} \operatorname{var}\left(\hat{\eta}_i^t\right) = \mathcal{O}\left(\frac{1}{k}\right)$$

# Simulation Results

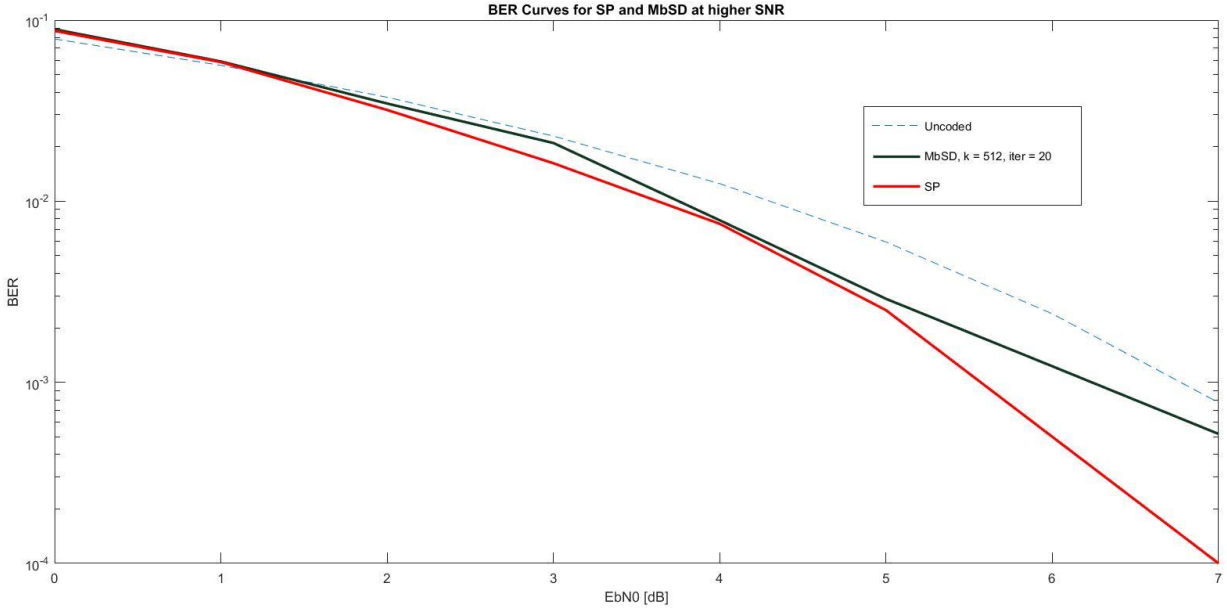
- Implementation of the MbSD algorithm
- Hamming (7,4) code.
- Exploration of the convergence with respect to k (the length of the Bernoulli sequence, divided by two)
- Asymptotic approach towards the SPA algorithm for the same Hamming code
- About twice as long as simulation for the SPA at same iterations. As k increases the running time also increases.
- Simulation was done for BERs 1 to 4

# Results: Convergence with k



# Result: Higher SNR

Note: What appears to be a floor at high SNR. This could be due to 1) low number of iterations (due to time constraint) or  2) latching



# Improvements and future work

- Latching: is when nodes get ‘locked’ in a state where they produce the same result for different inputs.
  - Happens at high SNR where probabilities near 0 or 1 % and there isn’t a lot of switching going on in the hardware elements, to which the bit streams become correlated, and convergence becomes very slow
- Noise Dependent Scaling: At high SNR the received likelihoods are scaled down by a factor proportional to the SNR
  - At high SNR the Markov sequences are likely to be all-one or all-zero
  - This significantly lowers the MbSD rate of convergence
  - This is a very effective way to get rid of latching

# Result of NDS (from paper)

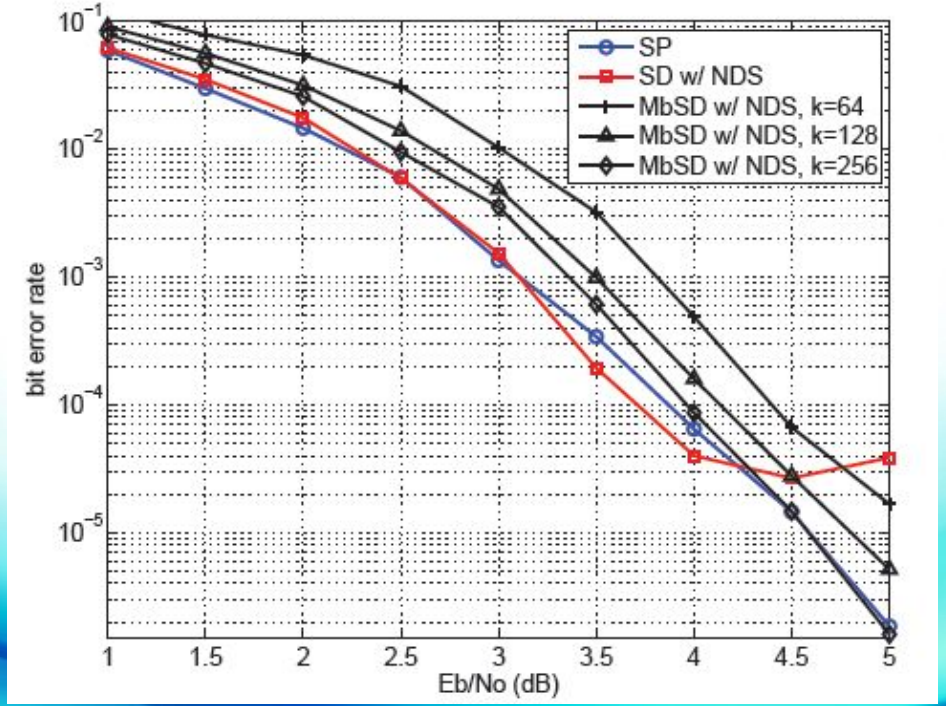
This is a:

(3,6)-LDPC code, with n=200 VNs

M = 100 CNs.

Note: SD is another stochastic

decoder for comparison



# Thank you!!


Noorshams, Nima, and Aravind Iyengar. **"A Novel Stochastic Decoding of LDPC Codes with Quantitative Guarantees."** *arXiv preprint arXiv:1405.6353* (2014).