One-Bit LDPC Message Passing Decoding Based on Maximization of Mutual Information

ZOU Sheng and Brian M. Kurkoski

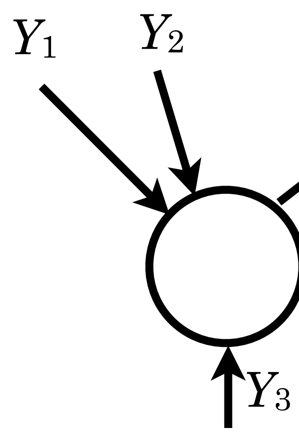
kurkoski@ice.uec.ac.jp

University of Electro-Communications
Tokyo, Japan

University of Science and Technology of China
Hefei, China

LDPC Workshop
September 29, 2009
Tokyo Institute of Technology, Tokyo

Conventional LDPC Message Quantization



Belief-propagation decoding of LDPC is well understood.

Variable node function:

$$Z = Y_1 + Y_2 + Y_3$$

Where Z, Y are continuous values:

$$\log \frac{\Pr[y|x=1]}{\Pr[y|x=0]}$$

VLSI Implementation

Y, Z are quantized using fixed-point representations

-2	-1.75	-1.5	-1.25	-1	-0.75	-0.5	-0.25	0	0.25	0.5	0.75	1	1.25	1.5	1.75
10.00	10.01	10.10	-10.11	11.00	11.01	11.10	11.11	00.00	00.01	00.10	00.11	01.00	01.01	01.10	01.11

Increasing the number of bits improves performance, but increases complexity

Typically, 6-7 bits per message are needed for floating-point performance

Can we do something better?

Break the wall between Theory and Practice

Theory

Compute fundamental limits

- Capacity, bounds

Coding theory:

- find good codes
- efficient decoding algorithms
- implement in C/Matlab

Practice

Circuits for mobile communications, storage, etc.

Implement in VLSI

- low power consumption
- high performance

Basic questions:

- How to quantize?
- Which decoding algorithm?

Broad Research Goal: Break this wall

~ Find the fundamental limits on implementation complexity ~

- **Theory**: Find and solve new information theoretic problems
- **Practice**: Improve the performance/complexity tradeoff
  Cheaper devices, longer battery life, etc.

History of Quantization of Messages-Passing Algorithms

Vector quantization

BCJR Algorithm vector quantization of the state metrics

- Convolutional codes, erasure channel: exact quantization [Globe 2003, ISIT 2004]
- Inter-symbol interference channel [ISIT 2005] **High complexity**

GF(q) LDPC codes Vector quantization of q-ary messages

- "Heuristic" vector quantization [ITA 2007] **Good only certain chan.**

Vector quantization is hard! Try scalar quantization

Scalar quantization

Binary LDPC codes quantize messages to maximize mutual information

- Channel quantization ≈ Message-passing decoding maps [Globecom 2008]
- Algorithm to quantize DMC [ITW 2010], proof of optimality [sub. IT 2011]
  - Typical VLSI 6-7 bits/message → our method 4 bits/message

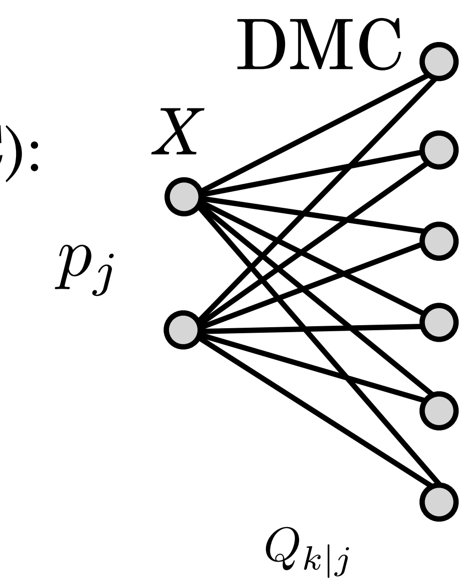
Finite-length binary codes (this talk):

- Show results hold for finite-length codes
- Look at one-bit per message LDPC decoding, compare with bit-flipping

Above papers are my joint work with P. Siegel, J. Wolf, K. Yamaguchi, K. Kobayashi and H. Yagi.

Background:
Maximizing Mutual Information

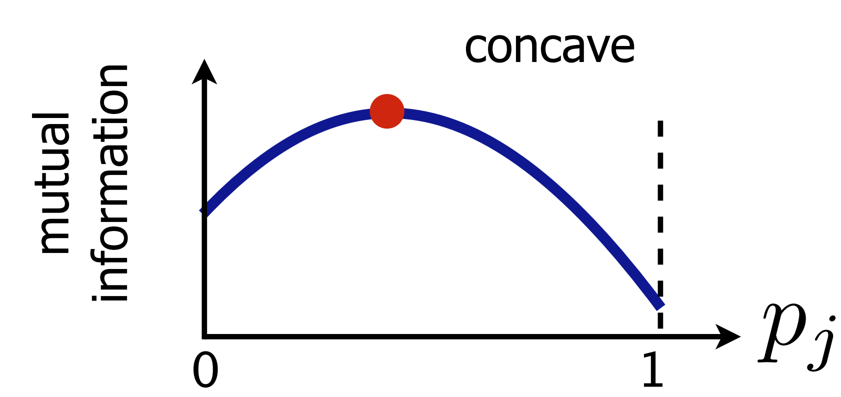
Mutual information of a discrete memoryless channel (DMC):



$$I(X;Z) \quad = \quad \sum_k \sum_j p_j Q_{k|j} \log \frac{Q_{k|j}}{\sum_j p_j Q_{k|j}}.$$

Channel capacity C is the maximization of mutual information (over input distribution p_j):

$$R \quad \leq \quad C = \max_{p_j} I(X;Z)$$

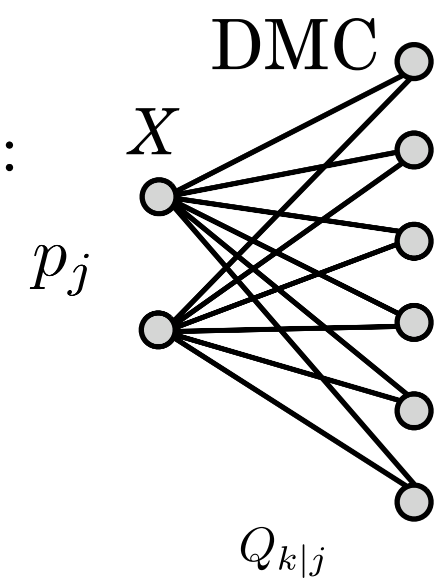


- Arimoto-Blahut algorithm computes the capacity.
- Mutual information gives highest achievable rate R

Thus:

Maximization of mutual information is an **excellent metric for quantization**!

Background:
Maximizing Mutual Information

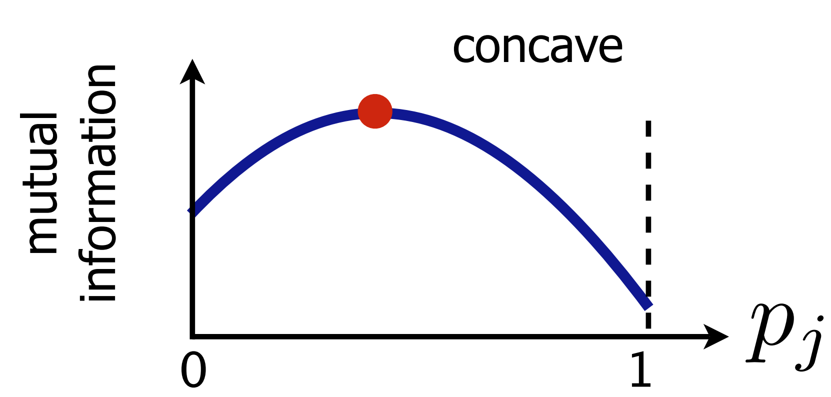


Mutual information of a discrete memoryless channel (DMC):

$$I(X;Z) \quad = \quad \sum_k \sum_j p_j Q_{k|j} \log \frac{Q_{k|j}}{\sum_j p_j Q_{k|j}}.$$

Channel capacity C is the maximization of mutual information (over input distribution p_j):

$$R \quad \leq \quad C = \max_{p_j} I(X;Z)$$



- Arimoto-Blahut algorithm computes the capacity.
- Mutual information gives highest achievable rate R

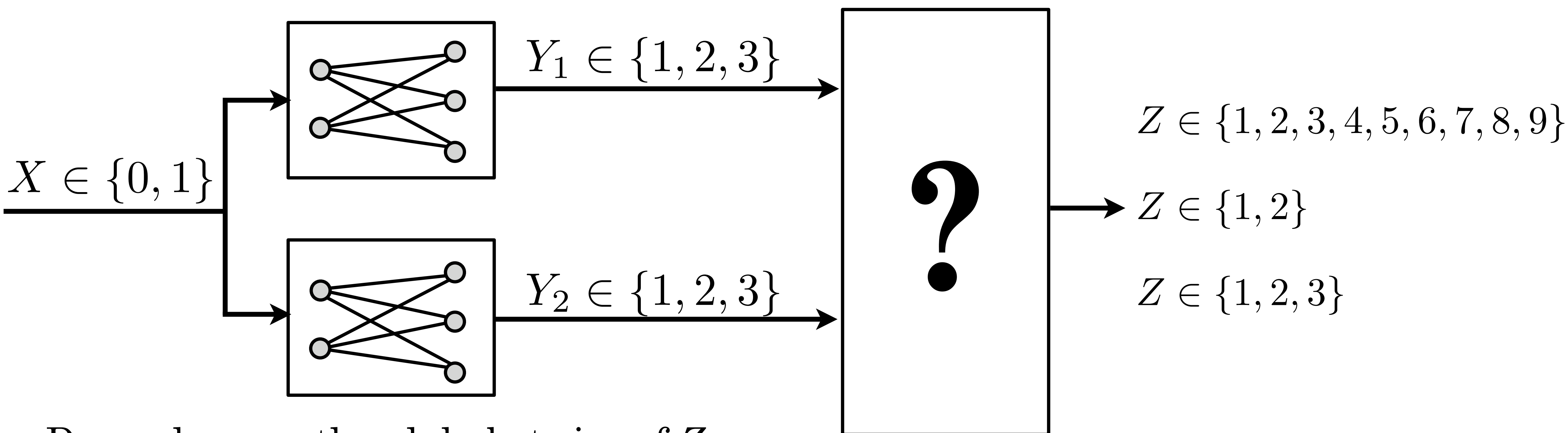
Thus:

Maximization of mutual information is an **excellent metric for quantization**!

A Question For You

Suppose a bit X is transmitted over two independent DMCs

- Goal: combine Y_1 and Y_2 into Z
- Want to maximize mutual information $I(X;Z)$
- How to combine?

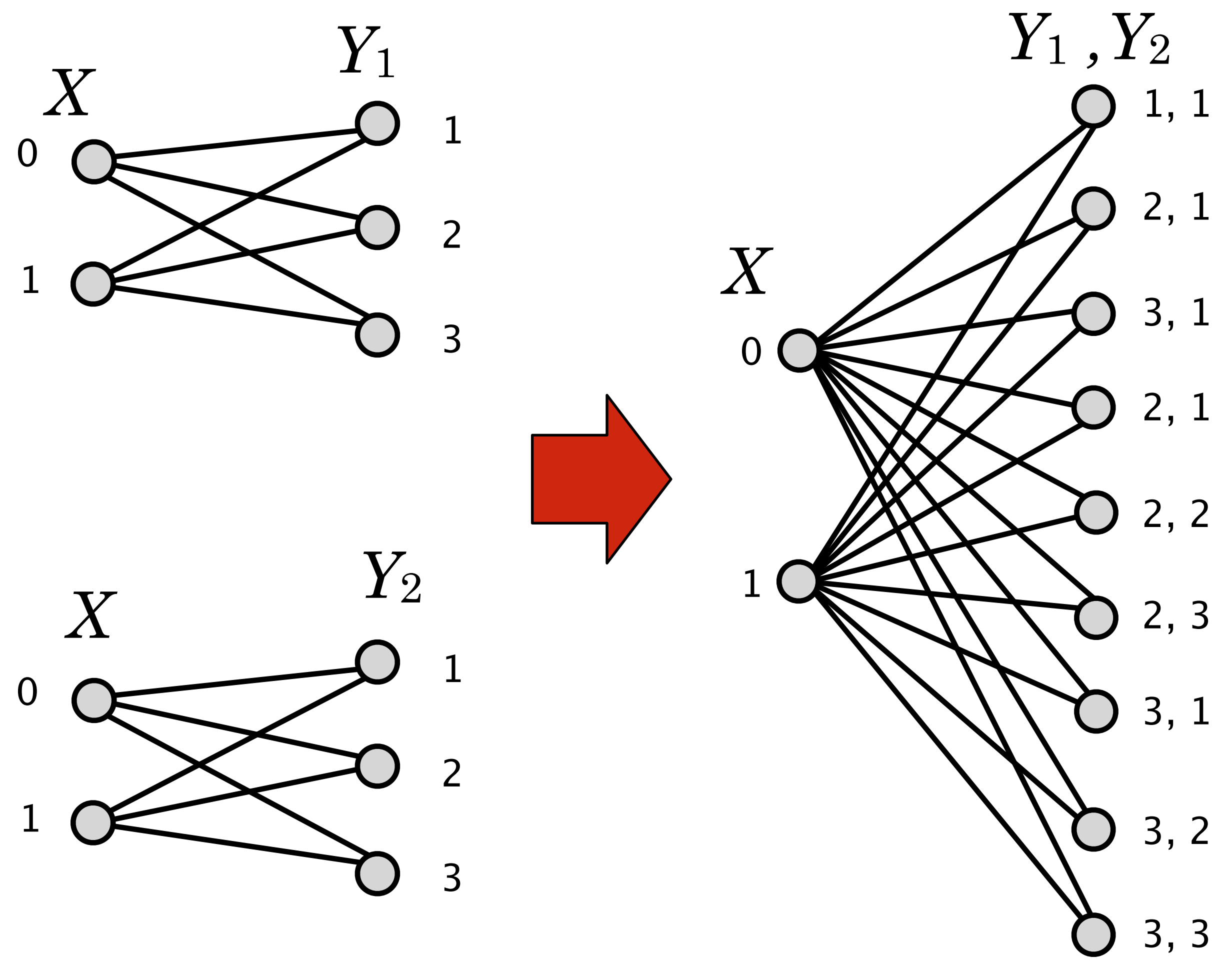


Depends upon the alphabet size of Z:

- **Easy** Size 9: trivial to get $I(X;Z) = I(X;Y_1,Y_2)$
- **Easy** Size 2: making hard decisions
- **Hard** Size 3: Let me tell you....

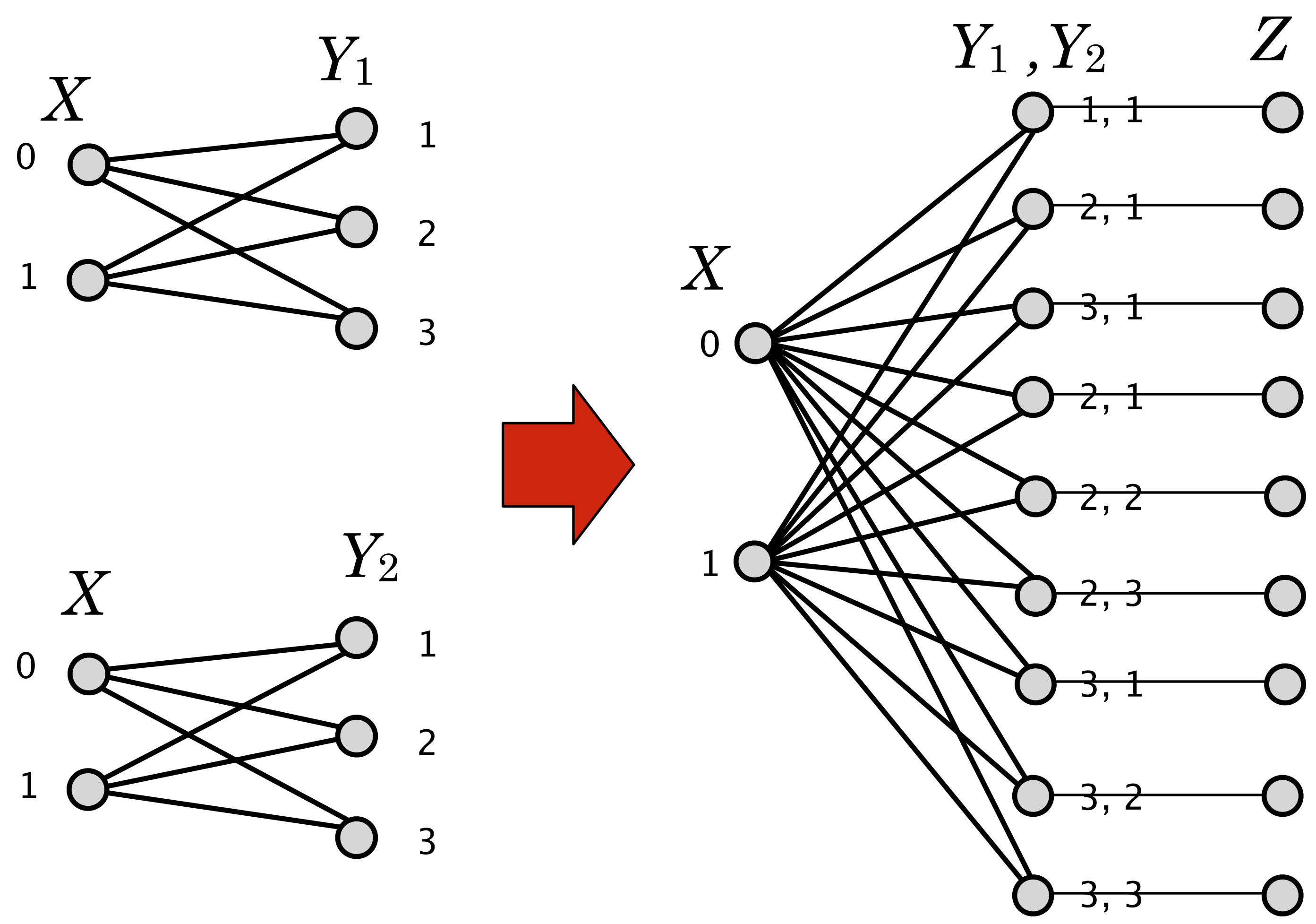
Answer: “DMC Quantization Algorithm”

- Create a “product channel”
- K: number of quantizer outputs
  - $K = 9$. A one-to-one mapping → no loss of mutual information
  - $K \leq 8$. “DMC Quantization Algorithm” finds the optimal quantizer [K. and Yagi, sub. IT 2011, http://arxiv.org/abs/1107.5637]



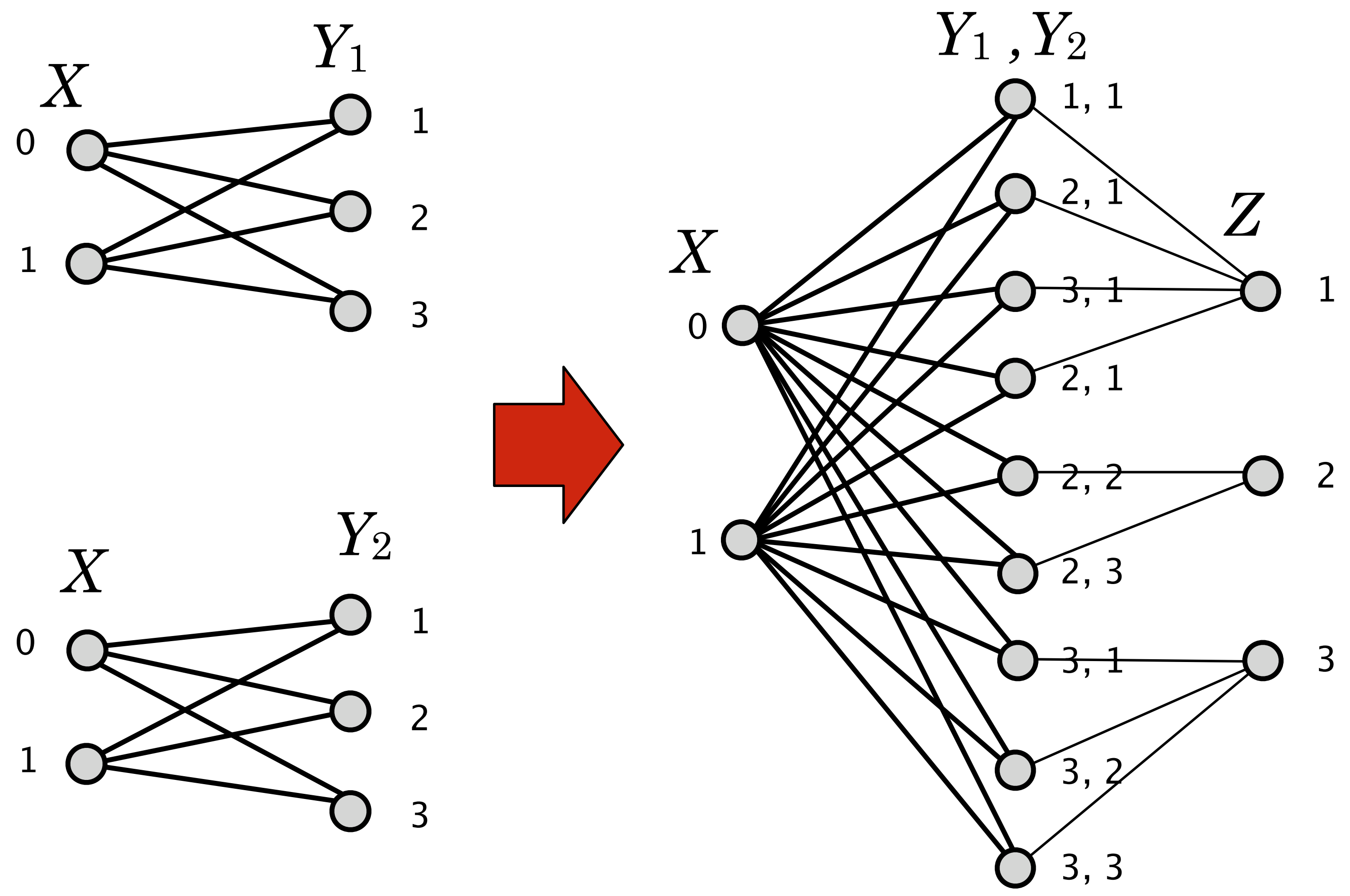
Answer: "DMC Quantization Algorithm"

- Create a "product channel"
- K: number of quantizer outputs
  - $K = 9$. A one-to-one mapping → no loss of mutual information
  - $K \leq 8$. "DMC Quantization Algorithm" finds the optimal quantizer [K. and Yagi, sub. IT 2011, http://arxiv.org/abs/1107.5637]



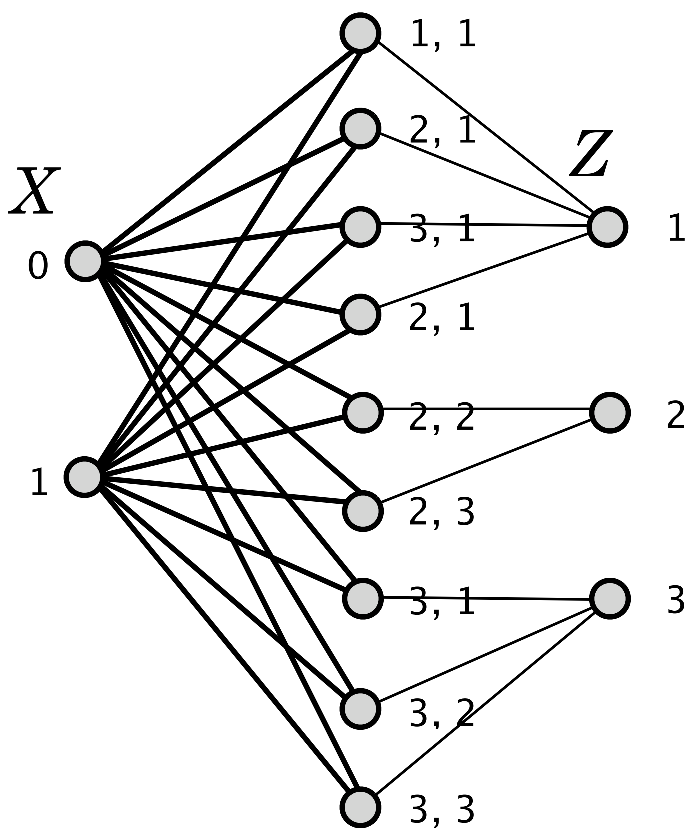
Answer: "DMC Quantization Algorithm"

- Create a "product channel"
- K: number of quantizer outputs
  - $K = 9$. A one-to-one mapping → no loss of mutual information
  - $K \leq 8$. "DMC Quantization Algorithm" finds the optimal quantizer [K. and Yagi, sub. IT 2011, http://arxiv.org/abs/1107.5637]



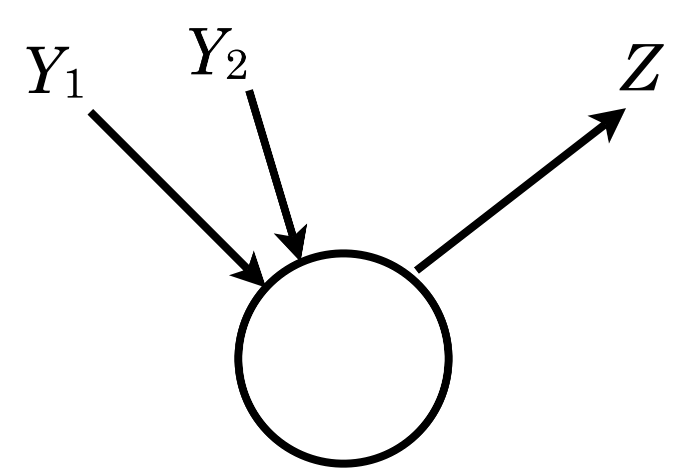
From Channel Quantizers to Decoding Algorithm

From the quantizer, can easily construct a table that gives Z from Y1 and Y2



values Z

Y_2 \ Y_1	**1**	**2**	**3**
1	1	1	1
2	1	2	2
3	3	3	3



This table is a decoding rule!

- Y_1, Y_2 are inputs at a variable node
- Z is the output

Easily extend to check node, multiple inputs, etc.

Message-passing decoding which maximizes mutual information

Quantization of a Binary-Input AWGN Channel

Before density evolution, we need to quantize the AWGN channel.

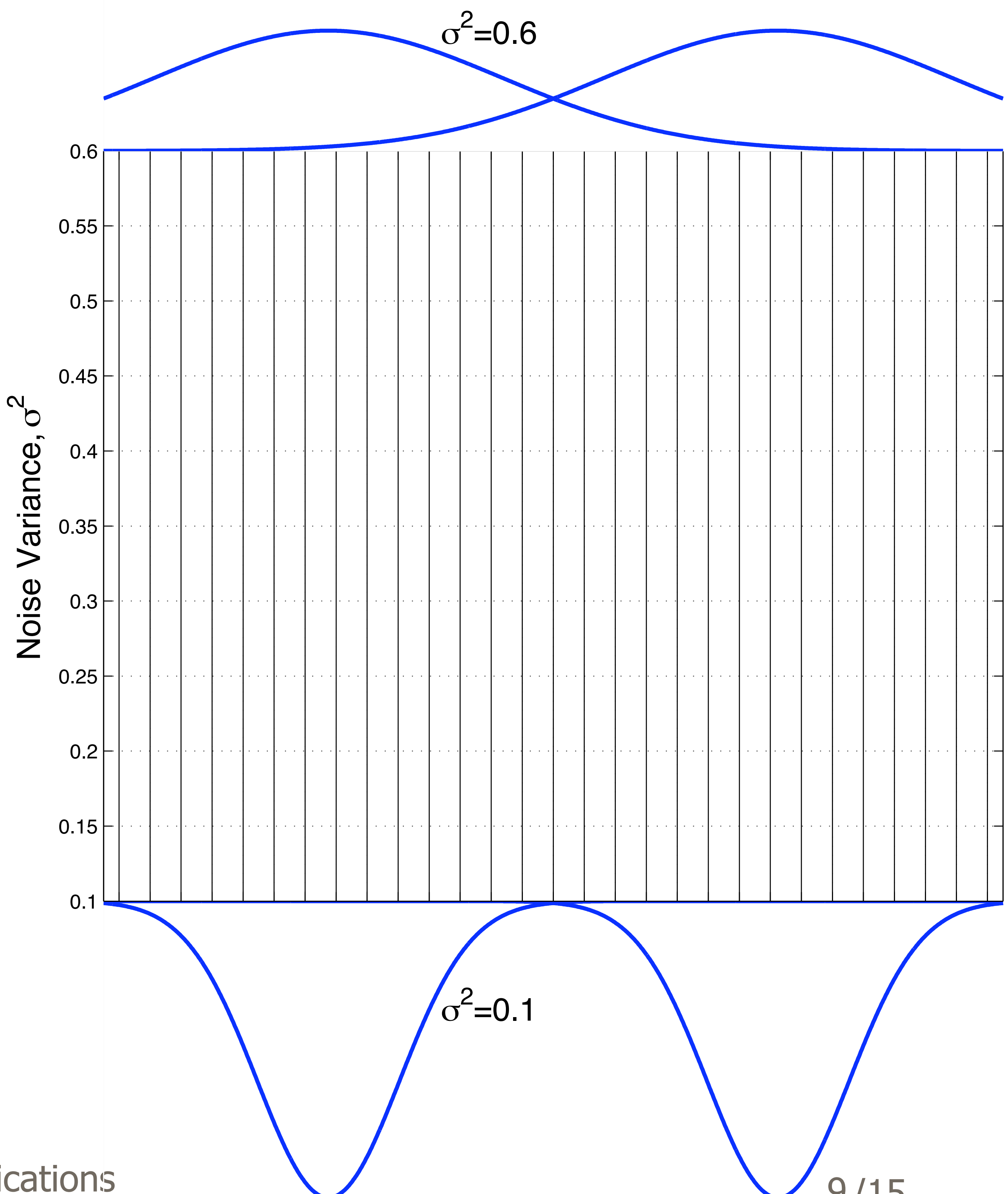
Use the Quantization Algorithm:

- Quantization Algorithm cannot operate on continuous output channels
- First create a DMC (using uniform quantization)
- Then apply the Quantization Algorithm

Example:

- AWGN various variances
- DMC with 30/500 outputs
- Quantized to 8 outputs

  (boundaries are shown)



Quantization of a Binary-Input AWGN Channel

Before density evolution, we need to quantize the AWGN channel.

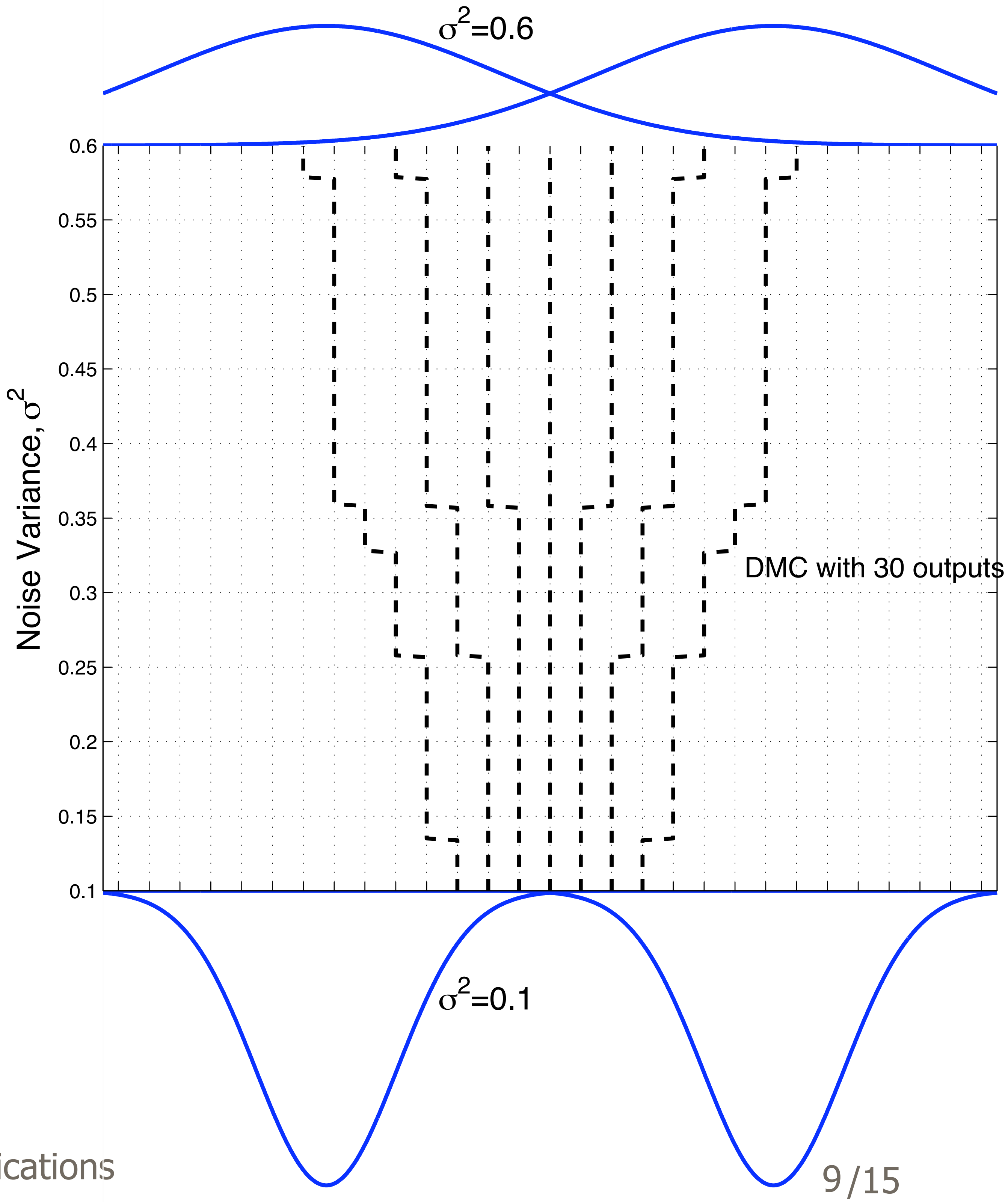
Use the Quantization Algorithm:

- Quantization Algorithm cannot operate on continuous output channels
- First create a DMC (using uniform quantization)
- Then apply the Quantization Algorithm

Example:

- AWGN various variances
- DMC with 30/500 outputs
- Quantized to 8 outputs

  (boundaries are shown)



Quantization of a Binary-Input AWGN Channel

Before density evolution, we need to quantize the AWGN channel.

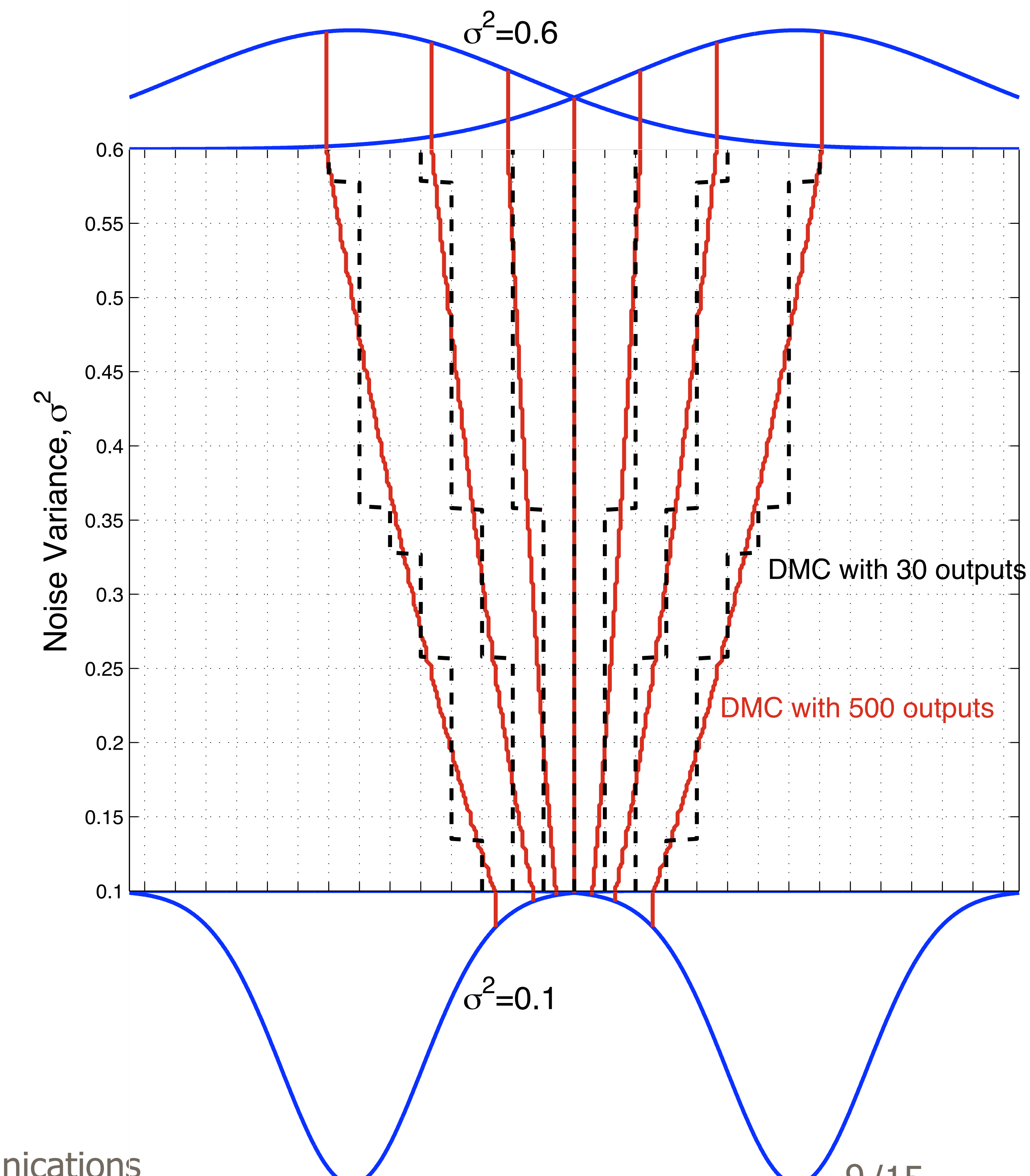
Use the Quantization Algorithm:

- Quantization Algorithm cannot operate on continuous output channels
- First create a DMC (using uniform quantization)
- Then apply the Quantization Algorithm

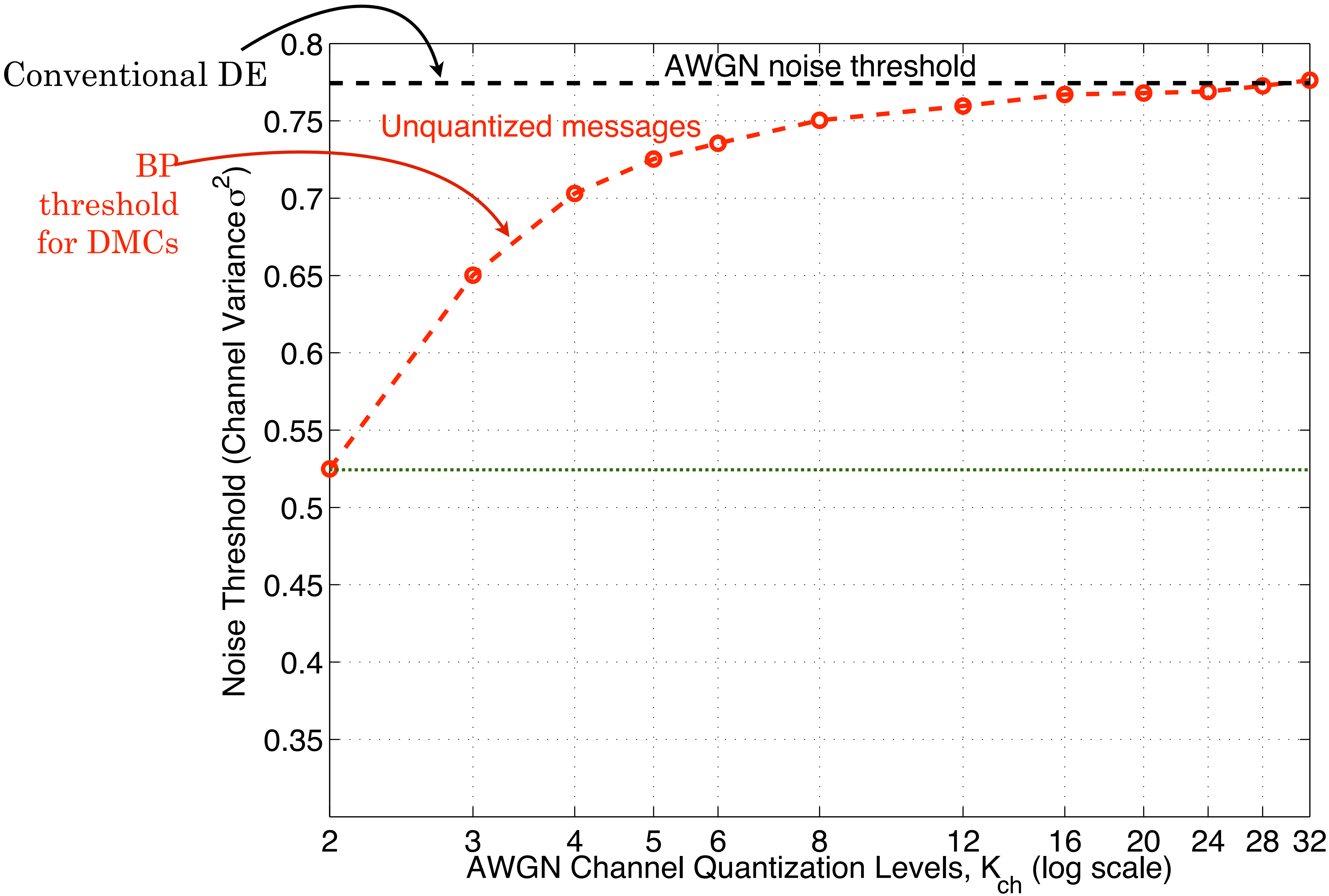
Example:

- AWGN various variances
- DMC with 30/500 outputs
- Quantized to 8 outputs

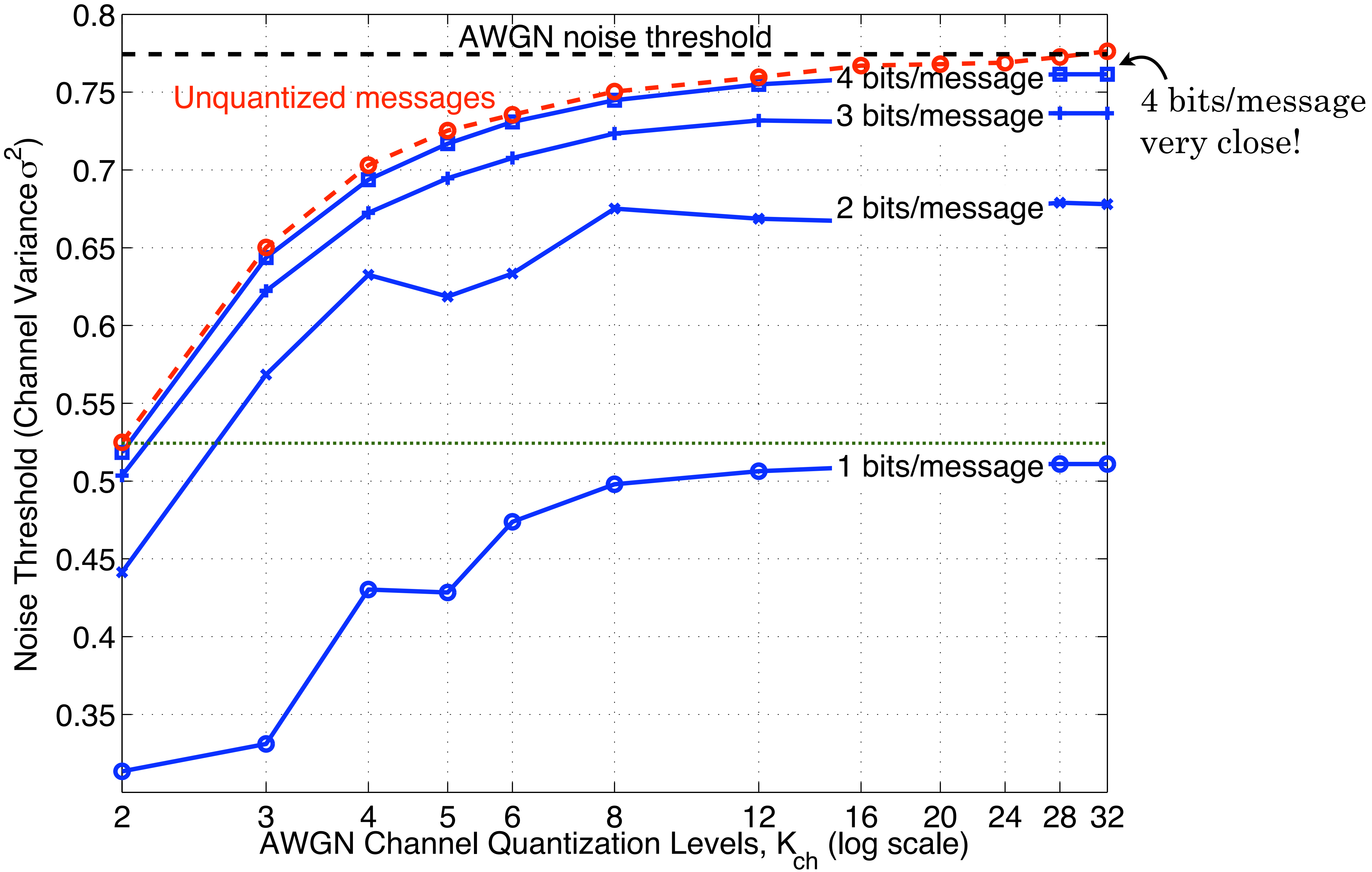
  (boundaries are shown)



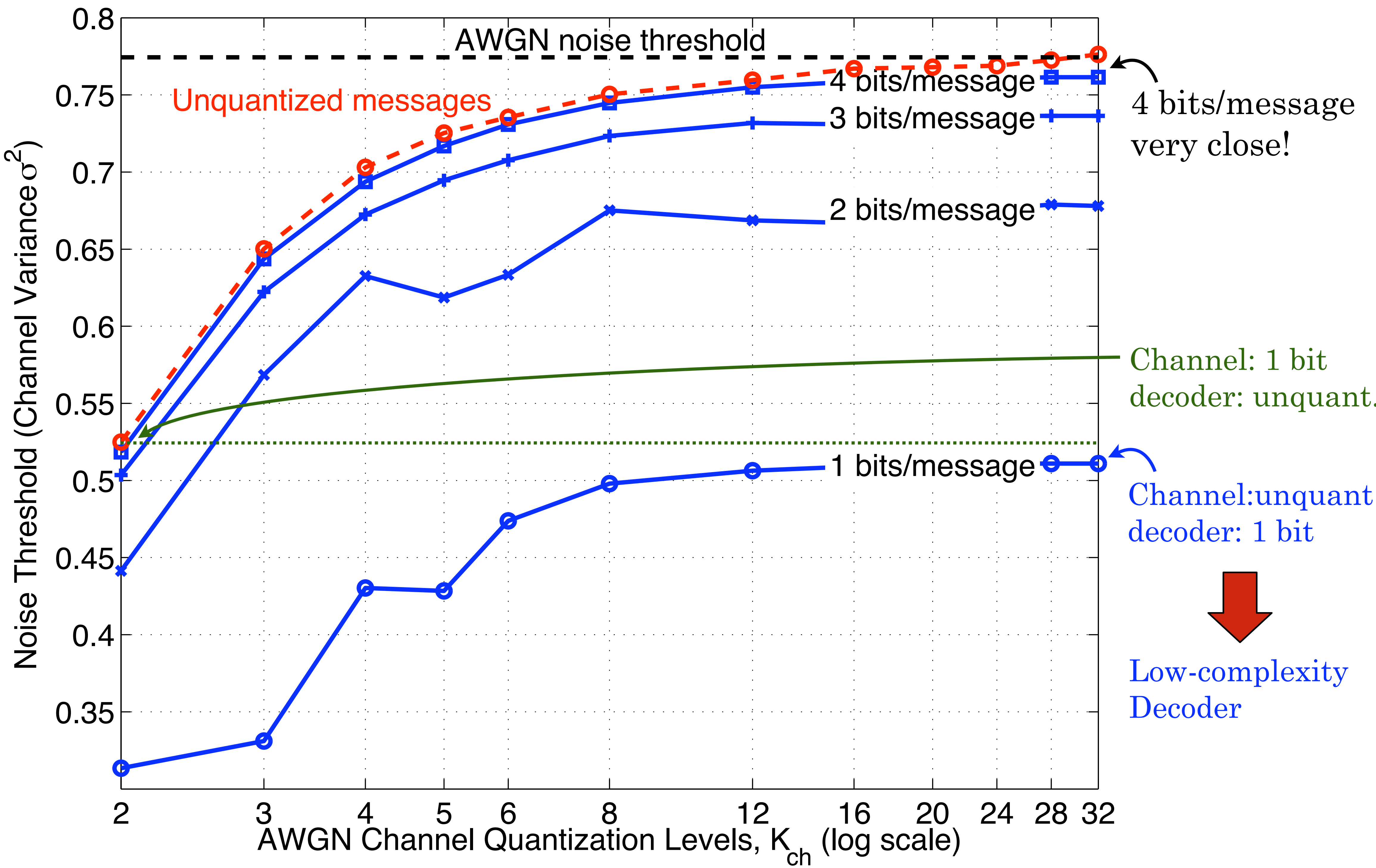
Infinite Block Length — (3,6) Regular LDPC Density Evolution Noise Thresholds



Infinite Block Length — (3,6) Regular LDPC Density Evolution Noise Thresholds



Infinite Block Length — (3,6) Regular LDPC Density Evolution Noise Thresholds



Proposed One Bit Message-Passing vs. Weighted Bit Flipping

What about finite-length codes?

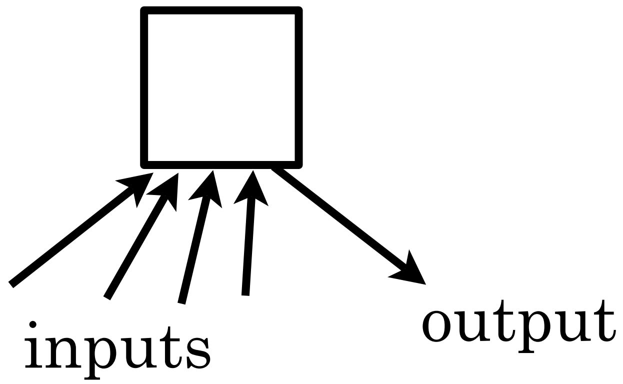
Investigate the proposed technique with one bit per message:

- Variable-check message consists of one bit
- Decoding maps found using "DMC Quantization Algorithm"
- Channel is AWGN quantized to 16 levels
- Compare with "Improved Modified Weighted Bit Flipping" (IMWBF) algorithm [Jiang et al, Comm Letters, 2005].

Check node map

- The map below is "obvious"
- But, it was obtained automatically, using optimization of mutual information

Number of 1's at input	Output
even	0
odd	1



Variable Node Map — SNR of 3 dB — Automatically Obtained Using DMC Quantizer

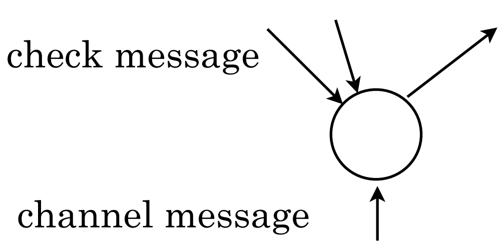
Degree 3 node

iteration 1

check message (number of 1's)	Channel Message															
	-8	**-7**	**-6**	**-5**	**-4**	**-3**	**-2**	**-1**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
2	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1

If message disagrees with channel, do not flip bit

If two messages disagree, use channel's hard decision



iteration 2-3

	-8	**-7**	**-6**	**-5**	**-4**	**-3**	**-2**	**-1**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
2	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1

As iterations increase, the influence of check message becomes stronger

The maps are almost always symmetrical

One-Bit Message Passing Decoding — Simulation

Channel

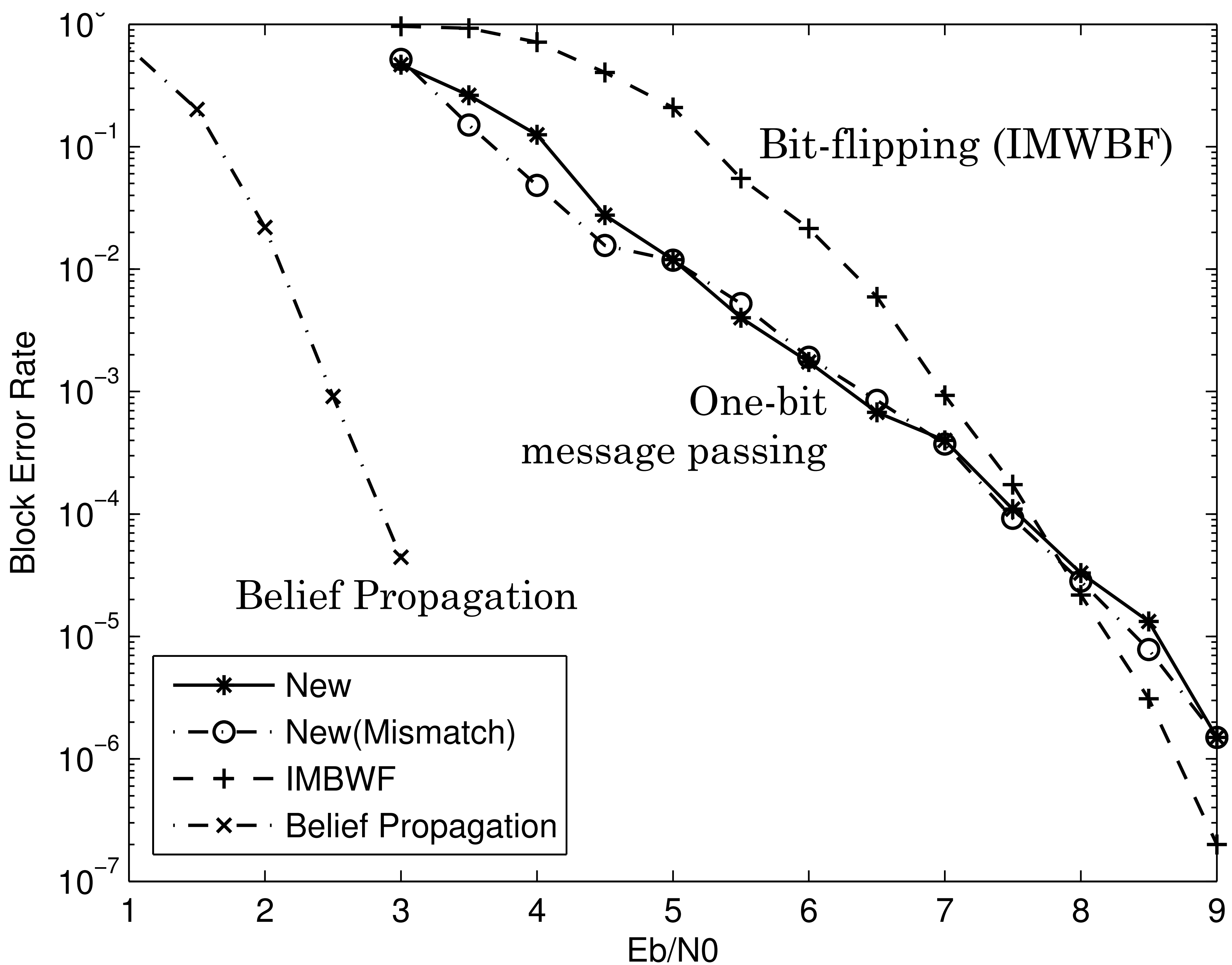
- AWGN quantized to 16 levels

Code

- Rate 1/2
- (816, 408) from MacKay's web site

One-bit message passing has about the same performance as bit flipping.

More complicated BP has better performance.



Complexity Comparison

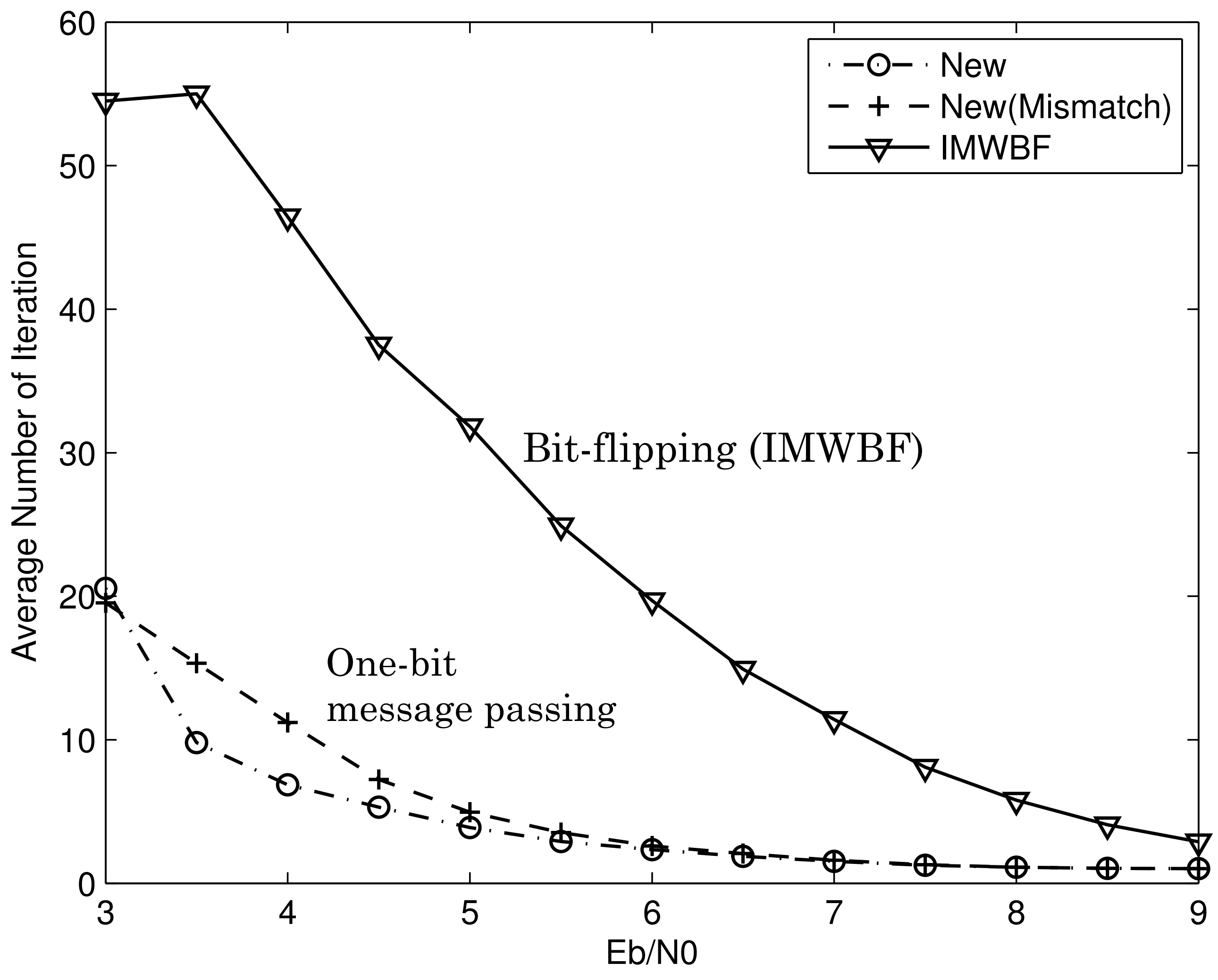
IMWBF algorithm must compute a flipping function:

$$w_m = \min_i |y_i|$$

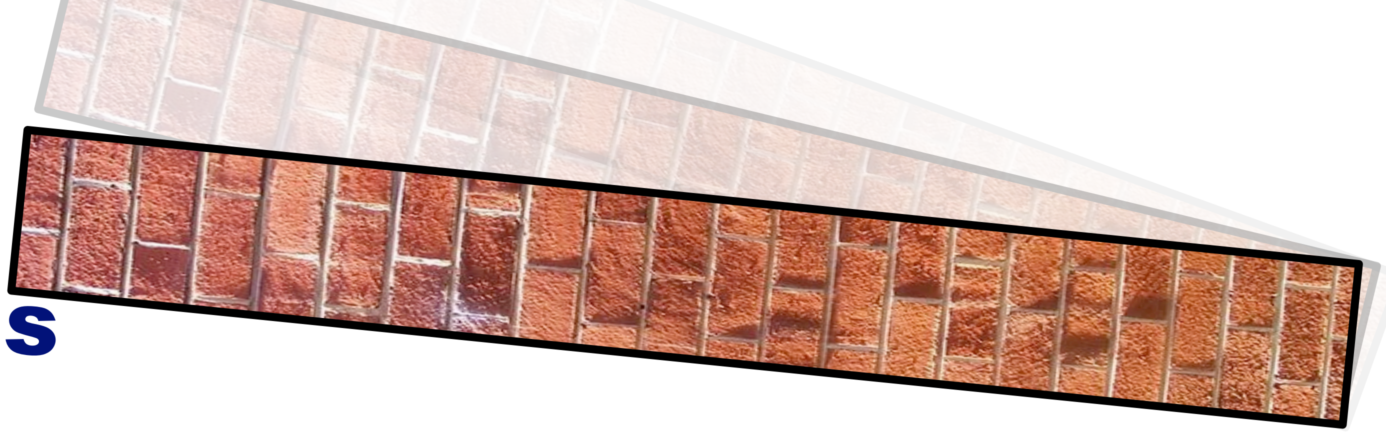
$$e = \sum_m (2s_m - 1) \cdot w_m - \alpha |y_n|$$

Same complexity as one iteration of min-sum decoding!

- At high SNR, only a few iterations needed.
- Flipping function is high fraction of total complexity.
- The two algorithms required about the same amount of computer time.



Conclusions



- There is a "wall" between information theory and VLSI implementation
- Quantization of messages is important for practical implementations
  - Reducing quantization can reduce power consumption, cost, etc.
- New perspective breaks the wall:
  - Implementation is an information theoretic problem
  - "DMC Quantization Algorithm" optimizes mutual information
- Already know:
  - How to optimally quantize channels
  - For infinite-length codes, reduce to 4 bits/message (from 6-7 bits)
- In this talk, showed:
  - For finite length codes, one-bit per message decoders perform as well as advanced bit-flipping algorithms
- Open questions:
  - Better understanding of performance/complexity trade-off
  - The role of symmetry
  - Implementation in VLSI