# On Optimum Decoding of Certain Product Codes

Enrico Paolini
University of Bologna, Italy

work with: Gianluigi Liva and Marco Chiani

Munich Workshop on Coding and Modulation
Munich, Germany, July 31st, 2015

# WS Topics

- Non-binary LDPC and Turbo codes
- Spatially-coupled codes
- Polar codes
- Lattice codes
- ✓ Decoding for short block lengths

# Outline

- ML decoding of product codes
- Reduced complexity decoder
- Numerical results
- Conclusion

# Product Codes

- Product code construction introduced in [Elias1954].
- Recent literature addresses efficient iterative decoding (e.g., [Tanner1981]–[Pyndiah1998]).
- [Wolf1978]: maximum-likelihood (ML) decoding can be performed very efficiently for some product codes.


- ⋆ [Elias1954] P. Elias, “Error-free coding,” *IRE T-IT*, 1954.
- ⋆ [Wolf1978] J. K. Wolf, “Efficient maximum likelihood decoding of linear block codes using a trellis,” *IEEE T-IT*, 1978.
- ⋆ [Tanner1981] R. Tanner, “A recursive approach to low complexity codes,” *IEEE T-IT*, 1981.
- ⋆ [Hagenauer1996] J. Hagenauer, E. Offer, and L. Papke, “Iterative decoding of binary block and convolutional codes,” *IEEE T-IT*, 1996.
- ⋆ [Pyndiah1998] R. Pyndiah, “Near optimum decoding of product codes: Block turbo codes,” *IEEE T-COM* 1998.

# Product Code Trellis

- [Wolf1978]: Trellis representation with a maximum number of states

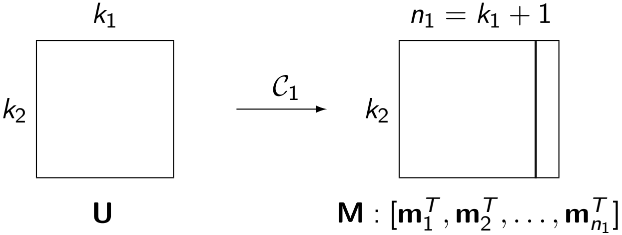
  $$\min\{2^{(n_1-k_1)k_2}, 2^{(n_2-k_2)k_1}\}$$

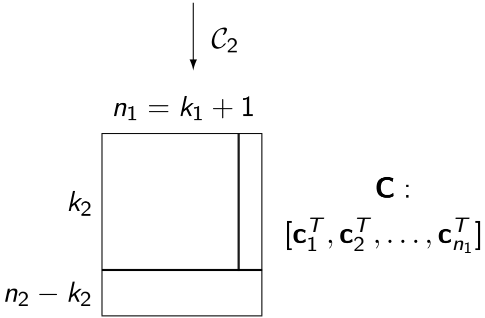
  per trellis section.
- Particularly beneficial when one component code has low dimension (e.g., small $k_2$) and one has a small number of parity bits (e.g., small $n_1 - k_1$).
- Particular case: the high rate code is a single parity-check code ($n_1 - k_1 = 1$). In this case:

  $$2^{k_2} \quad \text{states per trellis section.}$$

- Hereafter we focus on this class of product codes. Row code $\mathcal{C}_1$ is SPC, column code $\mathcal{C}_2$ is any linear block code.

# Some Notation (1/2)




$$\mathbf{m}_1^T + \mathbf{m}_2^T + \cdots + \mathbf{m}_{n_1}^T = \mathbf{0}^T$$

$$\mathbf{c} = [\mathbf{c}_1 \;\; \mathbf{c}_2 \;\; \cdots \;\; \mathbf{c}_{n_1}]$$

$$\mathbf{x} = [\mathbf{x}_1 \;\; \mathbf{x}_2 \;\; \cdots \;\; \mathbf{x}_{n_1}]$$

$$\mathbf{x}_i = \mathbf{1} - 2\mathbf{c}_i$$

$$\mathbf{y} = [\mathbf{y}_1 \;\; \mathbf{y}_2 \;\; \cdots \;\; \mathbf{y}_{n_1}]$$

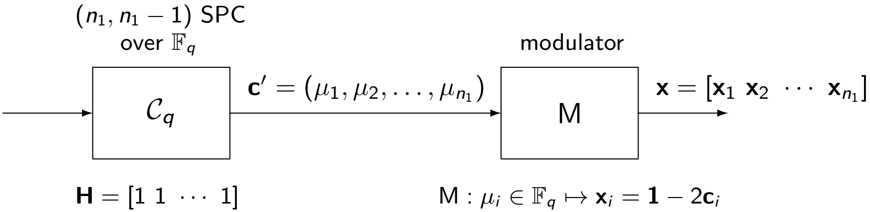
$$\mathbf{y}_i = \mathbf{x}_i + \boldsymbol{\nu}_i$$

# Some Notation (2/2)

- Length-$k_2$ vectors $\mathbf{m}_1^T, \mathbf{m}_2^T, \dots, \mathbf{m}_{n_1}^T$ regarded as the binary vector representations of the elements of finite field $\mathbb{F}_q$ with $q = 2^{k_2}$.
- $[\mu_1, \mu_2, \dots, \mu_{n_1}] := [\mathbf{m}_1^T, \mathbf{m}_2^T, \dots, \mathbf{m}_{n_1}^T]$, $\mu_i \in \mathbb{F}_q$, thus

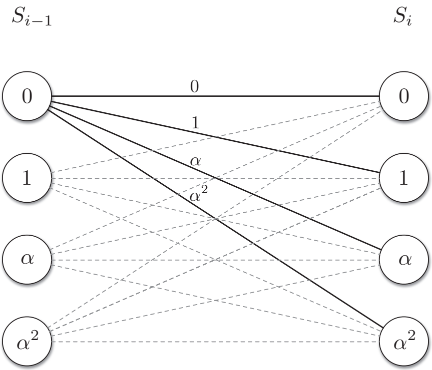
$$\mu_1 + \mu_2 + \cdots + \mu_{n_1} = 0$$

- Encoder (equivalent perspective):



# ML Decoding

- SPC code $\mathcal{C}_q$ Viterbi decoded over its trellis.
- The trellis comprises $q = 2^{k_2}$ states (apart from terminations). States are of subsequent layers are "fully connected". Example ($k_2 = 2$):

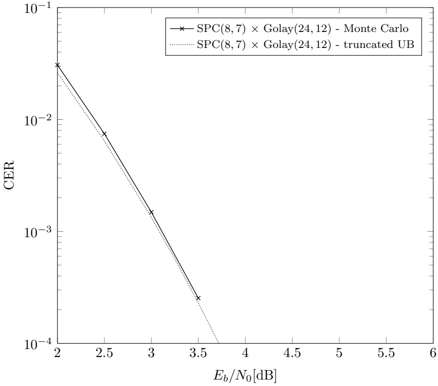



- Branch metrics $\langle \mathbf{x}_i, \mathbf{y}_i \rangle$.
- Label of edge from $S_{i-1} = s$ to $S_i = s'$: $s + s' \in \mathbb{F}_q$.

- Neglecting terminations, complexity proportional to $k_1 q^2 = k_1 2^{2k_2}$.

# Example

- Row code: $(8, 7)$ SPC code. Column code: $(24, 12)$ Golay code.
- $n = 192$, $k = 84$, $d = 16$, $A_{\text{min}} = 28 \times 759 = 21252$.
- Number of states: $2^{12}$. Number of edges per trellis section: $2^{24}$.




- **Q**: Possible to reduce complexity while preserving performance?

# Symbol-Wise MAP Decoding

- To reduce complexity:
  - Switch to symbol-wise optimum MAP decoding:

$$\hat{\mu}_i = \arg\max_{\omega \in \mathbb{F}_q} \Pr\{\mu_i = \omega | \mathbf{y}\}$$

  - Use fast Fourier transform.
- Using BCJR we have

$$L_i(\omega) := \Pr\{\mu_i = \omega | \mathbf{y}\} = \sum_{s,s' : s+s'=\omega} \varphi_{i-1}(s)\gamma_i(s,s')\beta_i(s')$$

with standard meaning for the forward ($\varphi$), backward ($\beta$), and branch transition ($\gamma$) metrics.

# Branch Metric

- Defining $s' = s + \omega$, the branch transition metric may be computed as

$$\begin{aligned} \gamma_i(s, s') &\propto (2\pi\sigma^2)^{n_2/2} \exp\left(-\frac{1}{2\sigma^2}\langle \mathbf{y}_i, \mathsf{M}(\omega)\rangle\right) \\ &=: \gamma_i(s + s') \\ &=: \gamma_i(\omega) \end{aligned}$$

- An inner product of length-$n_2$ vectors for each $\omega \in \mathbb{F}_q$.
- Complexity of branch metric calculation is $\mathcal{O}(k_1 k_2 2^{k_2})$.

# APP Calculation (1/2)

- As usual we have

$$\varphi_i(s) = \sum_{s'} \varphi_{i-1}(s')\gamma_i(s', s) \quad \text{and} \quad \beta_i(s) = \sum_{s'} \beta_{i+1}(s')\gamma_{i+1}(s, s')$$

- Let

$$\boldsymbol{\varphi}_i = (\varphi_i(0), \varphi_i(1), \ldots, \varphi_i(\alpha^{q-2}))$$
$$\boldsymbol{\beta}_i = (\beta_i(0), \beta_i(1), \ldots, \beta_i(\alpha^{q-2}))$$
$$\boldsymbol{\gamma}_i = (\gamma_i(0), \gamma_i(1), \ldots, \gamma_i(\alpha^{q-2}))$$
$$\mathbf{L}_i = (L_i(0), L_i(1), \ldots, L_i(\alpha^{q-2}))$$

then

$$\boldsymbol{\varphi}_i = \boldsymbol{\varphi}_{i-1} \circledast \boldsymbol{\gamma}_i$$
$$\boldsymbol{\beta}_i = \boldsymbol{\beta}_{i+1} \circledast \boldsymbol{\gamma}_{i+1}$$

where $\circledast$ denotes convolution.

# APP Calculation (2/2)

- Next

$$L_i(\omega) = \sum_{s,s':s+s'=\omega} \varphi_{i-1}(s)\gamma_i(s,s')\beta_i(s') = \gamma_i(\omega)\sum_s \varphi_{i-1}(s)\beta_i(s+\omega)$$

so ($\cdot$ denotes element-wise multiplication)

$$\mathbf{L}_i = \boldsymbol{\gamma}_i \cdot (\boldsymbol{\varphi}_{i-1} \circledast \boldsymbol{\beta}_i) = \boldsymbol{\gamma}_i \cdot \left(\left(\circledast_{j=1}^{i-1}\boldsymbol{\gamma}_j\right) \circledast \left(\circledast_{j=i+1}^{n_1}\boldsymbol{\gamma}_j\right)\right)$$

# Using FFT

- To calculate $\mathbf{L}_i$ we have to take the convolution of all vectors $\gamma_j$ but $\gamma_i$.
- In principle complexity of convolution scales as $\mathcal{O}(q^2)$.
- However, complexity reduced to $\mathcal{O}(q \log_2 q)$ by applying fast Fourier transform on finite Abelian groups (in this case equal to Hadamard transform):

$$\mathbf{L}_i = \gamma_i \cdot \mathcal{H}\left(\left(\mathop{\cdot}_{j=1}^{i-1} \mathcal{H}(\gamma_j)\right) \cdot \left(\mathop{\cdot}_{j=i+1}^{n_1} \mathcal{H}(\gamma_j)\right)\right)$$

- Complexity $\mathcal{O}(k_1 2^{2k_2})$ under Viterbi decoding is turned into $\mathcal{O}(k_1 k_2 2^{k_2})$.

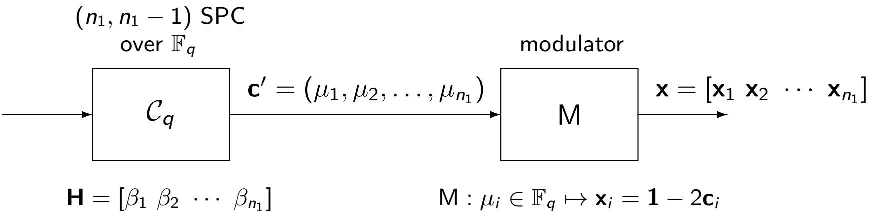
# Improving Multiplicity of Minimum Weight Codewords

- We have

$$d = 2d_2 \quad \text{and} \quad A_{\min} = \frac{n_1(n_1-1)}{2} A_{\min,2}$$

- To preserve $d = 2d_2$ while reducing $A_{\min}$, we replace $\mathbf{H} = [1\ 1\ \cdots\ 1]$ with $\mathbf{H}' = [\beta_1\ \beta_2\ \cdots\ \beta_{n_1}]$ with $\beta_i \in \mathbb{F}_q \setminus \{0\}$.
- Upon a uniformly random choice of $\beta_1, \beta_2, \ldots, \beta_{n_1}$ we expect

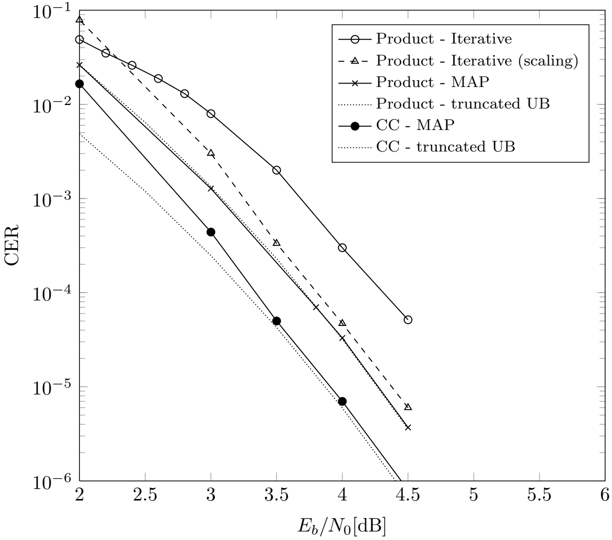
$$\bar{A}'_{\min} = \frac{n_1(n_1-1)}{2} \frac{A^2_{\min,2}}{2^{k_2}-1}$$



# Numerical Results

- Consider again the $(8,7)$ SPC $\times$ $(24,12)$ Golay code.
- Decoded by:
  - ▷ The BCJR algorithm to the component code trellises, iterating the soft information exchange (50 iterations max);
  - ▷ The BCJR algorithm, by weighting the soft-output of each component decoder by a factor $1/2$ [Pyndiah1998];
  - ▷ The symbol-wise MAP decoder.
- Additionally, we consider a second code (CC) with the same parameters but designed using the $\mathbf{H}' = [\beta_1 \ \beta_2 \ \cdots \ \beta_{n_1}]$ matrix approach.

# Numerical Results



# Conclusion

- Optimum decoding investigated for product codes given by concatenation of a binary SPC code with a low-dimension binary linear block code.
- Decoding complexity can be reduced further with respect to block-wise ML decoding by approaching the problem as a symbol-wise MAP decoding.
- A generalization of the code construction, enjoying the same low-complexity decoding principle is presented and analyzed, achieving additional coding gains.

G. Liva, E. Paolini, M. Chiani, “On optimum decoding of certain product codes,” *IEEE Commun. Lett.*, vol. 18, no. 6, pp. 905–908, June 2014.