Reinforcement Learning for Channel Coding: Learned Bit-Flipping Decoding

Fabrizio Carpi[1], Christian Häger[2],
Marco Martalò[3], Riccardo Raheli[3], and Henry D. Pfister[4]

[1]New York University, [2]Chalmers University of Technology,
[3]University of Parma, [4]Duke University,

57th Annual Allerton Conference on Communication, Control, and Computing

Allerton Park and Retreat Center, Monticello, IL – September 27th, 2019

Introduction and Motivation

medium.com

Classification/Regression problem

Supervised learning

Parameterized decoders
(message-passing, syndrome-based)

Decoding of error-correcting codes

Introduction and Motivation

medium.com

Decoding of error-correcting codes

Classification/Regression problem

Supervised learning

Parameterized decoders (message-passing, syndrome-based)

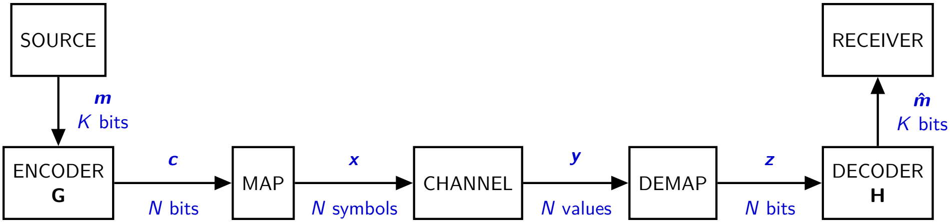
Decision-making problem

Reinforcement learning

?

google.com

Linear Block Codes



- $\mathcal{C}$ is a linear block code (N, K) described by a $M \times N$ parity check matrix $\mathbf{H}$
- Syndrome: $\boldsymbol{s} = \mathbf{H}\boldsymbol{z}$, where $\boldsymbol{z} \in \mathbb{F}_2^N$ is the received word
- Any codeword $\boldsymbol{c} \in \mathcal{C}$ satisfies $\mathbf{H}\boldsymbol{c} = \mathbf{0}$

Decoding Algorithms with Sequential Decision Processes

➡ Bit-Flipping (BF) decoding[1] $\leftarrow$ case study of this paper

- Basic idea: flip a bit that maximizes number of correct parity checks (on BSC)
  It can also be extended to AWGN channel (Weighed BF, WBF)

[1] W. Ryan and S. Lin, *Channel Codes Classical and Modern*. Cambridge University Press, 2009.

[2] G. Elidan, I. McGraw, and D. Koller, "Residual belief propagation: Informed scheduling for asynchronous message passing," in *Proc. Conf. Uncertainty in AI (UAI)*, Boston, MA, 2006.

[3] C. Häger and H. D. Pfister, "Approaching miscorrection-free performance of product codes with anchor decoding," *IEEE Trans. Commun.*, vol. 66, no. 7, pp. 2797–2808, July 2018.

Decoding Algorithms with Sequential Decision Processes

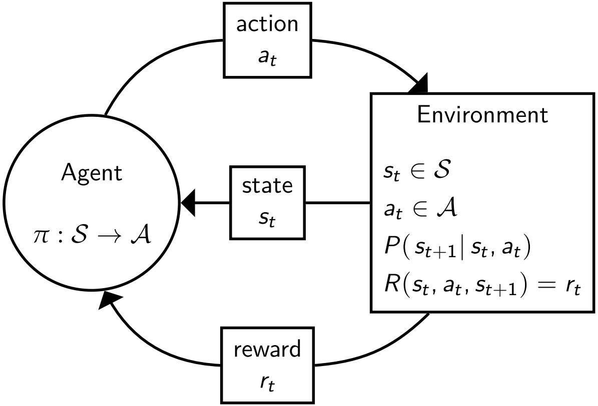
- Bit-Flipping (BF) decoding[1] $\leftarrow$ case study of this paper
  - Basic idea: flip a bit that maximizes number of correct parity checks (on BSC)
    It can also be extended to AWGN channel (Weighed BF, WBF)
- Residual Belief Propagation[2]
- Anchor Decoding of Product/Staircase Codes[3]

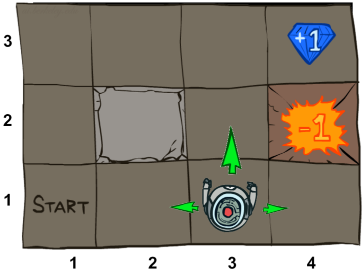
[1] W. Ryan and S. Lin, *Channel Codes Classical and Modern*. Cambridge University Press, 2009.

[2] G. Elidan, I. McGraw, and D. Koller, "Residual belief propagation: Informed scheduling for asynchronous message passing," in *Proc. Conf. Uncertainty in AI (UAI)*, Boston, MA, 2006.

[3] C. Häger and H. D. Pfister, "Approaching miscorrection-free performance of product codes with anchor decoding," *IEEE Trans. Commun.*, vol. 66, no. 7, pp. 2797–2808, July 2018.

Markov Decision Process (MDP)





ai.berkeley.edu

Q-Learning

Observable states and rewards ⇒ Solve with RL ⇒ Q-Learning[4]

[4] C. J. C. H. Watkins, "Learning from delayed rewards," Ph.D. dissertation, King's College, Cambridge, UK, 1989.

[5] C. J. C. H. Watkins, P. Dylan, "Technical Note: Q-Learning," *Machine Learning*, vol. 8, no. 3, pp. 279–292, May 1992.

Q-Learning

Observable states and rewards $\Rightarrow$ Solve with RL $\Rightarrow$ Q-Learning[4]

Policy

$$Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R} \qquad\qquad \pi^*(s) = \arg\max_{a \in \mathcal{A}} Q(s, a)$$

[4] C. J. C. H. Watkins, "Learning from delayed rewards," Ph.D. dissertation, King's College, Cambridge, UK, 1989.

[5] C. J. C. H. Watkins, P. Dylan, "Technical Note: Q-Learning," *Machine Learning*, vol. 8, no. 3, pp. 279–292, May 1992.

Q-Learning

Observable states and rewards $\Rightarrow$ Solve with RL $\Rightarrow$ Q-Learning[4]

Policy

$$Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R} \qquad \pi^*(s) = \arg\max_{a \in \mathcal{A}} Q(s, a)$$

Update (for learning rate α and discount factor γ)

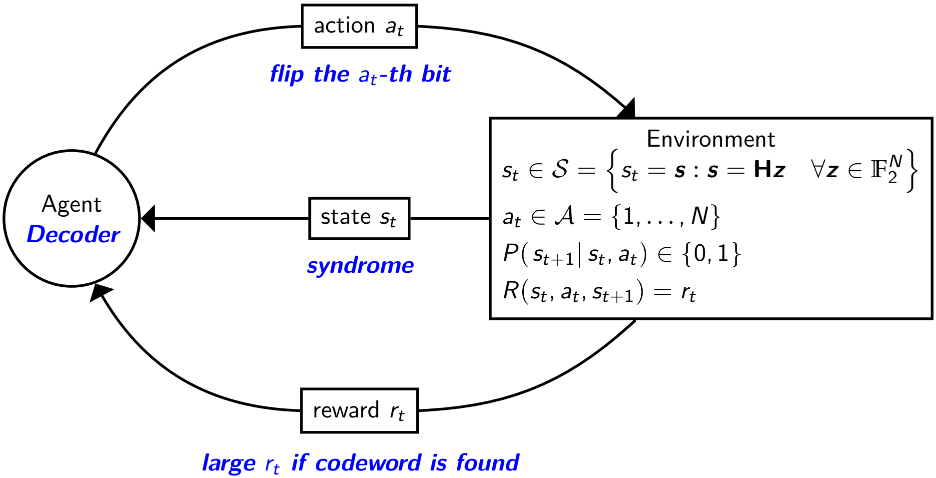
$$Q(s_t, a_t) \leftarrow (1 - \alpha) Q(s_t, a_t) + \alpha \left[r_t + \gamma \max_{a' \in \mathcal{A}} Q(s_{t+1}, a') \right]$$

Convergence[5]: if $|r_t| < \infty$ and $0 < \alpha, \gamma < 1$, then $Q(s, a) \xrightarrow{t \to \infty} Q^*(s, a)$

[4] C. J. C. H. Watkins, "Learning from delayed rewards," Ph.D. dissertation, King's College, Cambridge, UK, 1989.

[5] C. J. C. H. Watkins, P. Dylan, "Technical Note: Q-Learning," *Machine Learning*, vol. 8, no. 3, pp. 279–292, May 1992.

Bit-Flipping Interpreted as an MDP



Reward strategy

- Maximum likelihood decoding (λ_n is the log-likelihood ratio for the n-th bit)

$$\arg\max_{\boldsymbol{c}\in\mathcal{C}} \prod_{n=1}^{N} P_{Y_n|C_n}(y_n|c_n) = \cdots = \arg\max_{\boldsymbol{e}:\mathbf{H}\boldsymbol{e}=\boldsymbol{s}} \sum_{n=1}^{N} -e_n|\lambda_n|$$

- Considering the RL BF multi-stage process

$$\arg\max_{\tau, a_1,\ldots,a_\tau : \sum_{t=1}^{\tau} \boldsymbol{h}_{a_t}=\boldsymbol{s}} \sum_{t=1}^{\tau} -|\lambda_{a_t}|$$

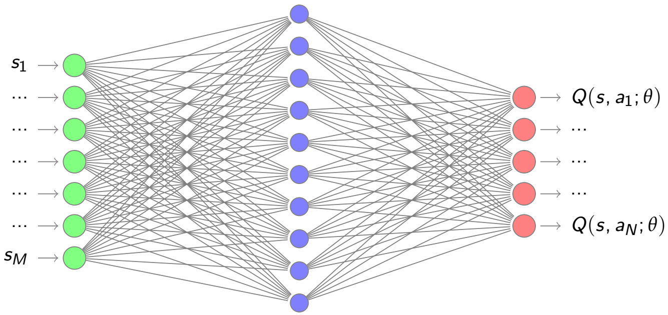
➡ We propose to interpret $-|\lambda_{a_t}|$ as a reward

$$R(s_t, a_t, s_{t+1}) = \begin{cases} -c|\lambda_{a_t}| + 1 & \text{if } s_{t+1} = \mathbf{0} \\ -c|\lambda_{a_t}| & \text{otherwise} \end{cases}$$

Q function

- For short codes: Q-table containing $Q(s, a)$ may be feasible (size $|\mathcal{S}| \cdot |\mathcal{A}|$)

➡ For large $\mathcal{S} \times \mathcal{A}$: use a neural network (NN) to approximate $Q(s, a) \approx Q(s, a; \theta)$



Exploration strategies

- Standard: ε-greedy exploration

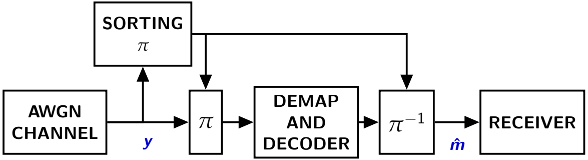
$$a = \begin{cases} \text{unif. random over } \mathcal{A} & \text{w.p. } \varepsilon \\ \arg\max_{a'} Q(s, a') & \text{w.p. } 1 - \varepsilon \end{cases}$$

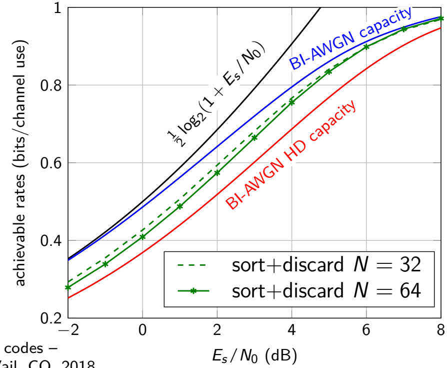
➡ We propose: $(\varepsilon, \varepsilon_{\mathrm{g}})$-goal exploration — where $\mathrm{supp}(\boldsymbol{e}) \triangleq \{i \in [N] \mid e_i = 1\}$

$$a = \begin{cases} \text{unif. random over } \mathcal{A} & \text{w.p. } \varepsilon \\ \text{unif. random over } \mathrm{supp}(\boldsymbol{e}) & \text{w.p. } \varepsilon_{\mathrm{g}} \\ \arg\max_{a'} Q(s, a') & \text{w.p. } 1 - \varepsilon - \varepsilon_{\mathrm{g}} \end{cases}$$

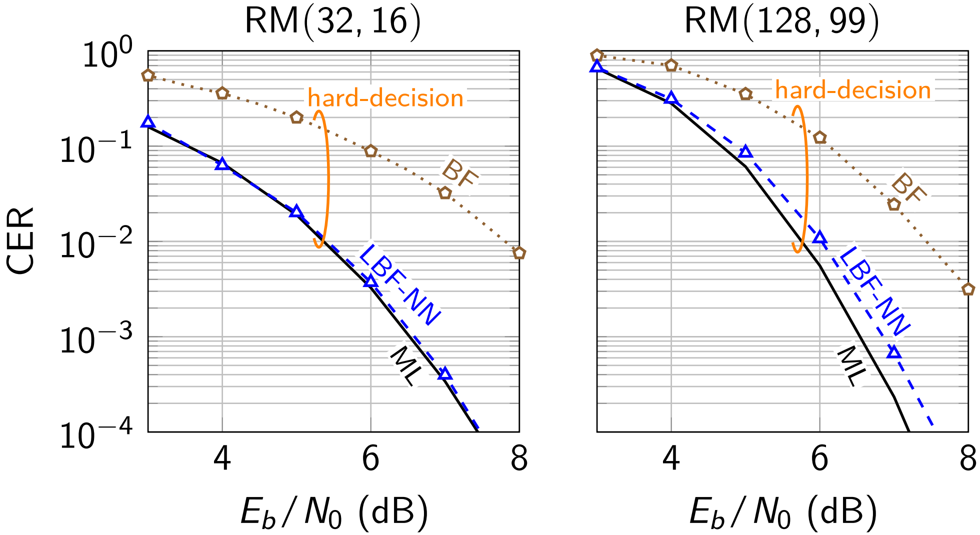
Decoding with Reliability-based Sorting

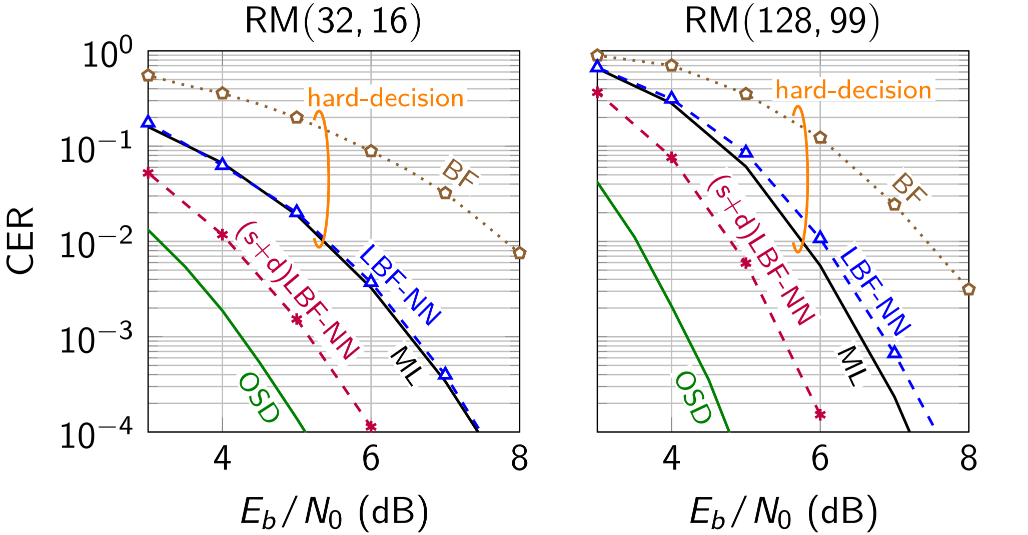
- Permutation automorphism group: $\mathrm{PAut}(\mathcal{C}) \triangleq \{\pi \in \mathcal{S}_N \mid \boldsymbol{x}^{\pi} \in \mathcal{C}, \forall \boldsymbol{x} \in \mathcal{C}\}$
  - Sorting strategy (BCH)[6]: the first K bits are the *most reliable*
- ➡ For RM, we move *least reliable* bits to positions $\{0, 1, 2, 4, \ldots, 2^{m-1}\} \triangleq \mathcal{B}$
  - Approximate Sort and Discard (s+d): sort the received bits + discard LLRs



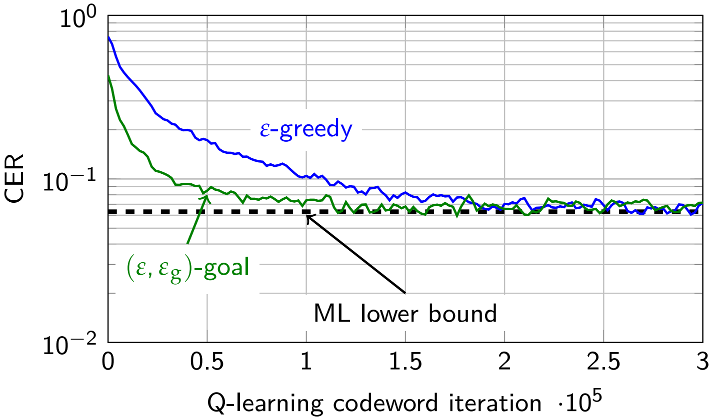


[6] A. Bennatan, Y. Choukroun, and P. Kisilev, "Deep learning for decoding of linear codes – a syndrome-based approach," in *Proc. IEEE Int. Symp. Information Theory (ISIT)*, Vail, CO, 2018.


RM(32, 16)
RM(128, 99)
CER
10^0
10^{-1}
10^{-2}
10^{-3}
10^{-4}
hard-decision
BF
LBF-NN
ML
4
6
8
E_b / N_0 (dB)


RM(32, 16)
RM(128, 99)
CER
10^0
10^{-1}
10^{-2}
10^{-3}
10^{-4}
4
6
8
E_b/N_0 (dB)
hard-decision
BF
LBF-NN
(s+d)LBF-NN
ML
OSD

Exploration Strategies and Convergence

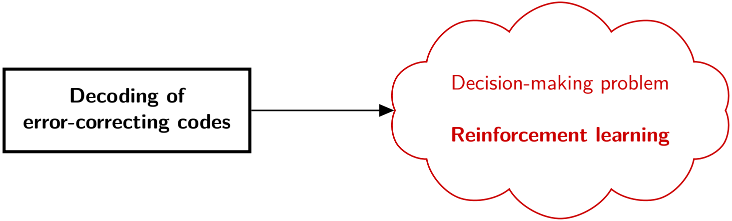


$RM(32,16)$ on BSC
@ $E_b / N_0 = 4\,dB$

ε-greedy: $\varepsilon = 0.9$

$(\varepsilon, \varepsilon_g)$-goal: $\begin{cases} \varepsilon = 0.6 \\ \varepsilon_g = 0.3 \end{cases}$

Conclusion



✓ RL framework for BF decoding

- ➡ BF is mapped to an MDP
  - The objective is ML decoding
  - Exploration can be biased towards "good" actions to speed-up convergence

- ➡ Table Q-learning and NN-based provide performance–complexity trade-offs

---

Simulation code available **Github: fabriziocarpi/RLdecoding**

Thank you! Q&A?

Reinforcement Learning for Channel Coding: Learned Bit-Flipping Decoding

Fabrizio Carpi[1], Christian Häger[2],
Marco Martalò[3], Riccardo Raheli[3], and Henry D. Pfister[4]

[1]New York University, [2]Chalmers University of Technology,
[3]University of Parma, [4]Duke University,

57th Annual Allerton Conference on Communication, Control, and Computing
Allerton Park and Retreat Center, Monticello, IL – September 27th, 2019