Reinforcement Learning for Channel Coding: Learned Bit-Flipping Decoding

Fabrizio Carpi\textsuperscript{1}, Christian Häger\textsuperscript{2}, Marco Martalò\textsuperscript{3}, Riccardo Raheli\textsuperscript{3}, and Henry D. Pfister\textsuperscript{4}

\textsuperscript{1}New York University, \textsuperscript{2}Chalmers University of Technology, \textsuperscript{3}University of Parma, \textsuperscript{4}Duke University,

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

Background
- Channel Coding
- Reinforcement Learning

Case Study: Reinforcement Learning for Bit-Flipping Decoding
- Problem Formulation
- Code Automorphism

Results
- Error Rate Performance
- Convergence Improvements

Conclusion
Introduction

Background

Case Study: Reinforcement Learning for Bit-Flipping Decoding

Results

Conclusion
Introduction and Motivation

Decoding of error-correcting codes

Parameterized decoders (message-passing, syndrome-based)

Classification/Regression problem

Supervised learning
Introduction and Motivation

Decoding of error-correcting codes

Classification/Regression problem

Supervised learning

Parameterized decoders (message-passing, syndrome-based)

Decision-making problem

Reinforcement learning

or

medium.com

google.com

F. Carpi, C. Häger, M. Martalò, R. Raheli, H.D. Pfister

Reinforcement Learning for Channel Coding: Learned Bit-Flipping Decoding
1 Introduction

2 Background

3 Case Study: Reinforcement Learning for Bit-Flipping Decoding

4 Results

5 Conclusion
Linear Block Codes

- $C$ is a linear block code $(N, K)$ described by a $M \times N$ parity check matrix $H$
- Syndrome: $s = Hz$, where $z \in \mathbb{F}_2^N$ is the received word
- Any codeword $c \in C$ satisfies $Hc = 0$
Decoding Algorithms with Sequential Decision Processes

→ Bit-Flipping (BF) decoding\(^1\) ← 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)

---


Decoding Algorithms with Sequential Decision Processes

- Bit-Flipping (BF) decoding\(^1\) ← 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\(^3\)

---


Markov Decision Process (MDP)

Agent
\[ \pi : S \to A \]

Environment
- state \( s_t \in S \)
- action \( a_t \in A \)
- transition probability \( P(s_{t+1} \mid s_t, a_t) \)
- reward \( R(s_t, a_t, s_{t+1}) = r_t \)
Q-Learning

Observable states and rewards ⇒ Solve with RL ⇒ Q-Learning


Q-Learning

Observable states and rewards ⇒ Solve with RL ⇒ Q-Learning

Policy

\[ Q : S \times A \rightarrow \mathbb{R} \]

\[ \pi^* (s) = \arg \max_{a \in A} Q(s, a) \]

---


Q-Learning

Observable states and rewards ⇒ Solve with RL ⇒ Q-Learning

Policy

\[ Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R} \]

\[ \pi^*(s) = \arg \max_{a \in \mathcal{A}} Q(s, a) \]

Update (for learning rate \( \alpha \) and discount factor \( \gamma \))

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

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

---


1 Introduction

2 Background

3 Case Study: Reinforcement Learning for Bit-Flipping Decoding

4 Results

5 Conclusion
Bit-Flipping Interpreted as an MDP

**Problem Formulation**

**Agent**

**Decoder**

**Environment**

- **State** $s_t \in S = \{ s_t = s : s = Hz \ \forall z \in \mathbb{F}_2^N \}$
- **Action** $a_t \in A = \{1, \ldots, N\}$
- **Transition** $P(s_{t+1} | s_t, a_t) \in \{0, 1\}$
- **Reward** $R(s_t, a_t, s_{t+1}) = r_t$

**Action** $a_t$

*flip the $a_t$-th bit*

**State** $s_t$

**Reward** $r_t$

*large $r_t$ if codeword is found*
Reward strategy

- Maximum likelihood decoding ($\lambda_n$ is the log-likelihood ratio for the $n$-th bit)

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

- Considering the RL BF multi-stage process

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

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

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

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

$\rightarrow$ For large $S \times A$: use a neural network (NN) to approximate $Q(s, a) \approx Q(s, a; \theta)$
Exploration strategies

- Standard: $\varepsilon$-greedy exploration

\[
a = \begin{cases} 
\text{unif. random over } A & \text{w.p. } \varepsilon \\
\text{arg max}_{a'} Q(s, a') & \text{w.p. } 1 - \varepsilon 
\end{cases}
\]

- We propose: $(\varepsilon, \varepsilon_g)$-goal exploration — where $\text{supp}(\mathbf{e}) \triangleq \{ i \in [N] | e_i = 1 \}$

\[
a = \begin{cases} 
\text{unif. random over } A & \text{w.p. } \varepsilon \\
\text{unif. random over } \text{supp}(\mathbf{e}) & \text{w.p. } \varepsilon_g \\
\text{arg max}_{a'} Q(s, a') & \text{w.p. } 1 - \varepsilon - \varepsilon_g 
\end{cases}
\]
Decoding with Reliability-based Sorting

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

\[ \frac{1}{2} \log_2(1 + \frac{E_s}{N_0}) \]

\( E_s/N_0 \) (dB)

achievable rates (bits/channel use)

\( \text{sort+discard } N = 32 \)
\( \text{sort+discard } N = 64 \)

1 Introduction

2 Background

3 Case Study: Reinforcement Learning for Bit-Flipping Decoding

4 Results

5 Conclusion
Results

Error Rate Performance

\begin{align*}
\text{CER} & \quad \text{RM}(32, 16) \\
E_b / N_0 (\text{dB}) & \quad 4 \quad 6 \quad 8
\end{align*}

\begin{align*}
\text{CER} & \quad \text{RM}(128, 99) \\
E_b / N_0 (\text{dB}) & \quad 4 \quad 6 \quad 8
\end{align*}
Results

Error Rate Performance

|ハードデシジョン | OSD | ML | BF | LBF-NN | (s+d)LBF-NN 
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Eb/N0 (dB)</td>
<td>4</td>
<td>6</td>
<td>8</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>CER</td>
<td>10^0</td>
<td>10^{-1}</td>
<td>10^{-2}</td>
<td>10^{-3}</td>
<td>10^{-4}</td>
</tr>
</tbody>
</table>

RM(32, 16)

RM(128, 99)
Exploration Strategies and Convergence

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

- $\varepsilon$-greedy: $\varepsilon = 0.9$

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

Background

Case Study: Reinforcement Learning for Bit-Flipping Decoding

Results

Conclusion
Conclusion

Decoding of error-correcting codes

Decision-making problem

Reinforcement learning

☑️ 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\textsuperscript{1}, Christian Häger\textsuperscript{2}, Marco Martalò\textsuperscript{3}, Riccardo Raheli\textsuperscript{3}, and Henry D. Pfister\textsuperscript{4}

\textsuperscript{1}New York University, \textsuperscript{2}Chalmers University of Technology, \textsuperscript{3}University of Parma, \textsuperscript{4}Duke University,