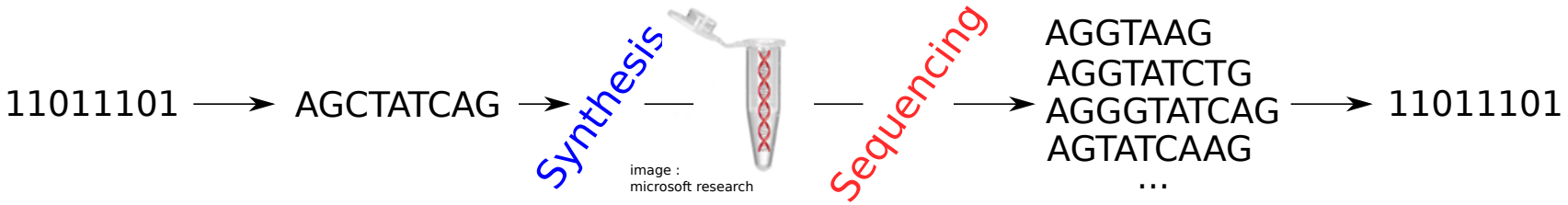
Channel Coding for storage on DNA molecules

Elsa Dupraz (IMT Atlantique/Lab-STICC)

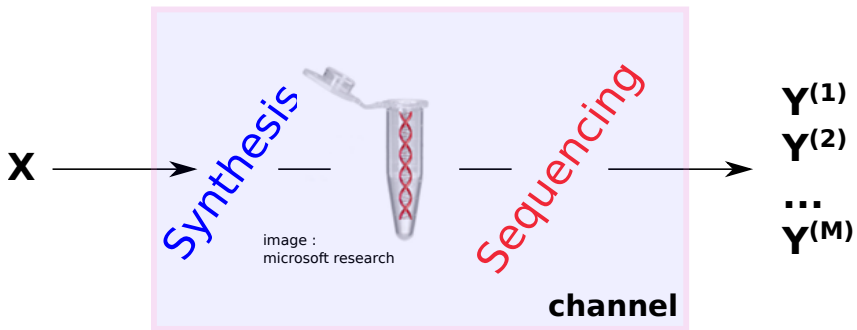
June 24th, 2025

The DNA data storage workflow



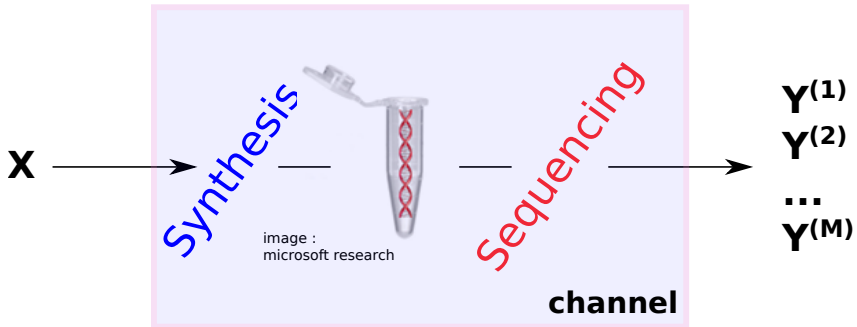
- Synthesis (chemical) is currently very expensive
- Sequencing (nanopore) introduces a large amount of errors

The DNA data storage workflow



- The (synthesis + sequencing) part can be seen as a **channel** which introduces errors in the read sequences

The DNA data storage workflow



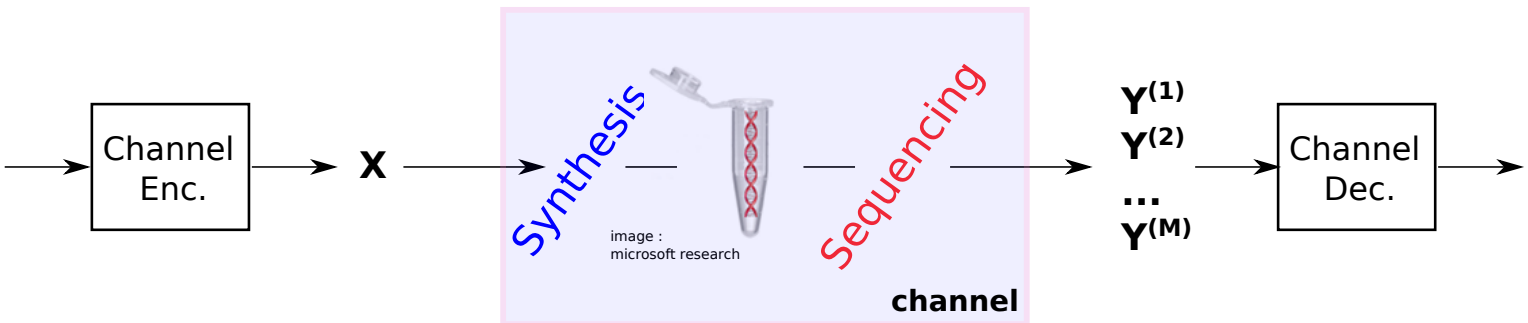
- The (synthesis + sequencing) part can be seen as a **channel** which introduces errors in the read sequences

- This channel introduces three types of errors

  ex: AGCTATCAG

  - **Substitutions**: AGCTAT**T**AG
  - **Deletions**: AGCTAT~~C~~AG
  - **Insertions**: AGCTATC**G**AG

Coding scheme for DNA data storage



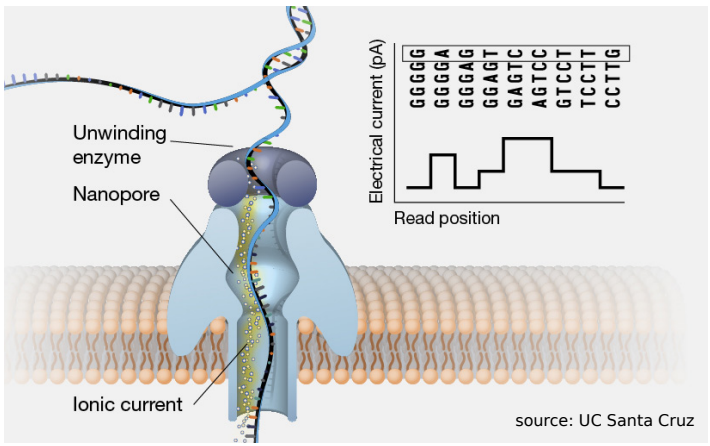
In this presentation

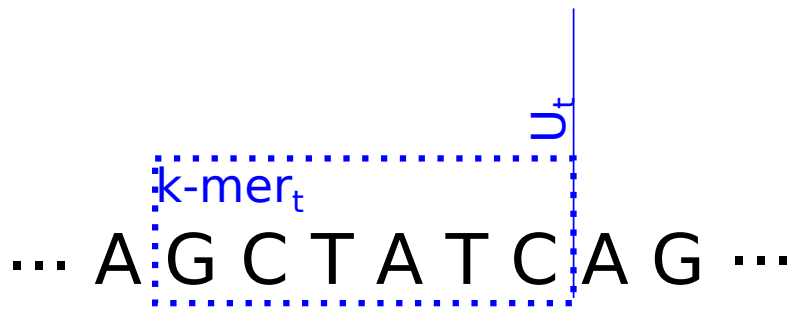
- How can we **model** errors introduced by the DNA storage process?
- How can we **correct** errors introduced by the DNA storage channel?
- How can we use **multiple reads** to improve error correction?

Outline

Principle of nanopore sequencing

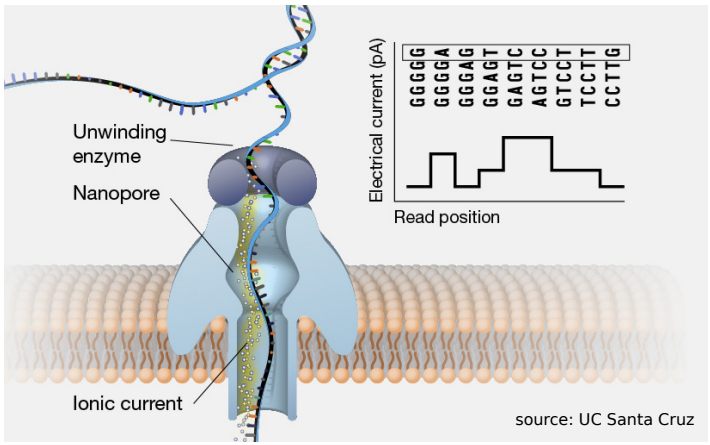
Nanopore sequencing:

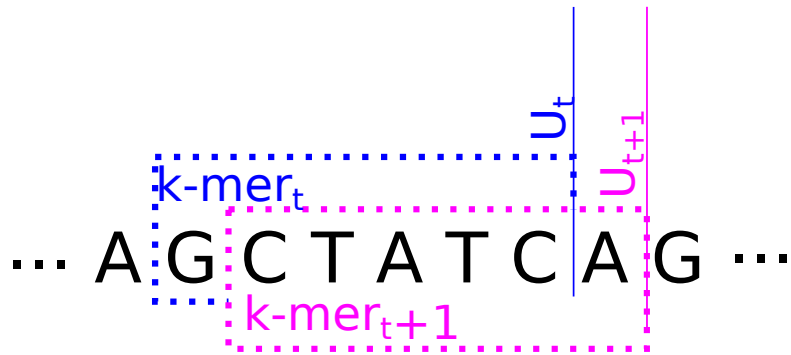




Principle of nanopore sequencing

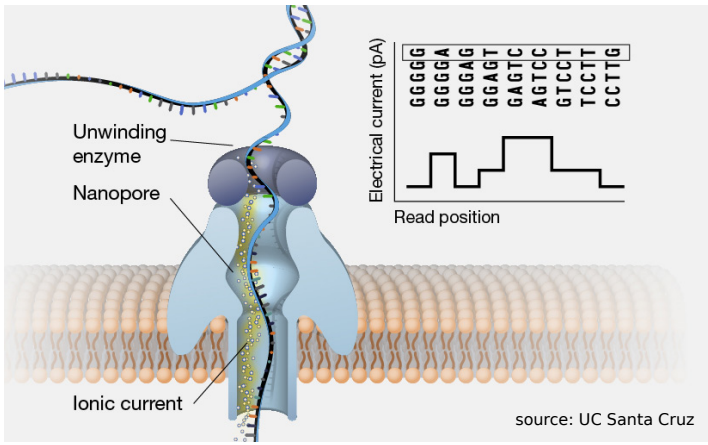
Nanopore sequencing:

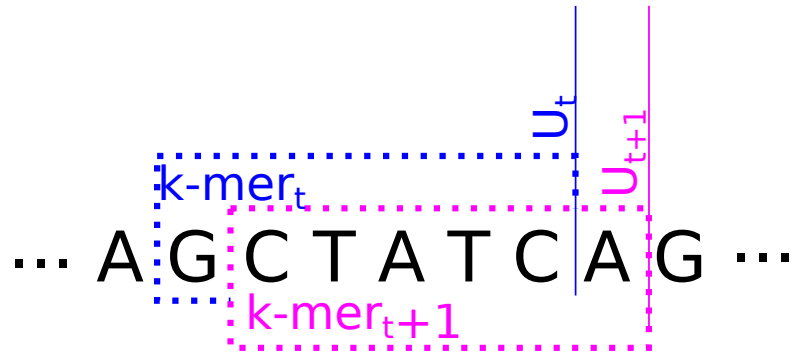




Principle of nanopore sequencing

Nanopore sequencing:

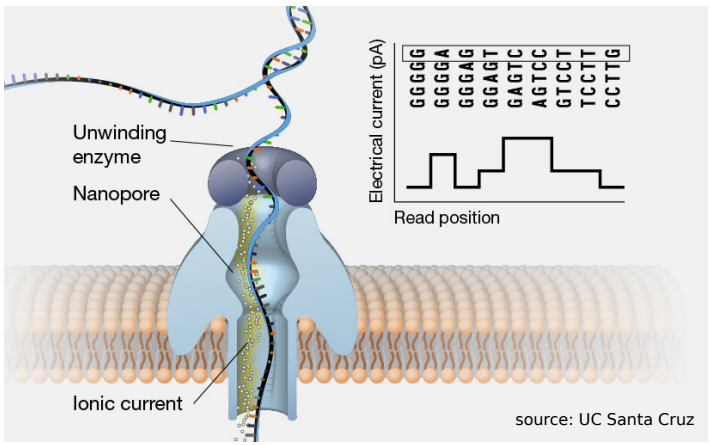


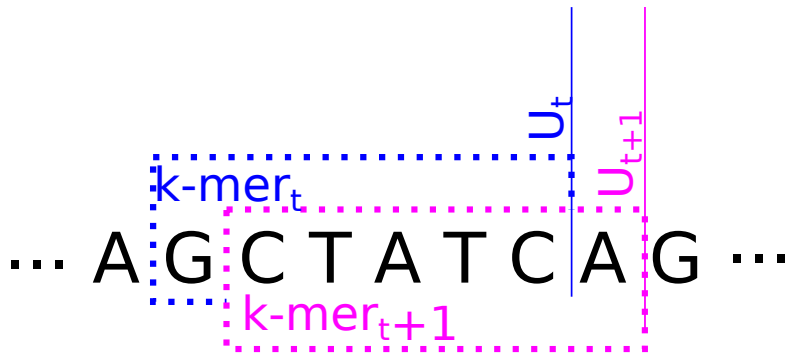


Basecaller: currents $(..U_{t-1}, U_t, U_{t+1}...) \rightarrow$ bases $(..Y_{t-1}, Y_t, Y_{t+1}...)$

Principle of nanopore sequencing

Nanopore sequencing:





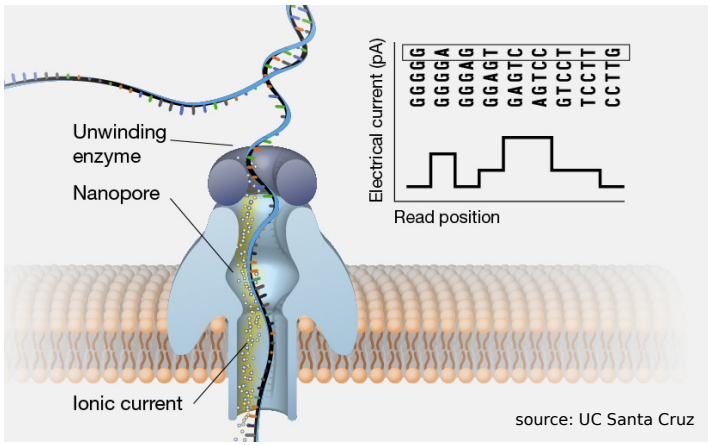
Basecaller: currents $(..U_{t-1}, U_t, U_{t+1}...) \rightarrow$ bases $(..Y_{t-1}, Y_t, Y_{t+1}...)$

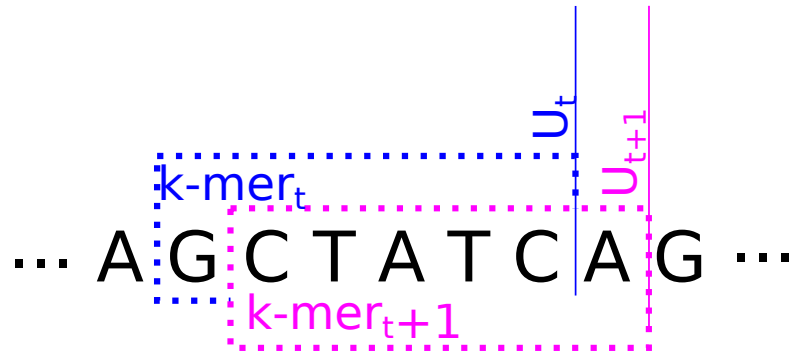
Model assumptions:

- Error probability depends on the underlying **k-mer**
- Error probability depends on the **previous error event**

Proposed channel model with memory[1]

- Nanopore sequencing:



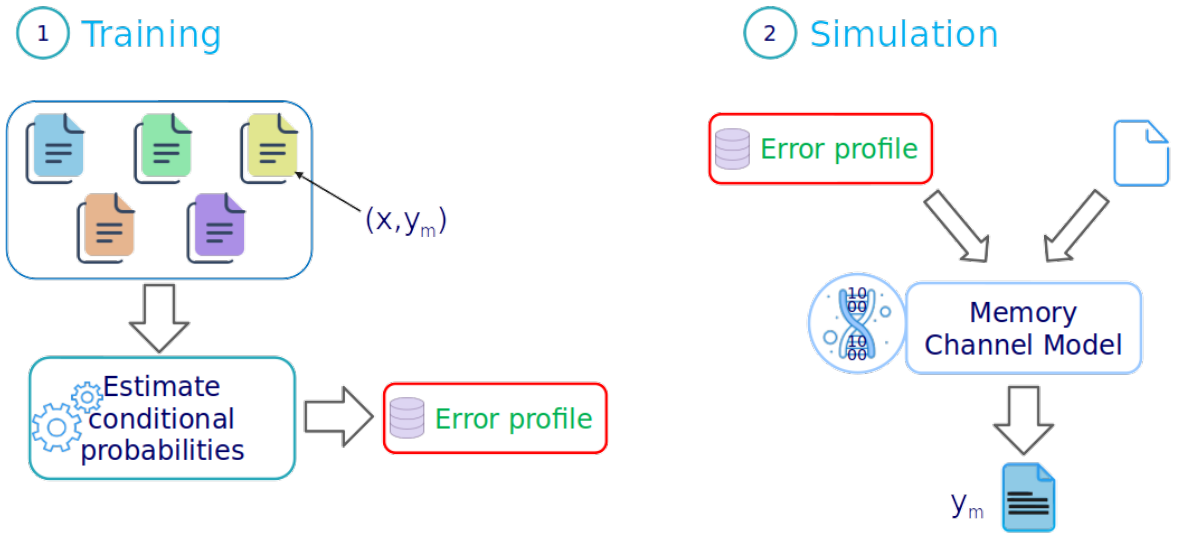


- Notation:
  - kmer$_t$: K-mer at position t
  - $E_t \in \{\text{Ins, Del, Sub, Match}\}$: error event at position t

- Probabilities:
  - $\mathbb{P}(E_t|\text{kmer}_t, E_{t-1})$: Probability of error event E_t
  - $\mathbb{P}(L|\text{kmer}_t, E_t = \text{Ins})$: Probability of insertion length L
  - $\mathbb{P}(B|\text{kmer}_t, E_t = \text{Sub})$: Probability of substitution by base B

[1] Belaid Hamoum, Elsa Dupraz, Laura Conde-Canencia, Dominique Lavenier, Channel Model with Memory for DNA Data Storage with Nanopore Sequencing, ISTC 2021

Training and simulations



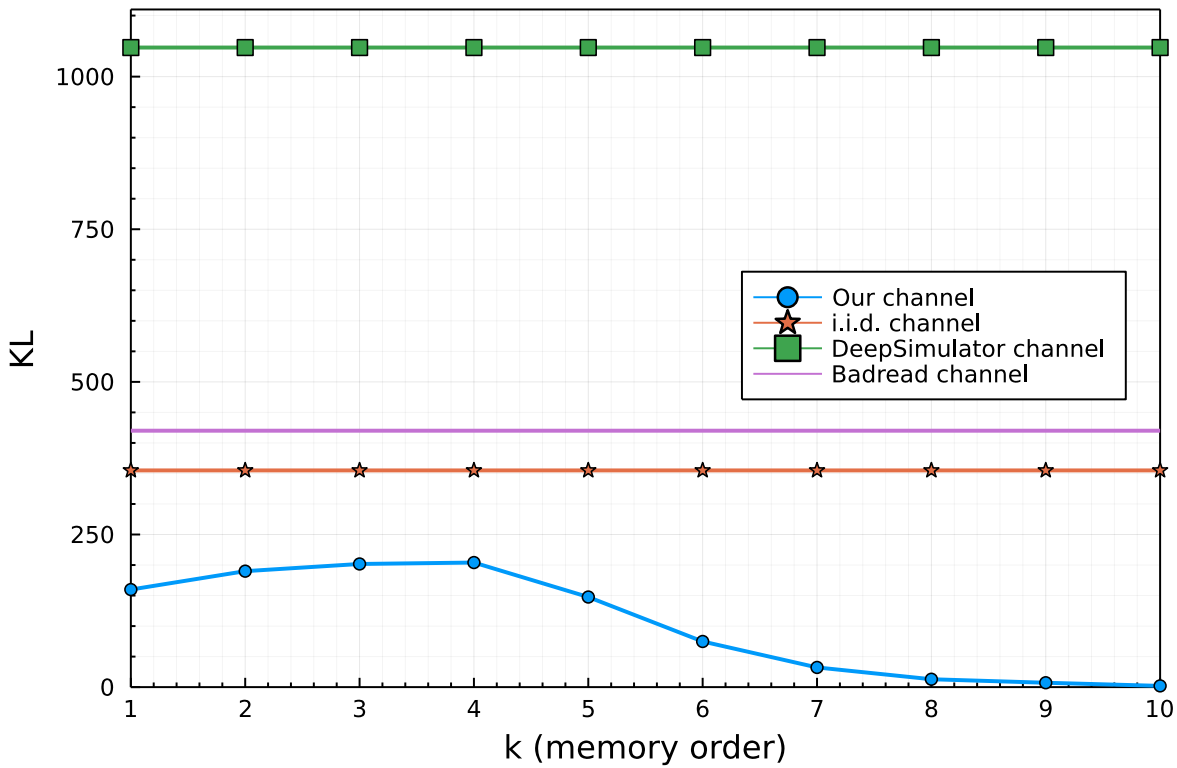
Training over different datasets:

Experimental data (2020): Thermo Fisher synthesis, **error rate**: 10%
Genomics data (2021): Streptococcus thermophilus bacteria, **error rate**: 3%
Experimental data (2021): Twist synthesis, Kodak Images, **Error rate**: 4%
Experimental data (2024): IDT synthesis, **Error rate**: 5%

Simulator available on GitHub: https://github.com/BHam-1/DNArSim

Kulback Leibler divergence[1]

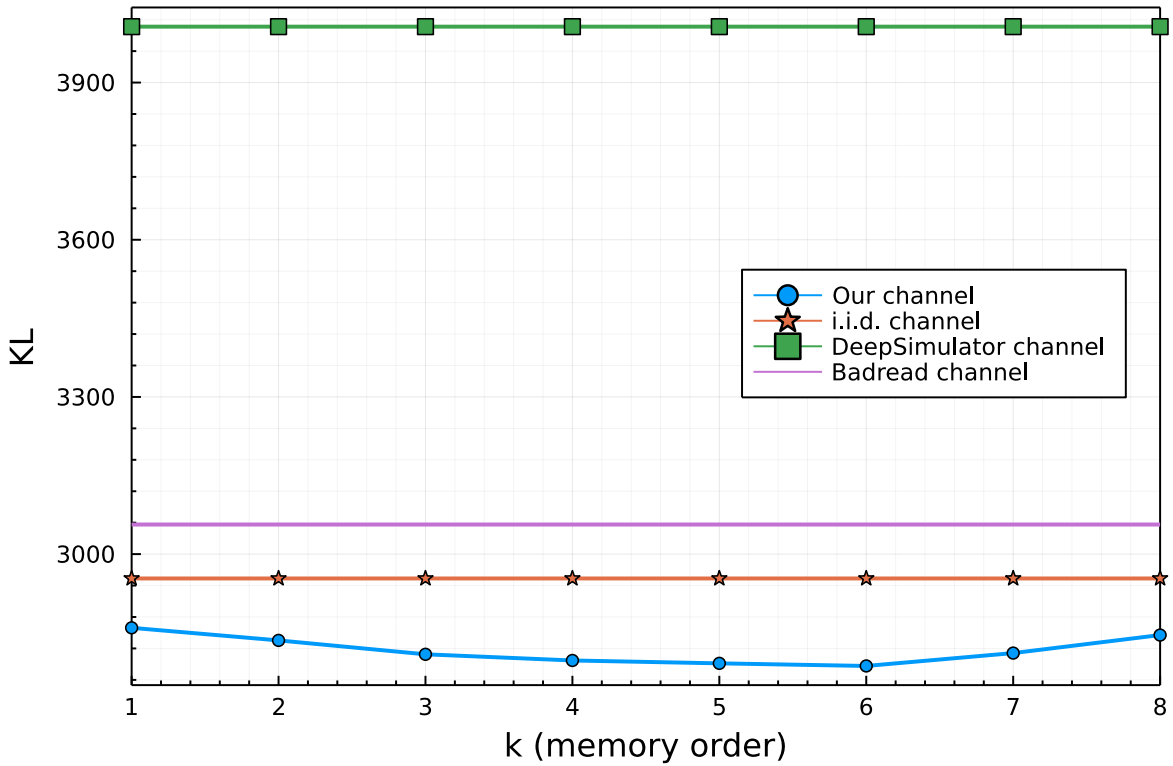
- Model trained on experimental data



[1] Belaid Hamoum, Elsa Dupraz, Laura Conde-Canencia, Dominique Lavenier, Channel Model with Memory for DNA Data Storage with Nanopore Sequencing, ISTC 2021

Kulback Leibler divergence[1]

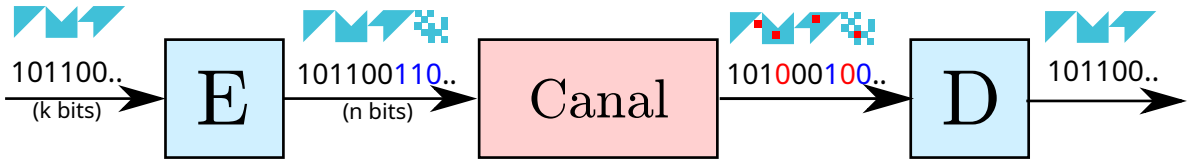
- Model trained on genomics data



[1] Belaid Hamoum, Elsa Dupraz, Laura Conde-Canencia, Dominique Lavenier, Channel Model with Memory for DNA Data Storage with Nanopore Sequencing, ISTC 2021

Outline

Channel coding



Channel coding:

- **(coding)** Structure redundancy in the data
- **(decoding)** Algorithm that leverages redundancy to correct errors

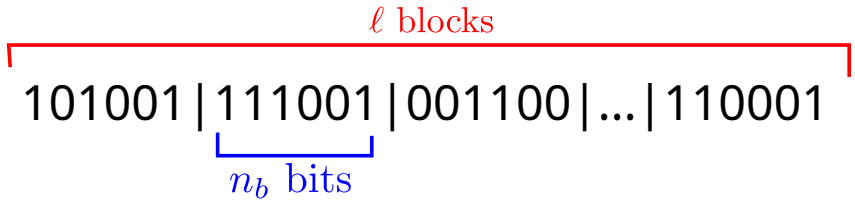
Coding rate:

$$R = \frac{k}{n}$$

Channel codes for DNA data storage

Adversarial models:

- Varshamov-Tenengolts (VT) codes can correct one deletion [Varshamov65]
- VT codes have been extended to the correction of a few insertions and/or deletions [Abroshan19,Xue20,KasHanna22,etc.]



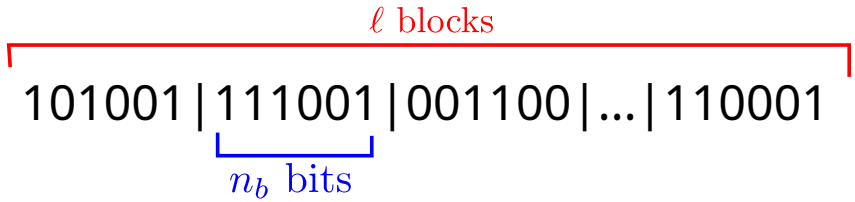
- Explicit bounds on the **redundancy**:

$$R \geq 1 - \lceil \log(n_b + 1) \rceil / n_b$$

Channel codes for DNA data storage

Adversarial models:

- Varshamov-Tenengolts (VT) codes can correct one deletion [Varshamov65]
- VT codes have been extended to the correction of a few insertions and/or deletions [Abroshan19,Xue20,KasHanna22,etc.]



- Explicit bounds on the **redundancy**:

$$R \geq 1 - \lceil \log(n_b + 1) \rceil / n_b$$

Statistical models:

- Pure coding:
  - **Convolutional codes** [Lenz21,etc.]
  - LDPC codes [Wang11,Shibata19,etc.]
- **Combination of consensus and coding**

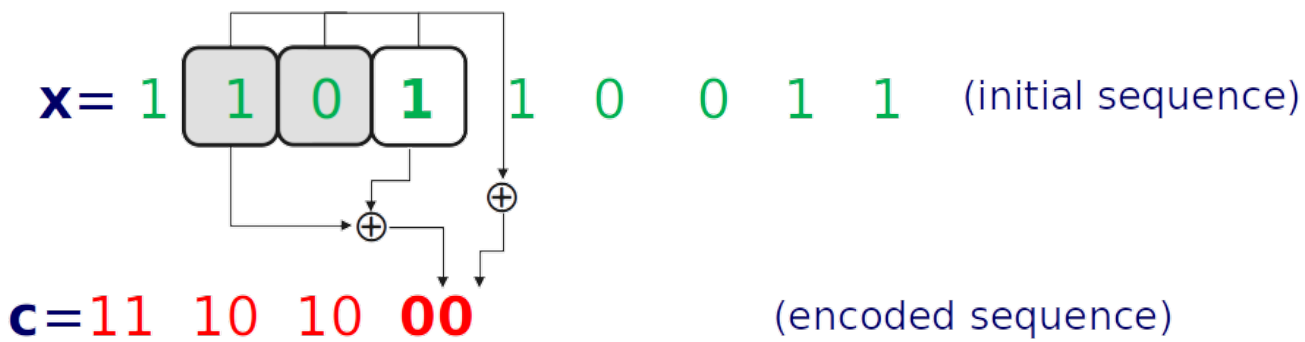
Outline

Standard convolutional codes

Convolutional codes:

- Were initially developped to correct **substitution errors**
- Encode blocks of arbitrary length

Example:
$(k_c = 1, n_c = 2, \nu = 2)$ convolutional code with $\Delta = [\Delta^2 + 1, \Delta^2 + \Delta + 1]$.

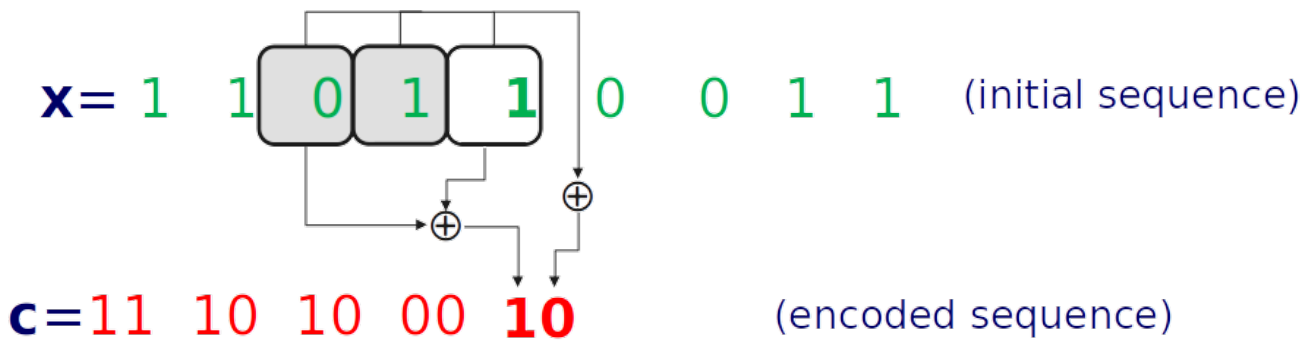


Standard convolutional codes

Convolutional codes:

- Were initially developped to correct **substitution errors**
- Encode blocks of arbitrary length

Example:
$(k_c = 1, n_c = 2, \nu = 2)$ convolutional code with $\Delta = [\Delta^2 + 1, \Delta^2 + \Delta + 1]$.

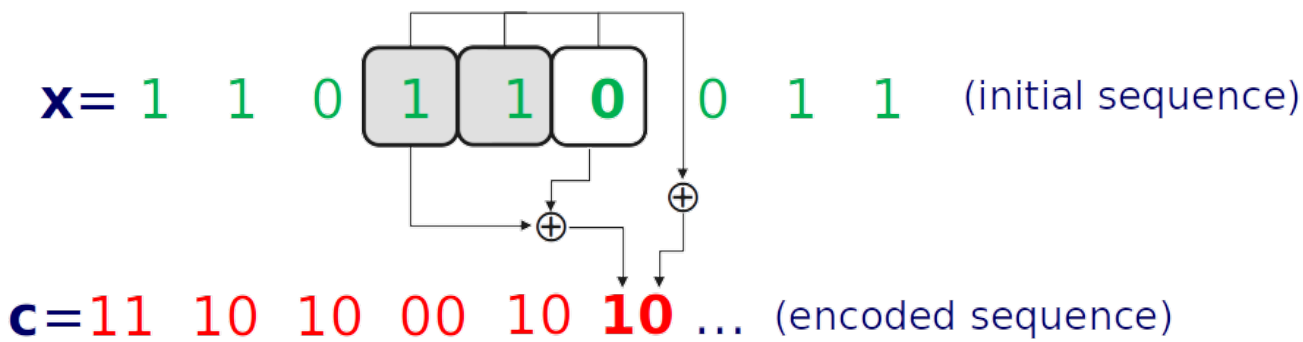


Standard convolutional codes

Convolutional codes:

- Were initially developped to correct **substitution errors**
- Encode blocks of arbitrary length

Example:
$(k_c = 1, n_c = 2, \nu = 2)$ convolutional code with $\Delta = [\Delta^2 + 1, \Delta^2 + \Delta + 1]$.

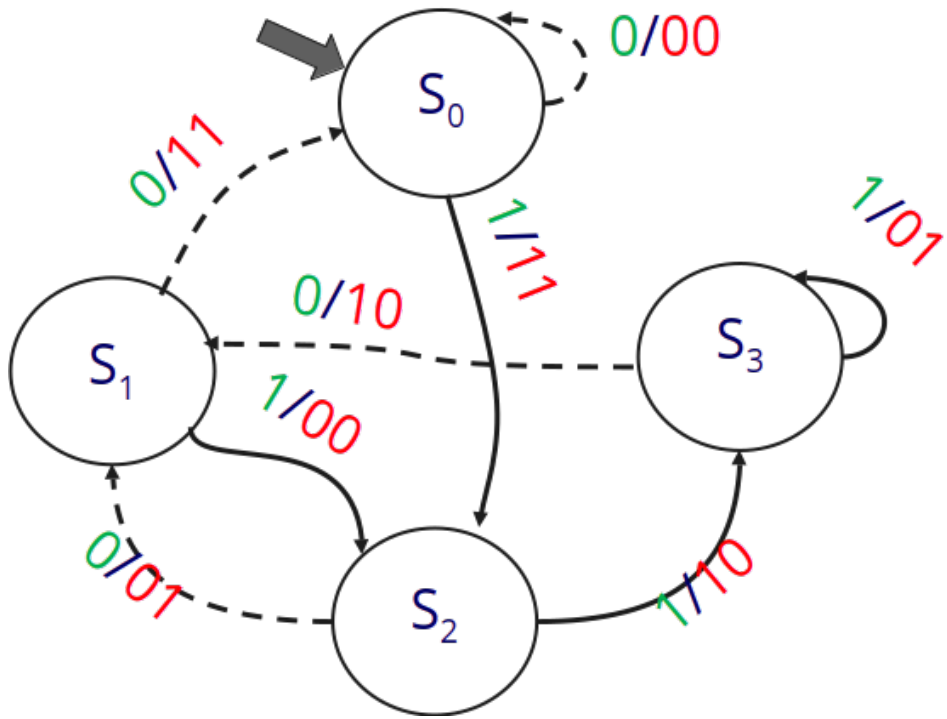


Standard CC decoding

- A CC is often represented as a state diagram
- **Example**:

$\mathbf{x} = 1 \quad 1 \quad 0 \quad 0..$

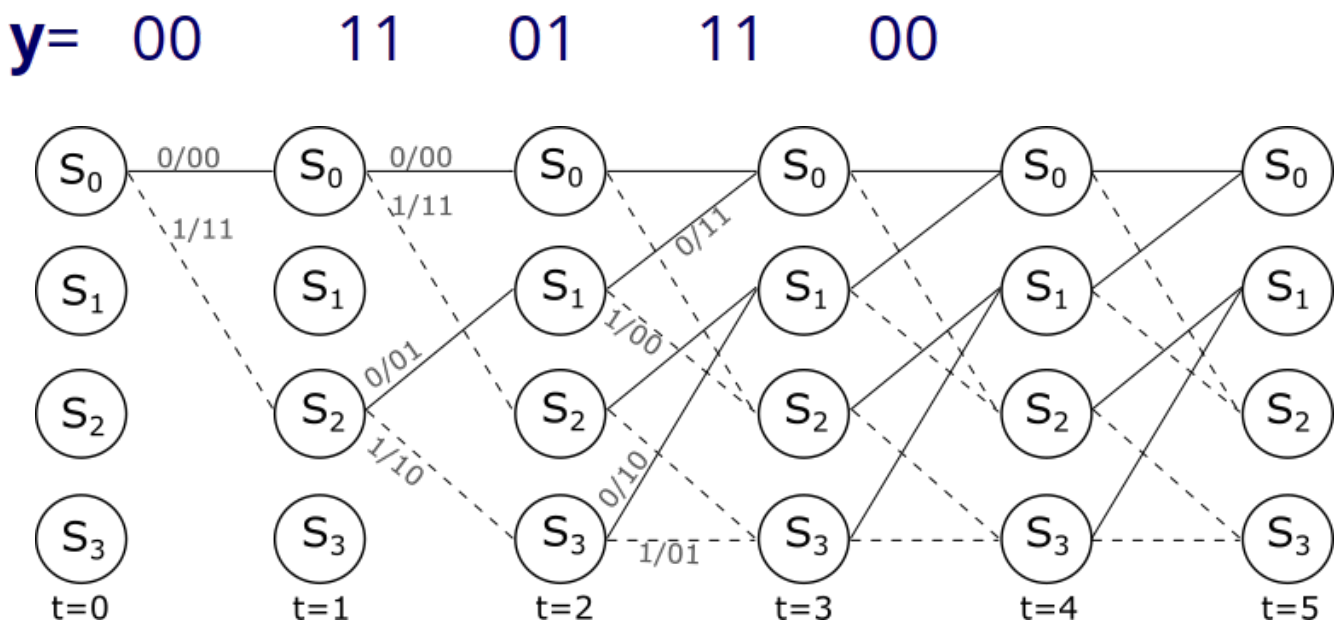
$\mathbf{c} = 11 \; 10 \; 10 \; 00..$



Standard CC decoding

- **Decoder task**: $\max_{\mathbf{x}^n} \mathbb{P}(\mathbf{x}^n|\mathbf{y}^n)$
- **(Markov property)** Each symbol (2 bits) y_t depends only on the states S_t and S_{t+1}

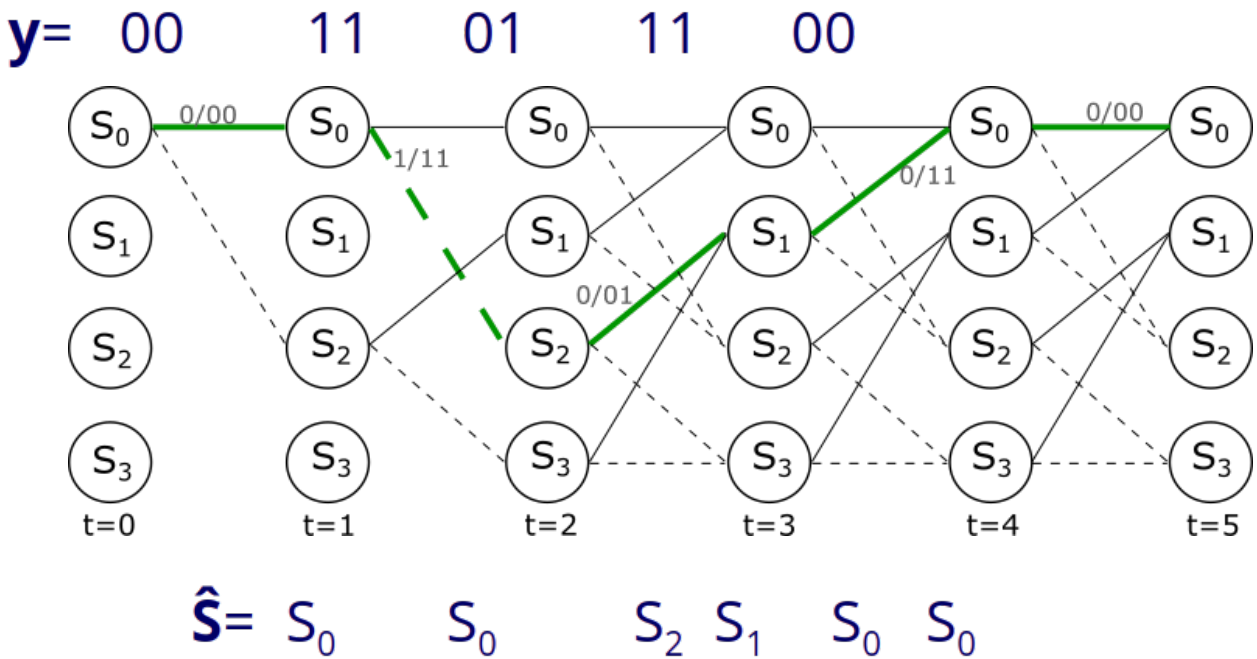
- Decoding from a trellis:



Standard CC decoding

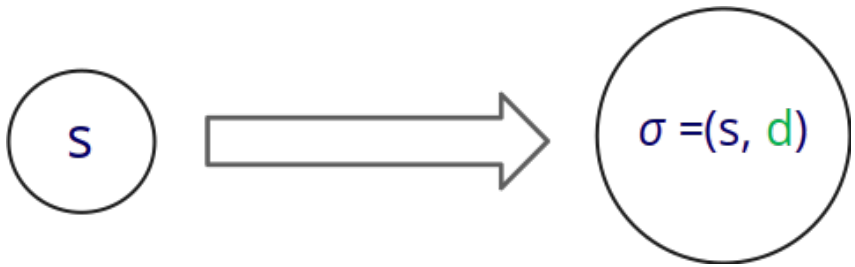
- **Decoder task**: $\max_{\mathbf{x}^n} \mathbb{P}(\mathbf{x}^n|\mathbf{y}^n)$
- **(Markov property)** Each symbol (2 bits) y_t depends only on the states S_t and S_{t+1}

- Decoding from a trellis:



CC decoding with insertions and deletions

- To consider synchronization errors, another variable, called Drift is added to the decoder states (S_0, S_1, S_2, S_3)

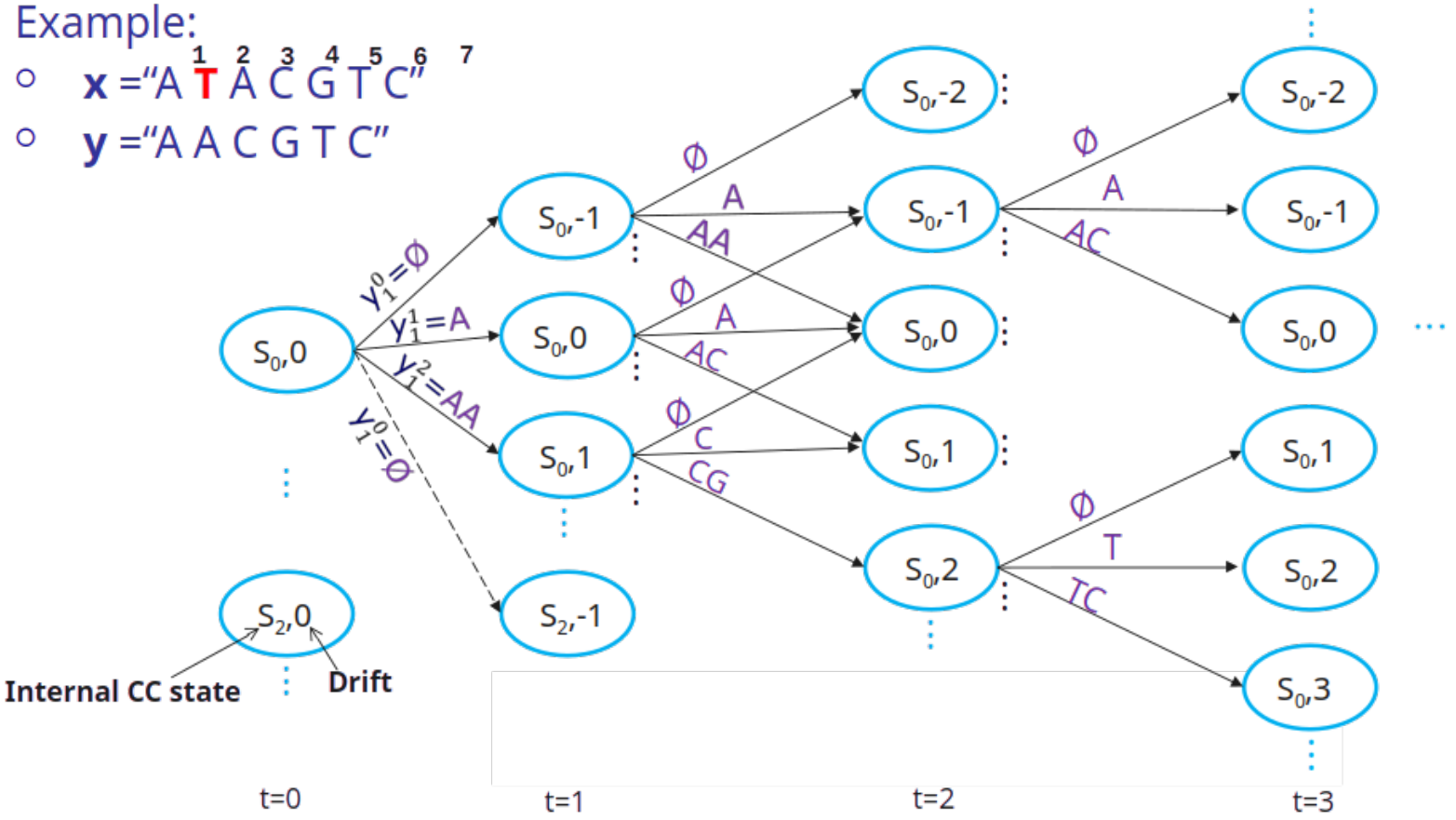


- $d_t = \text{Nb(Ins)}_t - \text{Nb(Del)}_t$ is the drift value before transmitting the symbol x_t.

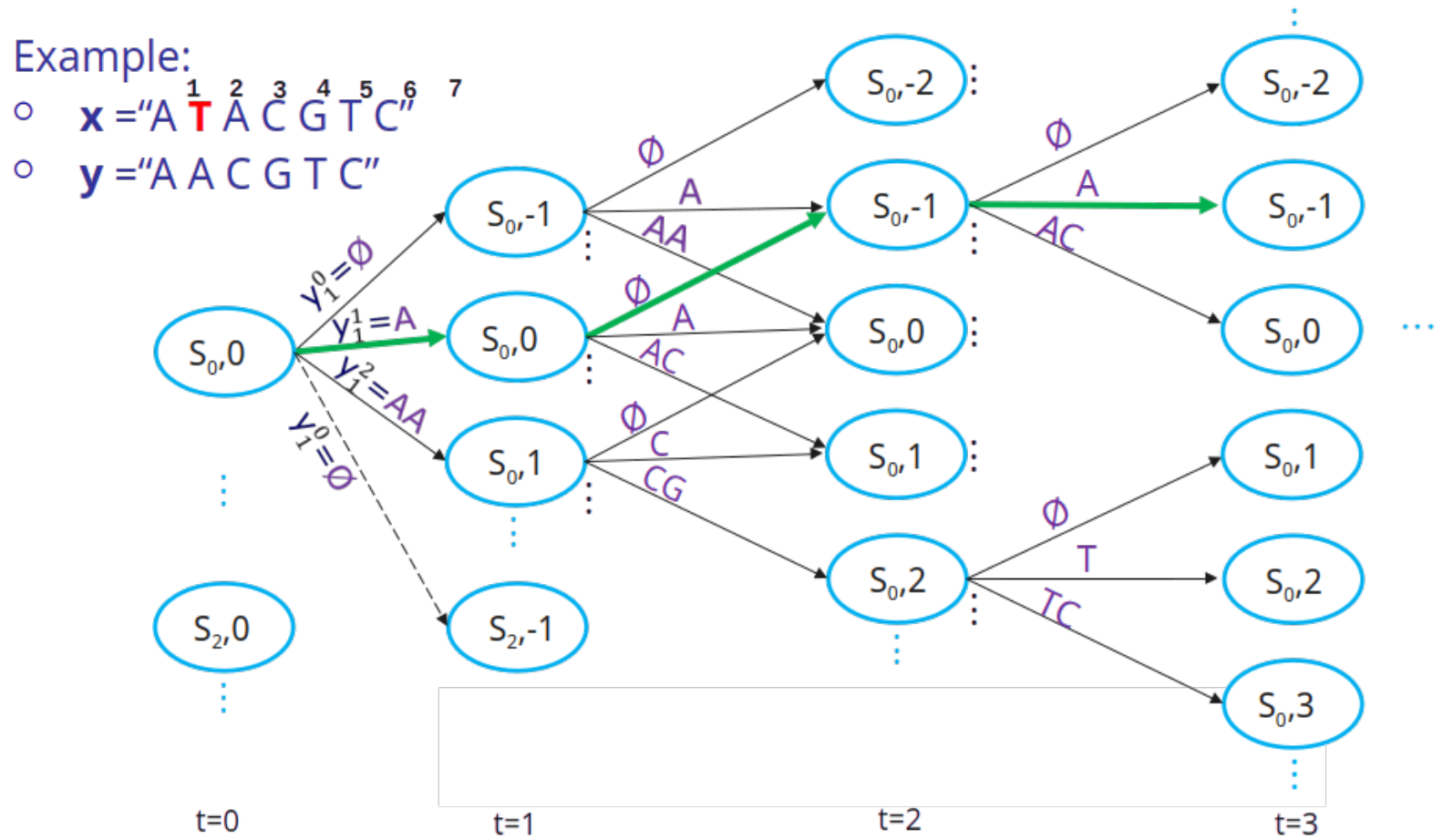
Example:

$$\mathbf{x} = ACAAT\_AGA$$
$$\mathbf{y} = A\_AATTAGA$$

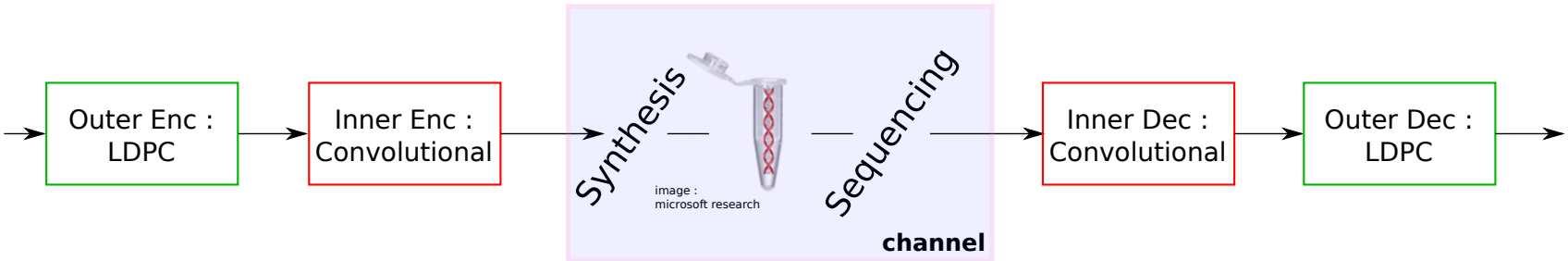
CC decoding with insertions and deletions



CC decoding with insertions and deletions



Full solution: Convolutionnal codes + LDPC codes[1]

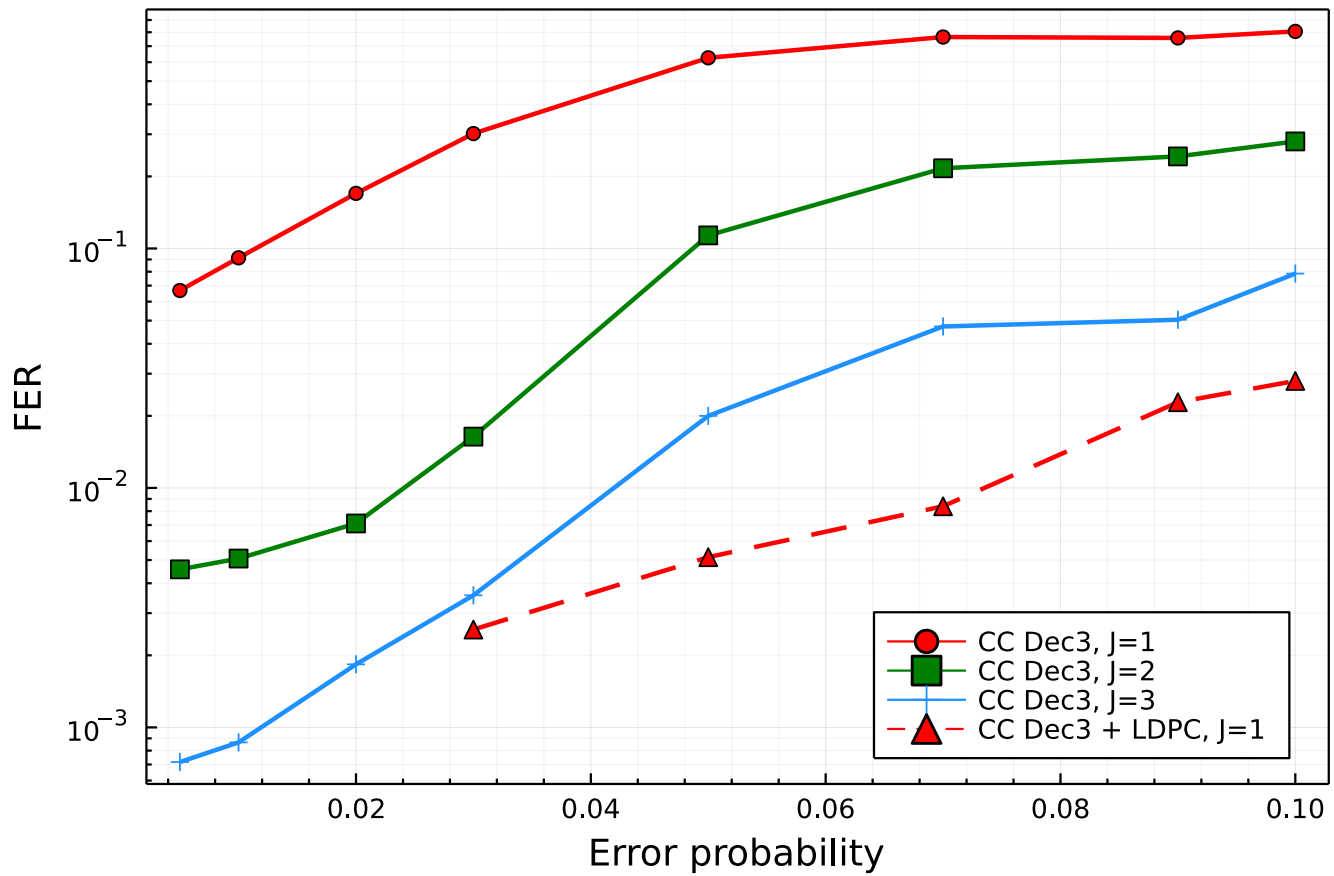


- Solution initially proposed in [Lenz21] for **i.i.d. channel models**
- It uses a modified BCJR algorithm for CC decoding to correct most synchronization errors
- We updated the BCJR algorithm to include the knowledge of our channel model

[1] Belaïd Hamoum, Elsa Dupraz, Channel Model and Decoder with Memory for DNA Data Storage with Nanopore Sequencing, IEEE Access, pp. 52075-52087, May 2023

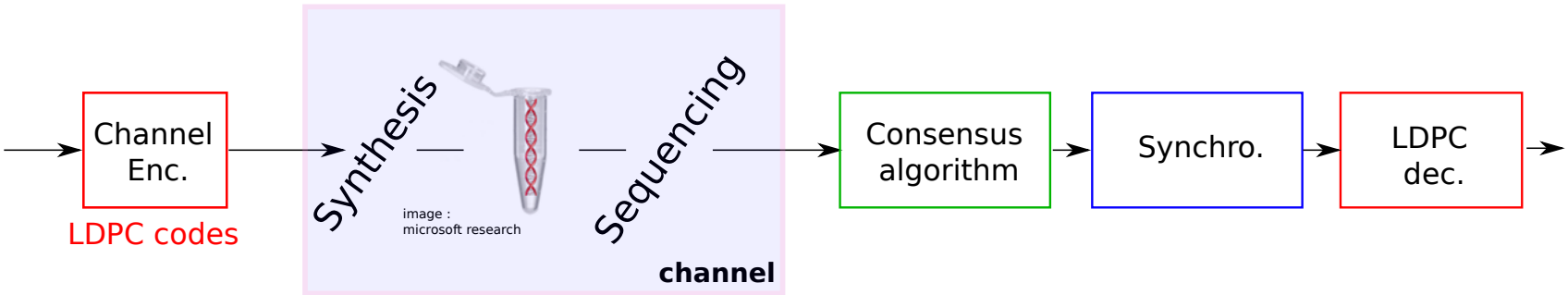
Results with the dynamic channel model

- Comparison with the concatenated construction, CC+LDPC, $R = 1/4$, $N = 204$



Outline

Mixed solution: Consensus algorithm + LDPC codes[1]



- The consensus algorithm [Lavenier21] takes **m sequences** as inputs
- A few errors remain after consensus
- The proposed synchronization method uses the LDPC code structure to correct (ins+sub) or (del+sub)

[1] Belaid Hamoum, Aref Ezzeddine, Elsa Dupraz, Synchronization algorithms from high-rate LDPC codes for DNA data storage, DSP 2023

LDPC codes

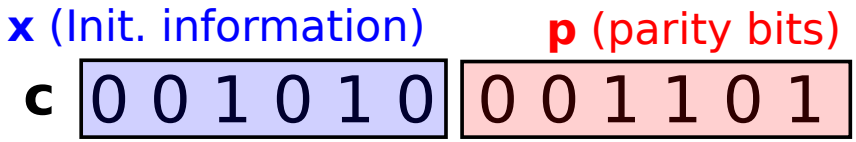
LDPC codes: linear block codes

x (Init. information)

0	0	1	0	1	0

LDPC codes

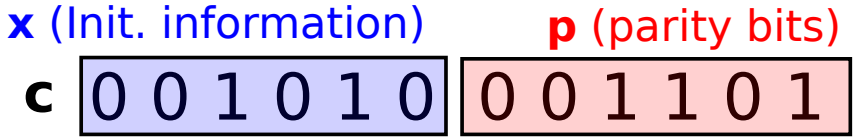
LDPC codes: linear block codes



Codeword: $\mathbf{c}^n = G\mathbf{x}^k$ such that $H\mathbf{c}^n = \mathbf{0}^m$ $\quad (HG = 0)$

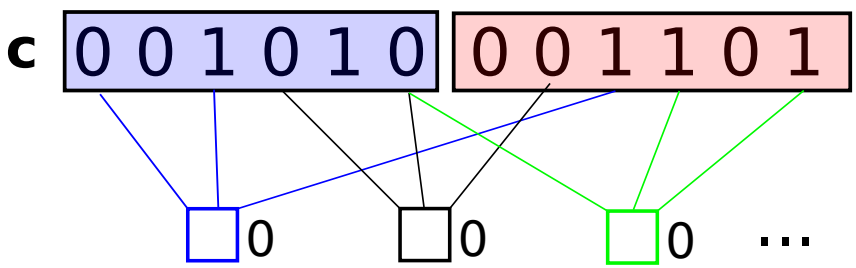
LDPC codes

LDPC codes: linear block codes



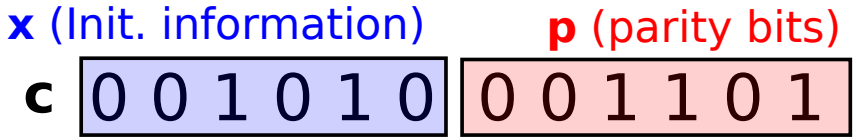
Codeword: $\mathbf{c}^n = G\mathbf{x}^k$ such that $H\mathbf{c}^n = \mathbf{0}^m$ $\quad (HG = 0)$

LDPC decoding by using the code parity check equations



LDPC codes

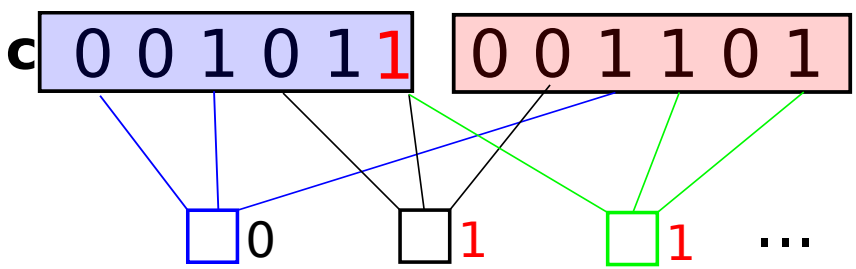
LDPC codes: linear block codes



Codeword: $\mathbf{c}^n = G\mathbf{x}^k$ such that $H\mathbf{c}^n = \mathbf{0}^m$ $\quad (HG = 0)$

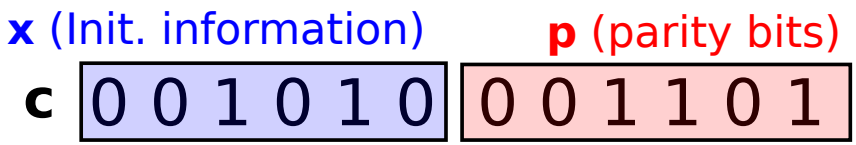
LDPC decoding by using the code parity check equations

Substitution error:



LDPC codes

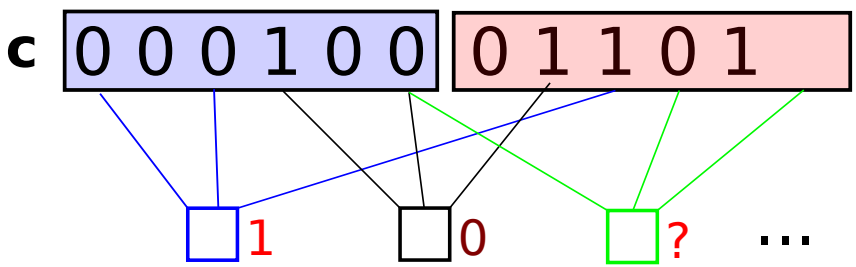
LDPC codes: linear block codes



Codeword: $\mathbf{c}^n = G\mathbf{x}^k$ such that $H\mathbf{c}^n = \mathbf{0}^m$ $\quad (HG = 0)$

LDPC decoding by using the code parity check equations

Deletion error:



Synchronization by testing all possible positions for deletions

Correcting 1 deletion

Original message:

$$0\,1\,0\,0\,1\,0\,1\,1\,\mathbf{1}\,\mathbf{1}\,1\,1\,0\,0\,0\,0\,1\,1$$

Synchronization method:

- Block length: L_s
- Try inserting two bits in each block so as to maximize the score
- Score: number of satisfied parity check equations

- **Example**: $(L_s = 4)$

$$0\,1\,0\,0 \mid 1\,0\,1\,1 \mid 1\,1\,0\,0 \mid 0\,0\,1\,1$$

Correcting 1 deletion

Original message:

$$0\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ \mathbf{1}\ \mathbf{1}\ 1\ 1\ 0\ 0\ 0\ 0\ 1\ 1$$

Synchronization method:

- Block length: L_s
- Try inserting two bits in each block so as to maximize the score
- Score: number of satisfied parity check equations

- **Example**: $(L_s = 4)$

$$0\ 0\ 0\ 1\ 0\ 0 \mid 1\ 0\ 1\ 1 \mid 1\ 1\ 0\ 0 \mid 0\ 0\ 1\ 1$$

score: 3

Correcting 1 deletion

Original message:

$$0\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ \mathbf{1}\ \mathbf{1}\ 1\ 1\ 0\ 0\ 0\ 1\ 1$$

Synchronization method:

- Block length: L_s
- Try inserting two bits in each block so as to maximize the score
- Score: number of satisfied parity check equations

- **Example**: $(L_s = 4)$

$$0\ 1\ 0\ 0\ |0\ 0\ 1\ 0\ 1\ 1\ |\ 1\ 1\ 0\ 0\ |\ 0\ 0\ 1\ 1$$

score: 4

Correcting 1 deletion

Original message:

$$0\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ \mathbf{1}\ \mathbf{1}\ 1\ 1\ 0\ 0\ 0\ 0\ 1\ 1$$

Synchronization method:

- Block length: L_s
- Try inserting two bits in each block so as to maximize the score
- Score: number of satisfied parity check equations

- **Example**: $(L_s = 4)$

$$0\ 1\ 0\ 0\ |\ 1\ 0\ 1\ 1\ |0\ 0\ 1\ 1\ 0\ 0\ |\ 0\ 0\ 1\ 1$$

score: 8

Correcting 1 deletion

Original message:

$$0\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ \mathbf{1}\ \mathbf{1}\ 1\ 1\ 0\ 0\ 0\ 1\ 1$$

Synchronization method:

- Block length: L_s
- Try inserting two bits in each block so as to maximize the score
- Score: number of satisfied parity check equations

- **Example**: $(L_s = 4)$

$$0\ 1\ 0\ 0 \mid 1\ 0\ 1\ 1 \mid 1\ 1\ 0\ 0 \mid 0\ 0\ 0\ 0\ 1\ 1$$

score: 4

Correcting t deletions

- Greedy approach: insert pairs of bit one after each other until t pairs have been inserted

$$0\,1\,0\,0 \mid 1\,0\,1\,1 \mid 1\,1\,0\,0 \mid 0\,0\,1\,1$$

- Exhaustive approach: insert the t pairs of bits at once

$$0\,1\,0\,0 \mid 1\,0\,1\,1 \mid 1\,1\,0\,0 \mid 0\,0\,1\,1$$

Correcting t deletions

- Greedy approach: insert pairs of bit one after each other until t pairs have been inserted

$$0\ 1\ 0\ 0\ |\ 1\ 0\ 1\ 1\ |0\ 0\ 1\ 1\ 0\ 0\ |\ 0\ 0\ 1\ 1$$

score: 8

- Exhaustive approach: insert the t pairs of bits at once

$$0\ 1\ 0\ 0\ |\ 1\ 0\ 1\ 1\ |\ 1\ 1\ 0\ 0\ |\ 0\ 0\ 1\ 1$$

Correcting t deletions

- Greedy approach: insert pairs of bit one after each other until t pairs have been inserted

$$0\ 1\ 0\ 0\ |\ 1\ 0\ 1\ 1\ |0\ 0\ 1\ 1\ 0\ 0\ |0\ 0\ 0\ 0\ 1\ 1$$

score: 14

- Exhaustive approach: insert the t pairs of bits at once

$$0\ 1\ 0\ 0\ |\ 1\ 0\ 1\ 1\ |\ 1\ 1\ 0\ 0\ |\ 0\ 0\ 1\ 1$$

Correcting t deletions

- Greedy approach: insert pairs of bit one after each other until t pairs have been inserted

  0 1 0 0 | 1 0 1 1 |0 0 1 1 0 0 |0 0 0 0 1 1

  score: 14

- Exhaustive approach: insert the t pairs of bits at once

  0 0 0 1 0 0 |0 0 1 0 1 1 | 1 1 0 0 | 0 0 1 1

  score: 6

Correcting t deletions

- Greedy approach: insert pairs of bit one after each other until t pairs have been inserted

  0 1 0 0 | 1 0 1 1 |0 0 1 1 0 0 |0 0 0 0 1 1

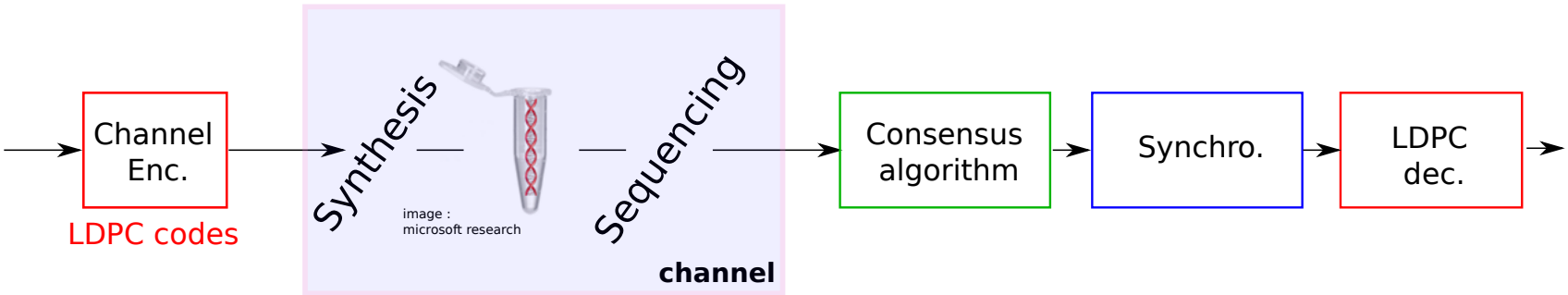
  score: 14

- Exhaustive approach: insert the t pairs of bits at once

  0 0 0 1 0 0 | 1 0 1 1 |0 0 1 1 0 0 | 0 0 1 1

  score: 12

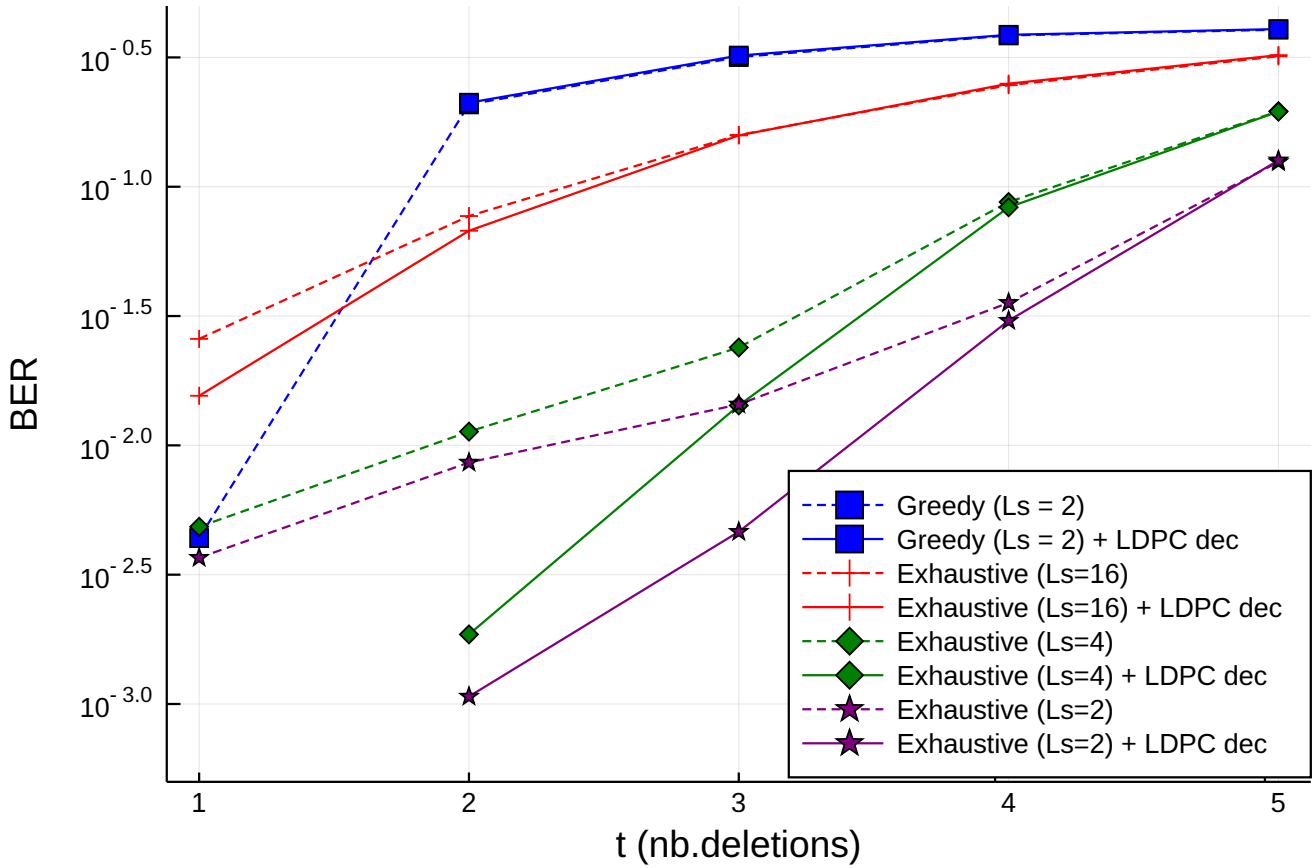
Mixed solution: Consensus algorithm + LDPC codes[1]



- The consensus algorithm [Lavenier21] takes **m sequences** as inputs
- A few errors remain after consensus
- The proposed synchronization method uses the LDPC code structure to correct (ins+sub) or (del+sub)

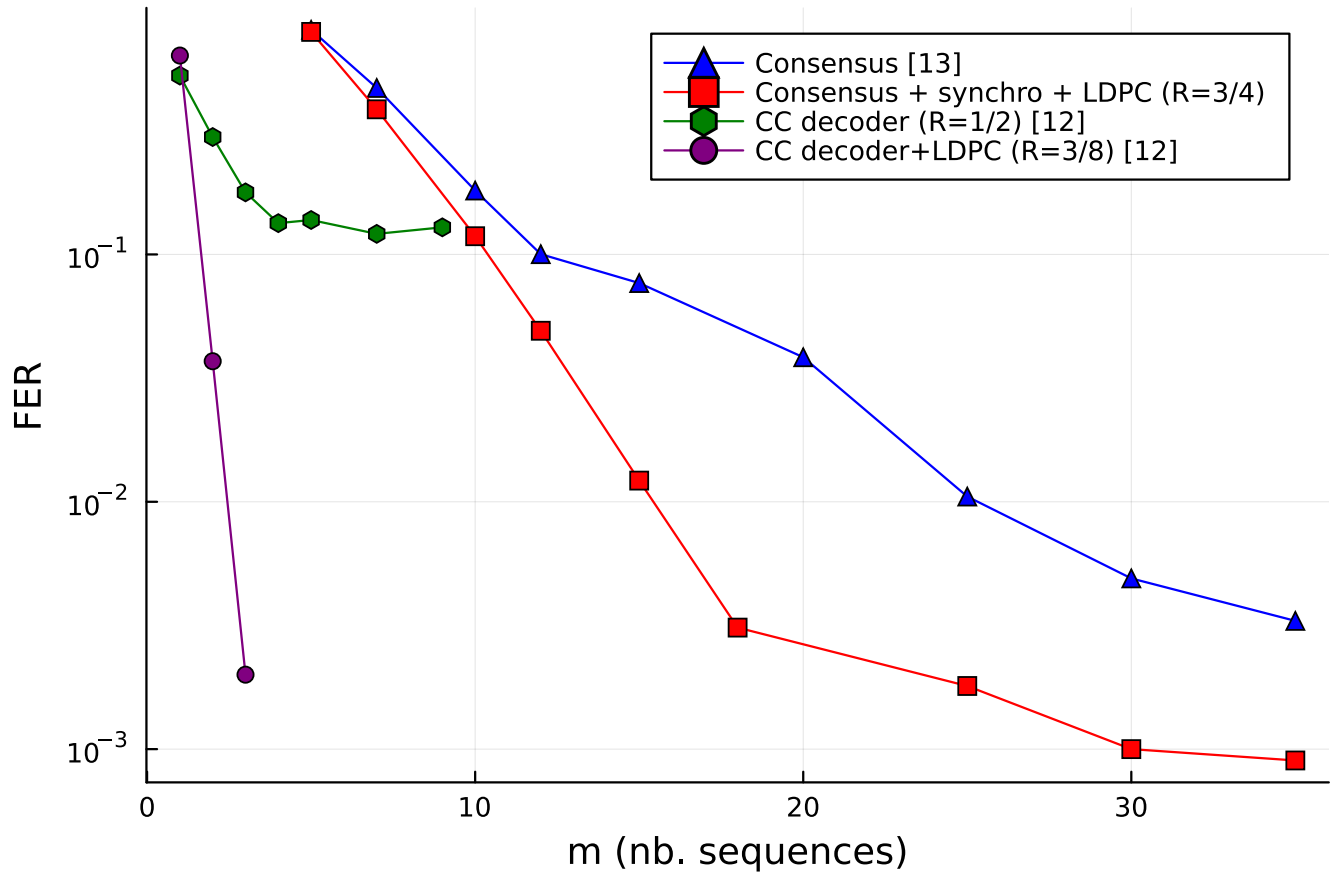
[1] Belaid Hamoum, Aref Ezzeddine, Elsa Dupraz, Synchronization algorithms from high-rate LDPC codes for DNA data storage, DSP 2023

Results for synchronization+LDPC



- Regular LDPC code, code rate $R = 3/4$, $N = 256$
- **Conclusion**: the exhaustive approach is more efficient but more complex

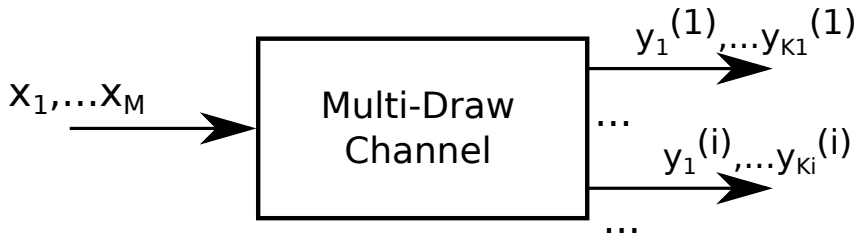
Results for the full solution



- Regular $(3, 4)$ LDPC codes with $N = 256 \rightarrow$ 192 bits of information
- Concatenated construction of [Lenz21]: $\rightarrow$ 96 bits of information

Outline

Multi-Draw channel model



Model

- Consider M sequences $\mathbf{x}_1, \cdots, \mathbf{x}_M$
- **Multi-draw channel**: for each $\mathbf{x}_i$, K_i outputs $\mathbf{y}_i^{(k)}$ (K_i is random)

Coding may help to:

- Cluster the reads $\mathbf{y}_i^{(k)}$
- Decode the M sequences $\mathbf{x}_i$
- Re-order them
- Retrieve non-observed inputs ($K_i = 0$)
- Address the coupon collector problem