Polar FEC Codes Running at Hundreds of Mbit/s in GNU Radio or On a Software Implementation of the Fast-SSC Algorithm

Pascal Giard[1] and Alexandre J. Raymond[2]

[1]McGill University,
Department of Electrical and Computer Engineering,
Montréal, Québec, Canada
pascal.giard@mail.mcgill.ca

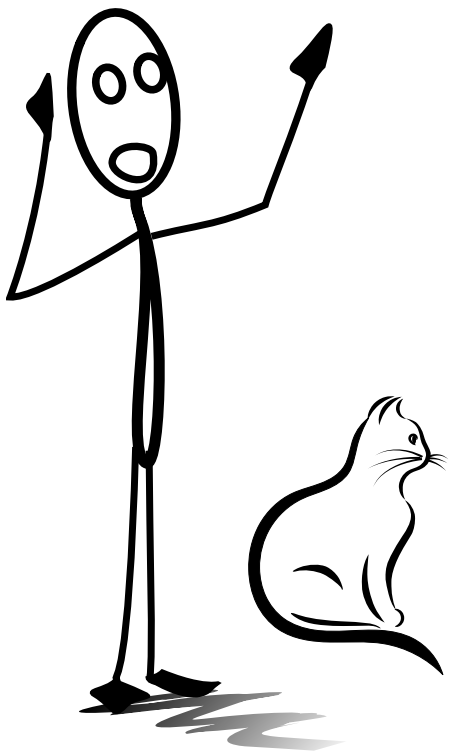
[2]École de technologie supérieure
Montréal, Québec, Canada
alexandre.raymond@lacime.etsmtl.ca

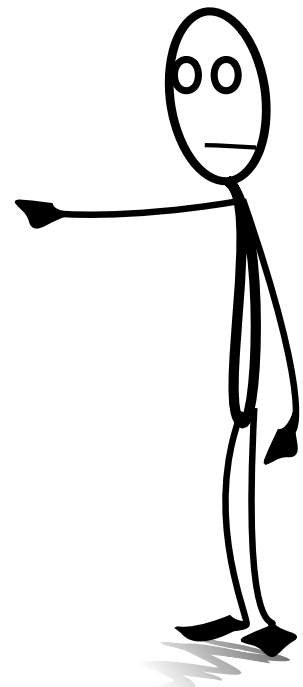
August 26, 2015

Who am I?

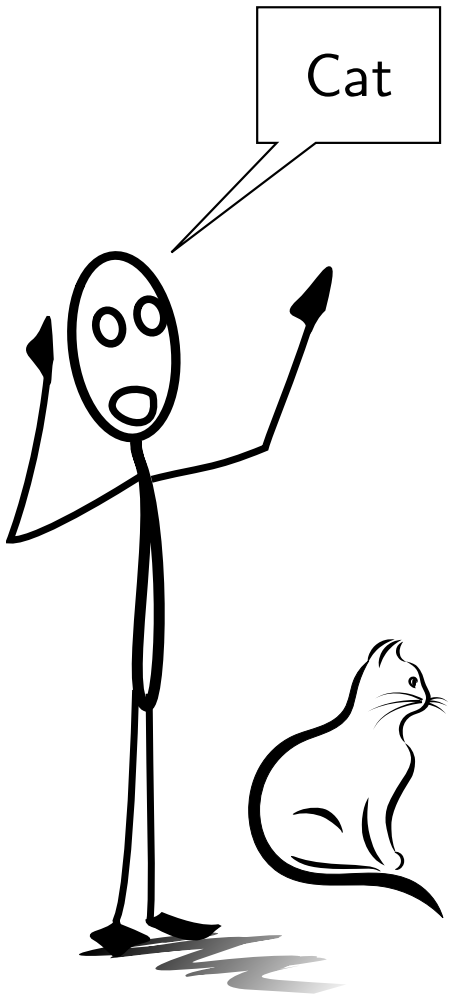
- Ph.D. candidate and research assistant at McGill University.
- Lecturer and research professional at ÉTS.
- Work on efficient algorithms and implementations for modern error-correction codes with a special focus on polar codes.
  - FPGA and ASIC
  - Intel/AMD x86-64 and ARM NEON SIMD
  - GPGPU

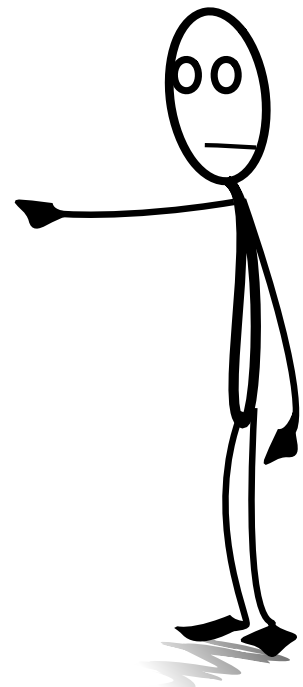
Introduction to Error Correction



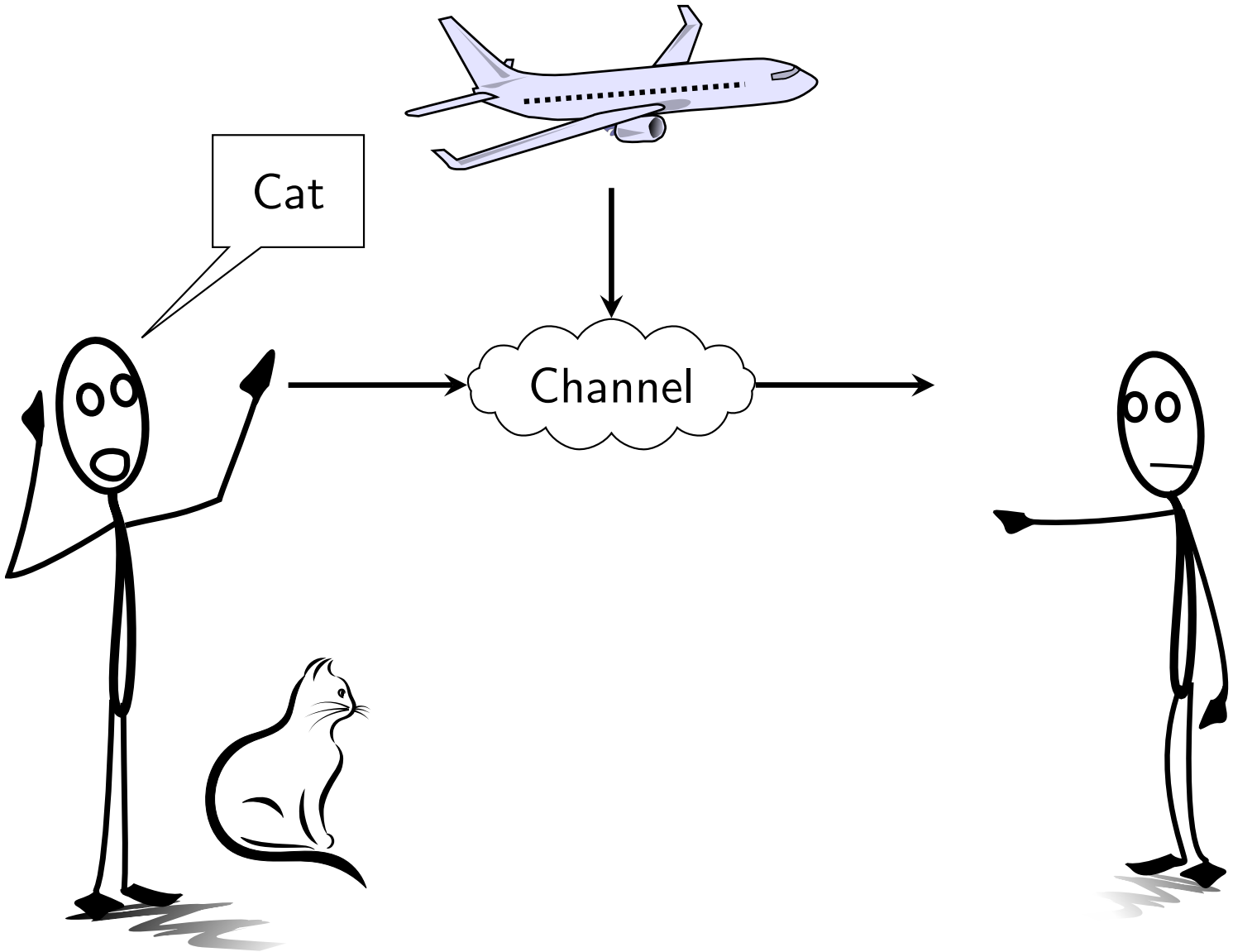


Introduction to Error Correction

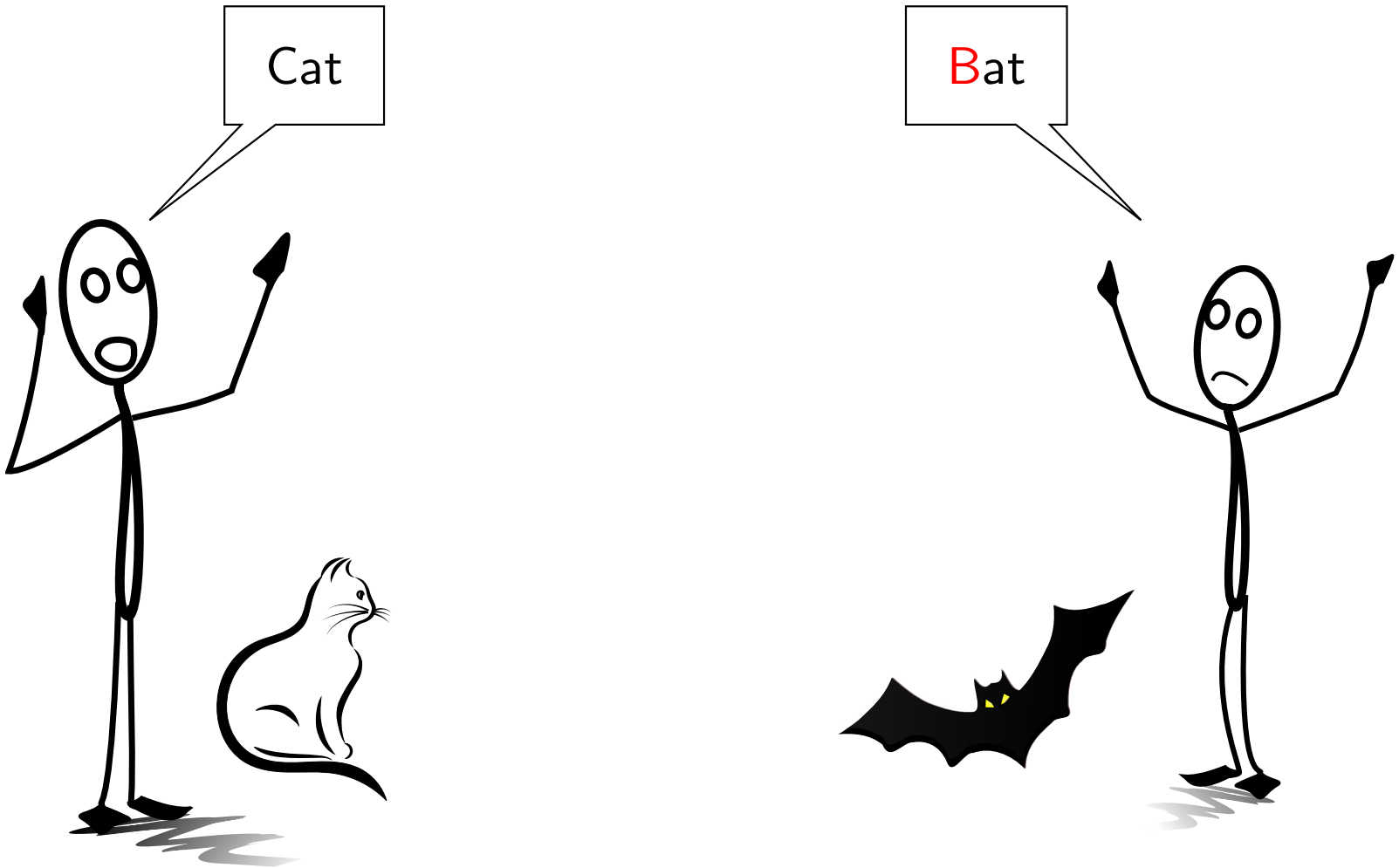




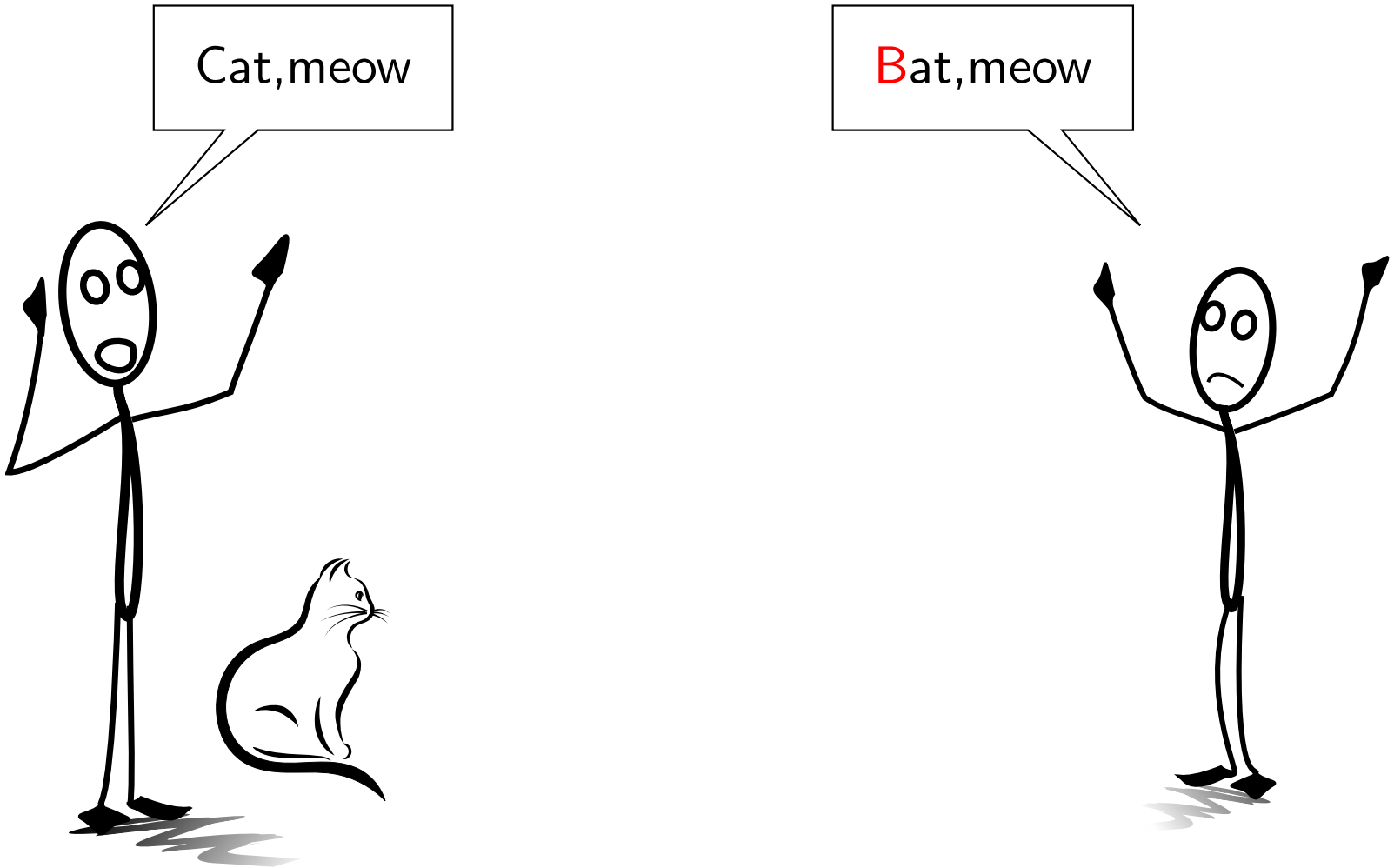
Introduction to Error Correction



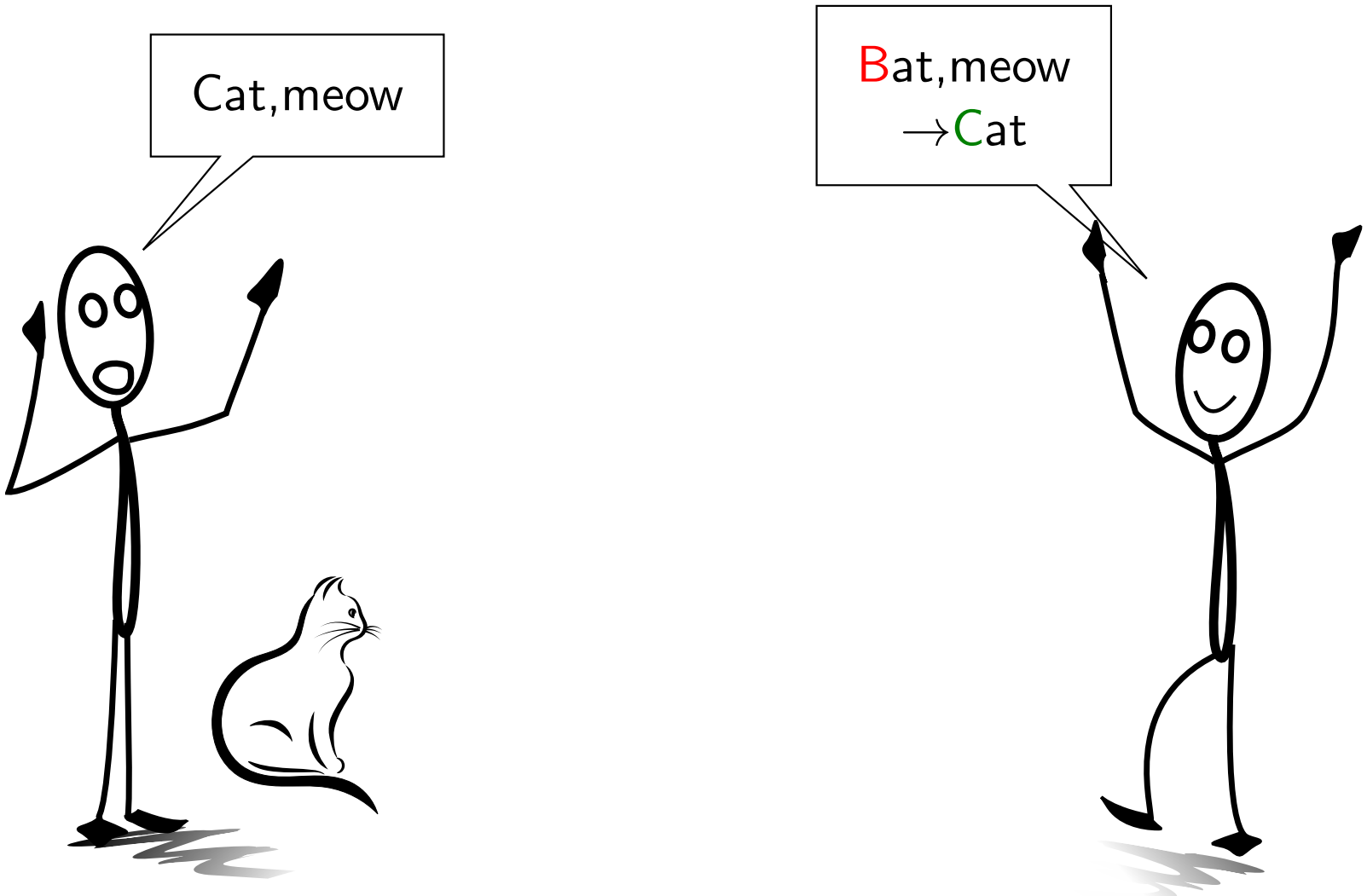
Introduction to Error Correction



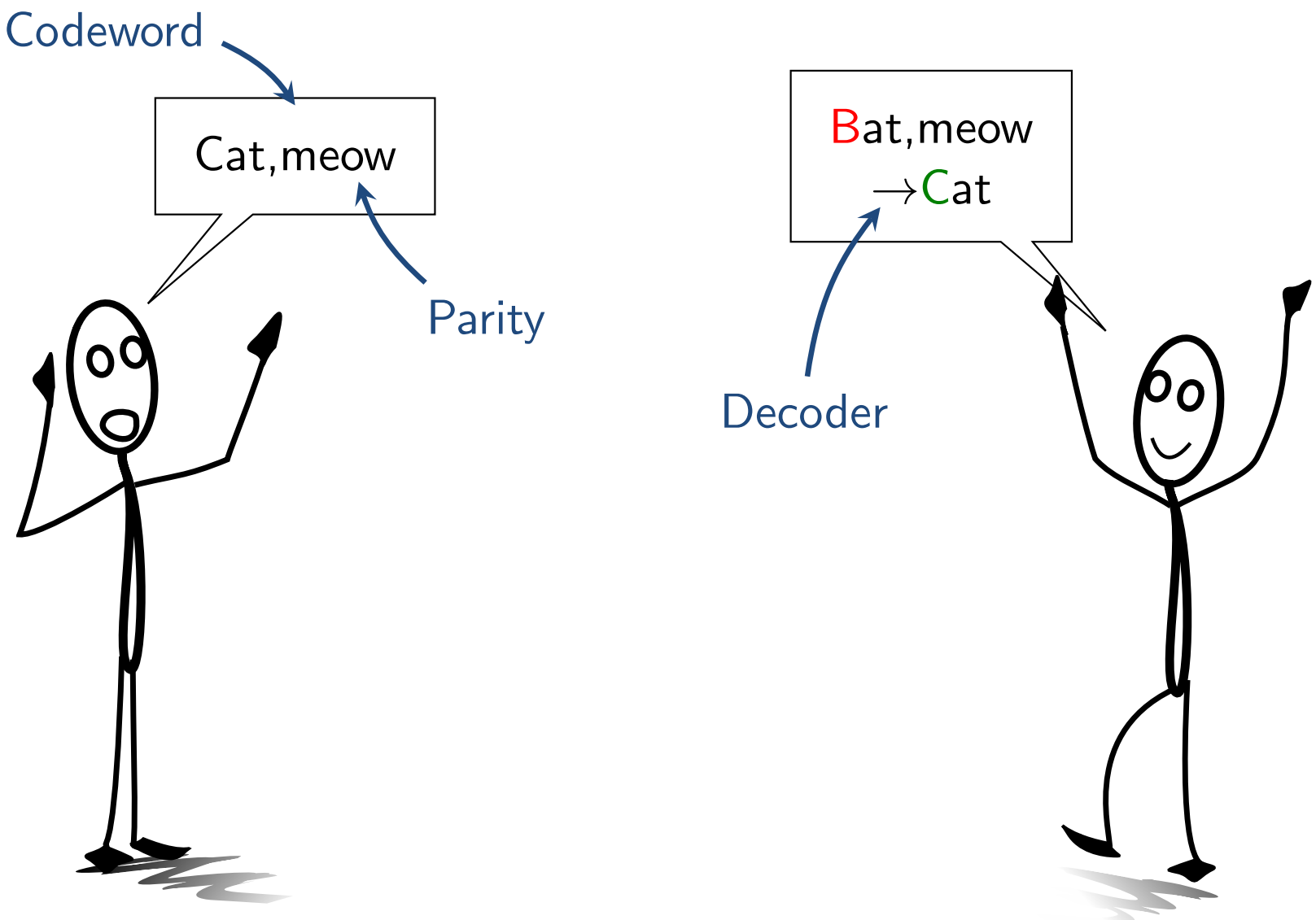
Introduction to Error Correction



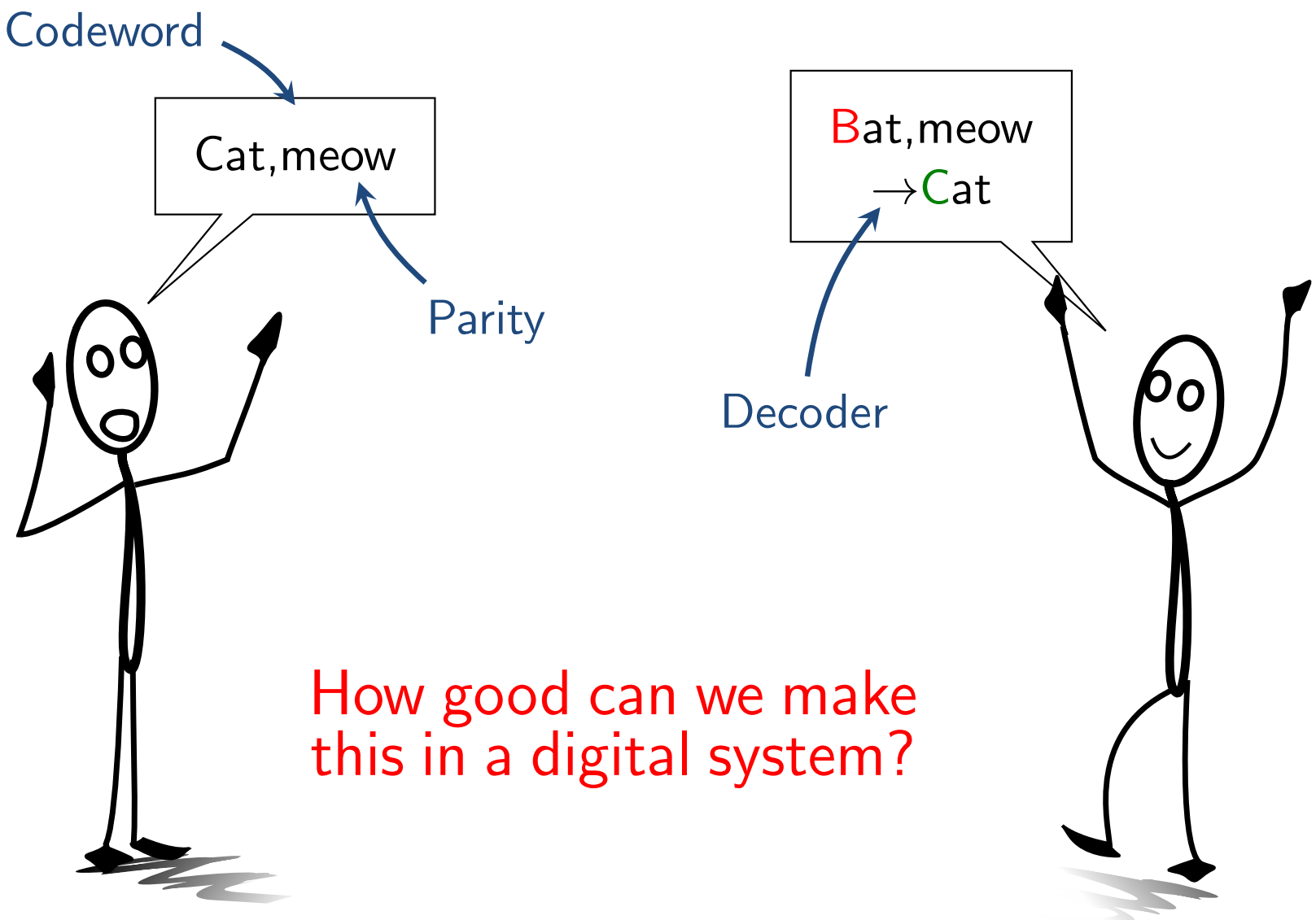
Introduction to Error Correction



Introduction to Error Correction

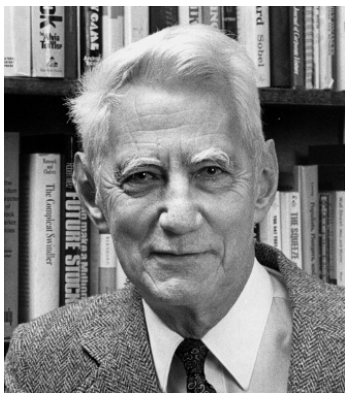


Introduction to Error Correction



How good can we make this in a digital system?

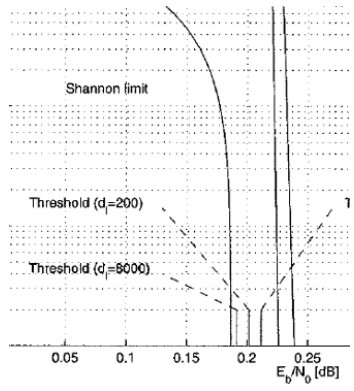
Achieving the Channel Capacity



Definition
1948

LDPC Codes
1960

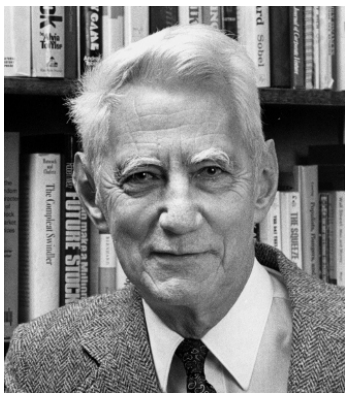
Turbo Codes
1993
0.7 dB



LDPC Codes
2004
0.04 dB

Polar Codes
2008
0 dB
using SC
as $N \to \infty$

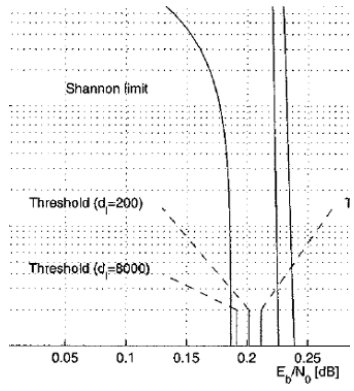
Achieving the Channel Capacity



Definition
1948

LDPC Codes
1960

Turbo Codes
1993
0.7 dB



LDPC Codes
2004
0.04 dB

Polar Codes
2008
0 dB
using SC
as $N \to \infty$

Very challenging in software.

Achieving the Channel Capacity

Successive cancellation $\rightarrow$ low throughput.

$\Rightarrow$ Use a better algorithm.

Polar Codes

2008

0 dB

using SC

as $N \rightarrow \infty$

Performance at moderate code-length is modest.
Becomes excellent with $N > 2^{20}$.

$\Rightarrow$ Still competitive under some conditions;
use long frames if required.

Fast-SSC Decoding

What better algorithm? Fast-SSC!

- Orders of magnitude more efficient than SC;
- Maps nicely into SIMD;
- Memory can be arranged in an efficient way;
- Natural fit for systematic coding.

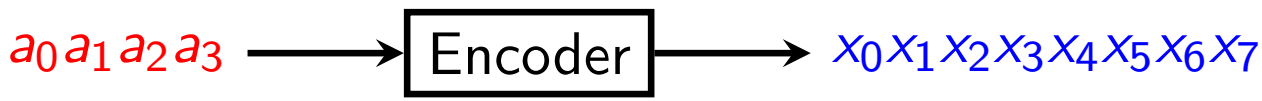
Wait! What is a systematic code?

Sarkis, Giard, Vardy, Thibeault and Gross, "Fast Polar Decoders: Algorithm and Implementation," IEEE JSAC, 2014.

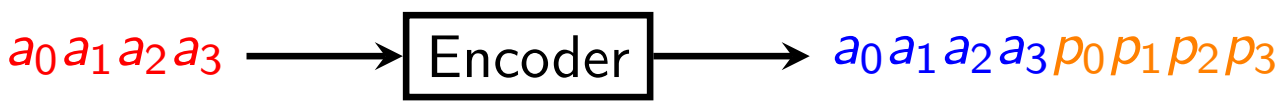
Non-systematic and Systematic Coding

(8, 4) code $\rightarrow$ 8-bit codeword: 4 information bits + 4 parity bits.

- Non-systematic:



- Systematic:



Non-systematic Polar Encoding

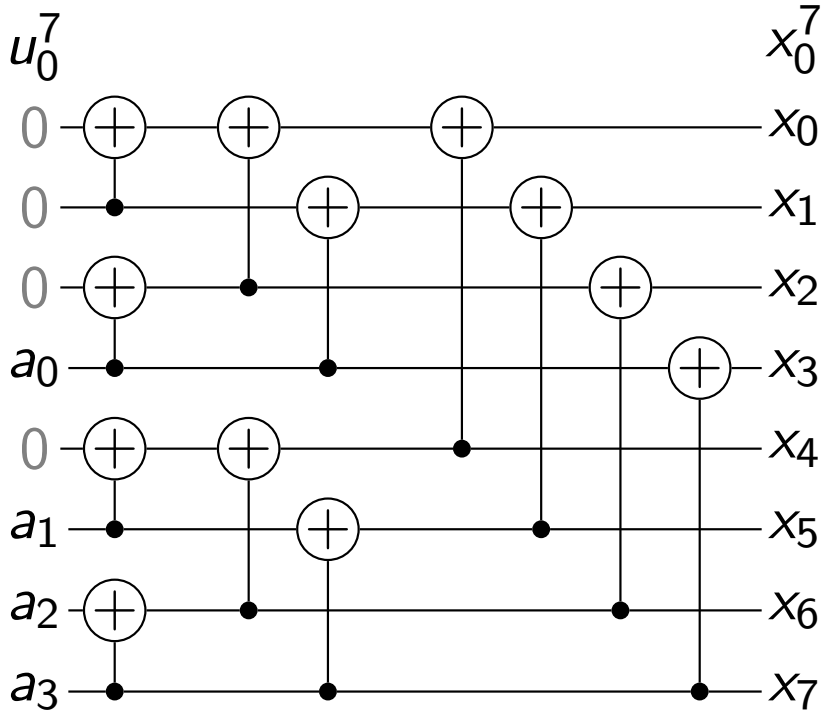


Figure : Structure for a $(8, 4)$ polar code.

Systematic Polar Encoding

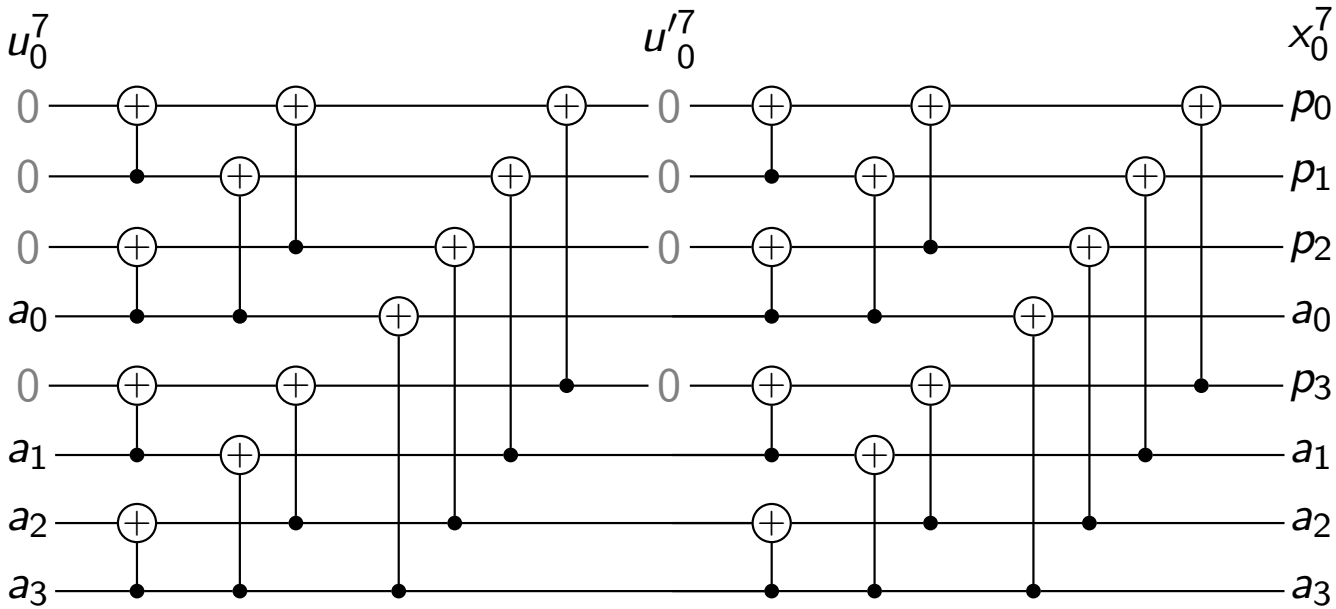


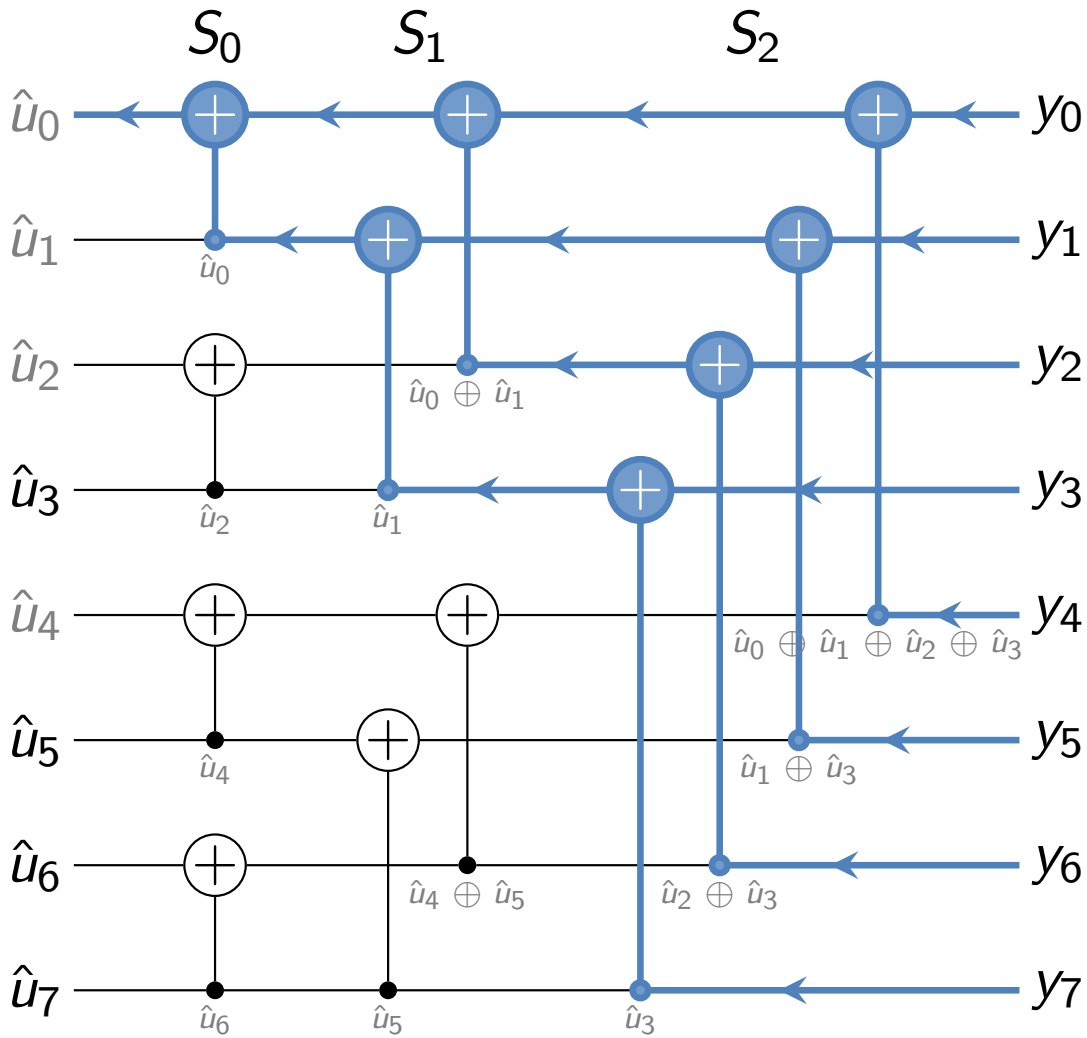
Figure : Low-complexity structure for a $(8, 4)$ polar code.

Alright, now how do we go from SC to Fast-SSC decoding?

Sarkis, Giard, Vardy, Thibeault and Gross, "Fast Polar Decoders: Algorithm and Implementation," IEEE JSAC, 2014.
Sarkis, Tal, Giard, Vardy, Thibeault and Gross "Flexible and Low-Complexity Encoding and Decoding of Systematic Polar Codes," *submitted to* IEEE TCOM and ArXiv, 2015.

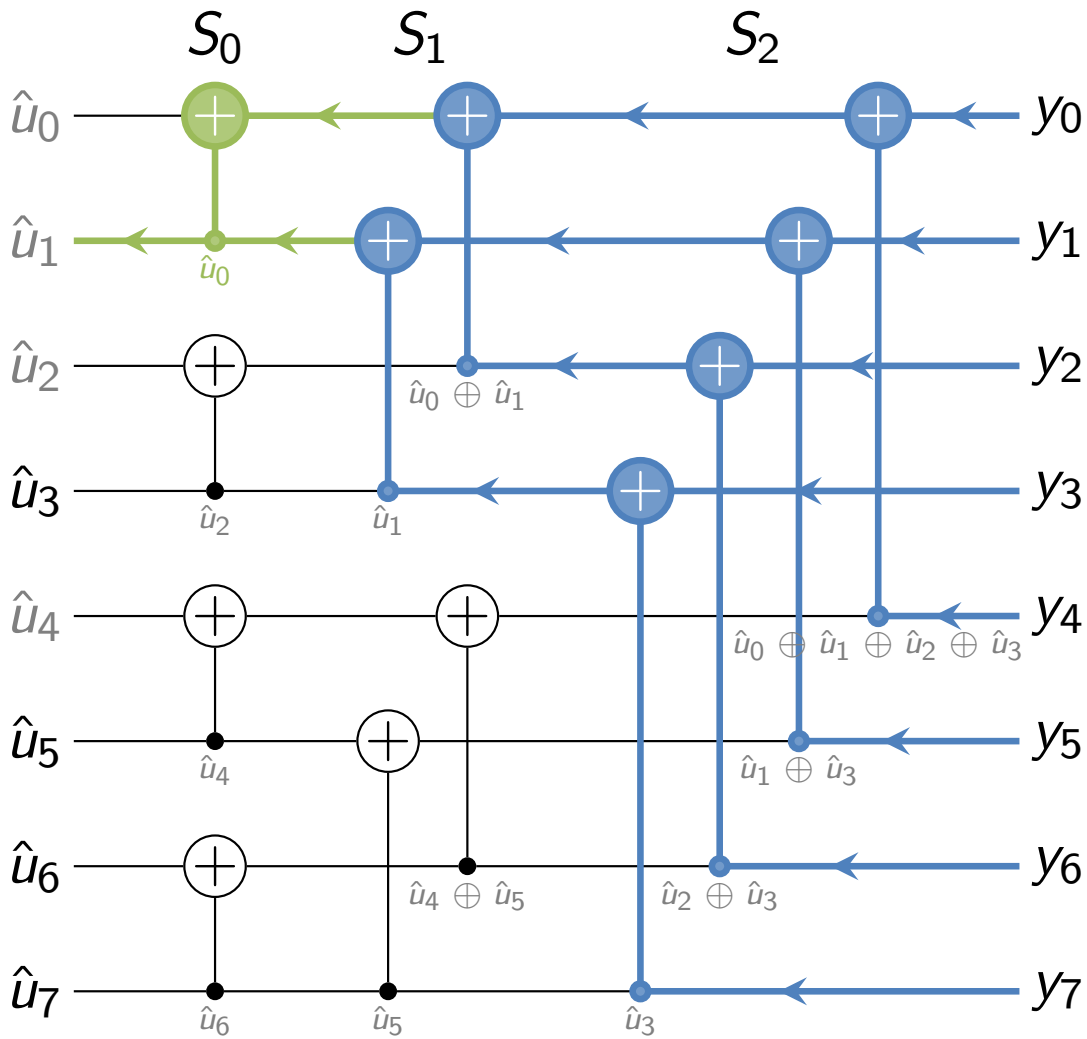
Successive-Cancellation Decoding

- Reminder of how SC decoding works.



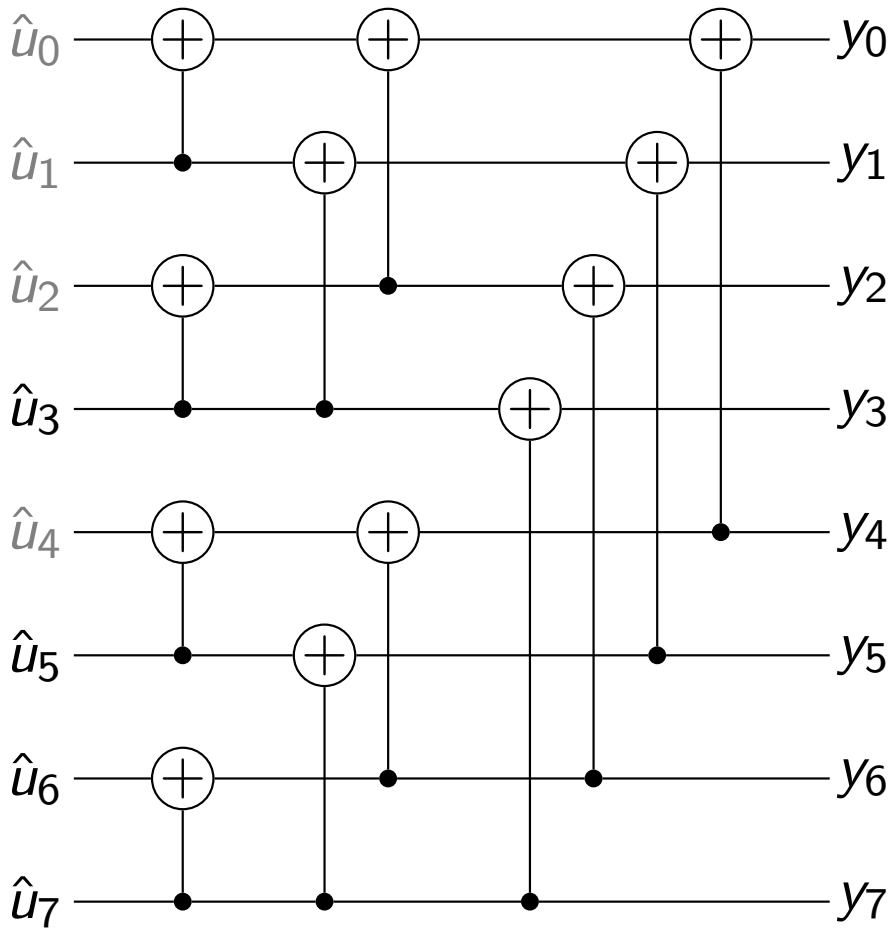
Successive-Cancellation Decoding

- Reminder of how SC decoding works.



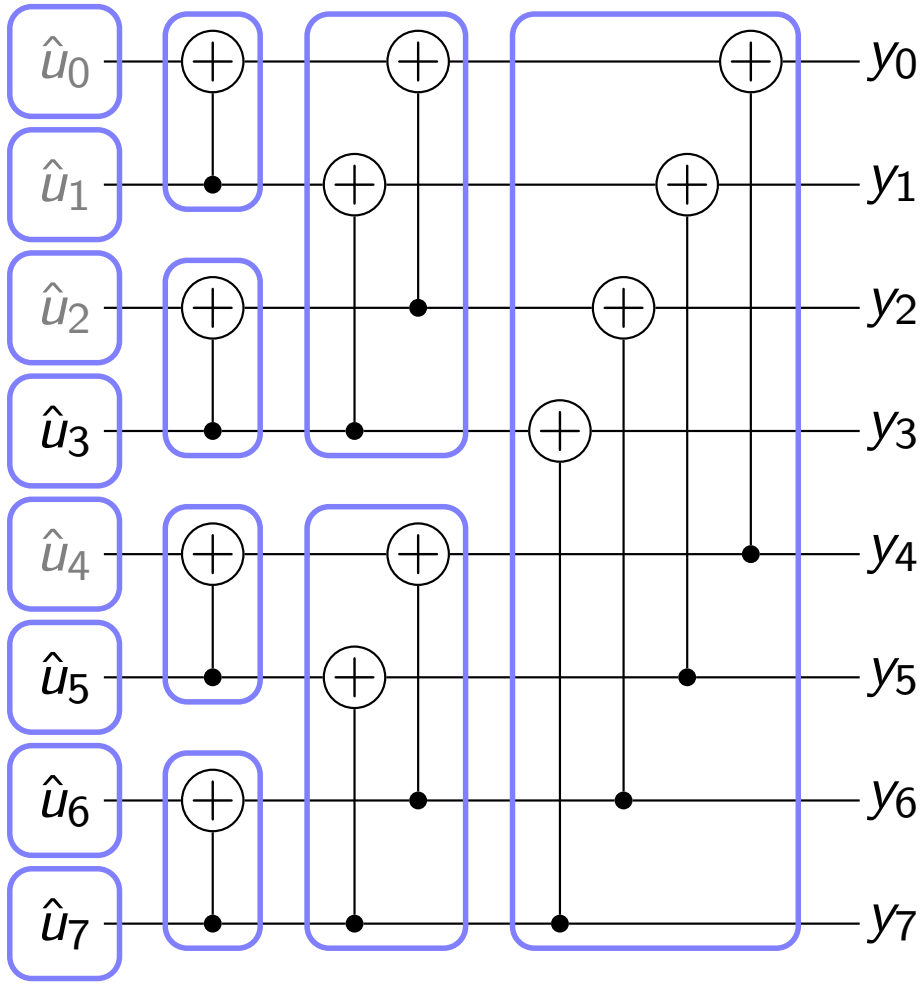
Successive-Cancellation Tree

- View the SC decoder graph as a tree.



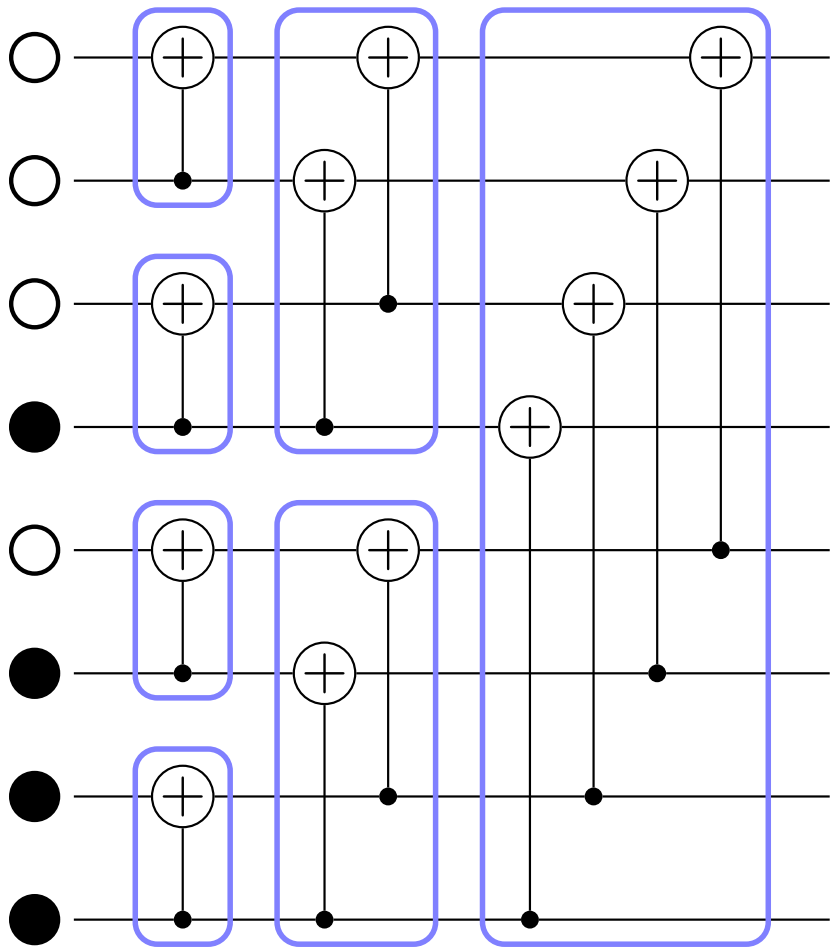
Successive-Cancellation Tree

- View the SC decoder graph as a tree.



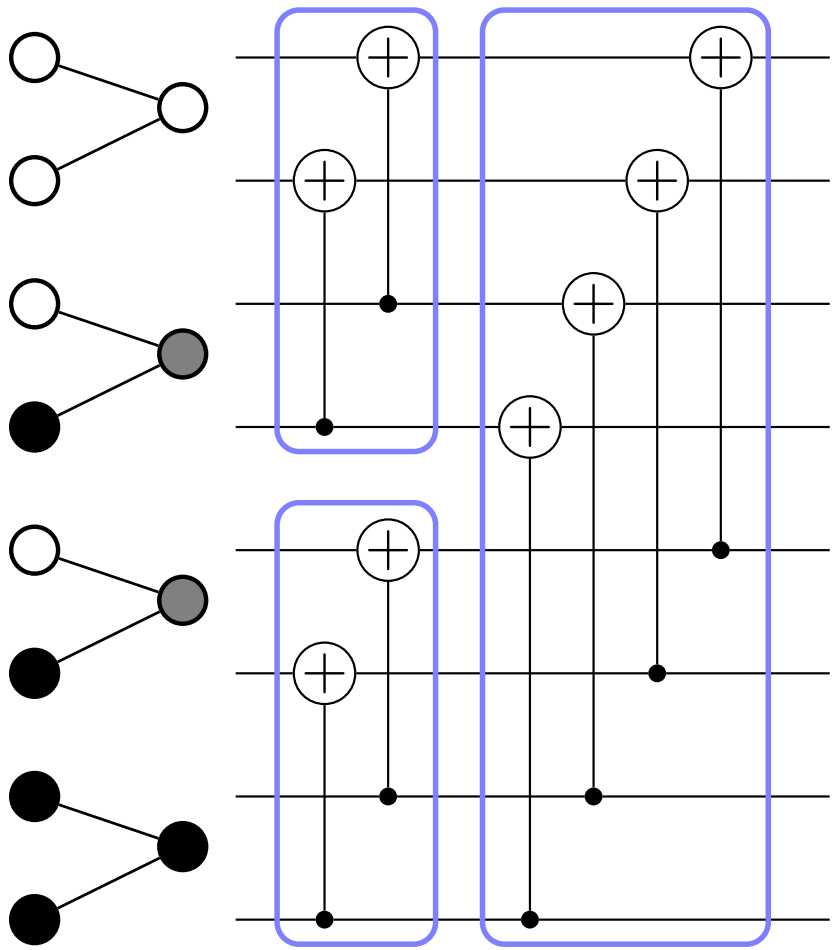
Successive-Cancellation Tree

- View the SC decoder graph as a tree.



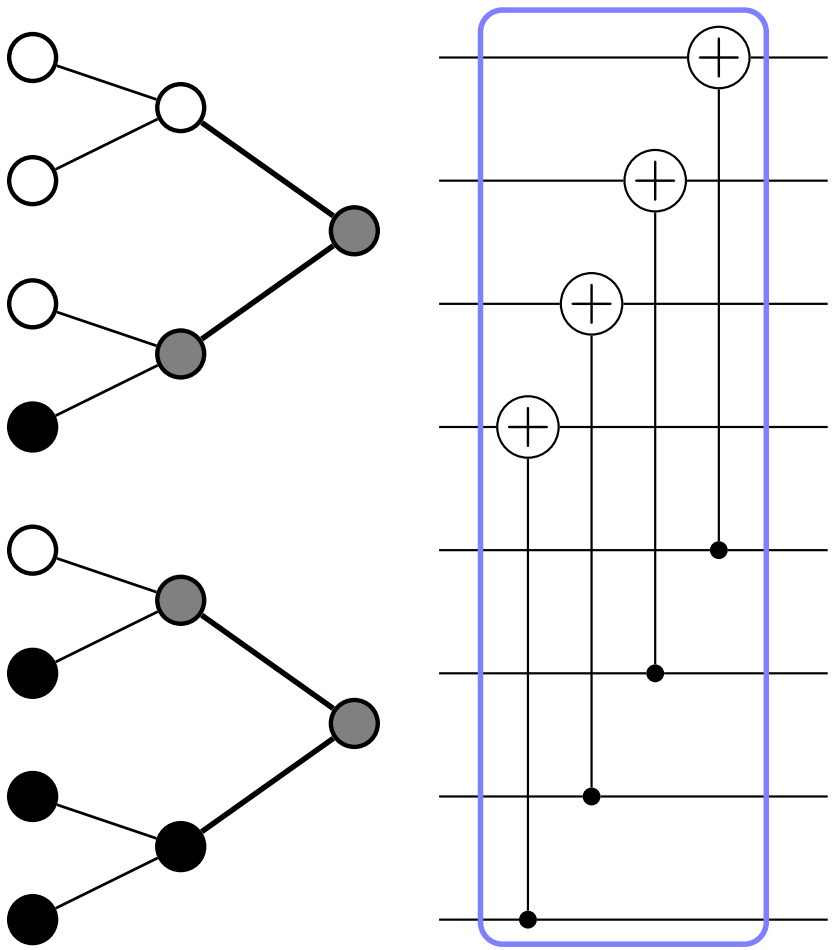
Successive-Cancellation Tree

- View the SC decoder graph as a tree.



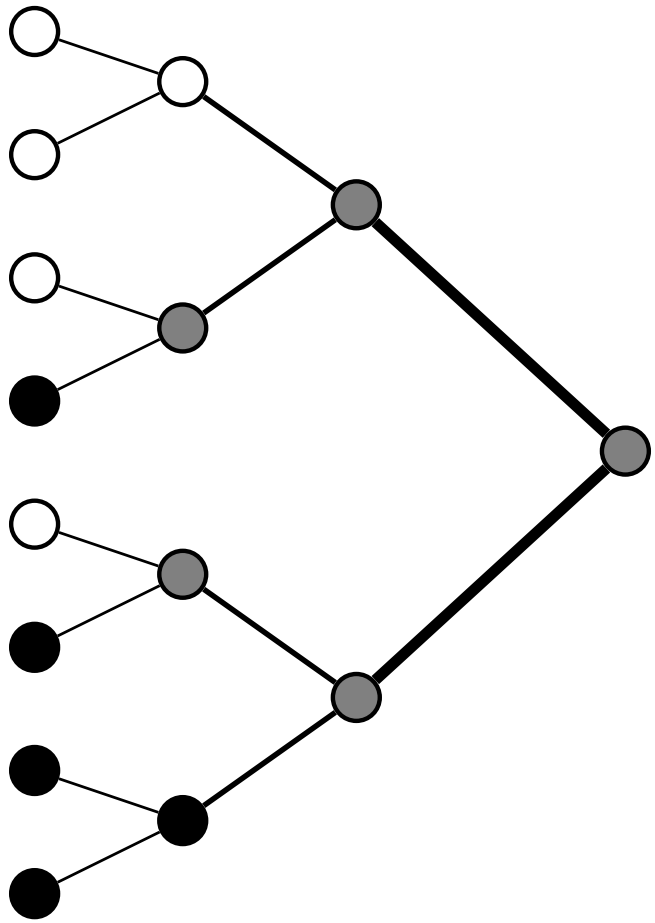
Successive-Cancellation Tree

- View the SC decoder graph as a tree.

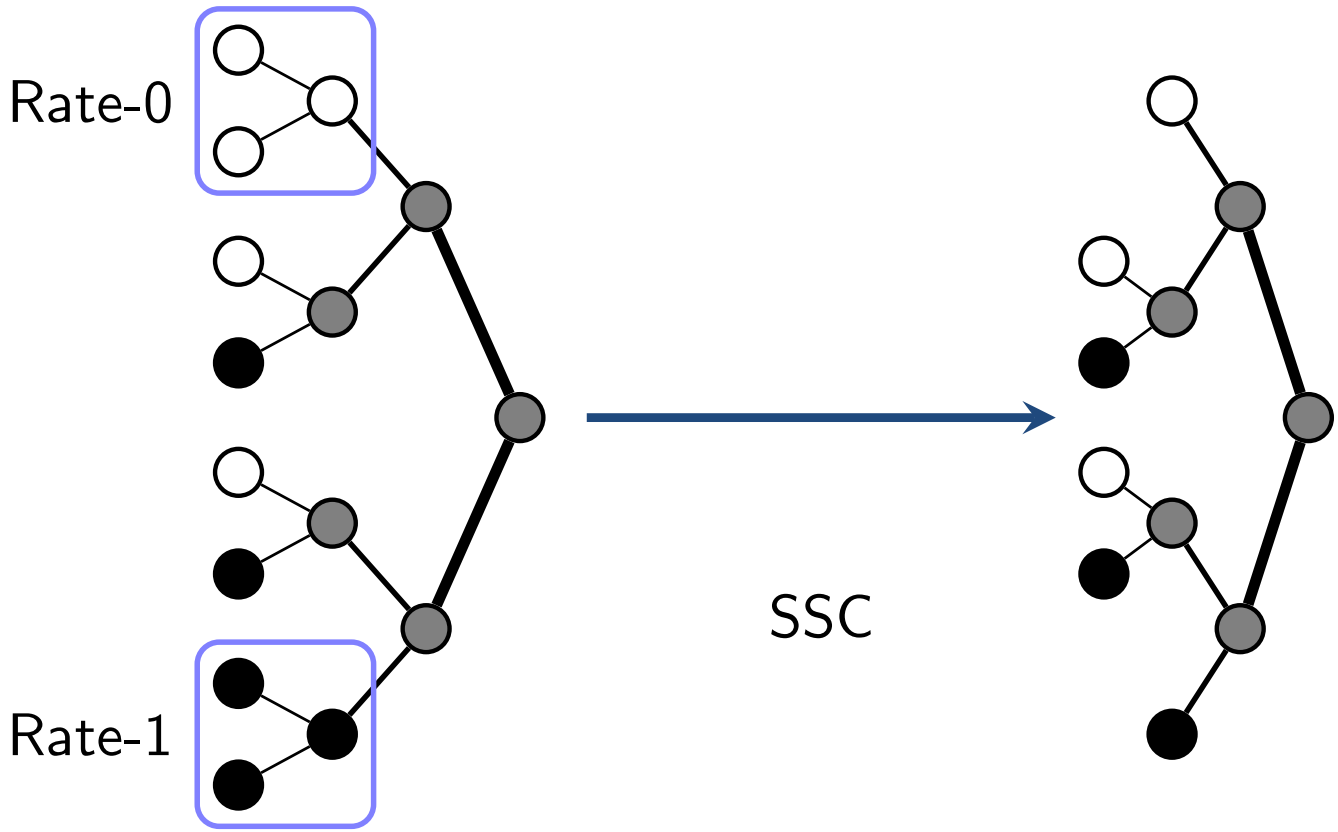


Successive-Cancellation Tree

- View the SC decoder graph as a tree.

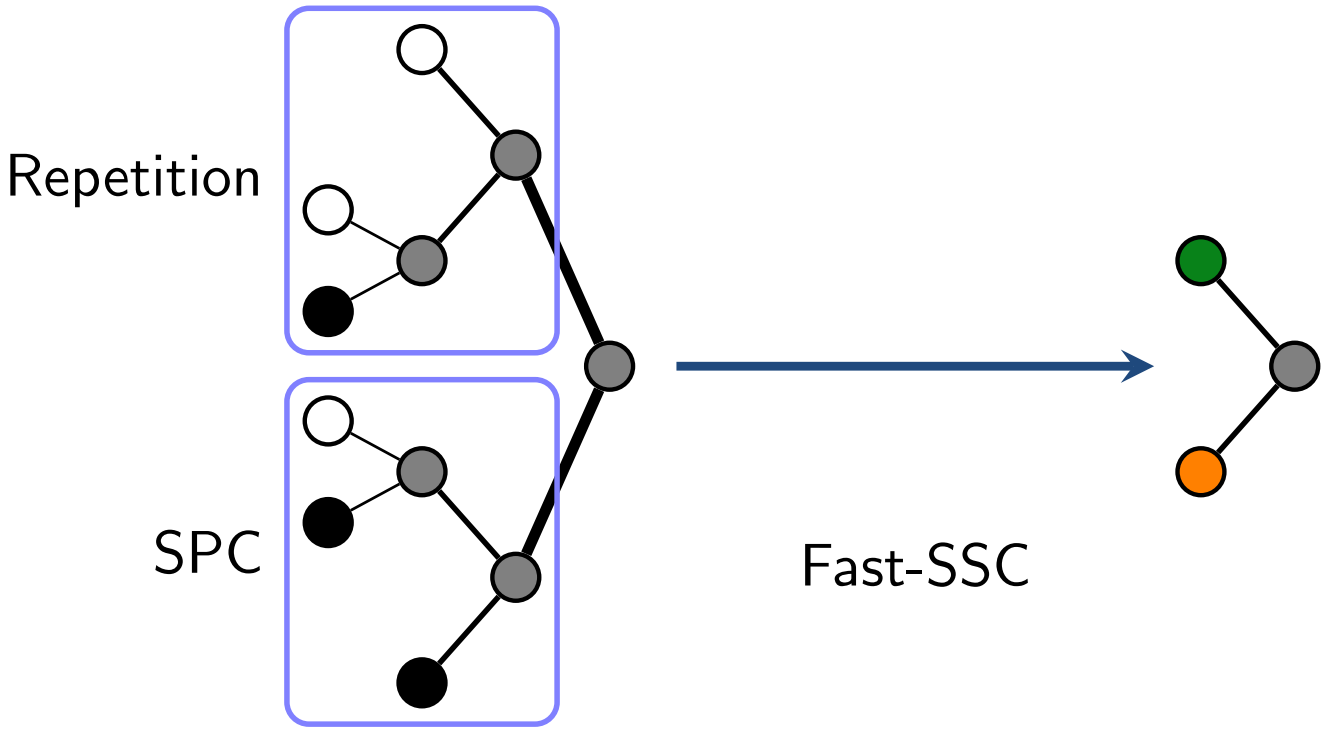


Simplified Successive Cancellation



Alamdar-Yazdi and Kschischang., "A Simplified Successive-Cancellation Decoder for Polar Codes," Comm. Lett., 2011.

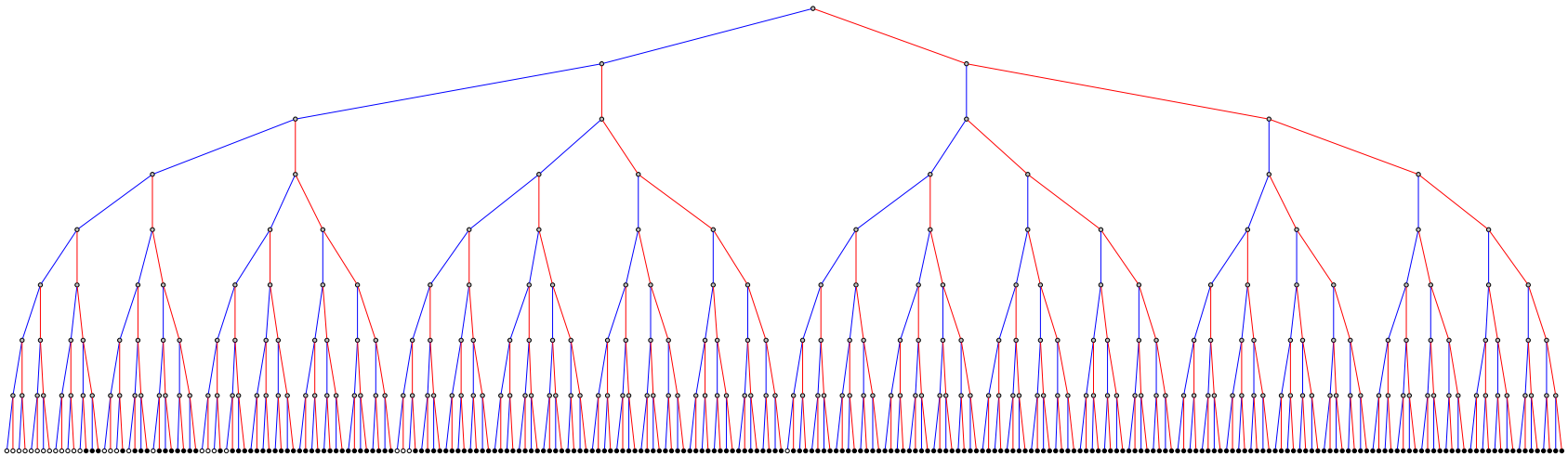
Fast-SSC Decoding



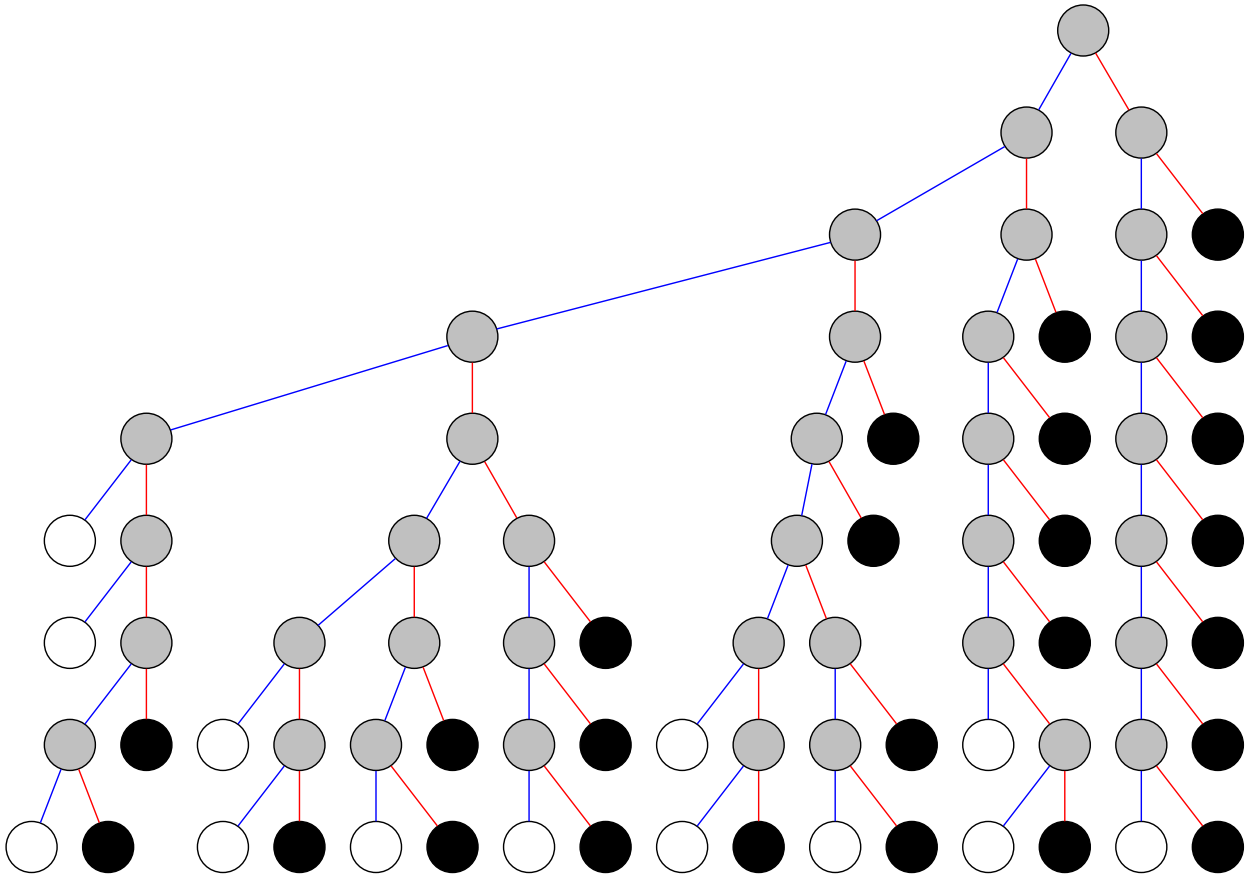
- Key idea: use more efficient algorithms on constituent codes.
- Fast-SSC supports more node types not covered here.

Sarkis, Giard, Vardy, Thibeault and Gross, "Fast Polar Decoders: Algorithm and Implementation," IEEE JSAC, 2014.

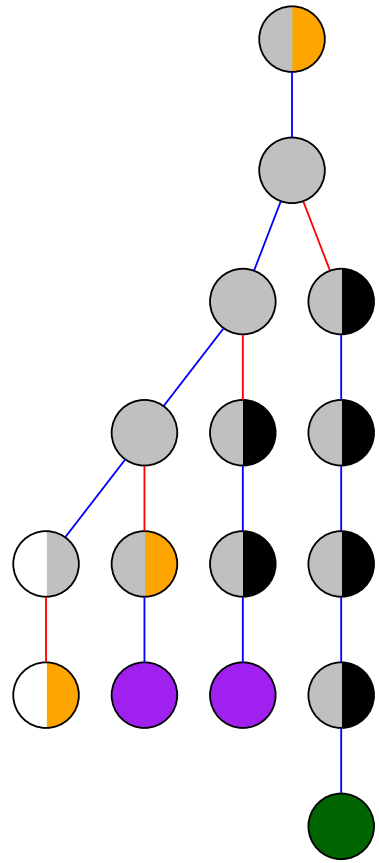
SC Tree—(256, 230)



SSC and Fast-SSC Trees—(256, 230)



SSC

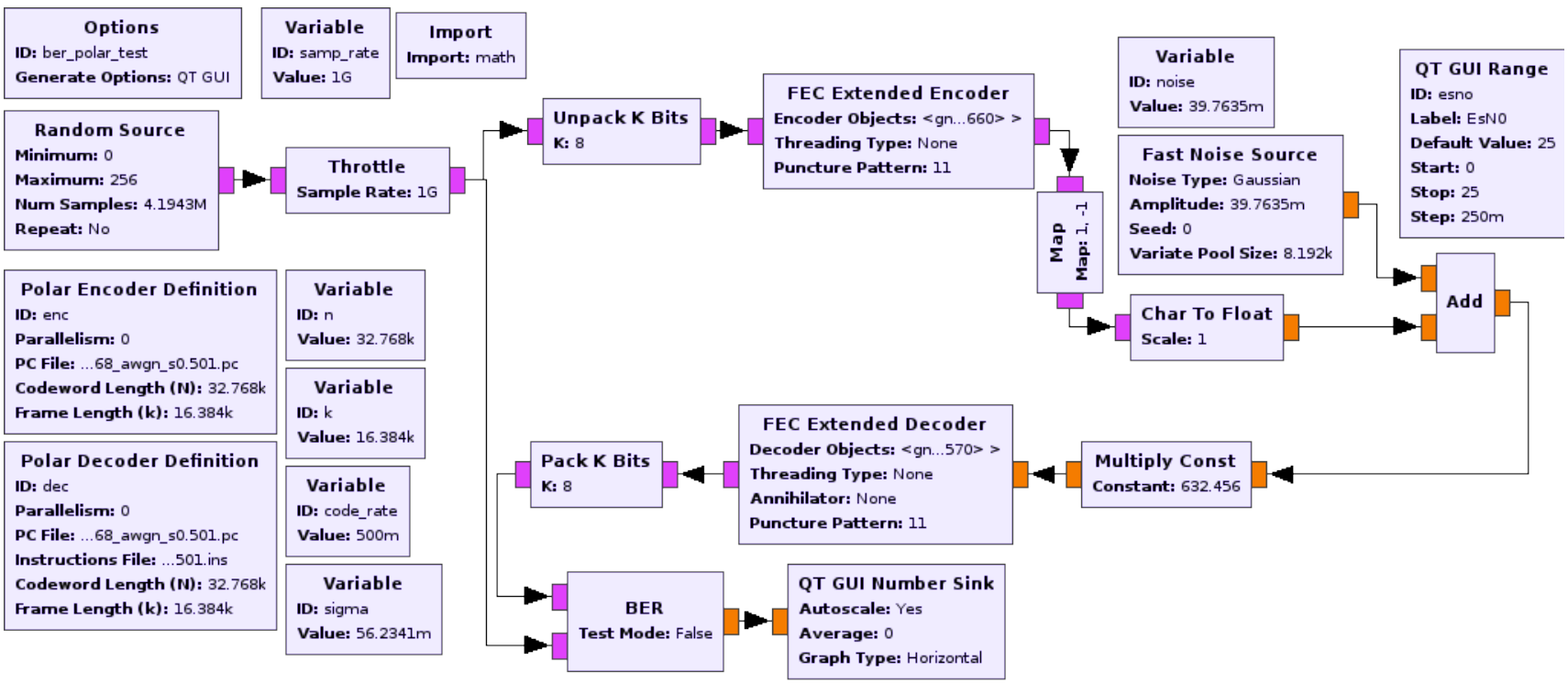


Fast-SSC

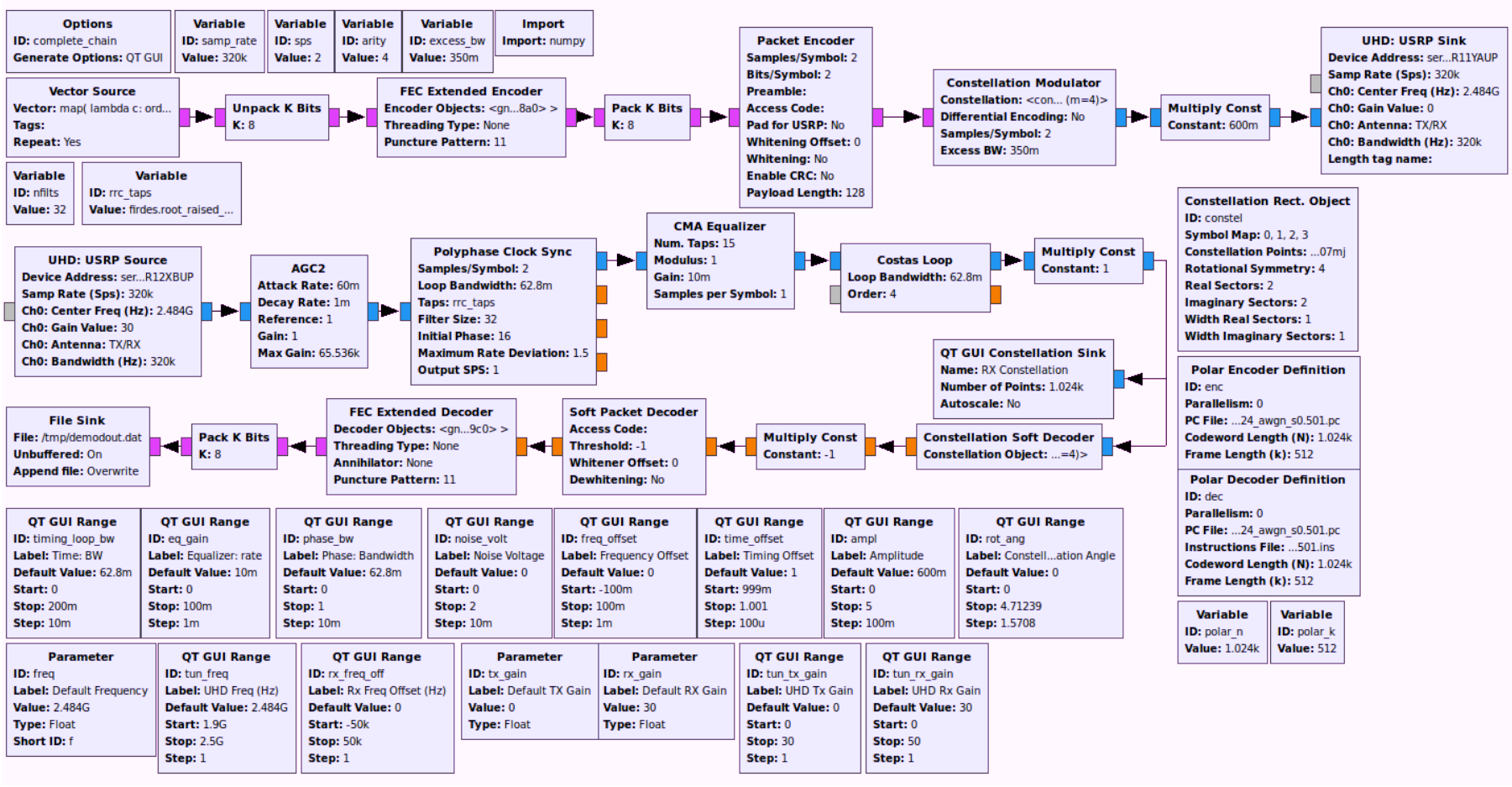
Implemention for GNU Radio

- Systematic polar coding for better BER.
- Supports a subset of the Fast-SSC nodes.
- Uses the FEC API.
- Supports soft inputs for greater error-correction performance.
- Uses fixed-point arithmetic internally.
- Highly-optimized VOLK kernels for SSE4, AVX2 and NEON.

Classical GRC Example



Complete Chain Example



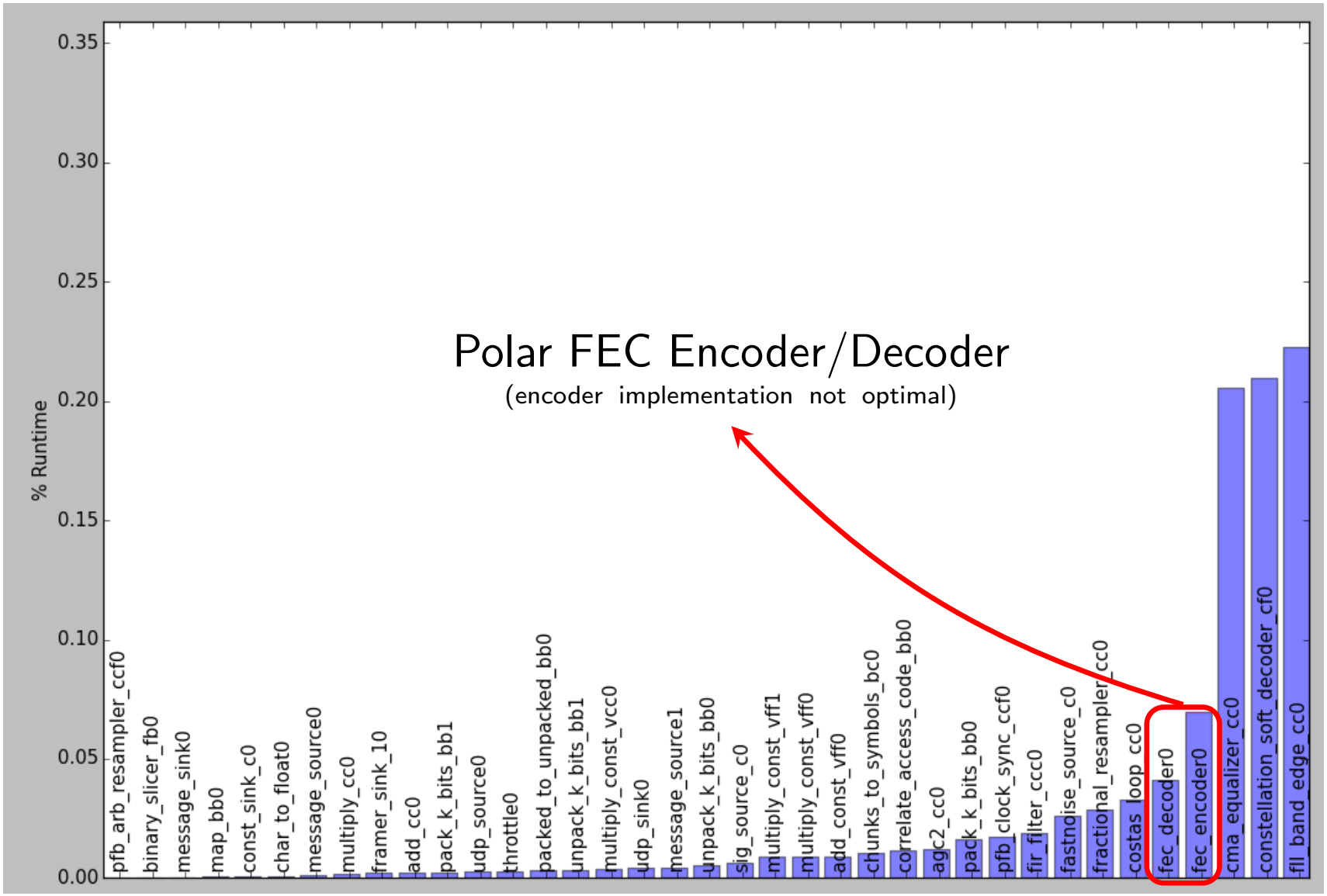
Performance Results

Using a single processor core:

- Under the worst conditions, a stable 50 Mbps on a low-power SandyBridge with SSE4.1.
- Under good conditions, an average coded throughput of 350 Mbps on a Haswell with AVX2 is typical.

$\Rightarrow$ In a complete communication chain, the modern FEC is NOT the bottleneck anymore.

Performance Results Example



Final Words

- Will this be upstreamed to GNU Radio/VOLK?
- Will we make the source code available?
- When can you start using it?
- What if you want to use polar codes outside GNU Radio?

Acknowledgement

The authors would like to warmly thank:

- Profs. Ghyslain Gagnon and Claude Thibeault of ETS for funding this trip.
- Prof. François Gagnon of ETS for HW access and helpful discussions.
- Gabi Sarkis of McGill university for helpful discussions and presentation material.

Contact information

- Pascal.Giard@mail.mcgill.ca
- Alexandre.Raymond@lacime.etsmtl.ca