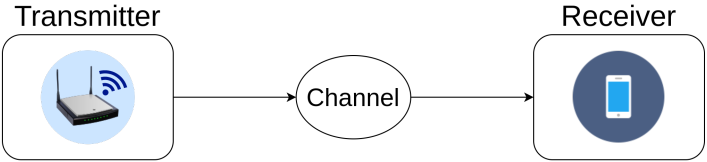
High-Throughput does not Compromise Energy Efficiency:

New Algorithms and Implementations for 5G Polar Codes

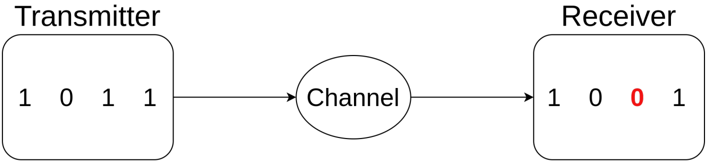
Furkan Ercan, Ph.D.
5G PHY Developer - Octasic Inc.
Vice Chair - IEEE Montréal Section
Montréal, QC, Canada

May 6, 2021

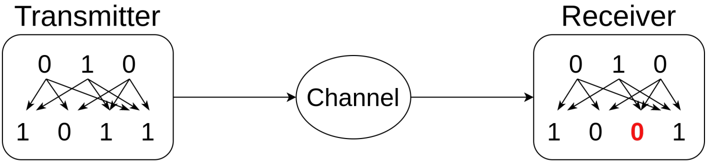
Communications & Channel Capacity



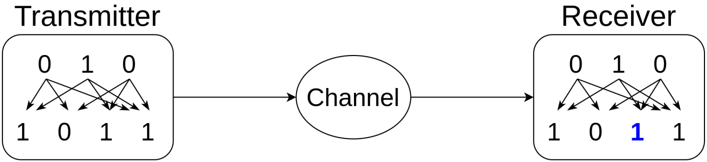
Communications & Channel Capacity



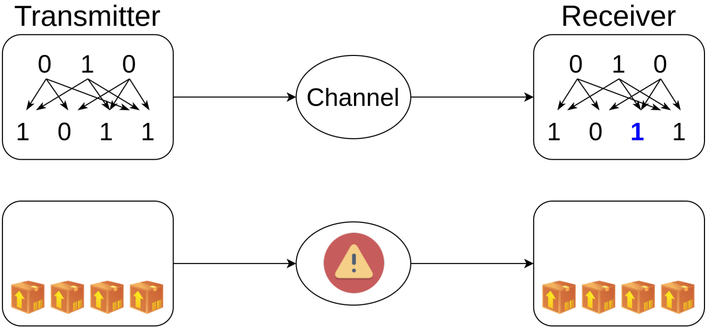
Communications & Channel Capacity



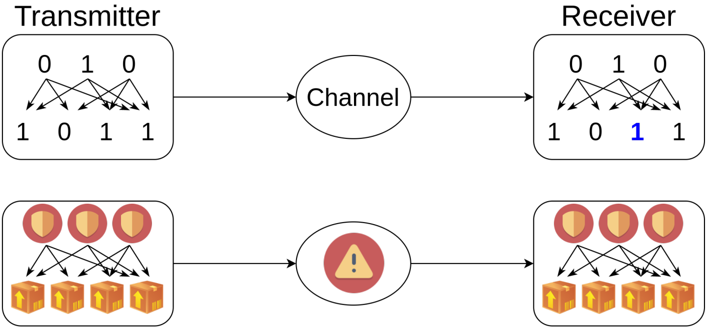
Communications & Channel Capacity



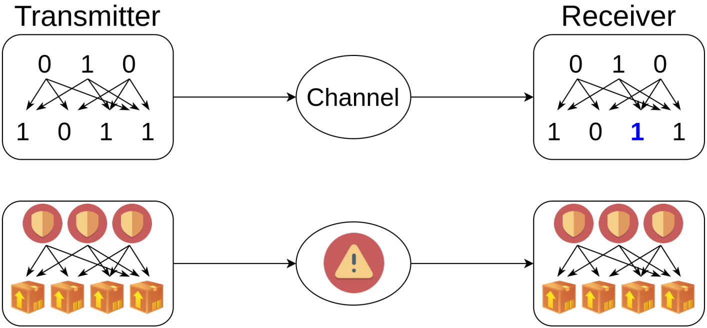
Communications & Channel Capacity



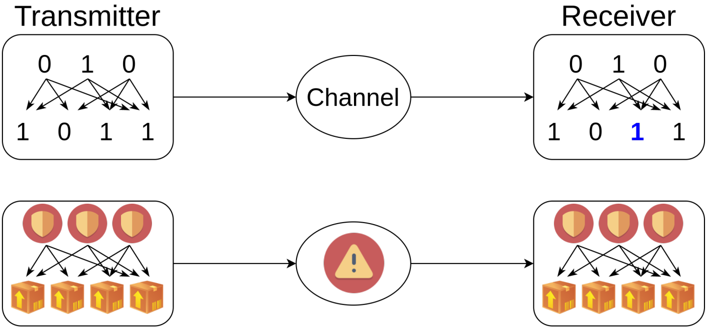
Communications & Channel Capacity

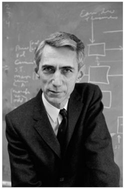


Communications & Channel Capacity



Communications & Channel Capacity





Channel Capacity:
Given channel condition, the maximum rate of information

Evolution of Digital Communications

- How to achieve the channel capacity?
- Various channel coding algorithms emerged (e.g. LDPC, Turbo).
- Polar codes provably achieve the channel capacity.
- They are involved as a coding scheme in 5G standard.

Evolution of Digital Communications

- How to achieve the channel capacity?
- Various channel coding algorithms emerged (e.g. LDPC, Turbo).
- Polar codes provably achieve the channel capacity.
- They are involved as a coding scheme in 5G standard.

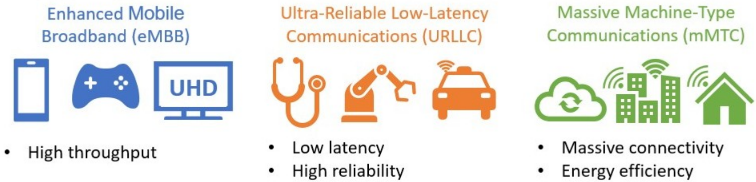
Evolution of Digital Communications

- How to achieve the channel capacity?
- Various channel coding algorithms emerged (e.g. LDPC, Turbo).
- Polar codes provably achieve the channel capacity.
- They are involved as a coding scheme in 5G standard.

Evolution of Digital Communications

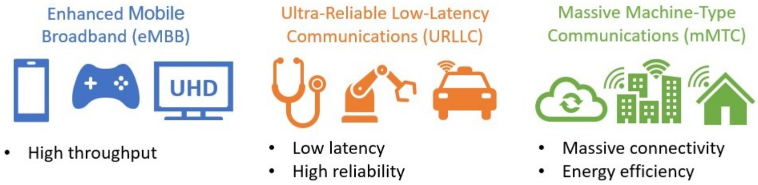
- How to achieve the channel capacity?
- Various channel coding algorithms emerged (e.g. LDPC, Turbo).
- Polar codes provably achieve the channel capacity.
- They are involved as a coding scheme in 5G standard.

5G Use Cases



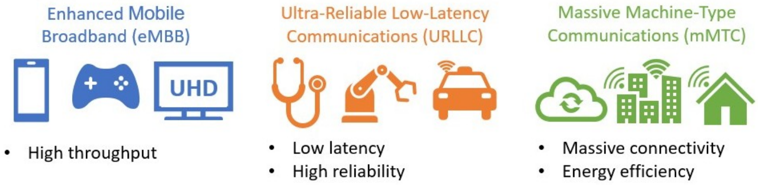
- 5G prioritizes various targets based on the use case.
- Polar codes are involved in 5G eMBB control channel.
- Currently, polar codes are considerable candidates for other use cases.
- Fast, practical, energy-efficient polar decoders are essential.

5G Use Cases



- 5G prioritizes various targets based on the use case.
- Polar codes are involved in 5G eMBB control channel.
- Currently, polar codes are considerable candidates for other use cases.
- Fast, practical, energy-efficient polar decoders are essential.

5G Use Cases



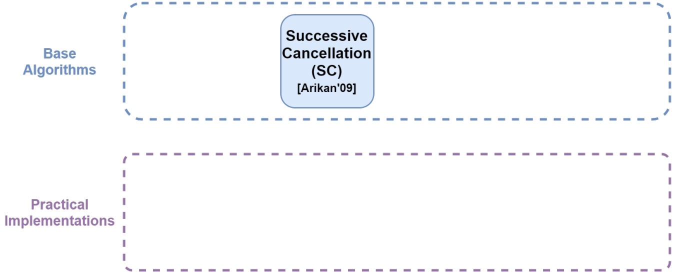
- 5G prioritizes various targets based on the use case.
- Polar codes are involved in 5G eMBB control channel.
- Currently, polar codes are considerable candidates for other use cases.
- Fast, practical, energy-efficient polar decoders are essential.

An Overview of Polar Decoders

Base
Algorithms

Practical
Implementations

An Overview of Polar Decoders

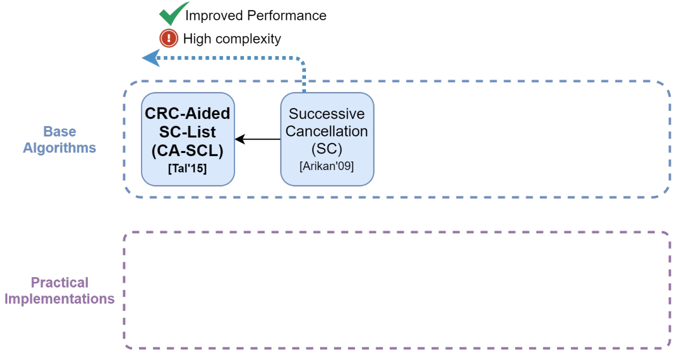


Successive Cancellation (SC) Decoding *

- ✓ Simple decoding
- ✗ Mediocre performance at practical lengths
- ✗ Sequential, long latency

* E. Arikan, "Channel Polarization: A Method for Constructing Capacity-Achieving Codes for Symmetric Binary-Input Memoryless Channels," in IEEE Tran. Inf. Theory, vol. 55, no. 7, pp. 3051-3073, July 2009.

An Overview of Polar Decoders



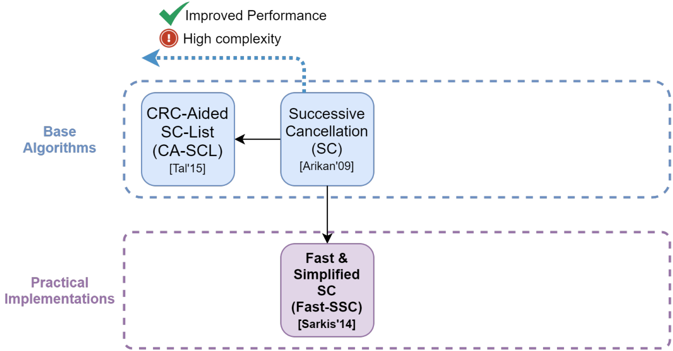
SC-List (SCL) Decoding *

✓ Improved error-correction performance

✗ Increased complexity

* I. Tal and A. Vardy, "List Decoding of Polar Codes," in IEEE TIT, May 2015.

An Overview of Polar Decoders



Fast-SSC Decoding *

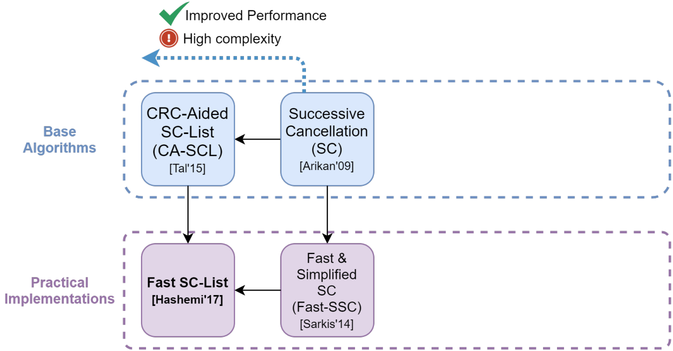
✓ $\approx 10\times$ less latency

- No error correction performance degradation

---

* G. Sarkis et al., "Fast Polar Decoders: Algorithm and Implementation," in IEEE JSAC, May 2014.

An Overview of Polar Decoders

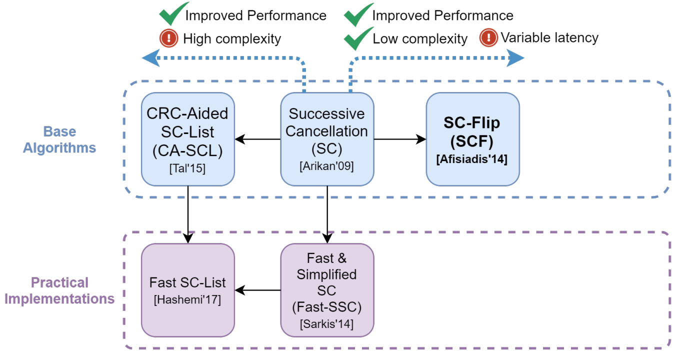


Fast-SC-List Decoding *

- ✓ Latency reduced further
- ✗ Very high complexity

* S. A. Hashemi et al. "Fast and Flexible Successive-Cancellation List Decoders for Polar Codes," in IEEE TSP, Nov. 2017.

An Overview of Polar Decoders



SC-Flip (SCF) Decoding *

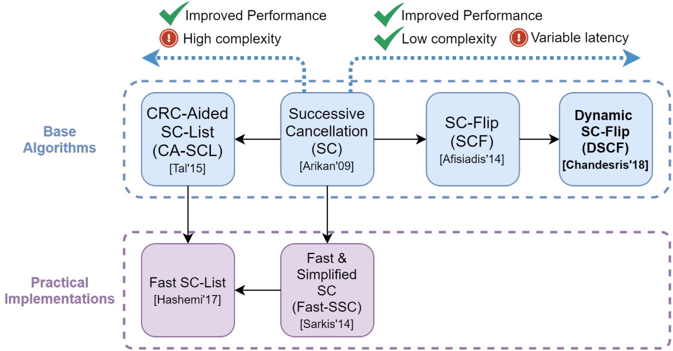
✓ Improved error-correction performance

✓ Low complexity

✗ Variable latency

* O. Afisiadis et al. "A low-complexity improved successive cancellation decoder for polar codes," 2014 48th Asilomar Conference, 2014.

An Overview of Polar Decoders

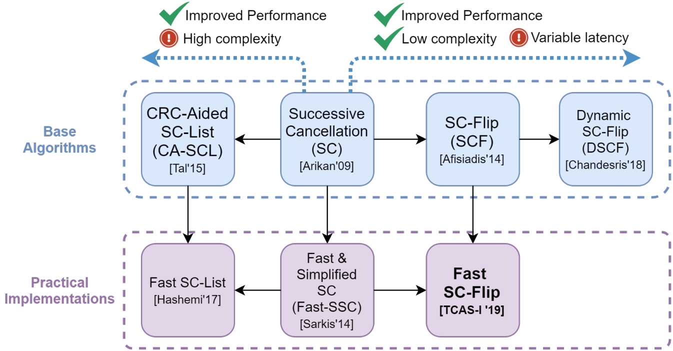


Dynamic SC-Flip (DSCF) Decoding *

- ✓ Improved error-correction performance
- ✗ Not practical due to complex computations

* L. Chandesris et al. "Dynamic-SCFlip Decoding of Polar Codes," in IEEE TCOM, June 2018.

An Overview of Polar Decoders



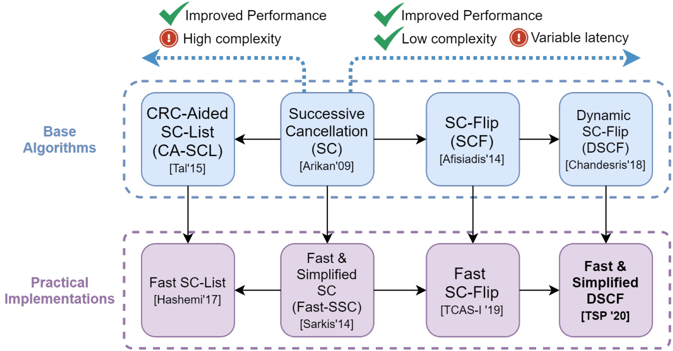
Fast SC-Flip Decoding *

✓ Introduced fast computations

✓ Energy-efficient implementation

* F. Ercan, et al. "Energy-Efficient Hardware Architectures for Fast Polar Decoders," in IEEE TCAS-I, Jan. 2020.

An Overview of Polar Decoders

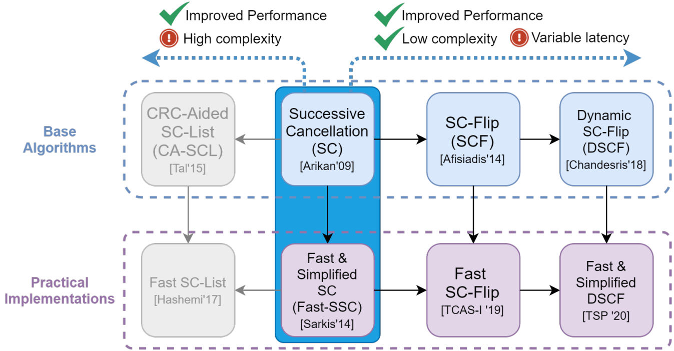


Practical Dynamic SC-Flip Decoding *

- ✓ Replaced complex computations with simple approximations
- ✓ Introduced fast computations
- ✓ First hardware implementation

* F. Ercan et al. "Practical Dynamic SC-Flip Polar Decoders: Algorithm and Implementation," in IEEE TSP, 2020.

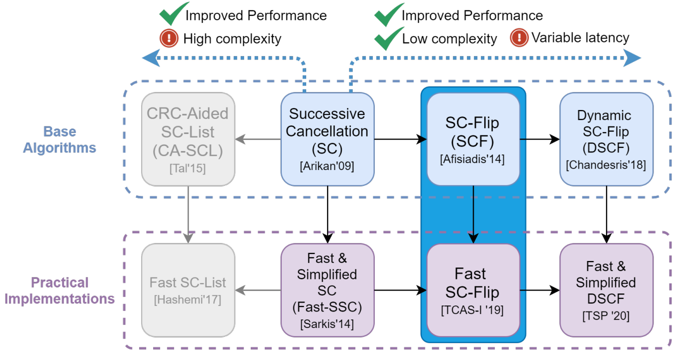
An Overview of Polar Decoders



Overview of This Talk

- Part I: Background on SC Decoding
-
-

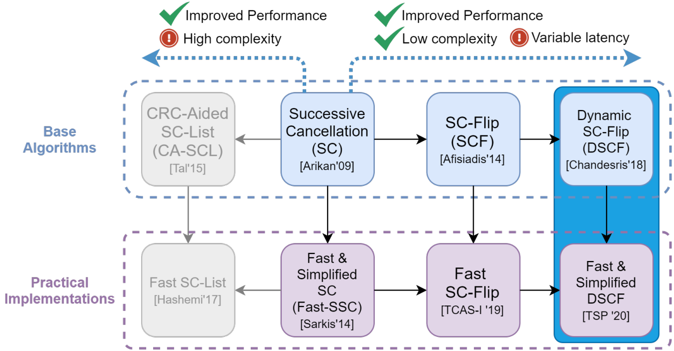
An Overview of Polar Decoders



Overview of This Talk

- Part I: Background on SC Decoding
- Part II: Realizing the SC-Flip Algorithm in Hardware
-

An Overview of Polar Decoders



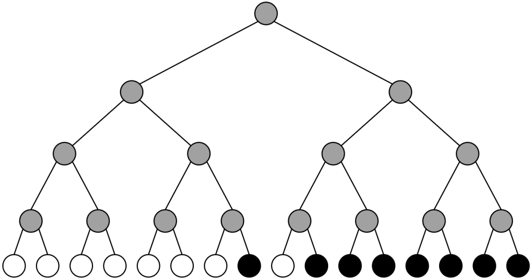
Overview of This Talk

- Part I: Background on SC Decoding
- Part II: Realizing the SC-Flip Algorithm in Hardware
- Part III: Making Dynamic SC-Flip Decoding Practical

Part I:
Preliminaries

SC Tree and Special Nodes

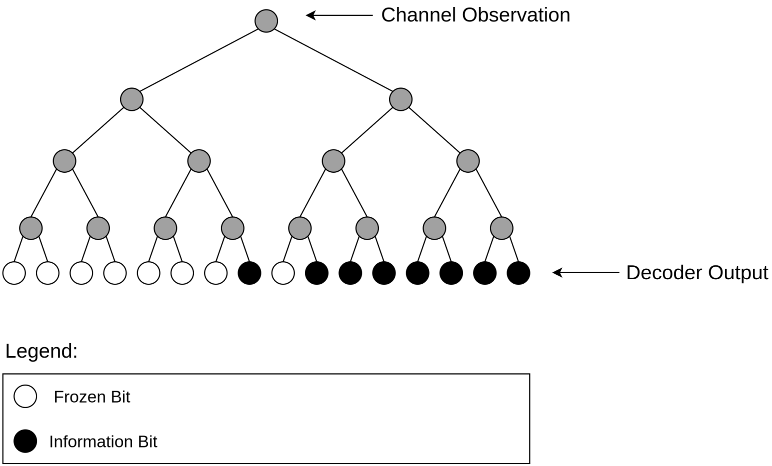
Example: PC(16,8)



Legend:

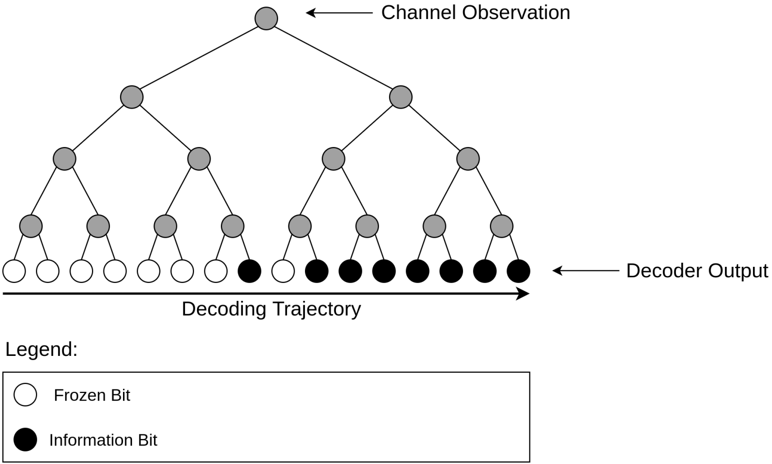
SC Tree and Special Nodes

Example: PC(16,8)



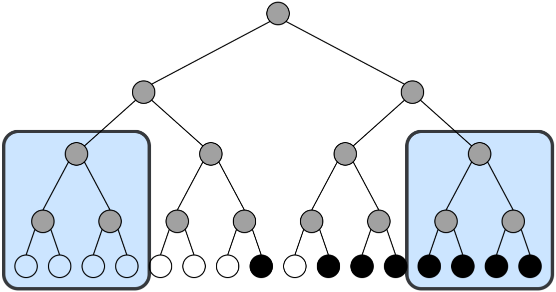
SC Tree and Special Nodes

Example: PC(16,8)



SC Tree and Special Nodes

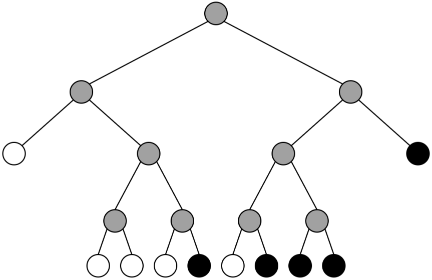
Example: PC(16,8)



Legend:

SC Tree and Special Nodes

Example: PC(16,8)



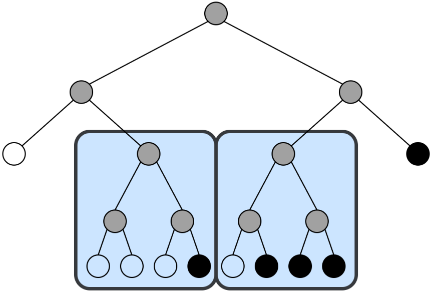
Legend:

Frozen Bit (Rate-0 Node)

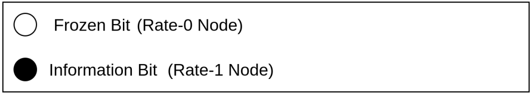
Information Bit (Rate-1 Node)

SC Tree and Special Nodes

Example: PC(16,8)

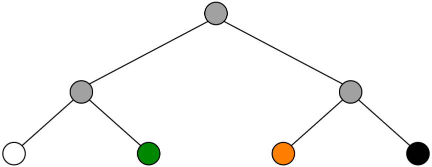


Legend:



SC Tree and Special Nodes

Example: PC(16,8)



Legend:

- Frozen Bit (Rate-0 Node)
- Repetition Node
- Information Bit (Rate-1 Node)
- Single Parity Check Node

Fast Decoding v Energy Efficiency

- Energy = Power $\times$ Delay

Fast Decoding v Energy Efficiency

- Energy = Power $\times$ Delay

Table: SC Decoding, 65nm CMOS, PC(1024,512)

	SC[1]	Fast-SSC[2]
Power (mW)	51	160
Latency (μs)	12.7	0.6
Throughput (Mbps)	81	1719
Energy (pJ/bit)	1262	188

* Normalized for 65 nm CMOS.

[1] Giard et al., "PolarBear: A 28-nm FD-SOI ASIC for Decoding of Polar Codes," in IEEE JETCAS, 2017.

[2] Sarkis et al.,"Fast Polar Decoders: Algorithm and Implementation," in IEEE JSAC, 2014.

Fast Decoding v Energy Efficiency

- Energy = Power $\times$ Delay

Table: SC Decoding, 65nm CMOS, PC(1024,512)

	SC[1]	Fast-SSC[2]
Power (mW)	51	160
Latency (μs)	12.7	0.6
Throughput (Mbps)	81	1719
Energy (pJ/bit)	1262	188

* Normalized for 65 nm CMOS.

$\leftarrow 3.1\times$

[1] Giard et al., "PolarBear: A 28-nm FD-SOI ASIC for Decoding of Polar Codes," in IEEE JETCAS, 2017.

[2] Sarkis et al.,"Fast Polar Decoders: Algorithm and Implementation," in IEEE JSAC, 2014.

Fast Decoding v Energy Efficiency

- Energy = Power × Delay

Table: SC Decoding, 65nm CMOS, PC(1024,512)

	SC[1]	Fast-SSC[2]	
Power (mW)	51	160	← 3.1×
Latency (μs)	12.7	0.6	← 21×
Throughput (Mbps)	81	1719	
Energy (pJ/bit)	1262	188	

* Normalized for 65 nm CMOS.

[1] Giard et al., "PolarBear: A 28-nm FD-SOI ASIC for Decoding of Polar Codes," in IEEE JETCAS, 2017.

[2] Sarkis et al.,"Fast Polar Decoders: Algorithm and Implementation," in IEEE JSAC, 2014.

Fast Decoding v Energy Efficiency

- Energy = Power $\times$ Delay

Table: SC Decoding, 65nm CMOS, PC(1024,512)

	SC[1]	Fast-SSC[2]	
Power (mW)	51	160	$\leftarrow 3.1\times$
Latency (μs)	12.7	0.6	$\leftarrow 21\times$
Throughput (Mbps)	81	1719	
Energy (pJ/bit)	1262	188	$\leftarrow 6.7\times$

* Normalized for 65 nm CMOS.

[1] Giard et al., "PolarBear: A 28-nm FD-SOI ASIC for Decoding of Polar Codes," in IEEE JETCAS, 2017.

[2] Sarkis et al.,"Fast Polar Decoders: Algorithm and Implementation," in IEEE JSAC, 2014.

Fast Decoding v Energy Efficiency

- Energy = Power $\times$ Delay

Table: CA-SCL Decoding, 65nm CMOS, PC(512,256), $L = 2$

	SCL[3]	Fast-SSCL[4]
Power (mW)	75.3	119.7
Latency (μs)	1.71	0.43
Throughput (Mbps)	300	1201
Energy (pJ/bit)	502	199

[3] Stimming et al., "LLR-Based Successive Cancellation List Decoding of Polar Codes," in IEEE TSP, 2015.

[4] Hashemi et al.,"Fast and Flexible Successive-Cancellation List Decoders for Polar Codes," in IEEE TSP, 2017.

Fast Decoding v Energy Efficiency

- Energy = Power $\times$ Delay

Table: CA-SCL Decoding, 65nm CMOS, PC(512,256), $L = 2$

	SCL[3]	Fast-SSCL[4]	
Power (mW)	75.3	119.7	$\leftarrow$ 59%
Latency (μs)	1.71	0.43	$\leftarrow 4\times$
Throughput (Mbps)	300	1201	
Energy (pJ/bit)	502	199	$\leftarrow 2.5\times$

- Fast decoding: Huge gains in latency $>$ Penalty in power

[3] Stimming et al., "LLR-Based Successive Cancellation List Decoding of Polar Codes," in IEEE TSP, 2015.

[4] Hashemi et al.,"Fast and Flexible Successive-Cancellation List Decoders for Polar Codes," in IEEE TSP, 2017.

Fast Decoding v Energy Efficiency

- Energy = Power $\times$ Delay

Table: CA-SCL Decoding, 65nm CMOS, PC(512,256), $L = 2$

	SCL[3]	Fast-SSCL[4]	
Power (mW)	75.3	119.7	$\leftarrow$ 59%
Latency (μs)	1.71	0.43	$\leftarrow 4\times$
Throughput (Mbps)	300	1201	
Energy (pJ/bit)	502	199	$\leftarrow 2.5\times$

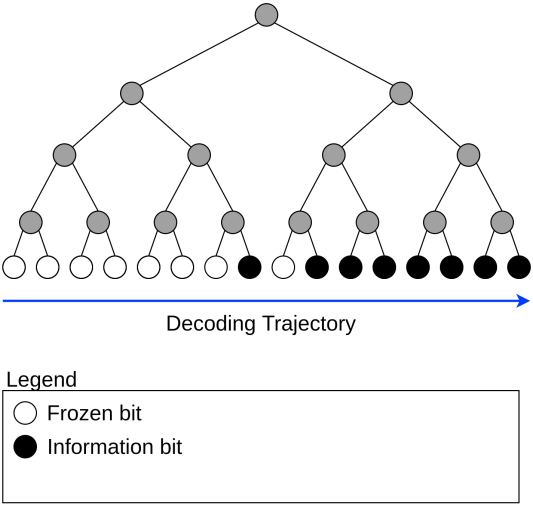
- Fast decoding: Huge gains in latency $>$ Penalty in power
- Motivation: Even more energy efficient with SCF
  - All while being competitive in error correction performance & T/P!

[3] Stimming et al., "LLR-Based Successive Cancellation List Decoding of Polar Codes," in IEEE TSP, 2015.

[4] Hashemi et al.,"Fast and Flexible Successive-Cancellation List Decoders for Polar Codes," in IEEE TSP, 2017.

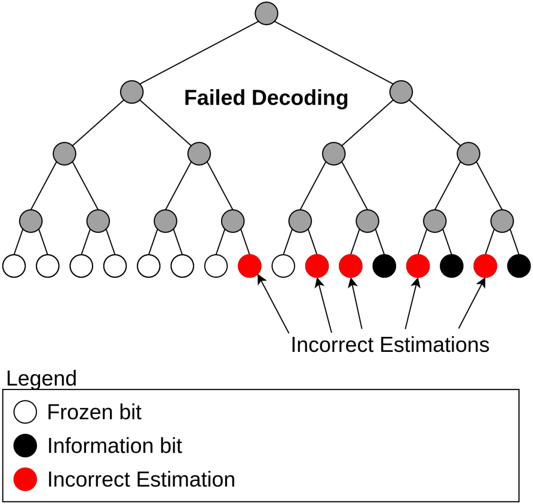
Part II:
Realizing the SC-Flip Algorithm

SC-Flip (SCF) Decoding



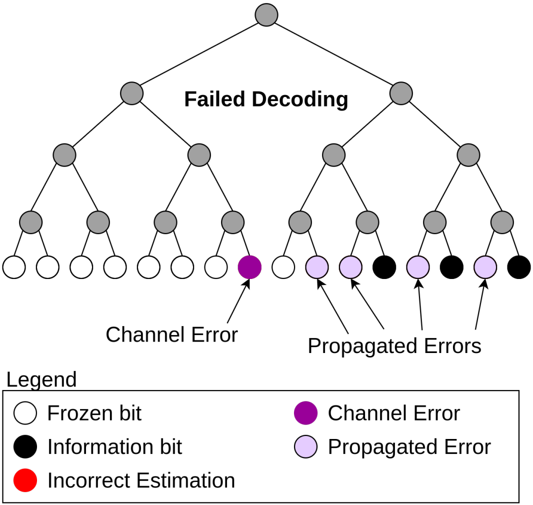
Afisiadis, et al., "A low-complexity improved successive cancellation decoder for polar codes," 48th Asilomar Conference, 2014.

SC-Flip (SCF) Decoding



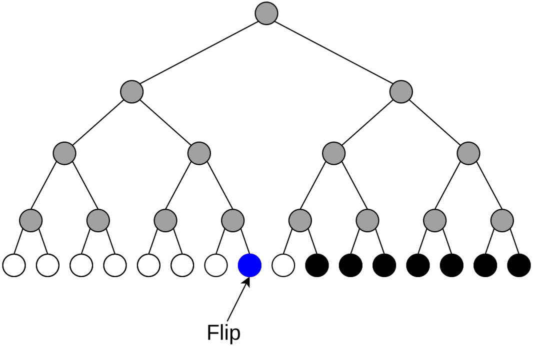
Afisiadis, et al., "A low-complexity improved successive cancellation decoder for polar codes," 48th Asilomar Conference, 2014.

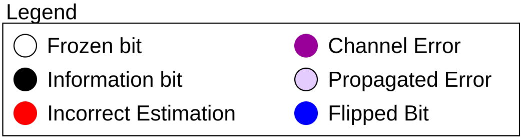
SC-Flip (SCF) Decoding



Afisiadis, et al., "A low-complexity improved successive cancellation decoder for polar codes," 48th Asilomar Conference, 2014.

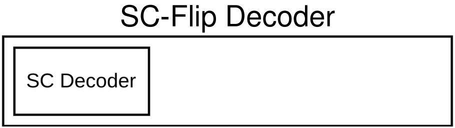
SC-Flip (SCF) Decoding





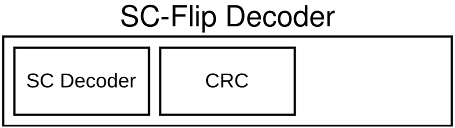
Afisiadis, et al., "A low-complexity improved successive cancellation decoder for polar codes," 48th Asilomar Conference, 2014.

Components of SC-Flip Algorithm



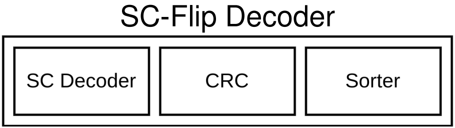
- SC decoder
- An outer Cyclic Redundancy Check (CRC) decoder
  - To tell if a decoding attempt fails
- A sorter for bit-flipping indices
  - To sort them w.r.t. their reliability metric
  - Reliability metric: Log-likelihood ratio (LLR) value at each index

Components of SC-Flip Algorithm



- SC decoder
- An outer Cyclic Redundancy Check (CRC) decoder
  - To tell if a decoding attempt fails
- A sorter for bit-flipping indices
  - To sort them w.r.t. their reliability metric
  - Reliability metric: Log-likelihood ratio (LLR) value at each index

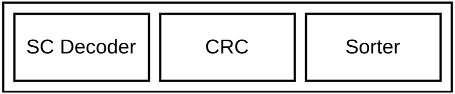
Components of SC-Flip Algorithm



- SC decoder
- An outer Cyclic Redundancy Check (CRC) decoder
  - To tell if a decoding attempt fails
- A sorter for bit-flipping indices
  - To sort them w.r.t. their reliability metric
  - Reliability metric: Log-likelihood ratio (LLR) value at each index

Previous Works on Fast SCF

SC-Flip Decoder



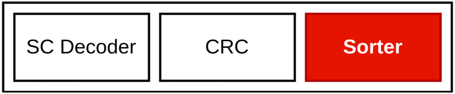
- Implementing fast nodes for SCF was proposed before [1],[2]
  - High sorting complexity
    - *i.e.* up to $n!$ bit-flipping candidates are evaluated in a node of size n
- No hardware architecture

[1] Giard et al., "Fast-SSC-flip decoding of polar codes," IEEE WCNCW 2018.

[2] Ardakani et al., "Fast Successive-Cancellation Based Decoders of Polar Codes," in IEEE TCOM 2019.

Previous Works on Fast SCF

SC-Flip Decoder

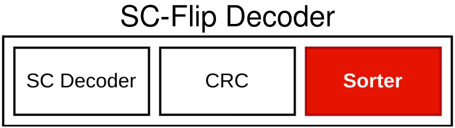


- Implementing fast nodes for SCF was proposed before [1],[2]
  - High sorting complexity
    - *i.e.* up to $n!$ bit-flipping candidates are evaluated in a node of size n
- No hardware architecture

[1] Giard et al., "Fast-SSC-flip decoding of polar codes," IEEE WCNCW 2018.

[2] Ardakani et al., "Fast Successive-Cancellation Based Decoders of Polar Codes," in IEEE TCOM 2019.

Previous Works on Fast SCF



- Implementing fast nodes for SCF was proposed before [1],[2]
    - High sorting complexity
        - *i.e.* up to $n!$ bit-flipping candidates are evaluated in a node of size n
- No hardware architecture

[1] Giard et al., "Fast-SSC-flip decoding of polar codes," IEEE WCNCW 2018.

[2] Ardakani et al., "Fast Successive-Cancellation Based Decoders of Polar Codes," in IEEE TCOM 2019.

Our Work on Fast-SCF Decoding

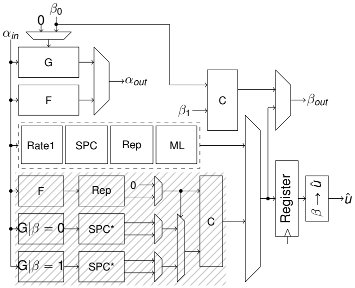
Fast-SSC	CRC	Sorter

- An energy-efficient Fast-SSC architecture, to be used by Fast-SCF
    - Introduced resource sharing for merged operations
    - Reduced the overall memory requirement for soft and hard decision memories
        - Example: LLR memory

Our Work on Fast-SCF Decoding

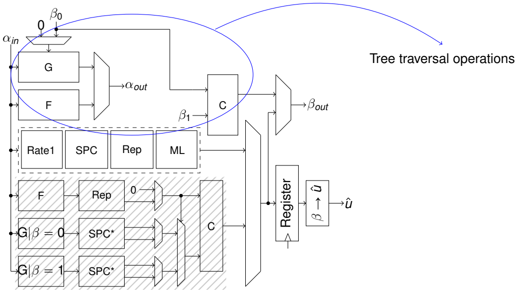
Fast-SSC	CRC	Sorter

- An energy-efficient Fast-SSC architecture, to be used by Fast-SCF
  - Introduced resource sharing for merged operations
  - Reduced the overall memory requirement for soft and hard decision memories
    - Example: LLR memory



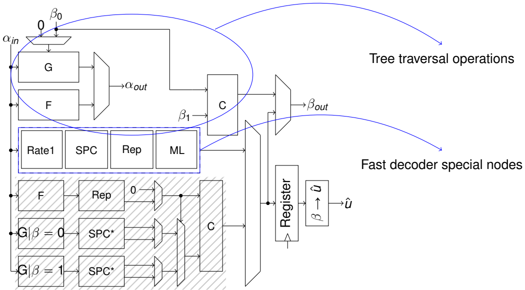
Our Work on Fast-SCF Decoding

- An energy-efficient Fast-SSC architecture, to be used by Fast-SCF
  - Introduced resource sharing for merged operations
  - Reduced the overall memory requirement for soft and hard decision memories
    - Example: LLR memory



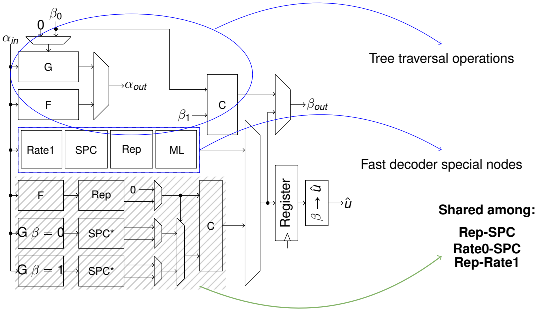
Our Work on Fast-SCF Decoding

- An energy-efficient Fast-SSC architecture, to be used by Fast-SCF
  - Introduced resource sharing for merged operations
  - Reduced the overall memory requirement for soft and hard decision memories
    - Example: LLR memory



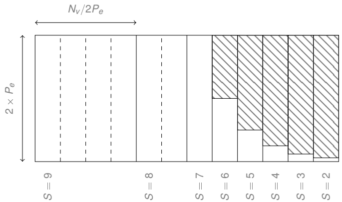
Our Work on Fast-SCF Decoding

- An energy-efficient Fast-SSC architecture, to be used by Fast-SCF
  - Introduced resource sharing for merged operations
  - Reduced the overall memory requirement for soft and hard decision memories
    - Example: LLR memory



Our Work on Fast-SCF Decoding

- An energy-efficient Fast-SSC architecture, to be used by Fast-SCF
  - Introduced resource sharing for merged operations
  - Reduced the overall memory requirement for soft and hard decision memories
    - Example: LLR memory

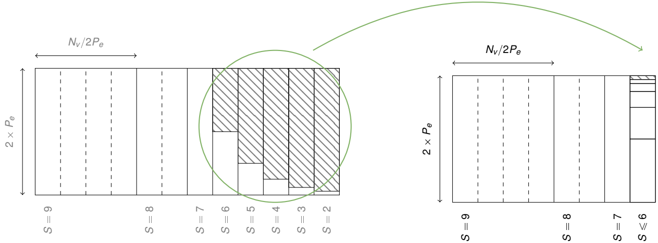


Old LLR memory architecture

Our Work on Fast-SCF Decoding

Fast-SSC | CRC | Sorter

- An energy-efficient Fast-SSC architecture, to be used by Fast-SCF
  - Introduced resource sharing for merged operations
  - Reduced the overall memory requirement for soft and hard decision memories
    - Example: LLR memory



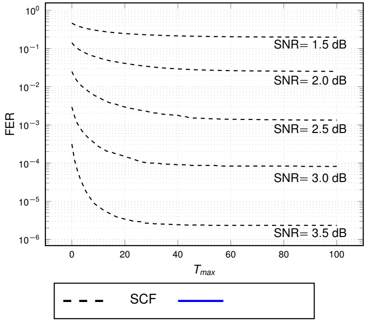
Old LLR memory architecture

New LLR memory architecture

Our Work on Fast-SCF Decoding

Fast-SSC	CRC	**Sorter**

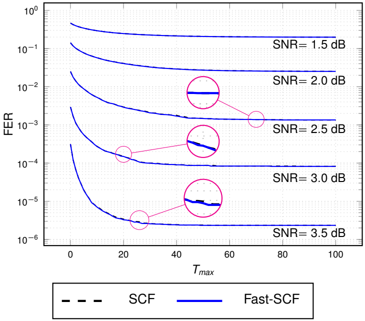
- How to select and sort bit-flipping indices for fast nodes?
  - Example: Performance v iterations, PC(1024,512)
  - Fast-SCF: Only index with minimum LLR is flipped in Rate-1 nodes



Our Work on Fast-SCF Decoding

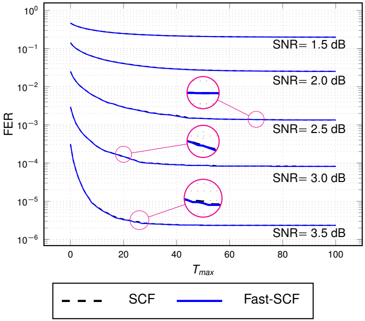
Fast-SSC	CRC	**Sorter**

- How to select and sort bit-flipping indices for fast nodes?
  - Example: Performance v iterations, PC(1024,512)
  - Fast-SCF: Only index with minimum LLR is flipped in Rate-1 nodes



Our Work on Fast-SCF Decoding

- How to select and sort bit-flipping indices for fast nodes?
  - Example: Performance v iterations, PC(1024,512)
  - Fast-SCF: Only index with minimum LLR is flipped in Rate-1 nodes

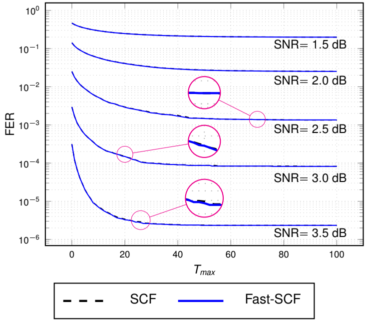


- One flipping index for Rate-1 nodes
- Up to two flipping indices for SPC nodes

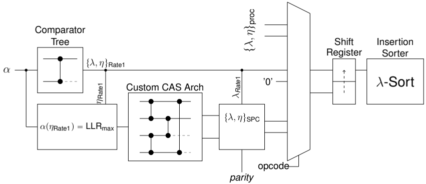
Our Work on Fast-SCF Decoding

Fast-SSC | CRC | **Sorter**

- How to select and sort bit-flipping indices for fast nodes?
  - Example: Performance v iterations, PC(1024,512)
  - Fast-SCF: Only index with minimum LLR is flipped in Rate-1 nodes



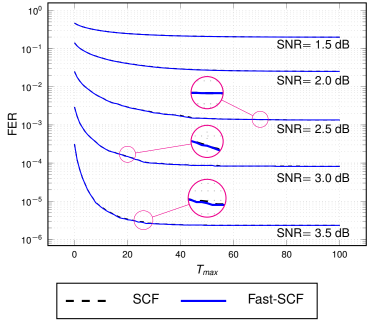
- One flipping index for Rate-1 nodes
- Up to two flipping indices for SPC nodes



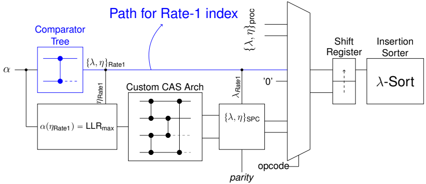
Our Work on Fast-SCF Decoding

Fast-SSC | CRC | **Sorter**

- How to select and sort bit-flipping indices for fast nodes?
  - Example: Performance v iterations, PC(1024,512)
  - Fast-SCF: Only index with minimum LLR is flipped in Rate-1 nodes

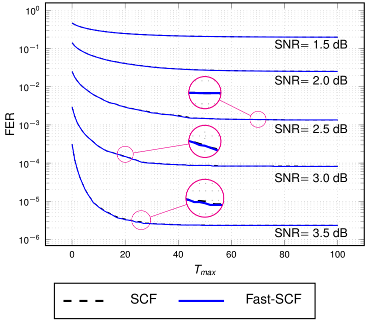


- One flipping index for Rate-1 nodes
- Up to two flipping indices for SPC nodes

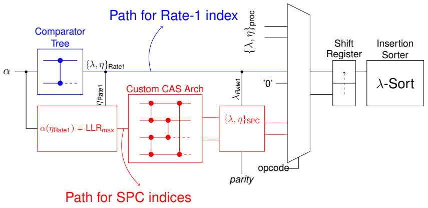


Our Work on Fast-SCF Decoding

- How to select and sort bit-flipping indices for fast nodes?
  - Example: Performance v iterations, PC(1024,512)
  - Fast-SCF: Only index with minimum LLR is flipped in Rate-1 nodes



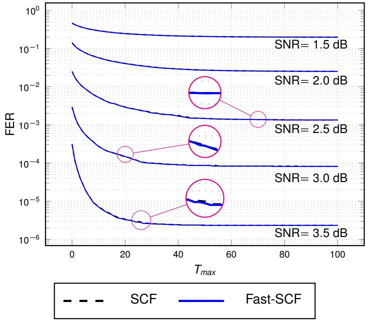
- One flipping index for Rate-1 nodes
- Up to two flipping indices for SPC nodes



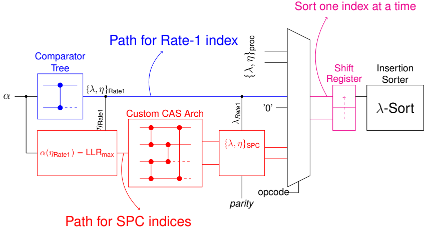
Our Work on Fast-SCF Decoding

Fast-SSC	CRC	**Sorter**

- How to select and sort bit-flipping indices for fast nodes?
  - Example: Performance v iterations, PC(1024,512)
  - Fast-SCF: Only index with minimum LLR is flipped in Rate-1 nodes



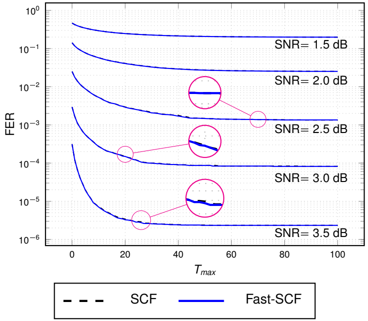
- One flipping index for Rate-1 nodes
- Up to two flipping indices for SPC nodes



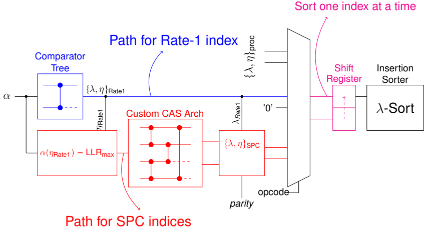
Our Work on Fast-SCF Decoding

Fast-SSC | **CRC** | Sorter

- How to select and sort bit-flipping indices for fast nodes?
  - Example: Performance v iterations, PC(1024,512)
  - Fast-SCF: Only index with minimum LLR is flipped in Rate-1 nodes



- One flipping index for Rate-1 nodes
- Up to two flipping indices for SPC nodes

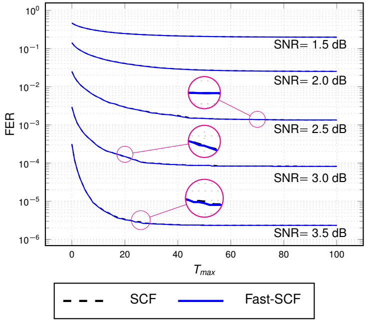


- Implemented a highly parallelized CRC datapath to support fast decoding

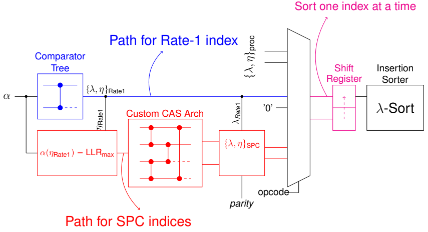
Our Work on Fast-SCF Decoding

Fast-SSC | CRC | Sorter

- How to select and sort bit-flipping indices for fast nodes?
  - Example: Performance v iterations, PC(1024,512)
  - Fast-SCF: Only index with minimum LLR is flipped in Rate-1 nodes



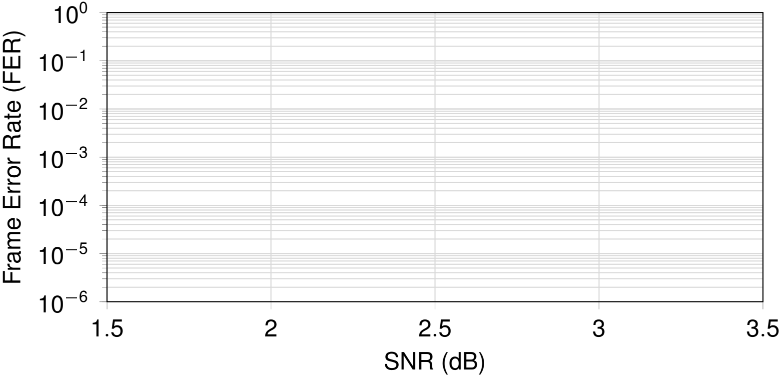
- One flipping index for Rate-1 nodes
- Up to two flipping indices for SPC nodes



- Implemented a highly parallelized CRC datapath to support fast decoding
- Putting it altogether: An energy-efficient Fast-SCF polar decoder architecture

Simulation Results: Fast-SCF Decoding

5G $PC(1024, 512)$, 11-bit 5G CRC, $T_{max} = 20$



---

[1] Afisiadis, et al., "A low-complexity improved successive cancellation decoder for polar codes," 48th Asilomar Conference, 2014.

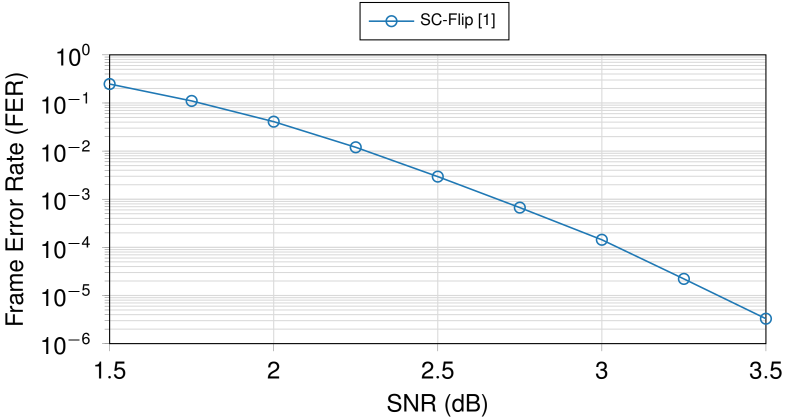
[2] Giard et al., "Fast-SSC-flip decoding of polar codes," IEEE WCNCW 2018.

[3] Ercan et al., "Energy-efficient hardware architectures for fast polar decoders," IEEE TCAS-I 2019.

[4] Hashemi et al., "A fast polar code list decoder architecture based on sphere decoding," IEEE TCAS-I 2016.

Simulation Results: Fast-SCF Decoding

5G $PC(1024, 512)$, 11-bit 5G CRC, $T_{max} = 20$



[1] Afisiadis, et al., "A low-complexity improved successive cancellation decoder for polar codes," 48th Asilomar Conference, 2014.

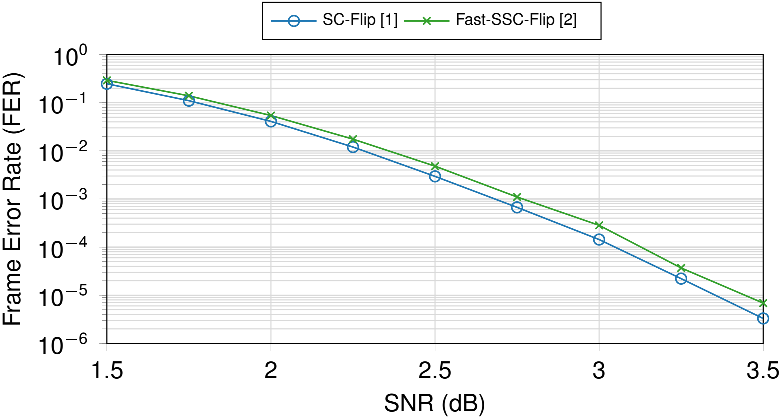
[2] Giard et al., "Fast-SSC-flip decoding of polar codes," IEEE WCNCW 2018.

[3] Ercan et al., "Energy-efficient hardware architectures for fast polar decoders," IEEE TCAS-I 2019.

[4] Hashemi et al., "A fast polar code list decoder architecture based on sphere decoding," IEEE TCAS-I 2016.

Simulation Results: Fast-SCF Decoding

5G $PC(1024, 512)$, 11-bit 5G CRC, $T_{max} = 20$



[1] Afisiadis, et al., "A low-complexity improved successive cancellation decoder for polar codes," 48th Asilomar Conference, 2014.

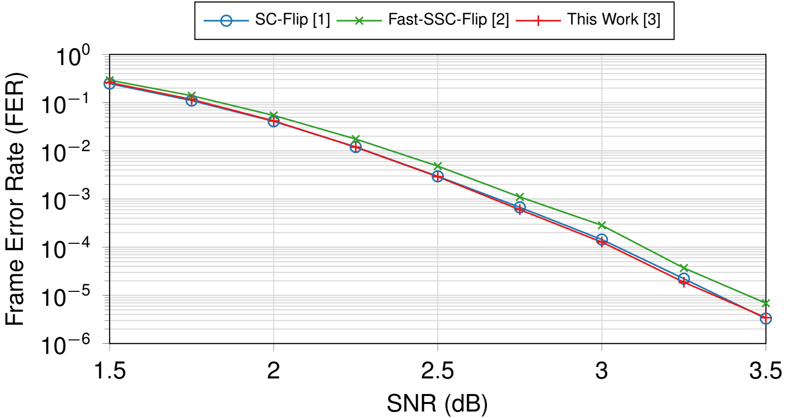
[2] Giard et al., "Fast-SSC-flip decoding of polar codes," IEEE WCNCW 2018.

[3] Ercan et al., "Energy-efficient hardware architectures for fast polar decoders," IEEE TCAS-I 2019.

[4] Hashemi et al., "A fast polar code list decoder architecture based on sphere decoding," IEEE TCAS-I 2016.

Simulation Results: Fast-SCF Decoding

5G $PC(1024, 512)$, 11-bit 5G CRC, $T_{max} = 20$



[1] Afisiadis, et al., "A low-complexity improved successive cancellation decoder for polar codes," 48th Asilomar Conference, 2014.

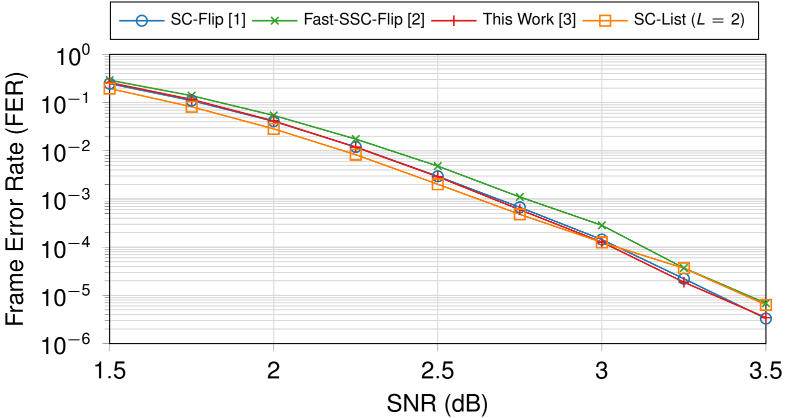
[2] Giard et al., "Fast-SSC-flip decoding of polar codes," IEEE WCNCW 2018.

[3] Ercan et al., "Energy-efficient hardware architectures for fast polar decoders," IEEE TCAS-I 2019.

[4] Hashemi et al., "A fast polar code list decoder architecture based on sphere decoding," IEEE TCAS-I 2016.

Simulation Results: Fast-SCF Decoding

5G $PC(1024, 512)$, 11-bit 5G CRC, $T_{max} = 20$



[1] Afisiadis, et al., "A low-complexity improved successive cancellation decoder for polar codes," 48th Asilomar Conference, 2014.

[2] Giard et al., "Fast-SSC-flip decoding of polar codes," IEEE WCNCW 2018.

[3] Ercan et al., "Energy-efficient hardware architectures for fast polar decoders," IEEE TCAS-I 2019.

[4] Hashemi et al., "A fast polar code list decoder architecture based on sphere decoding," IEEE TCAS-I 2016.

Implementation Results: Fast-SCF Decoder

- TSMC 65nm CMOS technology
- $PC(512, 256)$

	SCL [1]	Fast-SSCL [2]	This Work[3]
Power (mW)	75.2	119.7	**57.4**
Latency (μs)	1.71	0.43	**0.33**
Worst Case Latency (μs)	1.71	**0.43**	6.93
Throughput (Mbps)	300	1201	**1552**
Area Efficiency (Gbps/mm^2)	1.36	2.85	**4.31**
Energy (pJ/info. bit)	502	199	**74**

[1] Stimming et al., "LLR-Based Successive Cancellation List Decoding of Polar Codes," in IEEE TSP, 2015.

[2] Hashemi et al., "A fast polar code list decoder architecture based on sphere decoding," IEEE TCAS-I 2016.

[3] Ercan et al., "Energy-efficient hardware architectures for fast polar decoders," IEEE TCAS-I 2019.

Implementation Results: Fast-SCF Decoder

- TSMC 65nm CMOS technology
- $PC(512, 256)$

	SCL [1]	Fast-SSCL [2]	This Work[3]	
Power (mW)	75.2	119.7	**57.4**	$\leftarrow 2.1\times$
Latency (μs)	1.71	0.43	**0.33**	
Worst Case Latency (μs)	1.71	**0.43**	6.93	
Throughput (Mbps)	300	1201	**1552**	
Area Efficiency (Gbps/mm^2)	1.36	2.85	**4.31**	
Energy (pJ/info. bit)	502	199	**74**	

[1] Stimming et al., "LLR-Based Successive Cancellation List Decoding of Polar Codes," in IEEE TSP, 2015.

[2] Hashemi et al., "A fast polar code list decoder architecture based on sphere decoding," IEEE TCAS-I 2016.

[3] Ercan et al., "Energy-efficient hardware architectures for fast polar decoders," IEEE TCAS-I 2019.

Implementation Results: Fast-SCF Decoder

- TSMC 65nm CMOS technology
- $PC(512, 256)$

	SCL [1]	Fast-SSCL [2]	This Work[3]	
Power (mW)	75.2	119.7	**57.4**	$\leftarrow 2.1\times$
Latency (μs)	1.71	0.43	**0.33**	$\leftarrow 23\%$
Worst Case Latency (μs)	1.71	**0.43**	6.93	
Throughput (Mbps)	300	1201	**1552**	
Area Efficiency (Gbps/mm^2)	1.36	2.85	**4.31**	
Energy (pJ/info. bit)	502	199	**74**	

[1] Stimming et al., "LLR-Based Successive Cancellation List Decoding of Polar Codes," in IEEE TSP, 2015.

[2] Hashemi et al., "A fast polar code list decoder architecture based on sphere decoding," IEEE TCAS-I 2016.

[3] Ercan et al., "Energy-efficient hardware architectures for fast polar decoders," IEEE TCAS-I 2019.

Implementation Results: Fast-SCF Decoder

- TSMC 65nm CMOS technology
- $PC(512, 256)$

	SCL [1]	Fast-SSCL [2]	This Work[3]	
Power (mW)	75.2	119.7	**57.4**	$\leftarrow 2.1\times$
Latency (μs)	1.71	0.43	**0.33**	$\leftarrow 23\%$
Worst Case Latency (μs)	1.71	**0.43**	6.93	
Throughput (Mbps)	300	1201	**1552**	$\leftarrow 29\%$
Area Efficiency (Gbps/mm^2)	1.36	2.85	**4.31**	
Energy (pJ/info. bit)	502	199	**74**	

[1] Stimming et al., ”LLR-Based Successive Cancellation List Decoding of Polar Codes,” in IEEE TSP, 2015.

[2] Hashemi et al., ”A fast polar code list decoder architecture based on sphere decoding,” IEEE TCAS-I 2016.

[3] Ercan et al., ”Energy-efficient hardware architectures for fast polar decoders,” IEEE TCAS-I 2019.

Implementation Results: Fast-SCF Decoder

- TSMC 65nm CMOS technology
- $PC(512, 256)$

	SCL [1]	Fast-SSCL [2]	This Work[3]	
Power (mW)	75.2	119.7	**57.4**	$\leftarrow 2.1\times$
Latency (μs)	1.71	0.43	**0.33**	$\leftarrow 23\%$
Worst Case Latency (μs)	1.71	**0.43**	6.93	
Throughput (Mbps)	300	1201	**1552**	$\leftarrow 29\%$
Area Efficiency (Gbps/mm^2)	1.36	2.85	**4.31**	$\leftarrow 51\%$
Energy (pJ/info. bit)	502	199	**74**	

[1] Stimming et al., "LLR-Based Successive Cancellation List Decoding of Polar Codes," in IEEE TSP, 2015.

[2] Hashemi et al., "A fast polar code list decoder architecture based on sphere decoding," IEEE TCAS-I 2016.

[3] Ercan et al., "Energy-efficient hardware architectures for fast polar decoders," IEEE TCAS-I 2019.

Implementation Results: Fast-SCF Decoder

- TSMC 65nm CMOS technology
- $PC(512, 256)$

	SCL [1]	Fast-SSCL [2]	This Work[3]	
Power (mW)	75.2	119.7	**57.4**	$\leftarrow 2.1\times$
Latency (μs)	1.71	0.43	**0.33**	$\leftarrow 23\%$
Worst Case Latency (μs)	1.71	**0.43**	6.93	
Throughput (Mbps)	300	1201	**1552**	$\leftarrow 29\%$
Area Efficiency (Gbps/mm^2)	1.36	2.85	**4.31**	$\leftarrow 51\%$
Energy (pJ/info. bit)	502	199	**74**	$\leftarrow 2.7\times$

[1] Stimming et al., "LLR-Based Successive Cancellation List Decoding of Polar Codes," in IEEE TSP, 2015.
[2] Hashemi et al., "A fast polar code list decoder architecture based on sphere decoding," IEEE TCAS-I 2016.
[3] Ercan et al., "Energy-efficient hardware architectures for fast polar decoders," IEEE TCAS-I 2019.

Part III:
Making Dynamic SC-Flip Decoding Practical

Dynamic SC-Flip (DSCF) Decoding

- A better metric that can distinguish channel errors from propagated errors.
- This gives an opportunity to tackle more than one channel error.
- Error correction performance is greatly improved.
- Metric computation is impractical:
  - $M(\mathcal{E}_\omega) = \sum_{j\in\mathcal{E}_\omega} |L^0[\mathcal{E}_{\omega-1}]_j| + \sum_{\substack{j\leqslant i_\omega \\ j\in\mathcal{A}}} \frac{1}{\alpha} \log\left(1 + \exp(-\alpha|L^0[\mathcal{E}_{\omega-1}]_j|)\right)$

Dynamic SC-Flip (DSCF) Decoding

- A better metric that can distinguish channel errors from propagated errors.
- This gives an opportunity to tackle more than one channel error.
- Error correction performance is greatly improved.
- Metric computation is impractical:
  - $M(\mathcal{E}_\omega) = \sum_{j \in \mathcal{E}_\omega} |L^0[\mathcal{E}_{\omega-1}]_j| + \sum_{\substack{j \leqslant i_\omega \\ j \in \mathcal{A}}} \frac{1}{\alpha} \log\left(1 + \exp(-\alpha |L^0[\mathcal{E}_{\omega-1}]_j|)\right)$

Dynamic SC-Flip (DSCF) Decoding

- A better metric that can distinguish channel errors from propagated errors.
- This gives an opportunity to tackle more than one channel error.
- Error correction performance is greatly improved.
- Metric computation is impractical:
  - $M(\mathcal{E}_\omega) = \sum_{j \in \mathcal{E}_\omega} |L^0[\mathcal{E}_{\omega-1}]_j| + \sum_{\substack{j \leqslant i_\omega \\ j \in \mathcal{A}}} \frac{1}{\alpha} \log\left(1 + \exp(-\alpha |L^0[\mathcal{E}_{\omega-1}]_j|)\right)$

Dynamic SC-Flip (DSCF) Decoding

- A better metric that can distinguish channel errors from propagated errors.
- This gives an opportunity to tackle more than one channel error.
- Error correction performance is greatly improved.
- Metric computation is impractical:
  - $M(\mathcal{E}_\omega) = \sum_{j \in \mathcal{E}_\omega} |L^0[\mathcal{E}_{\omega-1}]_j| + \sum_{\substack{j \leqslant i_\omega \\ j \in \mathcal{A}}} \frac{1}{\alpha} \log \left(1 + \exp(-\alpha |L^0[\mathcal{E}_{\omega-1}]_j|)\right)$

Dynamic SC-Flip (DSCF) Decoding

$$M(\mathcal{E}_\omega) = \sum_{j\in\mathcal{E}_\omega} |L^0[\mathcal{E}_{\omega-1}]_j| + \sum_{\substack{j\leqslant i_\omega \\ j\in\mathcal{A}}} \frac{1}{\alpha}\log\left(1+\exp(-\alpha|L^0[\mathcal{E}_{\omega-1}]_j|)\right)$$

W. Gross and P. Gulak. "Simplified MAP algorithm suitable for implementation of turbo decoders," Electronics Letters, vol. 34, no. 16, pp. 1577–1578, Aug 1998.

Dynamic SC-Flip (DSCF) Decoding

$$M(\mathcal{E}_\omega) = \sum_{j\in\mathcal{E}_\omega} \boxed{|L^0[\mathcal{E}_{\omega-1}]_j|} + \sum_{\substack{j\leqslant i_\omega \\ j\in\mathcal{A}}} \frac{1}{\alpha} \log\left(1 + \exp(-\alpha|L^0[\mathcal{E}_{\omega-1}]_j|)\right)$$

LLRs of flipping indices
(Same as SCF)

W. Gross and P. Gulak. "Simplified MAP algorithm suitable for implementation of turbo decoders," Electronics Letters, vol. 34, no. 16, pp. 1577–1578, Aug 1998.

Dynamic SC-Flip (DSCF) Decoding

$$M(\mathcal{E}_\omega) = \sum_{j\in\mathcal{E}_\omega} \boxed{|L^0[\mathcal{E}_{\omega-1}]_j|} + \sum_{\substack{j\leqslant i_\omega \\ j\in\mathcal{A}}} \boxed{\frac{1}{\alpha}\log\left(1+\exp(-\alpha|L^0[\mathcal{E}_{\omega-1}]_j|)\right)}$$

LLRs of flipping indices
(Same as SCF)

Difficult part
$f_\alpha(|L|)$

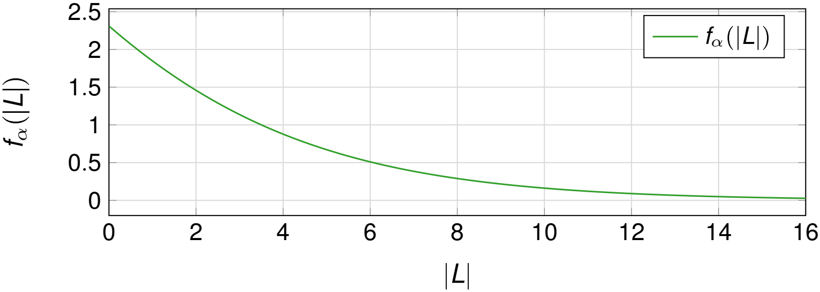
W. Gross and P. Gulak. "Simplified MAP algorithm suitable for implementation of turbo decoders," Electronics Letters, vol. 34, no. 16, pp. 1577–1578, Aug 1998.

Dynamic SC-Flip (DSCF) Decoding

$$M(\mathcal{E}_\omega) = \sum_{j\in\mathcal{E}_\omega} \boxed{|L^0[\mathcal{E}_{\omega-1}]_j|} + \sum_{\substack{j\leqslant i_\omega \\ j\in\mathcal{A}}} \boxed{\frac{1}{\alpha}\log\left(1+\exp(-\alpha|L^0[\mathcal{E}_{\omega-1}]_j|)\right)}$$

LLRs of flipping indices
(Same as SCF)

Difficult part
$f_\alpha(|L|)$



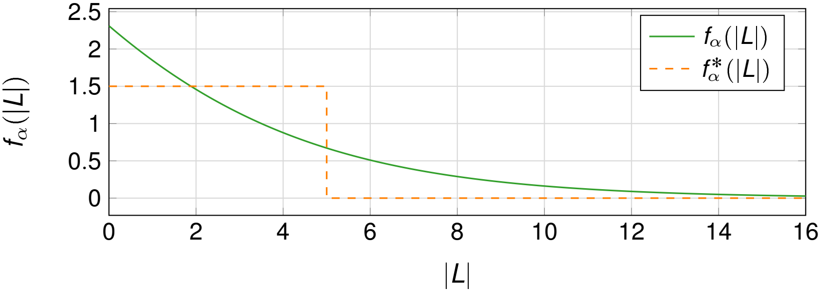
W. Gross and P. Gulak. ”Simplified MAP algorithm suitable for implementation of turbo decoders,” Electronics Letters, vol. 34, no. 16, pp. 1577–1578, Aug 1998.

Dynamic SC-Flip (DSCF) Decoding

$$M(\mathcal{E}_\omega) = \sum_{j\in\mathcal{E}_\omega} \boxed{|L^0[\mathcal{E}_{\omega-1}]_j|} + \sum_{\substack{j\leqslant i_\omega \\ j\in\mathcal{A}}} \boxed{\frac{1}{\alpha}\log\left(1+\exp(-\alpha|L^0[\mathcal{E}_{\omega-1}]_j|)\right)}$$

LLRs of flipping indices
(Same as SCF)

Difficult part
$f_\alpha(|L|)$



Replace $f_\alpha(x)$ with $f^*_\alpha(x) = \begin{cases} \frac{3}{2}, & \text{if } |x| \leqslant 5 \\ 0, & \text{otherwise.} \end{cases}$

W. Gross and P. Gulak. ”Simplified MAP algorithm suitable for implementation of turbo decoders,” Electronics Letters, vol. 34, no. 16, pp. 1577–1578, Aug 1998.

Towards Practical DSCF Decoding

Idea: Reformulate the simplified metric to be used in special nodes.
Recall the original DSCF metric:

$$M(\mathcal{E}_\omega) = \sum_{j \in \mathcal{E}_\omega} |L^0[\mathcal{E}_{\omega-1}]_j| + \sum_{\substack{j \leqslant i_\omega \\ j \in \mathcal{A}}} f_\alpha(|L^0[\mathcal{E}_{\omega-1}]_j|)$$

Towards Practical DSCF Decoding

Idea: Reformulate the simplified metric to be used in special nodes.
Recall the original DSCF metric:

$$M(\mathcal{E}_\omega) = \sum_{j \in \mathcal{E}_\omega} |L^0[\mathcal{E}_{\omega-1}]_j| + \sum_{\substack{j \leqslant i_\omega \\ j \in \mathcal{A}}} f_\alpha(|L^0[\mathcal{E}_{\omega-1}]_j|)$$

Let us split it as follows:

$$M_\alpha(\mathcal{E}_\omega) = |L^0[\mathcal{E}_{\omega-1}]_{i_\omega}| + \sum_{j \in \mathcal{E}_{\omega-1}} |L^0[\mathcal{E}_{\omega-1}]_j| + \sum_{\substack{j \leqslant i_\omega \\ j \in \mathcal{A}}} f_\alpha(|L^0[\mathcal{E}_{\omega-1}]_j|).$$

Towards Practical DSCF Decoding

Idea: Reformulate the simplified metric to be used in special nodes.
Recall the original DSCF metric:

$$M(\mathcal{E}_\omega) = \sum_{j \in \mathcal{E}_\omega} |L^0[\mathcal{E}_{\omega-1}]_j| + \sum_{\substack{j \leqslant i_\omega \\ j \in \mathcal{A}}} f_\alpha(|L^0[\mathcal{E}_{\omega-1}]_j|)$$

Let us split it as follows:

$$M_\alpha(\mathcal{E}_\omega) = \underbrace{|L^0[\mathcal{E}_{\omega-1}]_{i_\omega}|}_{M_1(L)} + \underbrace{\sum_{j \in \mathcal{E}_{\omega-1}} |L^0[\mathcal{E}_{\omega-1}]_j| + \sum_{\substack{j \leqslant i_\omega \\ j \in \mathcal{A}}} f_\alpha(|L^0[\mathcal{E}_{\omega-1}]_j|)}_{M_2(L)}.$$

- $M_1(L)$ is the *instantaneous component* obtained from the node, on the spot.
- $M_2(L)$ is the *accummulative component*, formed over the course of the decoding.

Incorporation of Special Nodes into DSCF

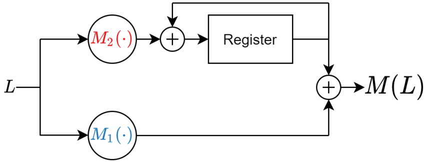
$$M(L) = M_1(L) + M_2(L).$$

- $M_1(L)$ is directly obtained from the LLR magnitude of the index.
- $M_2(L)$ is updated by the LLR magnitude of the index.
-

Incorporation of Special Nodes into DSCF

$$M(L) = M_1(L) + M_2(L).$$

- $M_1(L)$ is directly obtained from the LLR magnitude of the index.
- $M_2(L)$ is updated by the LLR magnitude of the index.

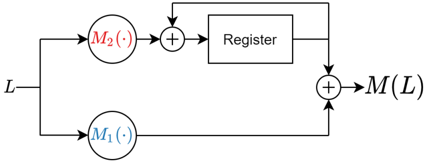


-

Incorporation of Special Nodes into DSCF

$$M(L) = M_1(L) + M_2(L).$$

- $M_1(L)$ is directly obtained from the LLR magnitude of the index.
- $M_2(L)$ is updated by the LLR magnitude of the index.



- **Main idea:** Obtain L from the special nodes.

More on Practical Fast-DSCF Decoding

- Algorithm-level improvements:
  - Performed a mathematical study to minimize the sorting complexity
- Hardware-level simplifications:
  - Normalized metric computation to avoid saturation
  - Custom sorter architecture to minimize delay
  - Reduced sorter length by 50% to reduce complexity

More on Practical Fast-DSCF Decoding

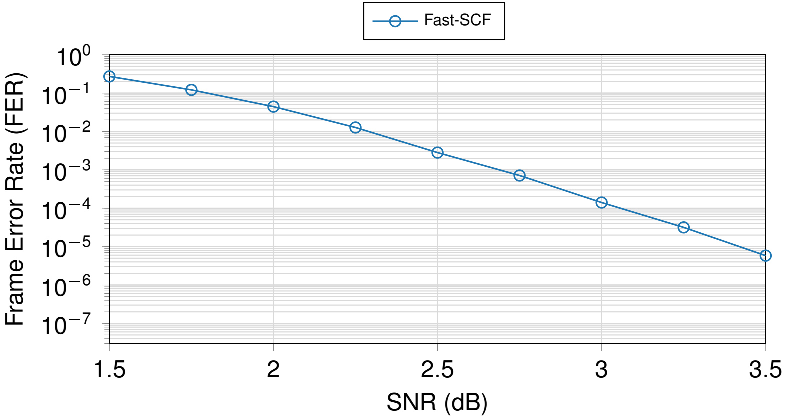
- Algorithm-level improvements:
  - Performed a mathematical study to minimize the sorting complexity
- Hardware-level simplifications:
  - Normalized metric computation to avoid saturation
  - Custom sorter architecture to minimize delay
  - Reduced sorter length by 50% to reduce complexity

More on Practical Fast-DSCF Decoding

- Algorithm-level improvements:
  - Performed a mathematical study to minimize the sorting complexity
- Hardware-level simplifications:
  - Normalized metric computation to avoid saturation
  - Custom sorter architecture to minimize delay
  - Reduced sorter length by 50% to reduce complexity

Simulation Results - Fast-DSCF Decoding

- 5G $PC(1024, 512)$, 16-bit 5G CRC, $T_{\max} = 400$.



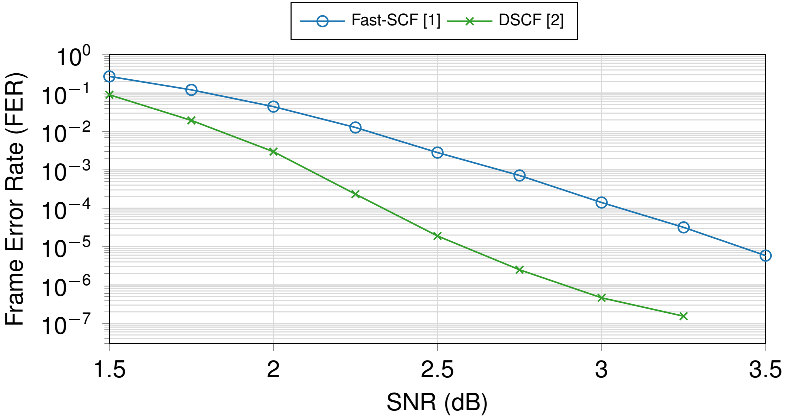
[1] Ercan et al., "Energy-efficient hardware architectures for fast polar decoders," IEEE TCAS-I 2019.

[2] Chandesris et al., "Dynamic-SCFlip decoding of polar codes," IEEE TCOM 2018.

[3] Ercan et al., "Practical dynamic SC-flip polar decoders: algorithm and implementation," IEEE TSP 2020.

Simulation Results - Fast-DSCF Decoding

- 5G $PC(1024, 512)$, 16-bit 5G CRC, $T_{\max} = 400$.



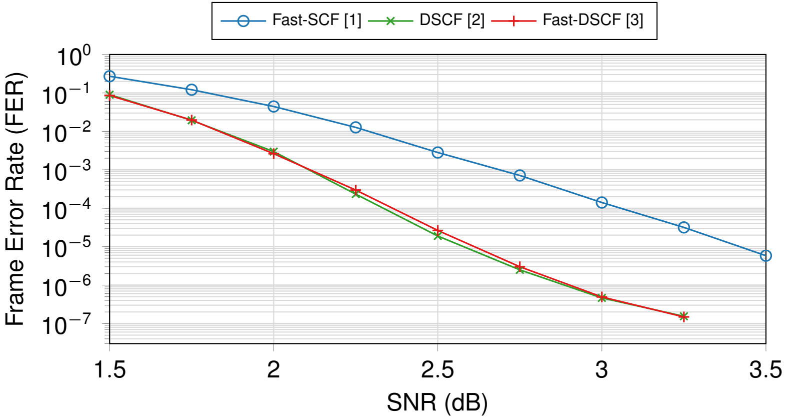
[1] Ercan et al., "Energy-efficient hardware architectures for fast polar decoders," IEEE TCAS-I 2019.

[2] Chandesris et al., "Dynamic-SCFlip decoding of polar codes," IEEE TCOM 2018.

[3] Ercan et al., "Practical dynamic SC-flip polar decoders: algorithm and implementation," IEEE TSP 2020.

Simulation Results - Fast-DSCF Decoding

- 5G $PC(1024, 512)$, 16-bit 5G CRC, $T_{\max} = 400$.



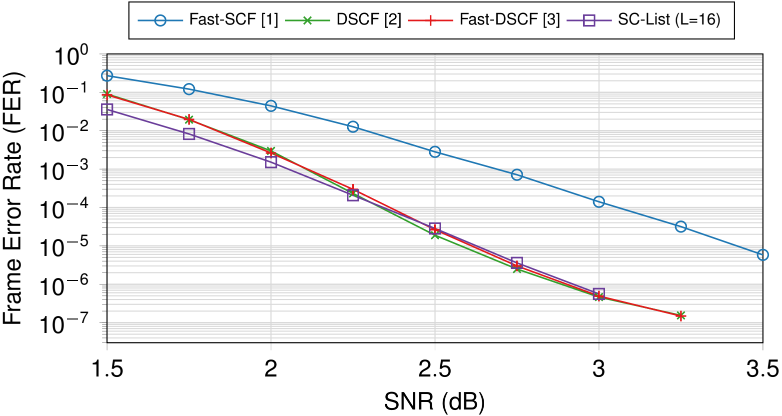
[1] Ercan et al., "Energy-efficient hardware architectures for fast polar decoders," IEEE TCAS-I 2019.

[2] Chandesris et al., "Dynamic-SCFlip decoding of polar codes," IEEE TCOM 2018.

[3] Ercan et al., "Practical dynamic SC-flip polar decoders: algorithm and implementation," IEEE TSP 2020.

Simulation Results - Fast-DSCF Decoding

- 5G $PC(1024, 512)$, 16-bit 5G CRC, $T_{\text{max}} = 400$.



[1] Ercan et al., "Energy-efficient hardware architectures for fast polar decoders," IEEE TCAS-I 2019.

[2] Chandesris et al., "Dynamic-SCFlip decoding of polar codes," IEEE TCOM 2018.

[3] Ercan et al., "Practical dynamic SC-flip polar decoders: algorithm and implementation," IEEE TSP 2020.

Implementation Results: Fast-DSCF Decoder

- TSMC 65nm CMOS technology
- $PC(1024, 512)$

	SC-List [1]	SC-List [2]	This Work[3]
Frequency (MHz)	676	911	410
Area (mm^2)	4.21	3.90	**0.55**
Latency (μs)	**0.76**	1.60	1.11
Worst Case Latency (μs)	**0.76**	1.60	447
Average Throughput (Mbps)	**1340**	637	935
Area Efficiency (Gbps/mm^2)	0.32	0.16	**0.94**
Power (mW)	–	–	201
Energy (pJ/info. bit)	–	–	439

[1] Xia et al., "A high-throughput architecture of list successive cancellation polar codes decoder with large list size," in IEEE TSP, 2018.

[2] Fan et al., "A low latency list successive-cancellation decoding implementation for polar codes," IEEE JSAC 2016.

[3] Ercan et al., "Practical dynamic SC-flip polar decoders: algorithm and implementation," IEEE TSP 2020.

Implementation Results: Fast-DSCF Decoder

- TSMC 65nm CMOS technology
- $PC(1024, 512)$

	SC-List [1]	SC-List [2]	This Work[3]	
Frequency (MHz)	676	911	410	
Area (mm^2)	4.21	3.90	**0.55**	$\leftarrow 7.1\times$
Latency (μs)	**0.76**	1.60	1.11	
Worst Case Latency (μs)	**0.76**	1.60	447	
Average Throughput (Mbps)	**1340**	637	935	
Area Efficiency (Gbps/mm^2)	0.32	0.16	**0.94**	
Power (mW)	–	–	201	
Energy (pJ/info. bit)	–	–	439	

[1] Xia et al., "A high-throughput architecture of list successive cancellation polar codes decoder with large list size," in IEEE TSP, 2018.

[2] Fan et al., "A low latency list successive-cancellation decoding implementation for polar codes," IEEE JSAC 2016.

[3] Ercan et al., "Practical dynamic SC-flip polar decoders: algorithm and implementation," IEEE TSP 2020.

Implementation Results: Fast-DSCF Decoder

- TSMC 65nm CMOS technology
- $PC(1024, 512)$

	SC-List [1]	SC-List [2]	This Work[3]	
Frequency (MHz)	676	911	410	$\leftarrow$ 50%
Area (mm^2)	4.21	3.90	**0.55**	$\leftarrow 7.1\times$
Latency (μs)	**0.76**	1.60	1.11	$\leftarrow$ 29%
Worst Case Latency (μs)	**0.76**	1.60	447	
Average Throughput (Mbps)	**1340**	637	935	$\leftarrow$ 30%
Area Efficiency (Gbps/mm^2)	0.32	0.16	**0.94**	
Power (mW)	–	–	201	
Energy (pJ/info. bit)	–	–	439	

[1] Xia et al., "A high-throughput architecture of list successive cancellation polar codes decoder with large list size," in IEEE TSP, 2018.

[2] Fan et al., "A low latency list successive-cancellation decoding implementation for polar codes," IEEE JSAC 2016.

[3] Ercan et al., "Practical dynamic SC-flip polar decoders: algorithm and implementation," IEEE TSP 2020.

Implementation Results: Fast-DSCF Decoder

- TSMC 65nm CMOS technology
- $PC(1024, 512)$

	SC-List [1]	SC-List [2]	This Work[3]	
Frequency (MHz)	676	911	410	$\leftarrow$ 50%
Area (mm^2)	4.21	3.90	**0.55**	$\leftarrow$ 7.1$\times$
Latency (μs)	**0.76**	1.60	1.11	$\leftarrow$ 29%
Worst Case Latency (μs)	**0.76**	1.60	447	
Average Throughput (Mbps)	**1340**	637	935	$\leftarrow$ 30%
Area Efficiency (Gbps/mm^2)	0.32	0.16	**0.94**	$\leftarrow$ 3$\times$
Power (mW)	–	–	201	
Energy (pJ/info. bit)	–	–	439	

[1] Xia et al., ”A high-throughput architecture of list successive cancellation polar codes decoder with large list size,” in IEEE TSP, 2018.

[2] Fan et al., ”A low latency list successive-cancellation decoding implementation for polar codes,” IEEE JSAC 2016.

[3] Ercan et al., ”Practical dynamic SC-flip polar decoders: algorithm and implementation,” IEEE TSP 2020.

Conclusions

- Algorithms and implementations for practical and energy-effcient polar decoders
- Energy-efficient Fast-SCF decoding:
  - Reduced memory, high throughput
  - Energy-efficient alternative
- Practical Fast-DSCF decoding:
  - Tackled barriers towards hardware implementation
  - Area- and energy-efficient alternative

Conclusions

- Algorithms and implementations for practical and energy-effcient polar decoders
- Energy-efficient Fast-SCF decoding:
  - Reduced memory, high throughput
  - Energy-efficient alternative
- Practical Fast-DSCF decoding:
  - Tackled barriers towards hardware implementation
  - Area- and energy-efficient alternative

Conclusions

- Algorithms and implementations for practical and energy-effcient polar decoders
- Energy-efficient Fast-SCF decoding:
  - Reduced memory, high throughput
  - Energy-efficient alternative
- Practical Fast-DSCF decoding:
  - Tackled barriers towards hardware implementation
  - Area- and energy-efficient alternative

Thank you!