Memory Reduction Techniques for Successive Cancellation Decoding of Polar Codes

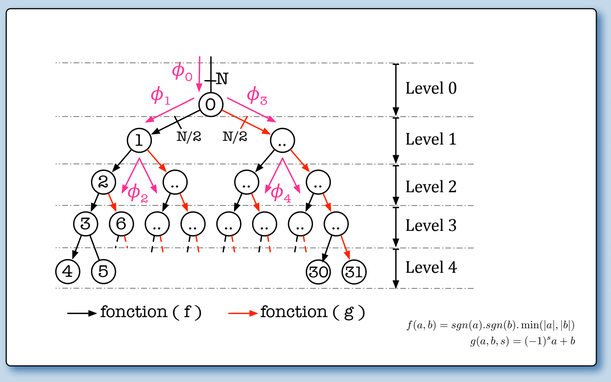
Bertrand LE GAL, Camille LEROUX and Christophe JEGO
IMS Laboratory, UMR CNRS 5218,
Bordeaux INP, University of Bordeaux
Talence, France

Abstract

Error correction codes (ECC) consist in the addition of redundancy to the binary information sequence before the transmission. They enable the FEC decoder to correct the effects of noise and interferences encountered during the transmission. Polar codes are based on a new coding scheme that asymptotically achieves the capacity of several communication channels. In terms of hardware implementation, architectural cost of SC decoders is limited by the memory complexity. In this work, two complementary methods are proposed to reduce the memory footprint of current state-of- the-art SC decoders.

SC Polar codes decoding

Literature approaches for Polar code SC decoder design



Polar codes are ECC having a decoding process described by a tree traversal.

At each graph level, M input values are processed by f (left edge) or g (right edge) functions to generate each M/2 output values.

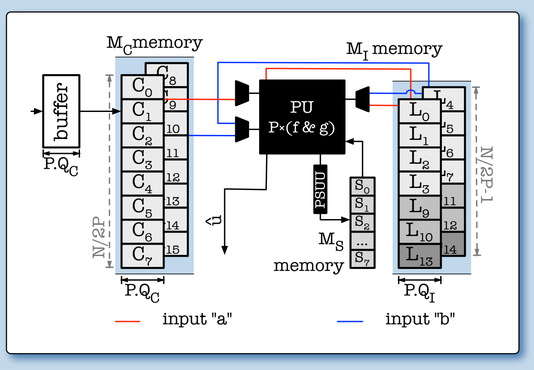
In state-of-the-art hardware architectures:

- one RAM to store N channel values,
- one RAM to store LLR values.

Channel LLRs are coded using Qc bits whereas internal LLRs are coded using Qi (Qc+Δ) bits.

Mem. cost of state-of-the-art architecture:

$$A_0 = Q_C \times N + (Q_C + \Delta) \times N + N$$



Recomputation technique (r1)

Observation on the data storage during the decoding

At the first graph level 0, N channel values are stored and N/2 values are produced first using the f function. When left sub-tree is processed N/2 values are then produced by the g function for right child node.

The N channel values are used only twice; once for f processing and once for g. They become useless when g processing is finished.

Both value sets (N channel values and N/2 LLR ones) are stored in distinct RAM memories to enable simultaneous read and write access during computations.

Data recomputing to discard some memory requirements

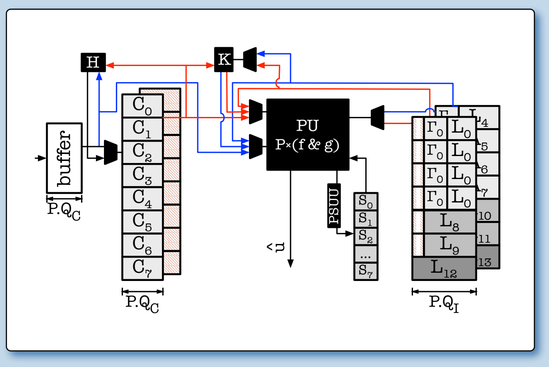
The key idea consists in removing the memory for channel values. Only (N/2) modified input values are stored. Original values can be recomputed on the fly from f results when g processing is done.

The f(a,b) function is inversible: from f result, max(a, b) value and an additional bit value, the (a, b) can be recovered to compute g(a, b). This slightly increases the hardware PU complexity.

$$\Gamma_i(a,b) = \begin{cases} 0 & \text{if } |f(a,b)| = |a| \\ 1 & \text{otherwise} \end{cases}$$

$$a = k_a(C_i, L_i) = \begin{cases} C_i & \text{if } \Gamma_i = 1 \\ sign(C_i).L_i & \text{otherwise} \end{cases}$$

$$b = k_b(C_i, L_i) = \begin{cases} C_i & \text{if } \Gamma_i = 0 \\ sign(C_i).L_i & \text{otherwise} \end{cases}$$



Memory cost is of this transformed architecture is:

$$A_1 = N/2(Q_C + 2(Q_C + \Delta) + 3)$$

Quantization reduction (r2)

The quantization format in SC hardware decoders

In first hardware SC decoders, a single quantization format was used in the overall decoder datapath. From an algorithm point of view, the f function does not increase data bit-width whereas the g function does (add 1 bit per graph level).

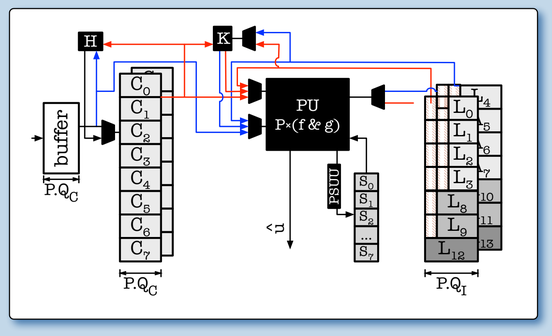
To reduce the memory cost and to improve error correction performance, channel and internal formats were dissociated. Depending on the code length, 2 to 4 bits are added to Qc to minimize the effect of saturation on the error correction performance. However, saturation mainly happens in (L > 2) graph nodes, by making some added bits useless.

The quantization format optimization

It is possible to increase progressively the bit width in the memory banks used to store internal LLRs up to Qi format.

Variable bit-width memory is easy to implement because memory banks associated to graph levels have power of 2 depths.

The memory cost of the adapted architecture is:



$$A_2 = N \times (Q_C + (Q_C + \Delta - \epsilon) + 1) \quad \text{with} \quad \epsilon = \sum_{l=1}^{\Delta-1} \frac{\Delta - l}{2^l}$$

FER impact evaluation

First quantization format do not lead to error performance impact. A higher reduction of the bit-width is also possible but leads to FER degradations.

For instance, starting with Qi=Qc for the first level provides interesting memory reduction with a slight performance degradation.

It was demonstrated by simulations over different code lengths, rates and Qc formats.

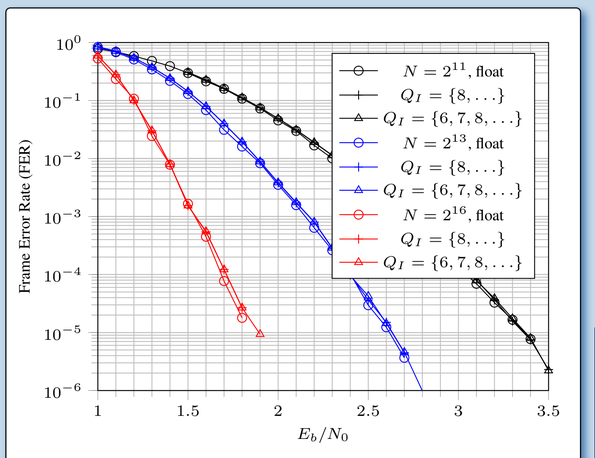


Fig. 4. FER curves for various N with $Q_C = 6$ bits.

Experimental results

The memory saving obtained for hardware SC decoders according to the proposed techniques depends on :

- the selected memory reduction technique(s);
- the Qc and Qi fixed-point formats.

An evaluation based on state of the art hardware SC decoders was performed.

Memory savings reach on total hardware SC memory cost vary from 16% to 35%.

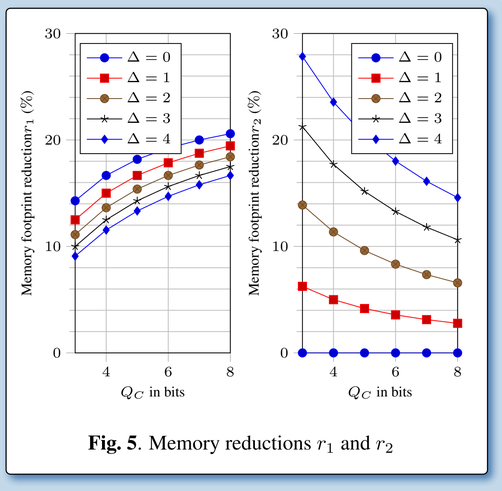


Fig. 5. Memory reductions r_1 and r_2

Parameters		A_1				A_2		$A_{1,2}$		A_2^*		$A_{1,2}^*$	
n	Q_C	Q_I	Ref	Extra LUTs	r_1'	Q_I	r_2'	$r_{1,2}'$	Q_I	Extra LUTs	$r_2^{*\prime}$	Extra LUTs	$r_{1,2}^{*\prime}$
16	{5}	{5}	[9]	+384	16%	{5, 5}	0%	0%	{5, 5}	+512	16%	+512	16%
17	{5}	{5}	[3]	+512	18%	{5, 5}	0%	0%	{5, 5}	+512	18%	+512	18%
17	{5}	{5}	[9]	+384	16%	{5, 5}	0%	0%	{5, 5}	+512	16%	+512	16%
20	{4}	{6}	[5]	+384	12%	{5, 6}	4%	10%	{4, 5, 6}	+384	16%	+384	22%
15	{4}	{7}	[5]	+384	11%	{5, 6, 7}	9%	15%	{4, 5, 6, 7}	+384	20%	+384	26%
15	{4}	{8}	[5]	+384	10%	{5, 6, 7, 8}	14%	21%	{4, 5, 6, 7, 8}	+384	25%	+384	31%
15	{4}	{9}	[5]	+384	9%	{5, 6, 7, 8, 9}	19%	26%	{4, 5, 6, 7, 8, 9}	+384	29%	+384	35%
15	{5}	{7}	[5]	+512	13%	{6, 7}	3%	9%	{5, 6, 7}	+512	17%	+512	22%
15	{5}	{8}	[9]	+512	13%	{6, 7, 8}	8%	13%	{5, 6, 7, 8}	+512	20%	+512	26%
21	{5}	{9}	[9]	+512	10%	{6, 7, 8, 9}	11%	16%	{5, 6, 7, 8, 9}	+512	21%	+512	26%

Conclusion and Perspectives

+ In terms of hardware implementation, the memory complexity is predominant in SC decoders
+ Two memory reduction methods were proposed:
  - The 1st reduces the channel memory thanks to data recomputing on the fly,
  - The 2nd reduces the LLR memory thanks to a reduction of the bit width in higher graph levels.
+ These two methods can be combined which leads to a significant memory reduction (16% to 35%).
+ These both memory reduction methods can be applied on the SCL decoding architectures.

ICASSP'16 : the 41st IEEE International Conference on Acoustics, Speech and Signal Processing, Shanghai, China, March 2016