Fountain Codes for Compression

Bertrand Ndzana Ndzana
LMA, EPFL

(0) Definitions:
The Burrows-Wheeler Transform (BWT)
[M. Burrows and D.J. Wheeler, 1994]

Operates on a sequence of size n.

Algorithm:

Do

Produces n cyclic shifts of the original sequence

Sorts these cyclic shifts lexicographically

End Do

Output: The result is the last column of the sorted table and the index position
Of the original sequence in the sorted table.

Complexity: O(n) using the prefix tree constructions

(o) Definitions: The BWT

(o) example: S = ‘aLgOLabO’, X = {a, b, g, L, O}, SBWT = ‘LOaLOabg’

	Step1
1	aLgOLabO
2	OaLgOLab
3	bOaLgOLa
4	abOaLgOL
5	LabOaLgO
6	OLabOaLg
7	gOLabOaL
8	LgOLabOa

	Step2
1	abOaLgOL
2	aLgOLabO
3	bOaLgOLa
4	gOLabOaL
5	LabOaLgO
6	LgOLabOa
7	OaLgOLab
8	OLabOaLg

	Step3
1	L
2	O
3	a
4	L
5	O
6	a
7	b
8	g

(0) Definitions: BWT

(1) example: S = 1010, X = {0, 1}

SBWT = 1001

	Step1
1	2
2	21
3	210
4	2101
5	21010

	Step2
1	21010
2	210
3	2101
4	21
5	2

	Step3
1	NULL
2	1
3	0
4	0
5	1

(0) Some properties: BWT

Definition : Given a random vector $G^n = (G_1, \ldots, G_n)$ and any integer $1 \leq D \leq n$, we call G^n D-piecewise independent and identically distributed (D-p.i.i.d.) if there exists some collection $\{q_1, \ldots, q_n\}$ of distributions on an alphabet A such that for any $x^n \in A^n$ there exists an integer transition pattern $(T_1, \ldots, T_{C+1})$, $1 = T_1 < T_2 < \ldots < T_{D+1} = n+1$, such that $\Pr(G^n = x^n) = \Pi_{j=1,\ldots,D} \Pi_{i=Tj,\ldots,Tj+1-1} p_j(x_i)$

Theorem: The output distribution of the BWT is approximately memoryless and piecewise stationary, in the sense that the normalized divergence between the ourput distribution and a memoryless and piecewise stationary distribution is small

[Karthik Visweswariah et al., 2000]

Generalization of the previous theorem: For memeory sources

[Michelle Effors et al, 2002]

(1) Definitions: The Move To Front transformation (MTF)

It operates on a sequence S of nsize n.

<u>Algorithm:</u>

<u>Do</u>

Initialize Y a list that contains all the symbol in the alphabet

<u>For</u> j = 0, ..., n-1

$S_{MTF}[j]$ = # symbols preceding symbol S[j] in Y

Move symbol symbol S[j] to the front of Y

<u>End For</u>

<u>End Do</u>

The output is a vector of n integers S_{MTF}.

<u>Complexity:</u> O(n*k), where k is the alphabet size

(1) Definitions: The MTF

(0) example: S = ‘LOaLOagb’, X ={a, b, g, L, O}

S_{MTF} = ‘3 4 2 2 2 2 4 4’

Y (List to update)	$S_{\_MTF}$ (Output)
[a b g L O]	‘3’
[L a b g O]	‘3 4’
[O L a b g]	‘3 4 2 ‘
[a O L b g]	‘3 4 2 2’
[L a O b g]	‘3 4 2 2 2 ‘
[O L a b g]	‘3 4 2 2 2 2‘
[a O L b g]	‘3 4 2 2 2 2 4’
[g a O L b]	‘3 4 2 2 2 2 4 4’

(2) Definitions: Tree sources

Consider:

- ❖ A finite ordered alphabet C of size $|C|$
- ❖ Length-N input sequences $x = x_1, \ldots, x_N$
- ❖ X^* the set of finite-length sequences over X

❑ A tree source is a finite set of sequences called states $S \subset X^*$ that is complete and proper and a set of conditional probabilities $p(\alpha/S)$ for each state $s \in S$ and each symbol $\alpha \in X$

❑ A sequence of symbols $x_{i-1}, \ldots x_{i-L}$ that uniquely determine the current state s are called the context and L is the context depth for state s.

Let $D = \max_{s \in S} |S|$, D is the maximum context depth.

(Example on the blackboard)

(0) Universal source coding
(0) Model estimation

Goal: Find a most efficient piecewise i.i.d. to attain the Merhav bound.

Idea: Model the source tree structure and estimate its state probability by processing the BWT output sequence. [Dror B., Yoram B., 2004]

The cost model: The source statistics model M is given by the number of segments S', the distinct transition points and the model segment distributions $\{Q_j(a): j = 1, \ldots, S', a \in A\}$. The cost of a model M to represent X_{BWT}:

C(XBWT, M) = S ‘ *(log k + (q-1)*b) + Σj Σa (Nj(a)log (1/Qj (a))) ,

Where $j = 1, \ldots, S'$ and $a \in A$

(0) Universal source coding
(1) Model estimation

Algorithm:

Depth-d_{max} segments are arranged as leaves of a q-ary tree where the root is the whole sequence (segment at depth 0), and for $0 < d < d_{max}$, the segment with Context $S_{dmax} = (S_d..., S_1)$ has at most q childeren segments with contexts (a, S_d)

Input: The output of the BWT transform X_{BWT}

Output. Estimation of piecewise i.i.d. (segmentation)

Do

Associate to each depth d_{max} segment its cost

Partition X_{BWT} into segments of symbols with common context for a certain maximum depth d_{max}

For $d = d_{max}-1$ to $d = 0$,

Compute the cost associate to the segment S_d by taking the minimum of the cost of representing its children segments and the cost of representing him directly

End For

End Do

(0) Linear channel codes in data compression

The Shannon-MacMilan theorem: For memoryless sources, there exist fixed length n-to-m compression codes of any rate m/n exceeding the entropy rate (H(S) +d) with vanishing block error probability as the blocklength goes to infinity

[Shannon, 1948]

Fact: Generalization of the theorem for general sources.

Problems of almost-noiseless fixed-length data compression and almost noiseless coding of an additive-noise discrete channel whose noise has the same statistics as the source are identical.

[G. Caire et al., 2004]

(0) Foutain codes: LT codes

[M. Luby, 2002]

Definition: Fountain code ensemble with parameters (l, Ω) is a map F_2 (k) →F_2 (N) represented by an ∞*l matrix where rows are chosen independently with identical distribution Ω. The symbols produced by a Fountain code are called output symbols, and the l symbols from which these output symbols are calculated are called input symbols

Encoding process: To generate an encoding symbol, randomly choose a degree v from distribution Ω. Choose uniformly at randomly v input symbols as neighbors of the encoding symbol. The value of the encoding symbol is the exclusive-or of the v neighbors.

Decoding process: Belief propagation algorithm

Extension: Raptor code [A. Shokrollahi, 2003]

(o) Belief propagation(BP) algorithm

With $m_{o,i}$ and $m_{i,o}$ messages sent from output symbols to their adjacent input symbols and the message sent from input symbols to their adjacent output symbols. At round 0 of the BP algorithm, the input nodes send to all their adjacent output nodes the value 0.

Algorithm:

Do

$T = \tanh(W/2)\prod_{i' \neq i} \tanh(m_{i'o}(r))$

$m_{o,i}(r) = \ln((1+T/(1-tT)$

$m_{i,o}(r+1) = \Sigma_{o' \neq o} m_{o',i}(r)$

Reliability of each input node: $R = \Sigma_o m_{oi}(r)$

Take a decision to stop or to continue

While (error)

Where W is the initial log-likelihood ratio at output

(0) Closed loop iterative doping (CLID) algorithm [G. Caire et al., 2004]

During the BP-algorithm, the input symbol with the smallest reliability is marke and its log-likelihood is set to +∞ or -∞ depending on whether its value is 0 or 1

<u>Do</u>

Every d iterations

Reliability sorting

Least-reliable symbol doping

<u>End of Do</u>

The BP will converge after d*n iterations !

(1) Closed loop iterative doping (CLID) algorithm

Qualities properties:

- The position of doped symbols need not be explicitly communicated to the decoder
- The algorithm never dopes twice the same symbol
- The algorithm stops in at most d*n iterations
- Good strategy to enforce the convergence of the BP

Bad encoding it self!: The longer the number of required doped bits, and the higher the resilience against channel error and /or erasures

(0) Universal source coding
(0) Coding

- The first approach: LT

Compressor: LT encoder, LT-decoder (BP)

Decompressor: LT-decoder incorporating the statistics of the source (BP)

- The second approach: LT-CLID

Compressor: LT-encoder, LT-decoder (BP and CLID algorithms).

Decompressor: LT-decoder incorporating the statistics of the source ((BP and CLID algorithms)

(1) Universal source coding: binary case
(1) Coding

- The third approach: Two-stage LT-codes

Compressor: The input here is an original k-data vector X

- Block sorting of sequence X
- Move To Front transform to X: X_{BWT}
- Modeling (X, X_{BWT}) : estimate marginal probabilities on each segment and empirical entropy H(S). Output is X_M
- An intermediate block Y of length k is calculated from XM
- A vector Z of m symbols is generated from Y through encoding with an LT code with porameters (k, W). A bipartite graph is set up between X_M, Y and Z
- The BP algorithm is applied to the graph created in the previous step
- The CLID algorithm is applied during the BP algorithm: A vector W of d symbols is generated

The output of the compressor is the sequence ZW

The choice of Ω is crucial; $m = k\,(H(S) + \Delta)$

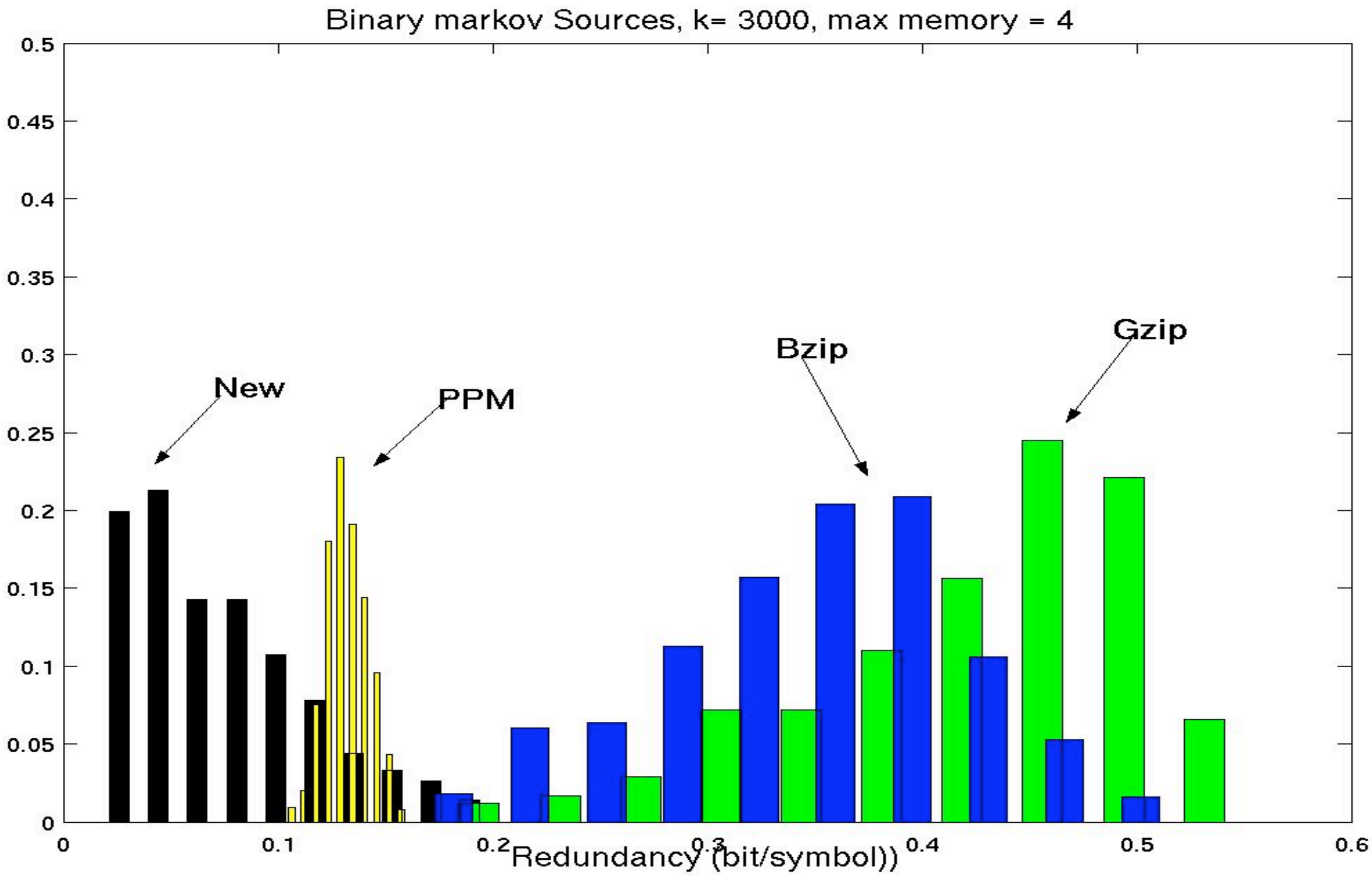
(2) Universal source coding
(2) Coding

- The third approach: Two-stage LT-codes

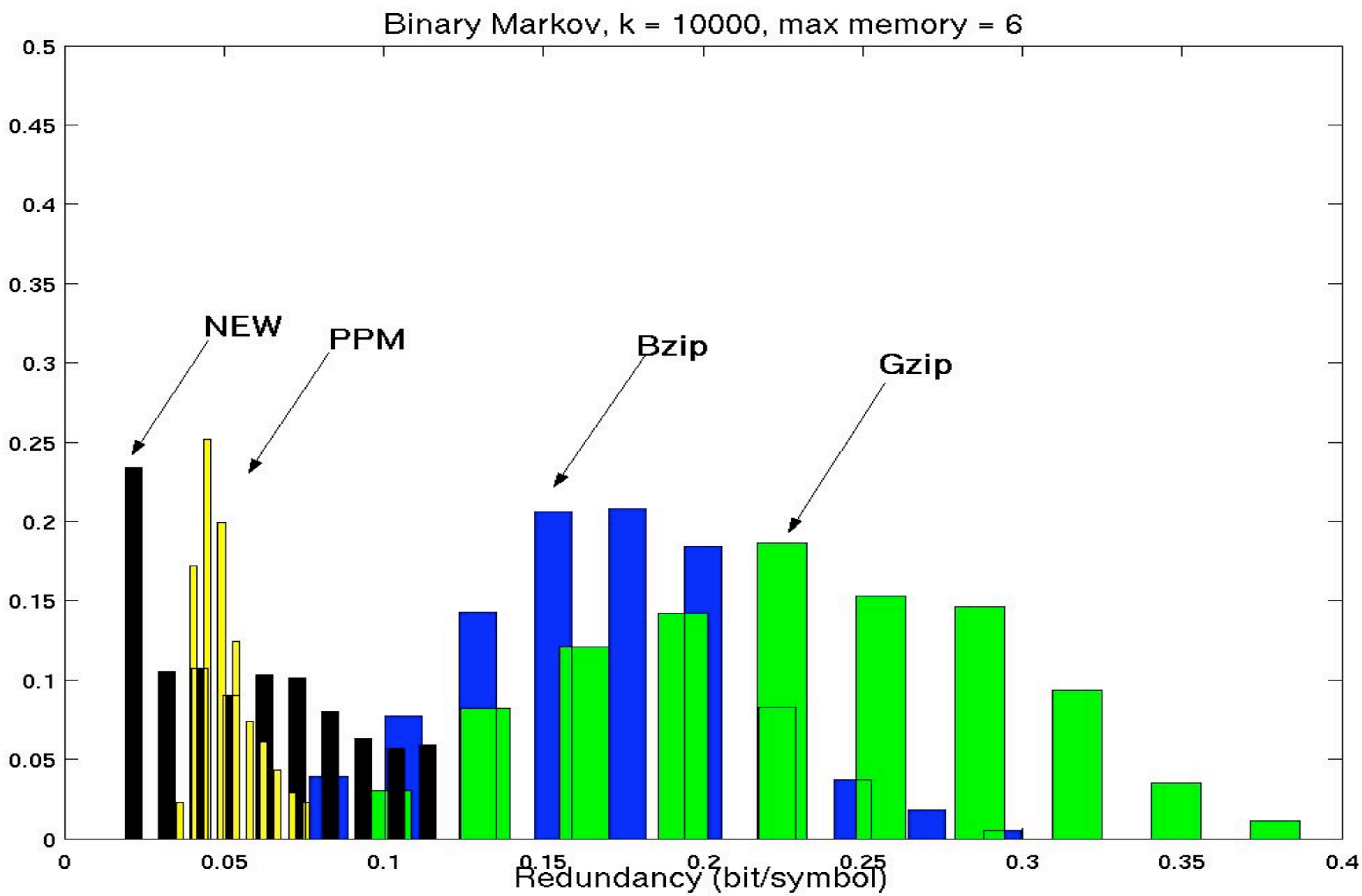
Decompressor: The input here is the compressed sequence C = ZW, the seed for generating the transformation from X_M Y, the seed for LT-encoder from Y to Z, the segmentation transitions points, the segments distribution, a flag to indicate if the MTF has been done after BWT

- From Z and W, the sequence of intermediate bits Y is reconstructed using a mirror image of the BP and CLID used at the compressor
- Applying the transformation inverse of generating intermediate symbols used at the compressor
- An inverse block sorting followed or not by the MTF transform recover the original data sequence

(3) Universal source coding: binary case
(0) Experiments



(4) Universal source coding: binary case
(1) Experiments



(5) Universal source coding
(0) Model estimation
[H. cai et al., 2004]

- Approaches: Uniform segmentation and adaptive segmentation

- Uniform segmentation: Partition the BWT output so that each segment contains an equal number of symbols w(n) from the sequence according to which we are segmenting. Taking w(n) to grow as sqrt (n) is a balanced choice.

  Theorem: For a sequence of length n generated from a stationary ergodic source, the entropy estimator using uniform segmentation with segment length $W(n) = c.\ n^x$ ($0 < x < 1$) converges to the entropy rate with probability one.

- Adaptive segmentation: uses a two-level hierarchical scheme to first obtain rough estimates for transition locations, followed by a second pass that refines the locations of the estimates

(6) Universal source coding (non Binary alphabets)

(0) Multilevel coding

[G. Caire et al. , 2004]

We suppose that an alphabet A is of cardinality 2^L and X a i.i.d. source with distribution P_X

Let f: $A \rightarrow GF(2)^L$ such that $f(x) = (b_1,\ldots,b_L)$ is the binary label corresponding to x.

The mapping f and the source probability P_X induce a probability assignment

$P_{B1,\ldots, BL}(f(x)) = P_X(x)$, where without loss of generalty, $B_1 ,\ldots, B_l$ are random variables in the natural order $l = 1,\ldots, L$

The <u>conditional probability</u> of $B_l = 1$ given $(B1, \ldots, B_L) = (b1, \ldots, b_L)$ is given by

$$P_l (b_1, \ldots, b_{l-1}) = P(b_l(X) = 1/b_1(X) = b_1, \ldots, b_{l-1} (X) = b_{l-1})$$

$$= (\Sigma_{x \in R:} P_X (x))/ (\Sigma_{x \in S:} P_X (x))$$

with $R = \{x \in A : b_1(x) = b_1, \ldots, b_{l-1} (x) = b_{l-1} , b_l(x) = 1 \}$, and

$S = \{x \in A : b_1(x) = b_1, \ldots, b_{l-1} (x) = b_{l-1}\}$,

The <u>entropy</u> is $H(X) = \Sigma_{l= 1, \ldots, l= L} H(b_l(X) /b_1(X) , \ldots, b_{l-1} (X))$

$$= \Sigma_{l= 1, \ldots, l= L} \Sigma_{x \in S:} P_X (x) h(P_l (b_1, \ldots, b_{l-})$$

With $h(p) = -p \log (p) - (1 - p) \log (1 - p)$

(7) Universal source coding (non Binary alphabets)
(1) Multilevel coding

Examples:

- $L = 1$, $A = \{0, 1\}$, X = "0 1 1 0 1 0 1 1 0 1 1", $P_X(0) = 4/11$, $P_X(1) = 7/11$

  $P_l(b_0) = P(b_1(X) = 1) = 7/11$

  $H(X) = P_X(0)\, h(P_l(b_0)) + P_X(1)\, h(P_l(b_0)) = h(P_l(b_0)) = h(4/11)$

- $L = 2$, $A = \{0, 1, 2, 3\}$, X = "3 1 0 2 2 3 0 1 2 1", f (X): b_0 = 1 1 0 0 0 1 0 1 0 0

  b_1 = 1 0 0 1 1 1 0 0 1 1

$P_X(0) = 0.2$, $P_X(1) = 0.3$, $P_X(2) = 0.3$, $P_X(3) = 0.2$

$P_l(b_0) = P(b_1(X) = 1) = 0.4$,

$P(b_2(X) = 1/ b_1(X) = 0) = 4/6$, $P(b_2(X) = 1/ b_1(X) = 1) = 2/4$

$H(X) = H(b_0(x)) + H(b_1(x)/ b_0(x))$

$= h(P_l(b_0)) + P_X(0)\, h(P(b_2(X) = 1/ b_1(X) = 0)) + P_X(1)\, h(P(b_2(X) = 1/ b_1(X) = 1))$

$+ \quad P_X(2)\, h(P(b_2(X) = 1/ b_1(X) = 0)) + P_X(3)\, h(P(b_2(X) = 1/ b_1(X) = 1))$

$= h(0.4) + 0.5\, h(4/6) + 0.5\, h(2/4)$

Further work

- Optimization of the decoding part of fountain codes to improve the compression and decompression time
- Slepian-Wolf problem with Fountain codes
- Fountain codes for lossy compression

(0) References

- G. caire, S. shamai, A. shokrollahi, S. verdu,
  Universal variable-length data compression of binary sources using fountain codes, IEEE, 2004
- M. Burrows, D.J.Wheeler, A block-sorting lossless data compression algorithm, SRC, 1994
- Karthik Visweswariah, Sanjeev Kulkarni, sergio Verdu, Output distribution of the Burrows Wheeler Transfrom", ISIT 2000
- Michelle Effors, Karthik Visweswariah, Sanjeev R. Kulkarni, Sergio verdu, Universal lossless Source Coding with the BWT transform, IEEE, 2002
- Dror Baron, Yoram Bresler, An O(N) semipredictive universal Encoder via the BWT, IEEE, 2004
- Giuseppe Caire, Shlomo Shamai, Sergio Verdu, Noiseless data compression with low-Density Parity-Check Codes, 2004

(1) References

- Amin Shokrollahi, Raptor Codes, Digital Foutain, 2004
- Michael Luby, LT Codes, Digital Foutain, 2002
- Haixiao Cai, Sanjeev R. Kulkarni, Sergio verdu, Universal entropy via block sorting, IEEE, 2004