Towards Better Compressed Representations

Michał Gańczorz

University of Wrocław, Poland

Uniwersytet Wrocławski

Motivation

Structures for the static random access problem:
Given a text $S \in \Sigma^*$, $\sigma = |\Sigma|$ it builds a compressed representation supporting only access(i) — returns $S[i]$.
Application: compressed RAM.

Previous results:

	Space used	Query time
	$\|S\|H_k(S) + \mathcal{O}\left(\|S\|\frac{k\log\sigma+\log\log\|S\|}{\log_\sigma\|S\|}\right)$	$\mathcal{O}(1)$
	$\|S\|H_k(S) + \mathcal{O}\left(\|S\|\frac{\log\log\|S\|}{\log_\sigma\|S\|} + \sigma^k\cdot\log\|S\|\right)$	$\mathcal{O}\left(\frac{\log\log\|S\|}{\log_\sigma\|S\|}\right)$

All such structures build a parsing:

$$Y_S = y_1 y_2 \dots y_{|Y_S|} \quad , y_i \in \Sigma^+$$

and show that:

$$|Y_S|H_0(Y_S) \le |S|H_k(S) + \mathcal{O}\left(|S|\frac{k\log\sigma+\log\log|S|}{\log_\sigma|S|}\right) \text{ or}$$

$$|Y_S|H_1(Y_S) \le |S|H_k(S) + \mathcal{O}\left(|S|\frac{\log\log|S|}{\log_\sigma|S|} + \sigma^k\cdot\log|S|\right) ,$$

i.e. output of $0/1$-order entropy coder achieves the bounds.

For most data structures: $|y_1| = |y_2| = \ldots = |y|_{|Y_S|} = \Theta(\log_\sigma|S|)$;
i.e. they do naive parsing $\implies$ small dictionary.
Interestingly, similar bounds hold for most parsings, [Gańczorz 2018].

Example (similar to [González, Navarro 2006])
Structure consists of: concatenation of prefix codes, structure for code borders, and a dictionary. access(i): read $\lceil i/3\rceil$-th codeword, decode the factor f, return $(i \bmod 3)$-th letter from f.

Y_S	acb	abc	abc	aaa	aaa	abb	aaa	aab	aaa	bac	aab	aab
Huffman	1100	10	10	00	00	1101	10	01	10	00	111	01
Rank/Select	0001	01	01	01	01	0001	01	01	01	01	001	01

Codeword	00	01	10	1100	1101	111
Factor	aaa	aab	abc	acb	abb	bac

Can we improve if we can choose the parsing? Yes!:

Theorem 2, [Gańczorz 2018] *Let S — string, then for any integer l we can construct a parsing Y_S of size $|Y_S| \le \lceil\frac{|S|}{l}\rceil + 1$ satisfying:*

$$|Y_S|H_0(Y_S) \le \frac{|S|}{l}\sum_{i=0}^{l-1} H_i(S) + \mathcal{O}(\log|S|).$$

All phrases have length l, except first and last, which can be shorter.

Theorem 4. *Let S — string, then for any integer l we can construct a parsing Y_S of size $|Y_S| \le \lceil\frac{|S|}{l}\rceil + 1$ satisfying:*

$$|Y_S|H_1(Y_S) \le \frac{|S|}{l}\sum_{i=l}^{2l-1} H_i(S) + \mathcal{O}(\log|S|).$$

All phrases have length l, except first and last, which can be shorter.

Observation: existing structures can be generalized to the case when factors have different but small lengths, i.e. $|y_i| \le m = \Theta(\log_\sigma|S|)$.

This motivates the problems:

Minimum Entropy Bounded-Factor Parsing

Given an integer m and a string S compute its parsing Y_S into phrases of length of at most m minimizing $|Y_S|H_0(Y_S)$ over all such parsings.

Minimum First-Order Entropy Bounded-Factor Parsing

Given an integer m and a string S compute its parsing Y_S into phrases of length of at most m minimizing $|Y_S|H_1(Y_S)$ over all such parsings.

Solution (Heuristic)

Idea for heuristics inspired by proof of Theorems 2, 4, which use:

Lemma 1, [Aczél 1973] *Let w be a string over alphabet Γ and $p : \Gamma \to \mathbb{R}^+$ be a function such that $\sum_{s\in\Gamma} p(s) \le 1$. Then: $|w|H_0(w) \le -\sum_{s\in\Gamma}|w|_s\log p(s)$, where $|w|_s$ is the number of occurrences of s in w.*

Intuition: each phrase among all different phrases is given a "probability" $p(y)$, this corresponds to "code" of length $-\log p(y)$, entropy is bounded by sum of "codes lengths" over all phrases of the parsing.

For Theorem 2 $p(y)$s are chosen in a way that: $\sum_{i=0}^{l-1}\sum_{y_j\in Y_S^i} p(y_j) = |S|\sum_{i=0}^{l-1} H_i(S)$, where $Y_S^0, Y_S^1, \ldots, Y_S^{l-1}$ are naive parsings, i.e. each phrase is equal to l, except first and last, which can be shorter. Thus one of the naive parsings can be encoded with $\frac{|S|}{l}\sum_{i=0}^{l-1} H_i(S)$ bits.

Theorem 4: similar argument, apply Lemma 1 separately to each string w_a made by concatenating letters of w occurring in one-letter context a.

Heuristic: use dynamic programming to calculate optimal parsing with respect to similar p — upper bound on either $|S|H_0(S)$, $|S|H_1(S)$.

Definition 1 *Let S — string, m-max. substring length, define p_{H_0} for a substring $y = a_1a_2\ldots a_j$ of S as*

$$p_{H_0}(y) = \frac{1}{m}\cdot p_1(a_1)\cdot p_2(a_2)\cdot\ldots\cdot p_j(a_j), \text{ where } p_i(a_i) = \mathbb{P}(a_i|a_1a_2\ldots a_{i-1}),$$

and p_{H_1} for a substring $y'y$ of S, $y = a_1a_2\ldots a_j$, as:

$$p_{H_1}(y, y') = \frac{1}{m}\cdot p_1(a_1)\cdot p_2(a_2)\cdot\ldots\cdot p_j(a_j), \text{ where } p_i(a_i) = \mathbb{P}(a_i|y'a_1a_2\ldots a_{i-1})$$

and phrases cost: $-\log p_{H_0}(y)$ and $-\log p_{H_1}(y'y)$. $\mathbb{P}(a|u) \approxeq \frac{|S|_{ua}}{|S|_u}$.

Intuition: p_{H_0} for i-th letter uses roughly same number of bits as i-th order entropy coder and p_{H_1} for i-th letter uses roughly same number of bits as $(i+|y'|)$-th order entropy coder.

Variant	Bound	Construction time
$\|Y_S\|H_0(Y_S) \le$	$\frac{\|S\|}{m}\sum_{0\le i\le m-1} H_i(S) + \|Y_S\|\log m$ $\|S\|H_k(S) + \|Y_S\|\log m + \|Y_S\|k\log\sigma$	$\mathcal{O}(m\cdot\|S\|)$
$\|Y_S\|H_1(Y_S) \le$	$\frac{\|S\|}{m}\sum_{m\le i\le 2m-1} H_i(S)$ $\|S\|H_m(S) + \|Y_S\|\log m$	$\mathcal{O}(m^2\cdot\|S\|)$

Experimental Results

File	m	$\frac{1}{m}\sum_{i<m} H_i(S)$	$\frac{\|N\|}{\|S\|}H_0(N)$	$\frac{\|S\|}{\|N\|}$	$\|\Sigma_N\|$	$\frac{\|A\|}{\|S\|}H_0(A)$	$\frac{\|S\|}{\|A\|}$	$\|\Sigma_A\|$	**Entropy Gain**
english	2	4.0677	4.0676	2.	$5.3\cdot10^3$	4.0610	1.831	$3.1\cdot10^3$	**0.16%**
	4	3.3607	3.3570	4.	$2.0\cdot10^5$	3.1928	3.629	$.93\cdot10^5$	**4.89%**
	6	2.8698	2.8431	6.	$1.1\cdot10^6$	2.6457	5.447	$.44\cdot10^6$	**6.94%**
	8	2.5166	2.4383	8.	$2.3\cdot10^6$	2.2773	7.302	$.97\cdot10^6$	**6.60%**
dblp.xml	2	4.2616	4.2615	2.	$5.7\cdot10^3$	4.1718	1.849	$2.5\cdot10^3$	**2.10%**
	4	2.9622	2.9569	4.	$2.9\cdot10^5$	2.8095	3.759	$1.4\cdot10^5$	**4.98%**
	6	2.2562	2.2328	6.	$8.3\cdot10^5$	2.1294	5.775	$4.9\cdot10^5$	**4.63%**
	8	1.8431	1.8025	8.	$1.1\cdot10^6$	1.6911	7.735	$.73\cdot10^6$	**6.18%**
sources	6	2.9813	2.9341	6.	$1.7\cdot10^6$	2.7636	5.504	$.81\cdot10^6$	**5.81%**
sources	8	2.4884	2.3994	8.	$2.5\cdot10^6$	2.2843	7.450	$1.4\cdot10^6$	**4.80%**
dna	2	1.9584	1.9584	2.	$1.0\cdot10^2$	2.0317	1.941	$.87\cdot10^2$	**-3.74%**
	8	1.9158	1.9149	8.	$6.6\cdot10^4$	1.9899	7.501	$7.1\cdot10^4$	**-3.92%**

Table 1. Entropy comparison for H_0 variant, A — parsing generated by heuristic, N — naive parsing (from Theorem 2).

File	m	$\sum_{m\le i<2m}\frac{H_i(S)}{m}$	$\frac{\|N\|}{\|S\|}H_1(N)$	$\frac{\|S\|}{\|N\|}$	$\|\text{pairs}(N)\|$	$\frac{\|A\|}{\|S\|}H_1(A)$	$\frac{\|S\|}{\|A\|}$	$\|\text{pairs}(A)\|$	**Entropy Gain**
english	2	2.6537	2.6510	2.	$2.54\cdot10^5$	2.6286	1.89	$2.03\cdot10^5$	**0.84%**
	3	2.0540	2.0285	3.	$1.53\cdot10^6$	1.9819	2.85	$1.29\cdot10^6$	**2.30%**
	4	1.6726	1.5882	4.	$3.64\cdot10^6$	1.5161	3.83	$3.23\cdot10^6$	**4.54%**
dblp.xml	2	1.6628	1.6589	2.	$3.72\cdot10^5$	1.5205	1.92	$3.05\cdot10^5$	**8.34%**
	3	1.0030	0.9802	3.	$1.26\cdot10^6$	0.8509	2.90	$1.07\cdot10^6$	**13.19%**
	4	0.7240	0.6818	4.	$1.82\cdot10^6$	0.6102	3.89	$1.65\cdot10^6$	**10.50%**
sources	4	1.2764	1.1848	4.	$4.03\cdot10^6$	1.0658	3.85	$3.86\cdot10^6$	**10.04%**
dna	2	1.9220	1.9220	2.	$1.02\cdot10^3$	1.9795	1.97	$1.09\cdot10^3$	**-2.99%**
	4	1.8914	1.8906	4.	$.68\cdot10^5$	1.9633	3.86	$1.14\cdot10^5$	**-3.85%**

Table 2. Entropy comparison for H_1 variant, A — parsing generated by the heuristic, N — naive parsing (from Theorem 4), pairs$(\cdot)\approx$ #of entries in the dictionary.

File	gzip	bzip	ppmdi	H0						H1					
					naive			algorithm			naive			algorithm	
				m	total	string	dict	total	string	dict	m	total	string	dict	total	string	dict
english	3.002	2.272	1.948	7	3.360	2.64	0.72	2.817	2.46	0.36	3	2.523	2.04	0.48	2.449	2.00	0.45
dblp.xml	1.379	0.898	0.737	7	2.480	2.00	0.48	2.182	1.86	0.32	3	1.479	1.05	0.42	1.326	0.94	0.38
sources	1.863	1.583	1.337	7	3.751	2.65	1.10	3.172	2.51	0.67	3	2.632	1.74	0.89	2.523	1.66	0.86
dna	2.164	2.078	1.945	7	1.935	1.93	0.01	2.013	2.00	0.01	3	1.924	1.92	0.00	1.989	1.99	0.00

Table 3. Compression comparison, values in bps, *algorithm* — parsing generated by heuristics (H_0/H_1), *naive* — parsing from Theorem 2/Theorem 4.

file	m	uncompressed			naive				algorithm			
		bps	T_r	T_b	bps	δ_s	T_r	T_b	bps	δ_s	T_r	T_b
english	7	8	0.008	0.003	**4.205**	0.845	3.14	16.06	**3.911**	1.094	14.38	18.88
	8				**4.332**	0.775	3.16	14.36	**3.870**	1.013	14.61	16.82
dblp	7	8	0.008	0.003	**3.215**	0.735	3.20	16.49	**3.174**	0.992	12.76	16.31
	8				**3.101**	0.668	3.20	14.55	**3.006**	0.872	13.37	15.12
sources	7	8	0.008	0.003	**4.692**	0.940	3.18	15.94	**4.308**	1.136	14.26	18.11
	8				**4.743**	0.845	3.24	14.70	**4.293**	1.036	16.63	16.47
dna	7	8	0.009	0.003	**2.573**	0.637	2.90	14.92	**2.926**	0.913	12.36	15.91
	8				**2.582**	0.637	3.02	12.28	**2.874**	0.851	12.22	13.84

Table 4. Structure for H_0, comparison of bps/time[sec] for operations, δ_s — difference between compression and structure [bps], T_r — read time for a random list of 10^6 letters, T_b — read time for 10^3 blocks of 50KB.

mga@cs.uni.wroc.pl