TSXor

A Simple Time Series Compression Algorithm

Andrea Bruno

Franco M. Nardini

Giulio E. Pibiri

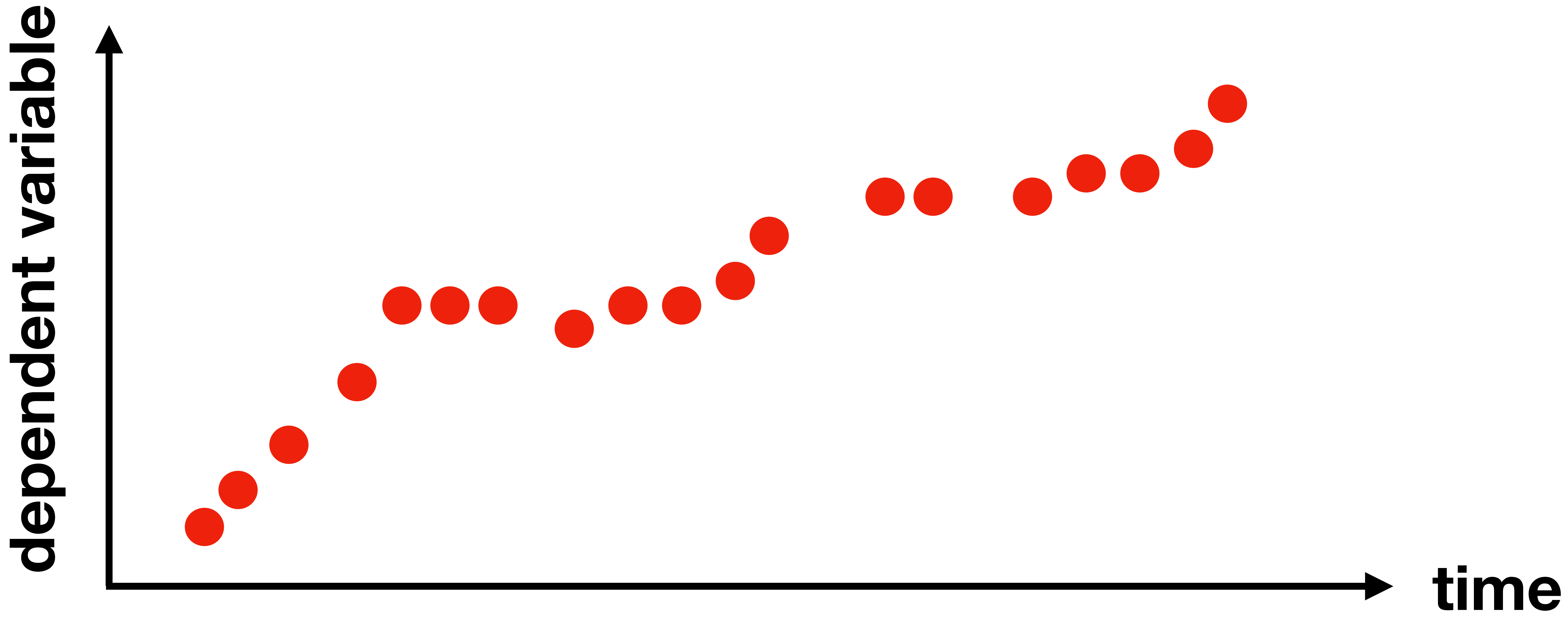
Roberto Trani

Rossano Venturini

University of Pisa and ISTI-CNR, Italy

28th International Symposium on String Processing and Information Retrieval (SPIRE), October 2021

Time Series



A sequence of key-value pairs (t_n, v_n).

The “de-facto” data format for the Internet of Things.
Heavily used in Machine Learning analytics.

TSXor — Overview

A lossless time series compression algorithm, achieving good compression ratios and fast **sequential** decoding speed.

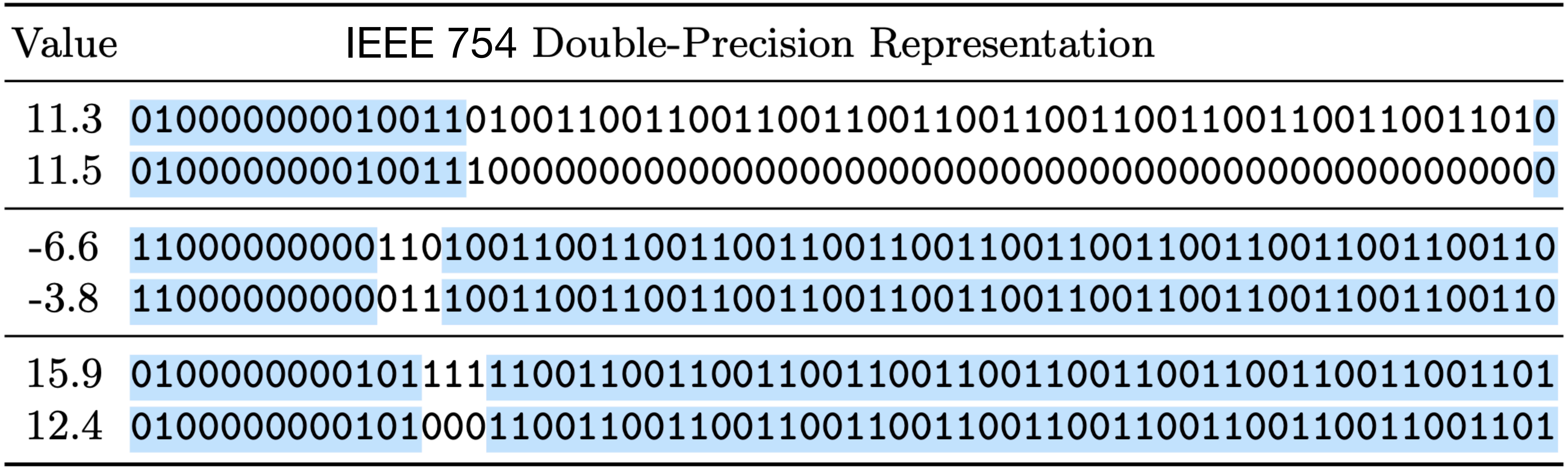
TSXor — Overview

A lossless time series compression algorithm, achieving good compression ratios and fast **sequential** decoding speed.

Empirical property of time series —
close-in-time values are very similar if not exactly the same.

Overall idea —
compress v_n by reference/similarity to the last W compressed values, for a suitable $W > 0$.

TSXor — Core



Value	IEEE 754 Double-Precision Representation
11.3	0100000000100110100110011001100110011001100110011001100110011010
11.5	0100000000100111000000000000000000000000000000000000000000000000
-6.6	1100000000011010011001100110011001100110011001100110011001100110
-3.8	1100000000001110011001100110011001100110011001100110011001100110
15.9	0100000000101111110011001100110011001100110011001100110011001101
12.4	0100000000101000110011001100110011001100110011001100110011001101

It is **not always effective** to compress v_n relative to v_{n-1}.

Compare v_n to its **preceding** $W \leq 127$ values, in the time range $[t_{n-W}, t_{n-1}]$.

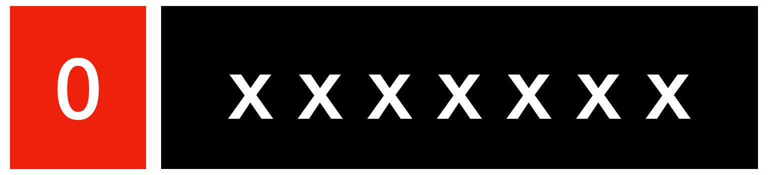
TSXor — 3 Cases

Compare v_n to its preceding $W \leq 127$ values, in the time range $[t_{n-W}, t_{n-1}]$:

TSXor — 3 Cases

Compare v_n to its preceding $W \leq 127$ values, in the time range $[t_{n-W}, t_{n-1}]$:

- **Reference.** If v_n is found in the window, write its position in the window. (1 byte)


0

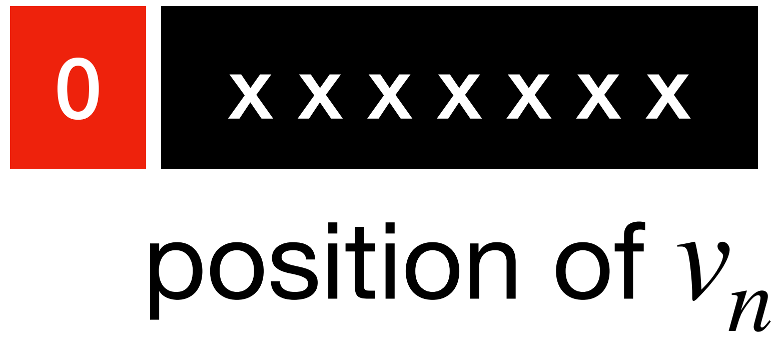
x x x x x x x

position of v_n

TSXor — 3 Cases

Compare v_n to its preceding $W \leq 127$ values, in the time range $[t_{n-W}, t_{n-1}]$:

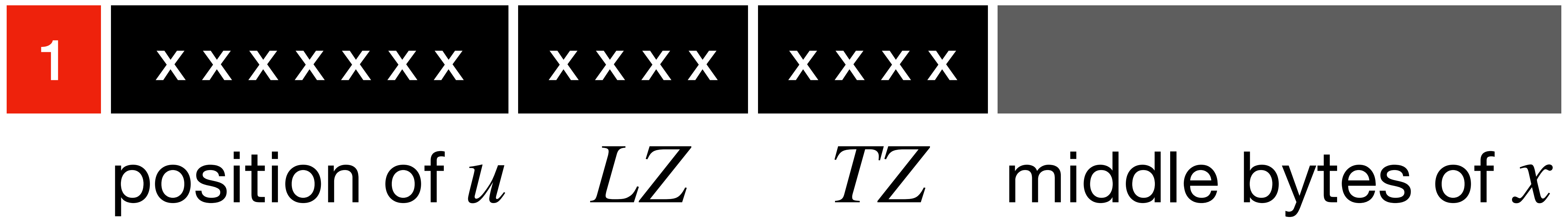
- **Reference.** If v_n is found in the window, write its position in the window. (1 byte)



0 x x x x x x x

position of v_n

- **Xor.** If v_n is *not* found in the window, determine the value u in the window such that $x = v_n \oplus u$ has the largest number of leading and trailing 0 bytes. ($>$ 2 bytes)



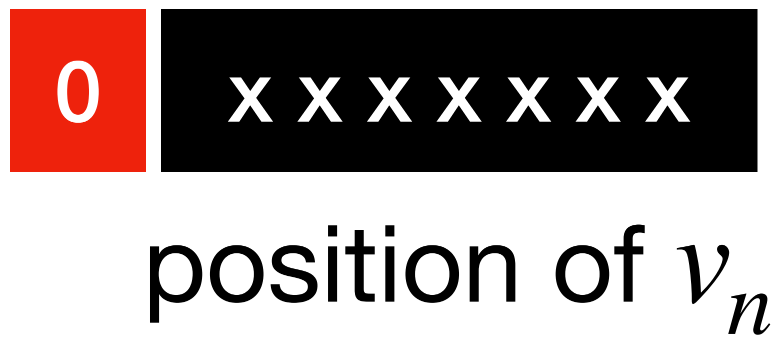
1 x x x x x x x x x x x x x x x

position of u LZ TZ middle bytes of x

TSXor — 3 Cases

Compare v_n to its preceding $W \leq 127$ values, in the time range $[t_{n-W}, t_{n-1}]$:

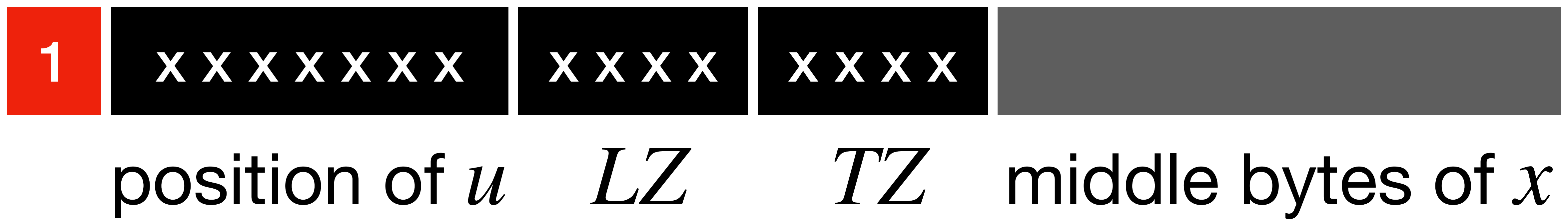
- **Reference.** If v_n is found in the window, write its position in the window. (1 byte)



0 | x x x x x x x

position of v_n

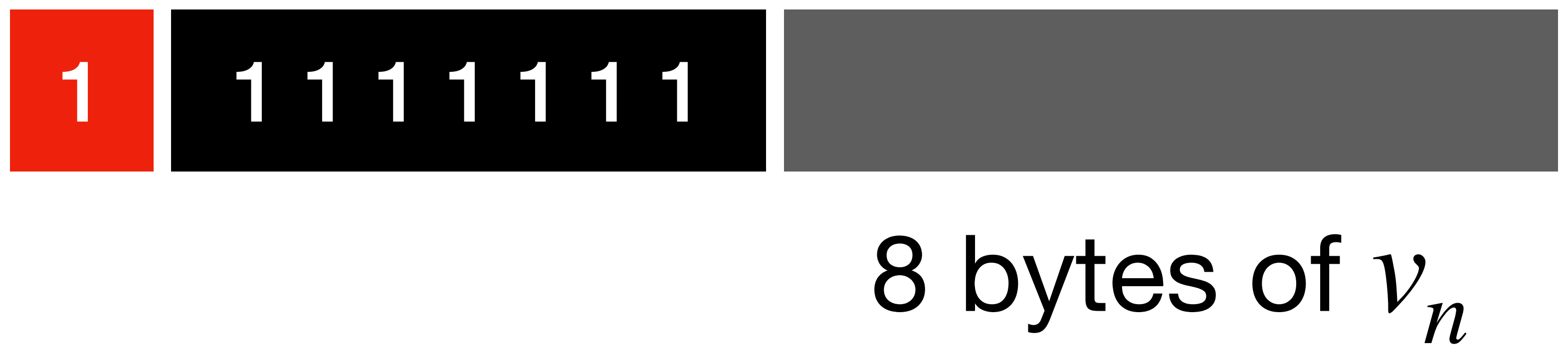
- **Xor.** If v_n is *not* found in the window, determine the value u in the window such that $x = v_n \oplus u$ has the largest number of leading and trailing 0 bytes. ($>$ 2 bytes)



1 | x x x x x x x | x x x x | x x x x

position of u | LZ | TZ | middle bytes of x

- **Exception.** If $LZ + TZ < 2$, output and exception. (9 bytes)



1 | 1 1 1 1 1 1 1

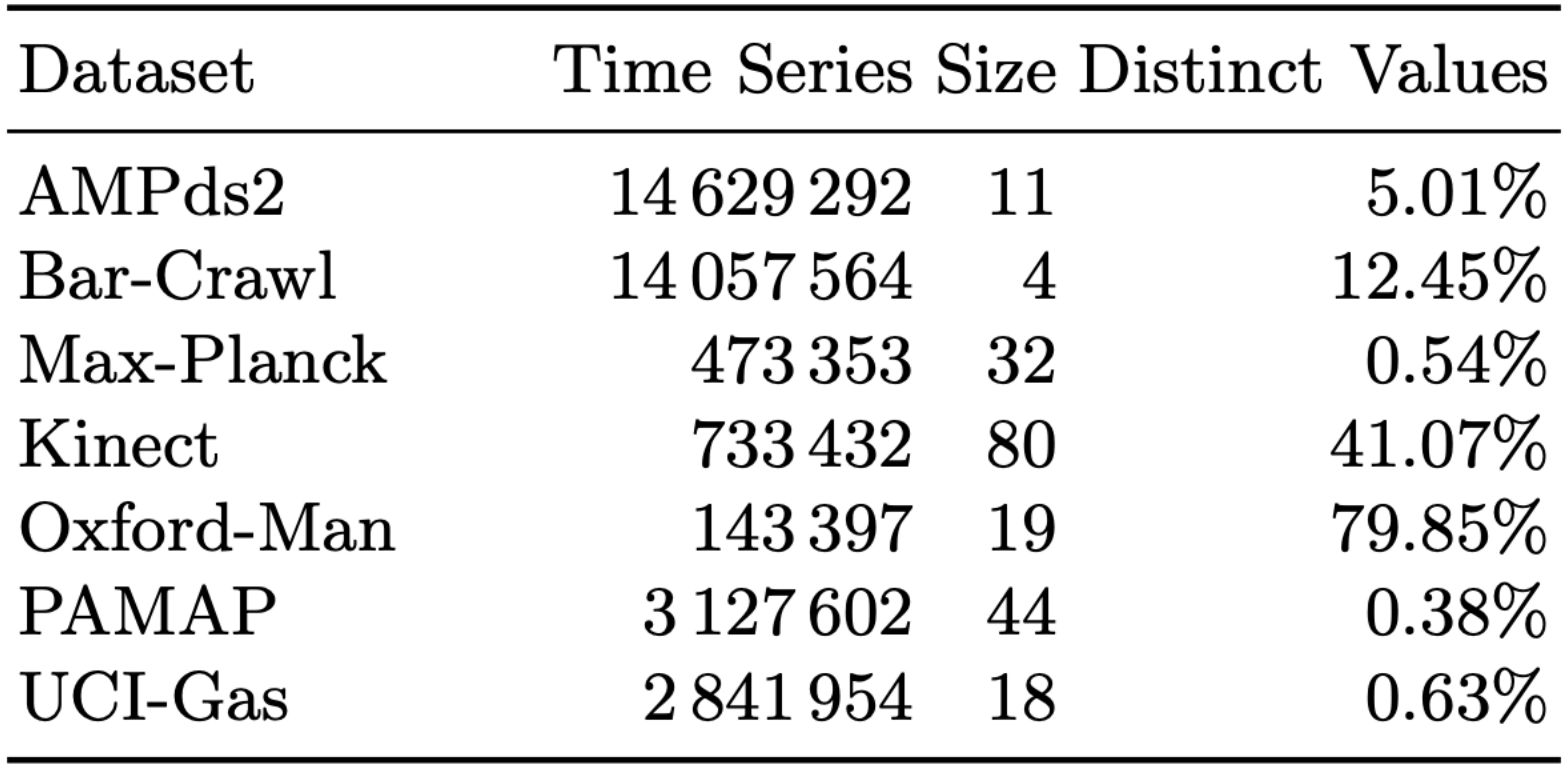
8 bytes of v_n

TSXor — Experiments

Setup

- Processor: Intel 17-7700 @ 3.6 GHz
- OS: Ubuntu 18.04
- C++ Code: https://github.com/andybbruno/TSXor
- Compiler: gcc 9.1.0, with flags `-march=native -O3`
- Details: experiments run in internal memory, single thread, ***W* = 127**

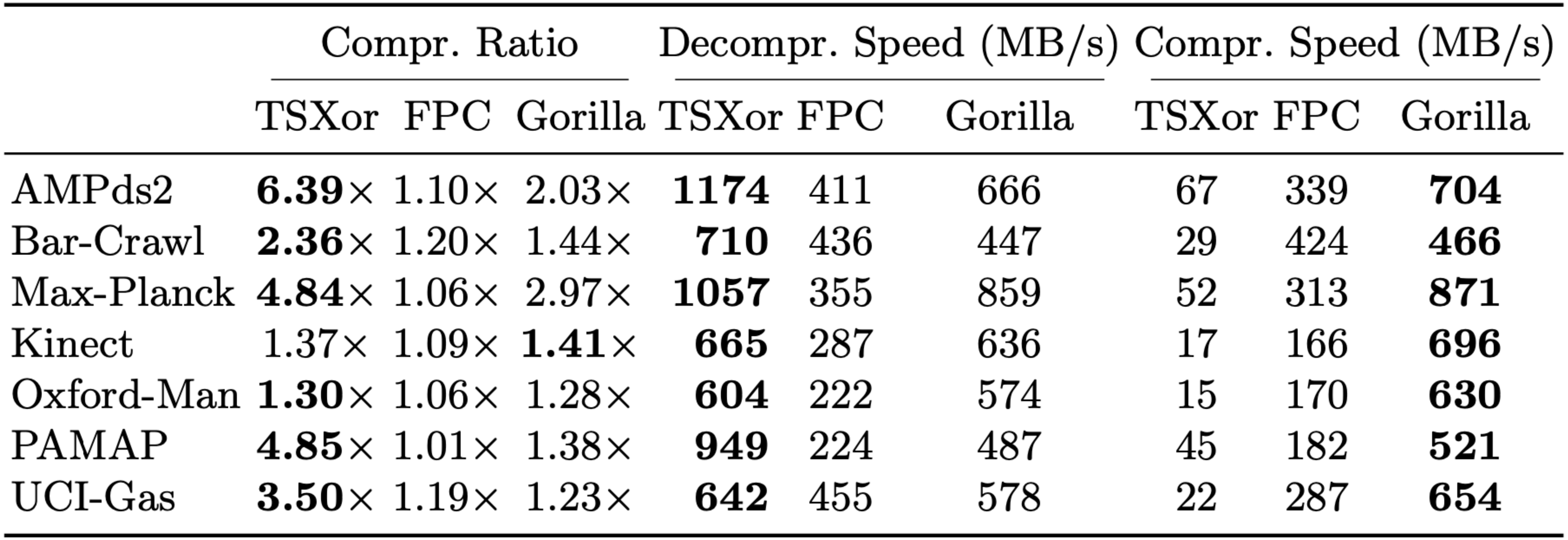
Datasets



Dataset	Time Series	Size	Distinct Values
AMPds2	14 629 292	11	5.01%
Bar-Crawl	14 057 564	4	12.45%
Max-Planck	473 353	32	0.54%
Kinect	733 432	80	41.07%
Oxford-Man	143 397	19	79.85%
PAMAP	3 127 602	44	0.38%
UCI-Gas	2 841 954	18	0.63%

TSXor — Experiments

Compression ratio, decompression speed, and compression speed



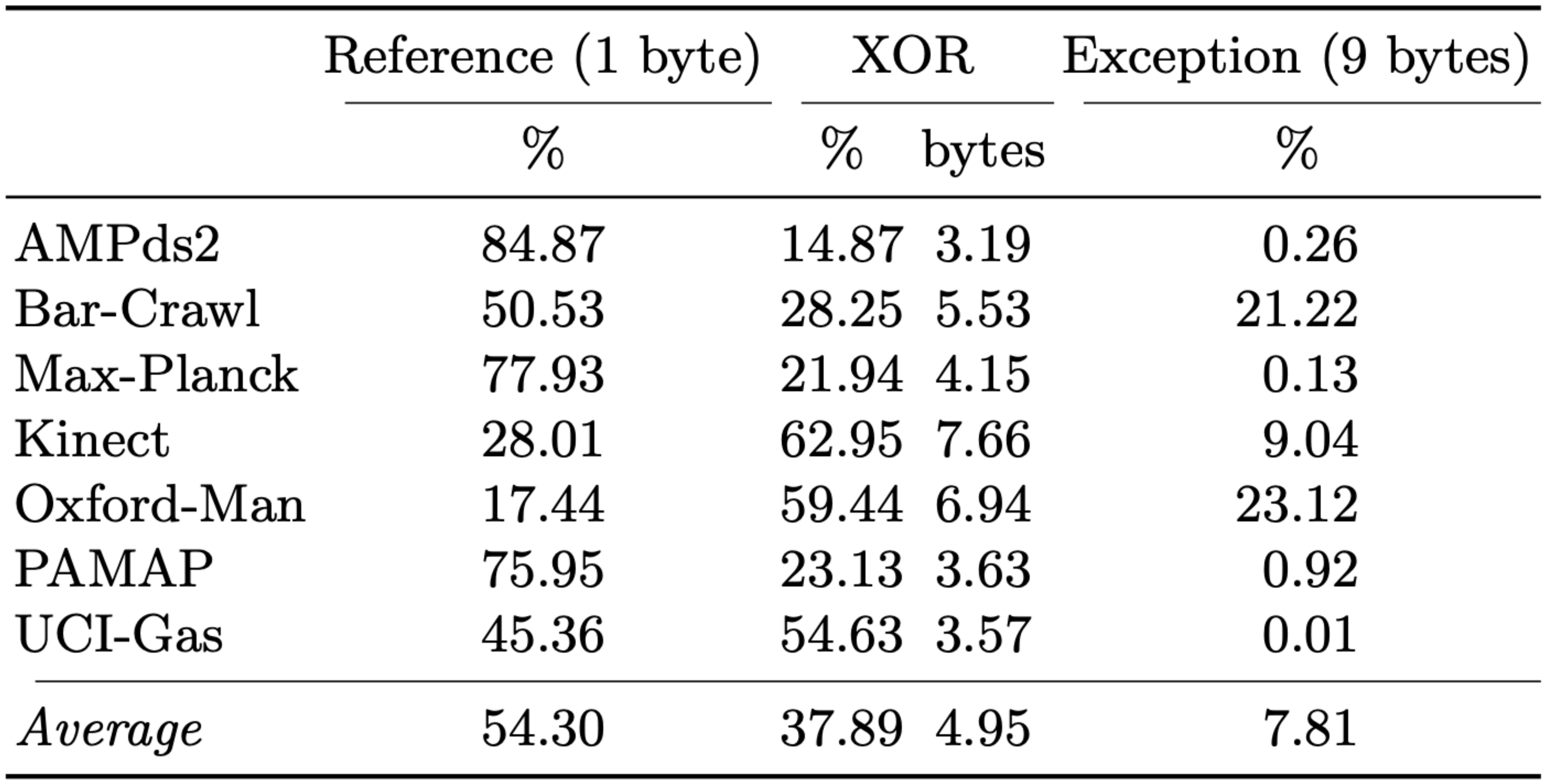
	Compr. Ratio			Decompr. Speed (MB/s)			Compr. Speed (MB/s)		
	TSXor	FPC	Gorilla	TSXor	FPC	Gorilla	TSXor	FPC	Gorilla
AMPds2	**6.39**×	1.10×	2.03×	**1174**	411	666	67	339	**704**
Bar-Crawl	**2.36**×	1.20×	1.44×	**710**	436	447	29	424	**466**
Max-Planck	**4.84**×	1.06×	2.97×	**1057**	355	859	52	313	**871**
Kinect	1.37×	1.09×	**1.41**×	**665**	287	636	17	166	**696**
Oxford-Man	**1.30**×	1.06×	1.28×	**604**	222	574	15	170	**630**
PAMAP	**4.85**×	1.01×	1.38×	**949**	224	487	45	182	**521**
UCI-Gas	**3.50**×	1.19×	1.23×	**642**	455	578	22	287	**654**

Martin Burtscher and Paruj Ratanaworabhan. **FPC**: A High-Speed Compressor for Double-Precision Floating-Point Data. IEEE Transactions on Computers, 58(1):18–31, January 2009. ISSN 0018-9340. https://doi.org/10.1109/TC.2008.131.

Tuomas Pelkonen, Scott Franklin, Justin Teller, Paul Cavallaro, Qi Huang, Justin Meza, and Kaushik Veeraraghavan. **Gorilla**: a fast, scalable, in-memory time series database. Proceedings of the VLDB Endowment, 8(12):1816–1827, August 2015. ISSN 2150-8097. https://doi.org/10.14778/2824032.2824078. **(Developed at Facebook.)**

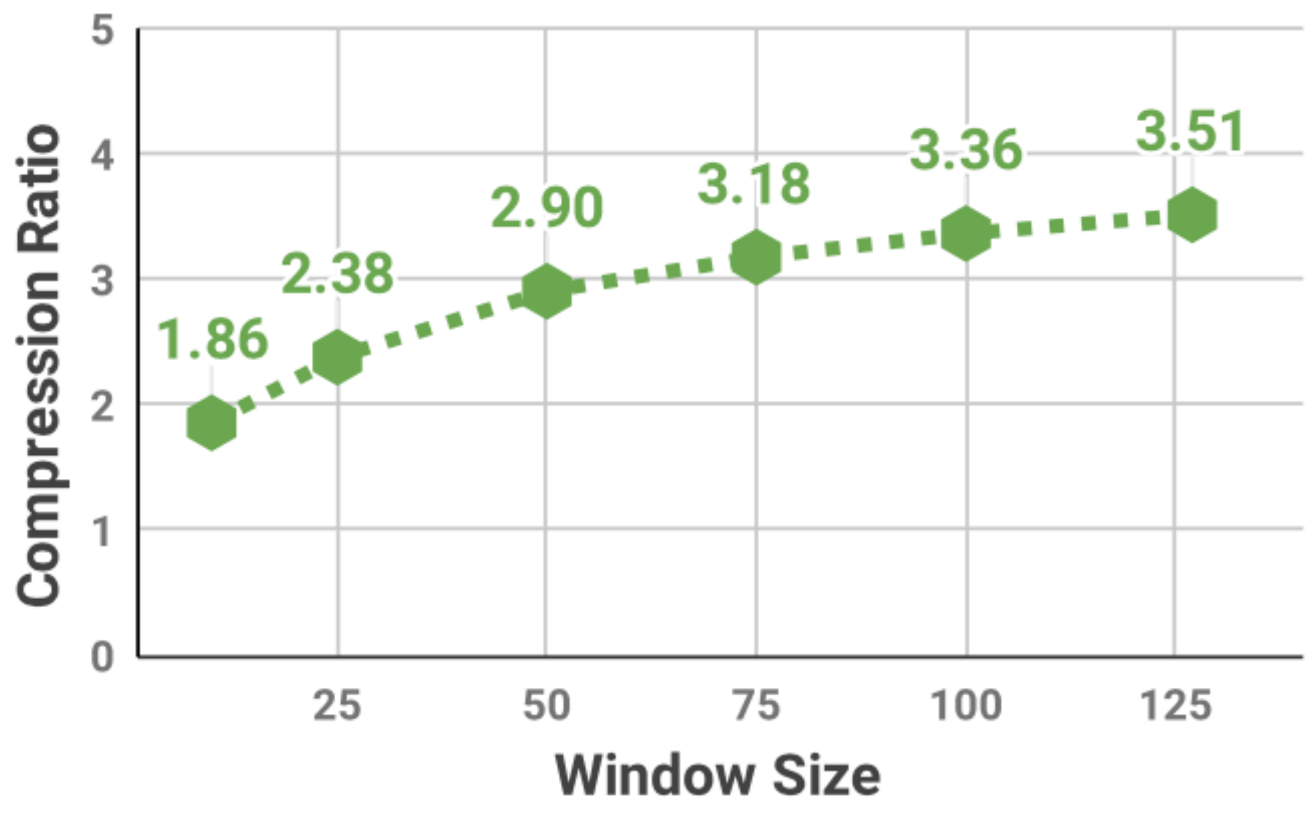
TSXor — Experiments

TSXor cases

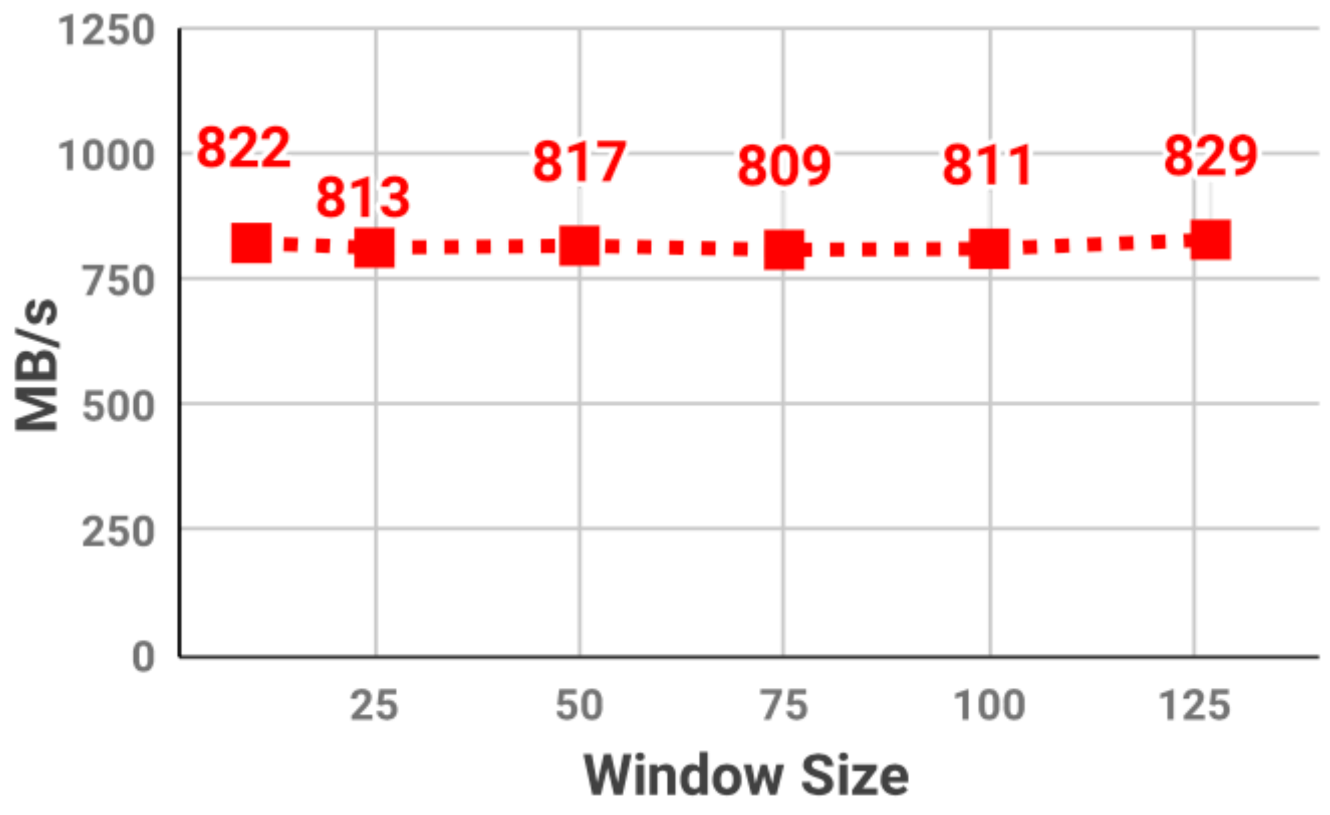


	Reference (1 byte)	XOR		Exception (9 bytes)
	%	%	bytes	%
AMPds2	84.87	14.87	3.19	0.26
Bar-Crawl	50.53	28.25	5.53	21.22
Max-Planck	77.93	21.94	4.15	0.13
Kinect	28.01	62.95	7.66	9.04
Oxford-Man	17.44	59.44	6.94	23.12
PAMAP	75.95	23.13	3.63	0.92
UCI-Gas	45.36	54.63	3.57	0.01
Average	54.30	37.89	4.95	7.81

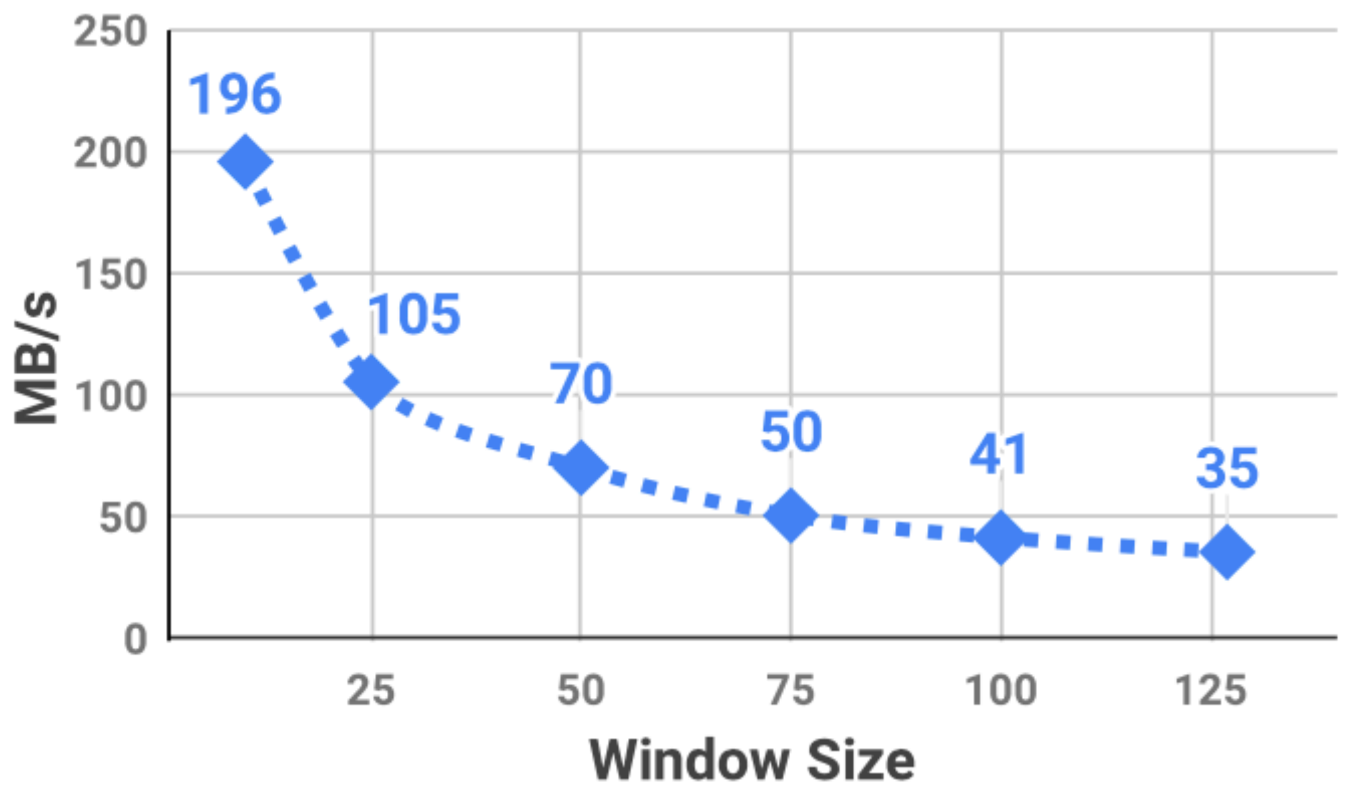
TSXor — Experiments



(a) Compr. Ratio



(b) Decompr. Speed



(c) Compr. Speed

Conclusions

- TSXor is a simple, yet effective, lossless time series compressor that achieves up to 3X better compression and up to 2X better decoding speed compared to the state-of-the-art.
  C++ code is open-source.

- TSXor trades-off compression effectiveness for encoding speed.

- Overall, TSXor provides good results, thus we are led to think that room for improvement is possible with more sophisticated mechanisms.

Thanks for your attention!

Our lab is hiring!
Please get in touch
if you are interested in working with us.

giulio.pibiri@di.unipi.it
rossano.venturini@unipi.it
francomaria.nardini@isti.cnr.it