Compression of Large Graphs: Theory and Practice

Luca Versari

April 26, 2021

Supervisor:

Roberto Grossi

Università di Pisa

Introduction and Outline

Compression and Graphs

- Why compression?
  - Makes the Internet as we know it possible
  - Strongly tied to information theory
  - Ties to artificial intelligence and machine understanding
- Why graphs?
  - Extremely well-studied mathematical object
  - Applications to multiple fields
  - Ever-increasing in size
    - 90+ billion edges
    - 360+ GB uncompressed, $\approx$ 8 GB compressed

Outline

This thesis improves the state of the art of graph compression with new compression techniques. We also show optimal compression schemes for some graph models.

- Improving the SotA of graph compression
  - Hybrid Integer Encoding and Zuckerli
- Denser (but slower) graph compression
  - New context modeling techniques and high-density Zuckerli
- Understanding optimality of graph compression
  - Theoretical results on graph compression

Hybrid Integer Encoding And Zuckerli

Entropy coding of Large Integers

- Entropy coding has per-symbol overhead
- For graphs, integers can be very large ($> 10^9$) →many symbols
- **Hybrid Integer Encoding** uses a mix of entropy-coded symbols and raw bits to reduce logarithmically the number of symbols
- Three parameters k, i, j:
  - k: number of “direct” integers
  - i: number of most-significant bits in symbols
  - j: number of least-significant bits in symbols

Entropy coding of Large Integers

$x \to (s, t)$. s is entropy coded, t has fixed length (depends on s)

$$x \quad \to \quad 1 \overbrace{b_{p-1} \ldots b_{p-i}}^{m} \overbrace{b_{p-i-1} \ldots b_{j+1}}^{t} \overbrace{b_j \ldots b_1}^{l}$$

$$s = 2^k + (p - k - 1) \cdot 2^{i+j} + m \cdot 2^j + l$$

	4, 2, 0			4, 1, 1			2, 1, 0			2, 0, 2		
	s	*n*	*t*	*s*	*n*	*t*	*s*	*n*	*t*	*s*	*n*	*t*
0	0	0	–	0	0	–	0	0	–	0	0	–
1	1	0	–	1	0	–	1	0	–	1	0	–
2	2	0	–	2	0	–	2	0	–	2	0	–
3	3	0	–	3	0	–	3	0	–	3	0	–
4	4	0	–	4	0	–	4	1	0	4	0	–
5	5	0	–	5	0	–	4	1	1	5	0	–
6	6	0	–	6	0	–	5	1	0	6	0	–
7	7	0	–	7	0	–	5	1	1	7	0	–
8	8	0	–	8	0	–	6	2	00	8	1	0
9	9	0	–	9	0	–	6	2	01	9	1	0
10	10	0	–	10	0	–	6	2	10	10	1	0
11	11	0	–	11	0	–	6	2	11	11	1	0
12	12	0	–	12	0	–	7	2	00	8	1	1
13	13	0	–	13	0	–	7	2	01	9	1	1
14	14	0	–	14	0	–	7	2	10	10	1	1
15	15	0	–	15	0	–	7	2	11	11	1	1
16	16	2	00	16	2	00	8	3	000	12	2	00
17	16	2	01	17	2	00	8	3	001	13	2	00

Hybrid Integer Encoding: Analysis

Theorem

The redundancy of Hybrid Integer Encoding with parameters $k, i, 0$ on a stream of i.i.d. random variables, where value v has probability p_v, can be bounded from above by

$$r = \max_{a \geq k} \left\{ 1, \max_{0 \leq b < 2^i} \left\{ \max_{v \in l_{a,b}} \frac{a - i - \log \sum_{v' \in l_{a,b}} p_{v'}}{-\log p_{v'}} \right\} \right\}$$

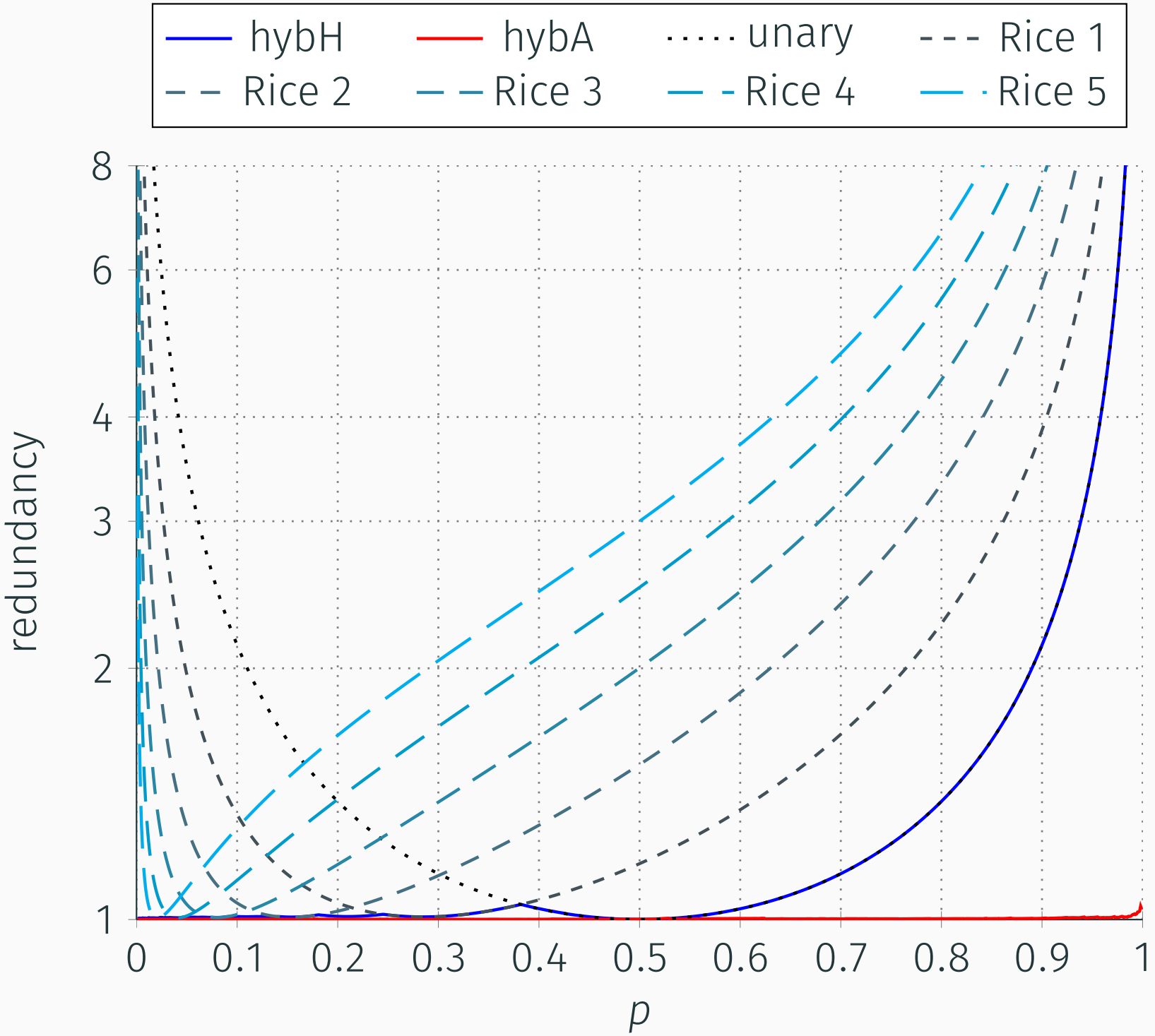
where $l_{a,b} = [2^a + b \cdot 2^{a-i}, 2^a + (b+1) \cdot 2^{a-i})$.

Hybrid Integer Encoding: Geometric distributions

Theorem

The redundancy of Hybrid Integer Encoding with parameters $k, i, 0$ when encoding a geometric distribution with probability of failure p is at most

$$r_{\alpha,k,i} \leq \begin{cases} 1 + \frac{2^{-(i+1)}}{\log \frac{1}{p}} & \text{if } k \leq i+2 \\ 1 + \frac{(k-i)2^{-k}}{\log \frac{1}{p}} & \text{otherwise} \end{cases}$$

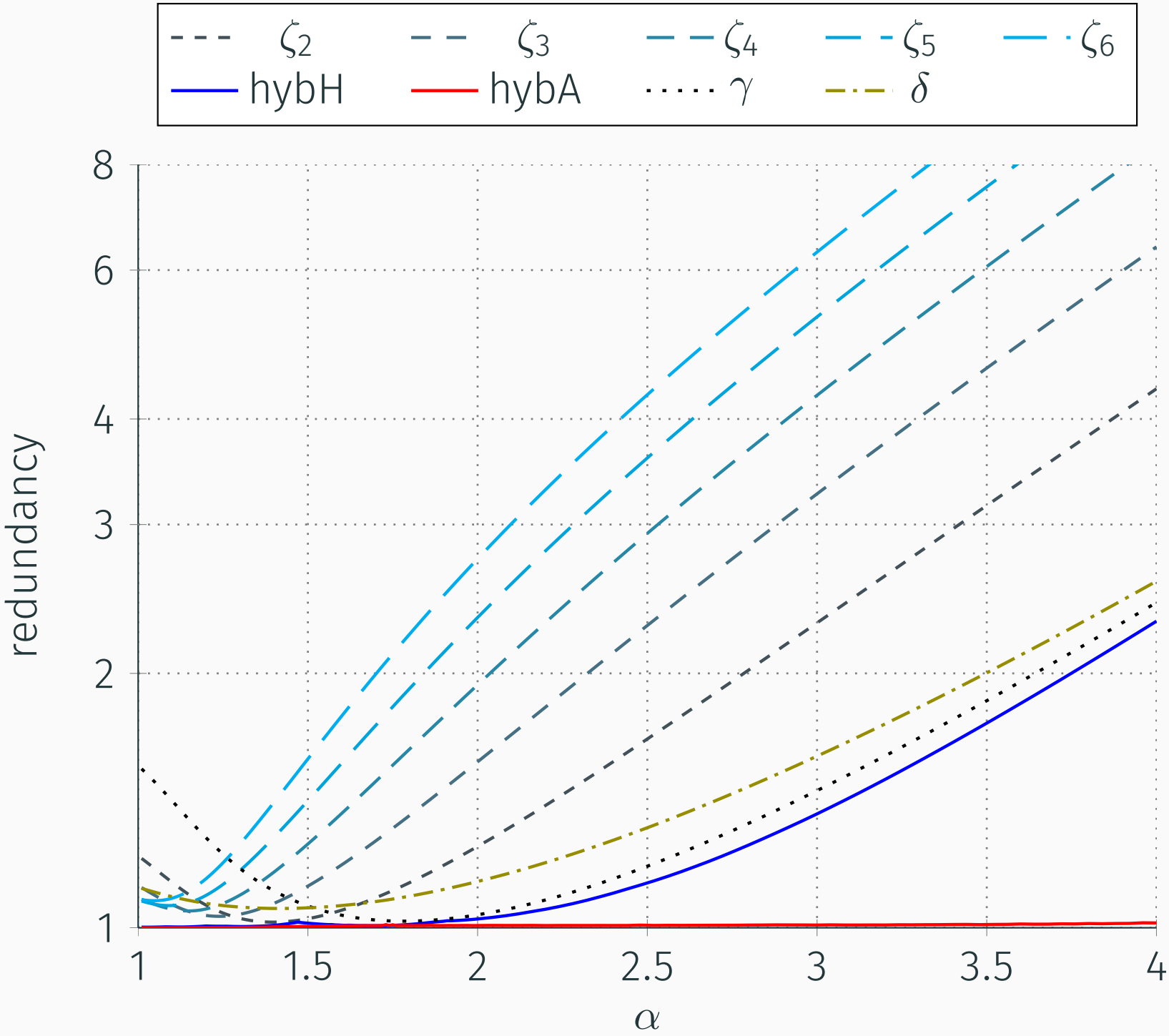

hybH
hybA
unary
Rice 1
Rice 2
Rice 3
Rice 4
Rice 5
redundancy
8
6
4
3
2
1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
p

Hybrid Integer Encoding: Zipf distributions

Theorem

The redundancy of Hybrid Integer Encoding with parameters $k, i, 0$ when encoding a Zipf distribution with parameter α is at most

$$r_{\alpha,k,i} \leq 1 + \frac{\log(1 + 2^{-i})}{k + \frac{\log \zeta(\alpha)}{\alpha}}$$


ζ2
ζ3
ζ4
ζ5
ζ6
hybH
hybA
γ
δ
redundancy
8
6
4
3
2
1
1
1.5
2
2.5
3
3.5
4
α

Application of Hybrid Integer Encoding to graph compression.

Similar model to WebGraph[1], but without intervals; for each node, store:

- Degree
- Reference node (0 for no reference, 1 for previous node, ...)
- Number of blocks to copy/skip and their lengths
- All other edges

[1] Paolo Boldi and Sebastiano Vigna. "The WebGraph framework I: compression techniques". In: *Proceedings of the 13th international conference on World Wide Web*. 2004, pp. 595–602.

Zuckerli: reference selection for random access.

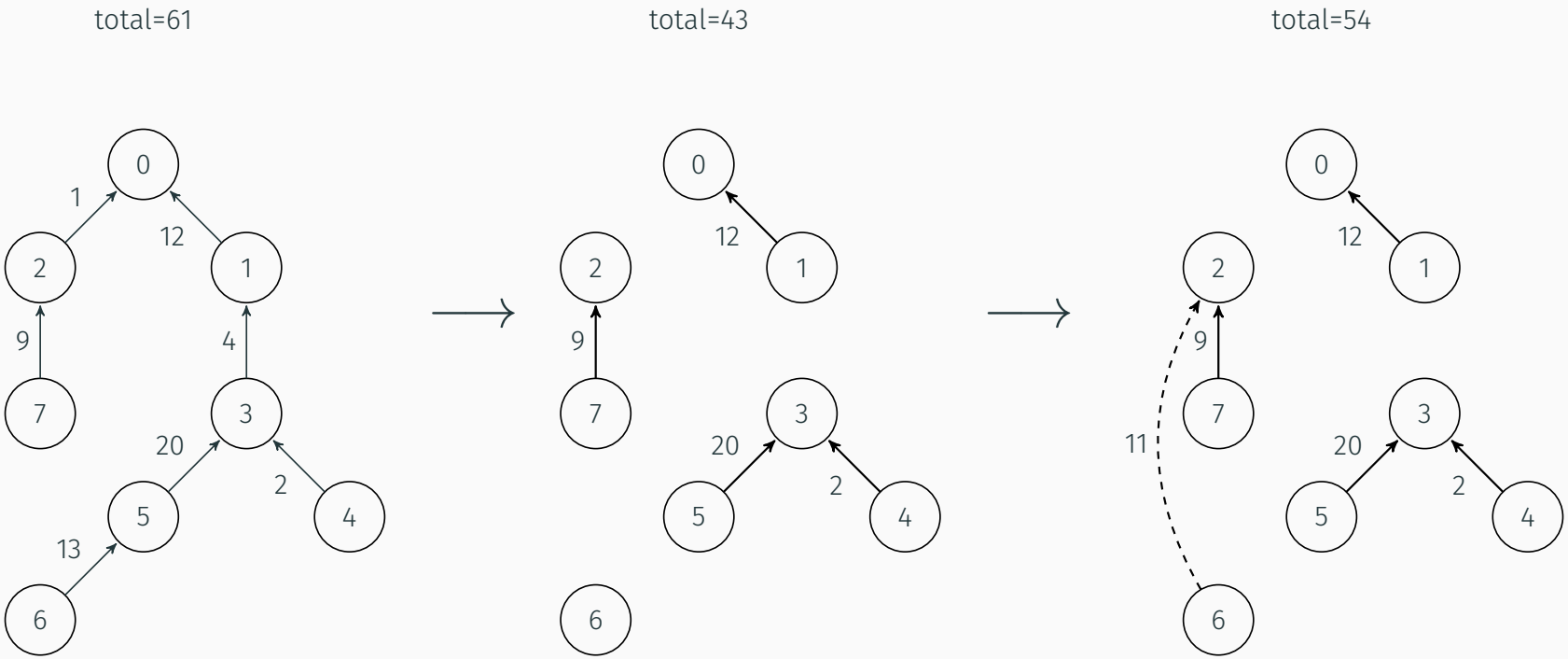
We want to limit the length of “reference chains”.
Approximation algorithm:

- Find the best references with no chain length limit
- Use dynamic programming on the resulting forest to find an optimal sub-forest with desired maximum length
- Find new references to use that don’t create long chains

For maximum length R, this is a $\frac{R}{R+1}$-approximation of the gains of using references.

Zuckerli: reference selection example.

Example with maximum length 2.



name	bits/edge											
	Zuckerli	WebGraph		GCBFS		k^2-trees		LM		Backlinks	
cnr-2000-hc	1.84	2.45	133%	1.88	102%		-	2.28	123%		-
cnr-2000	2.19	3.12	142%	2.72	124%	3.12	142%	3.46	158%	10.51	480%
in-2004-hc	1.31	1.76	134%	1.42	108%		-	1.74	132%		-
in-2004	1.65	2.15	130%	2.16	131%	2.25	136%	2.60	157%	9.62	583%
eu-2005-hc	2.46	3.16	128%	2.81	114%		-	2.34	95%		-
eu-2005	2.83	3.72	131%	3.50	124%	3.23	114%	3.48	123%	10.26	362%
bn-jung-hc	2.07	2.57	124%	4.97	240%		-	2.86	138%		-
bn-jung	2.41	2.96	123%	4.78	200%		-	3.21	133%	3.52	146%
uk-2002-hc	1.36	1.80	132%	1.71	126%		-	1.78	130%		-
uk-2002	1.72	2.24	130%	2.43	141%	2.26	131%	2.66	154%	10.54	613%
hw-2009-hc	4.26	4.80	113%	7.29	171%		-	5.11	120%		-
hw-2009	4.47	4.94	111%	7.51	168%	6.76	151%	5.83	124%	4.76	106%
tw-2010-hc	11.97	13.89	116%	15.34	128%		-	13.75	115%		-
tw-2010	12.12	14.46	119%	15.21	125%	19.53	161%	15.70	129%	15.43	127%
pp-miner-hc	3.48	3.80	109%	4.60	132%		-	4.34	124%		-
pp-miner	3.81	4.37	114%	4.60	120%		-	5.19	136%	3.73	98%
uk-2007-02-hc	0.91	1.18	130%	1.28	141%		-	1.06	116%		-
uk-2007-02	1.18	1.56	132%	1.80	152%	2.58	219%	1.62	137%	8.03	681%
eu-2015-hc	0.74	0.89	120%		-		-		-		-
eu-2015	0.92	1.20	130%		-		-		-		-

Advanced context modeling

Higher-order context modeling

High order context modeling uses the previous values of the encoded data to pick a distribution for subsequent data.

000000000000000000011111111111111111

For order-1 contexts, separate distribution for after-0 and 1. It can be impractical in its “raw” form: too many distributions!

Context clustering

Main idea: make clusters of contexts that share the same distribution

Likely NP-hard

Heuristic inspired by k-means++[2] initialization: pick maximally-separated centers according to

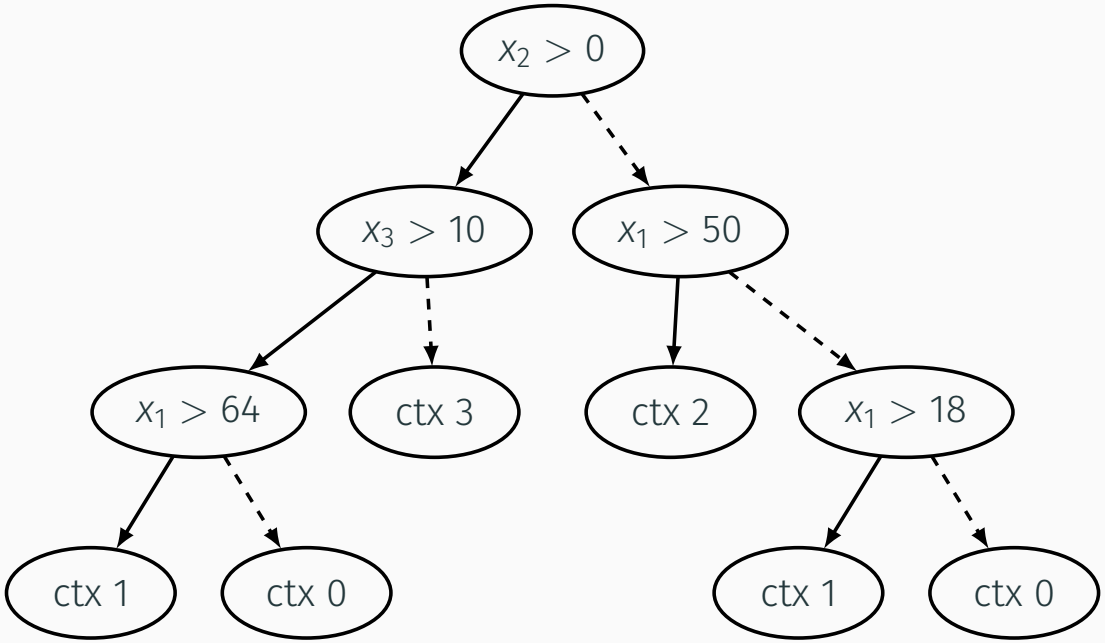
$$D(h(A), h(B)) = H(A \cup B) - H(A) - H(B)$$

[2] David Arthur and Sergei Vassilvitskii. *k-means++: The advantages of careful seeding*. Tech. rep. Stanford, 2006.

Context Trees: description

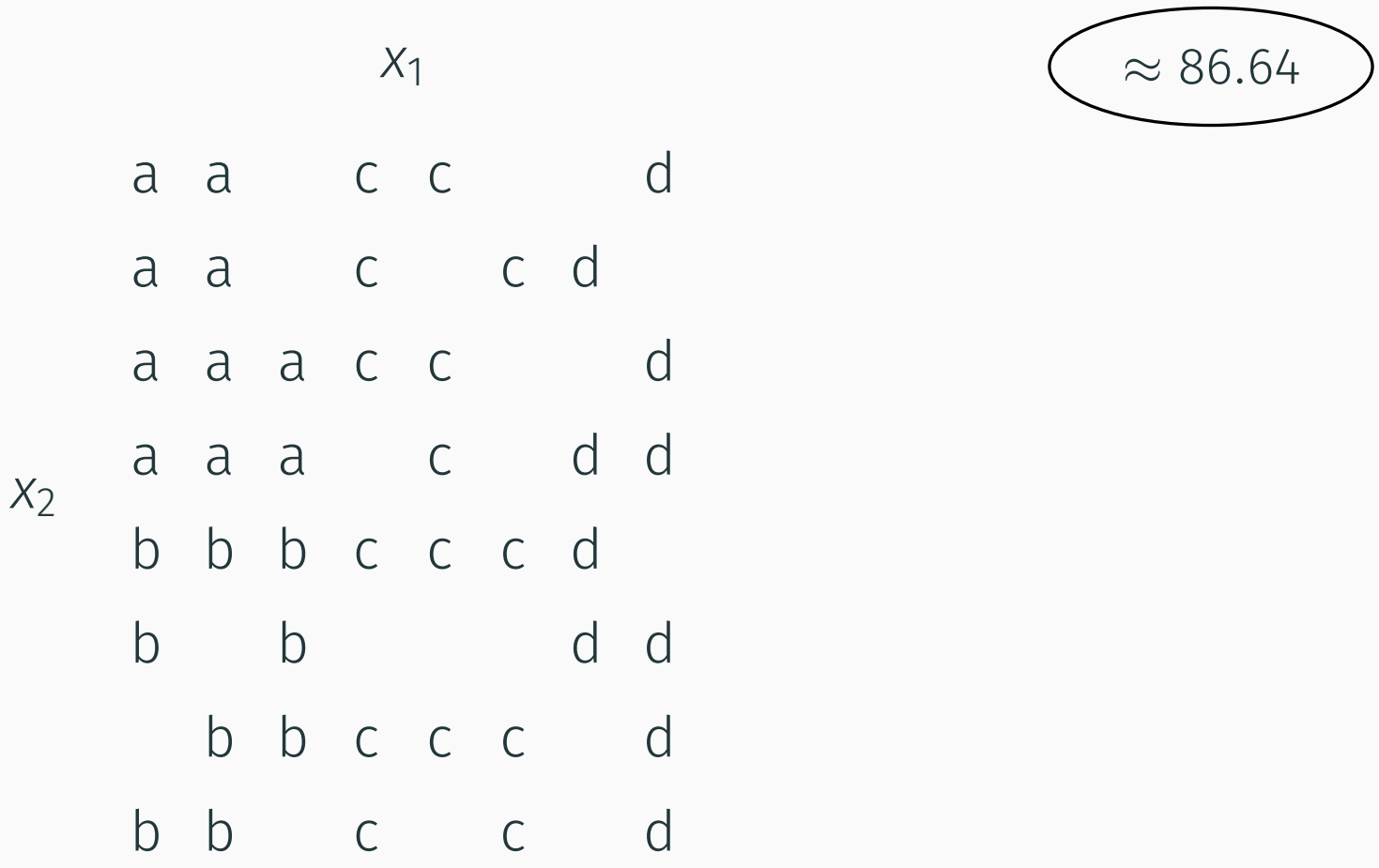
We assume that the context is given by an array $(x_1, x_2, \dots)$

To find the distribution id, we navigate a tree like the one represented here.



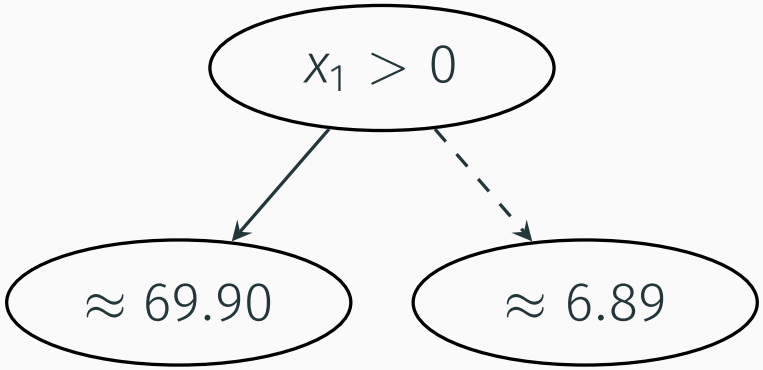
Finding optimal trees is probably NP-Hard.

Context Trees: construction example

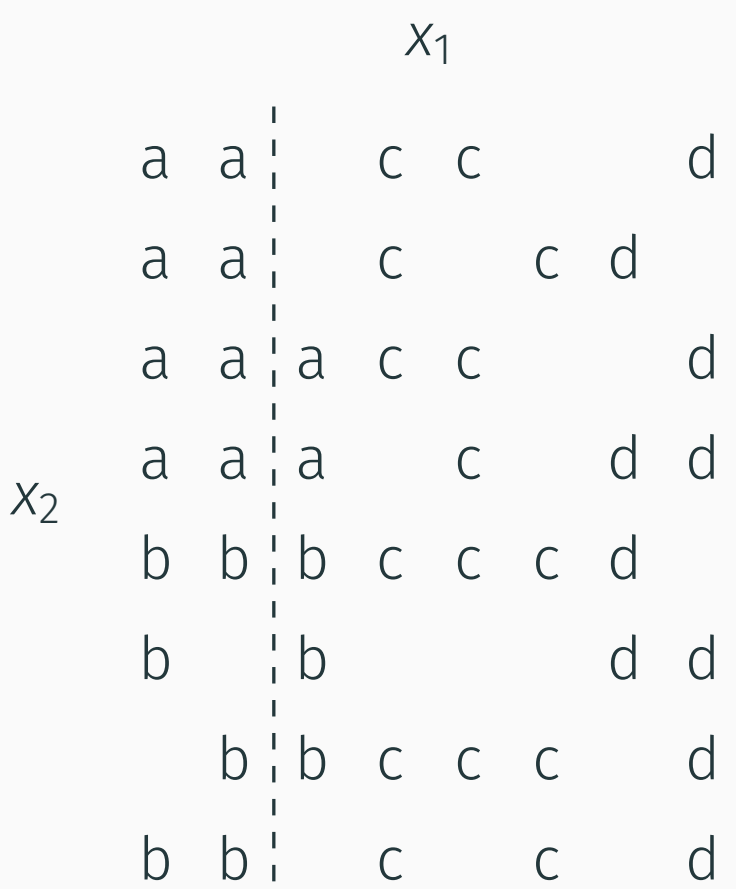


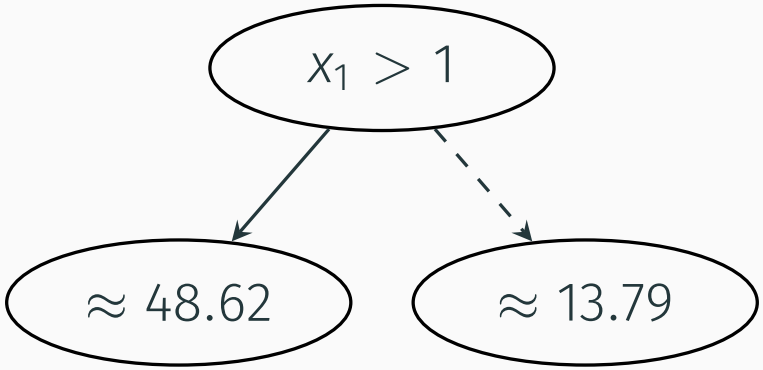
Context Trees: construction example

x_2 \ x_1	a	a	a	c	c	c	d	d
a	a			c	c			d
a	a			c		c	d	
a	a	a	a	c	c			d
a	a	a		c			d	d
b	b	b	b	c	c	c	d	
b		b					d	d
	b	b	b	c	c	c		d
b	b			c		c		d



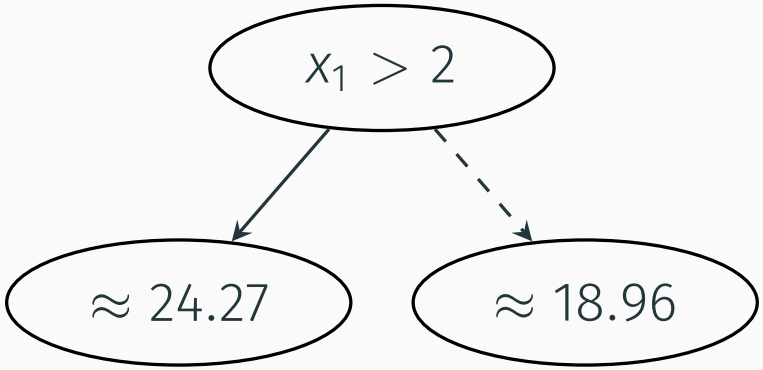
Context Trees: construction example



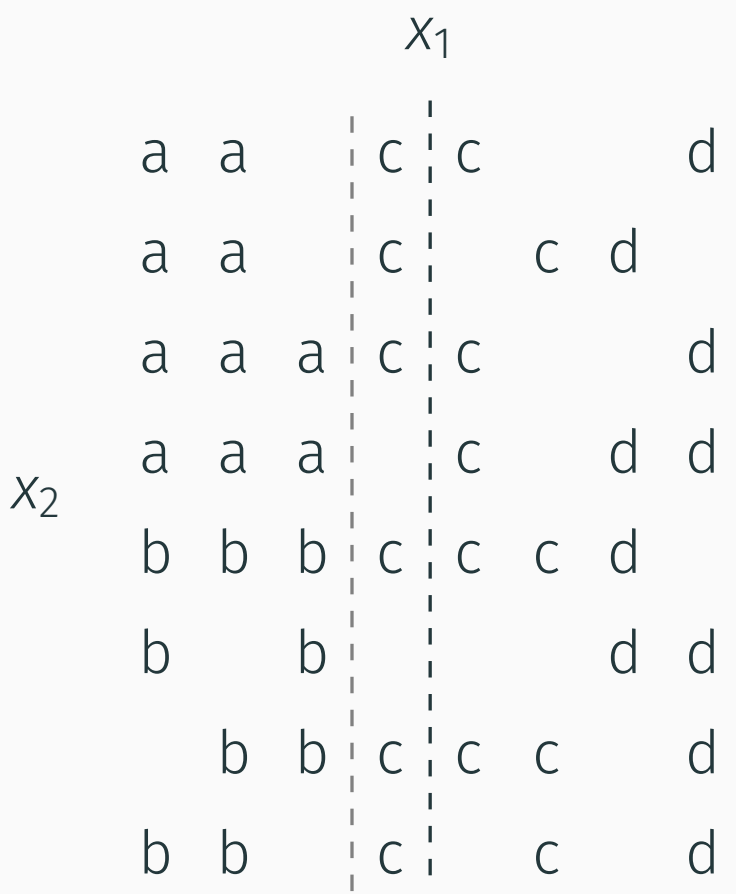


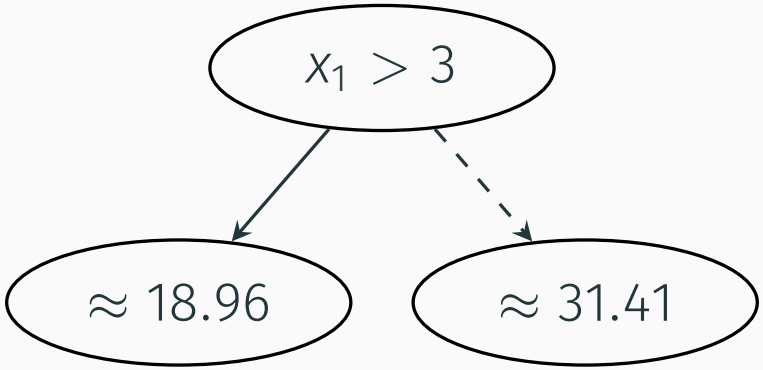
Context Trees: construction example

		x_1					
	a	a		c	c		d
	a	a		c		c	d
	a	a	a	c	c		d
	a	a	a		c		d	d
x_2	b	b	b	c	c	c	d
	b		b				d	d
		b	b	c	c	c		d
	b	b		c		c		d



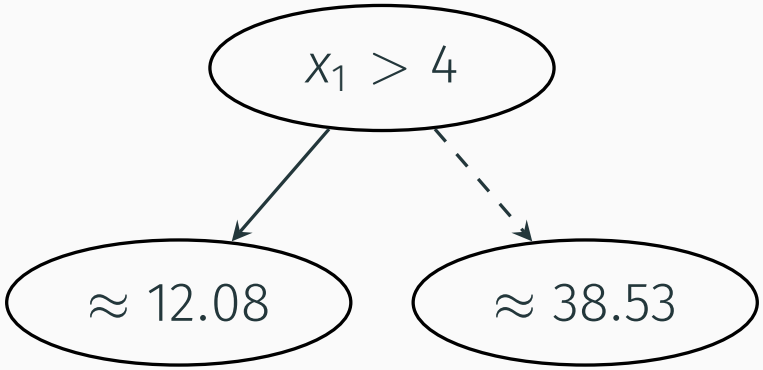
Context Trees: construction example





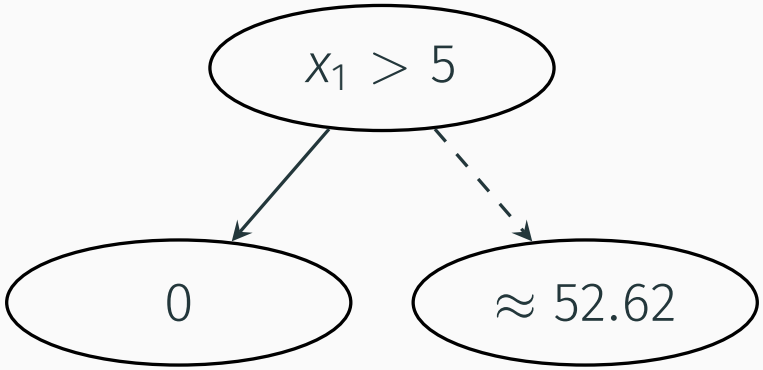
Context Trees: construction example

x_2 \ x_1								
	a	a		c	c			d
	a	a		c		c	d	
	a	a	a	c	c			d
	a	a	a		c		d	d
	b	b	b	c	c	c	d	
	b		b				d	d
		b	b	c	c	c		d
	b	b		c		c		d



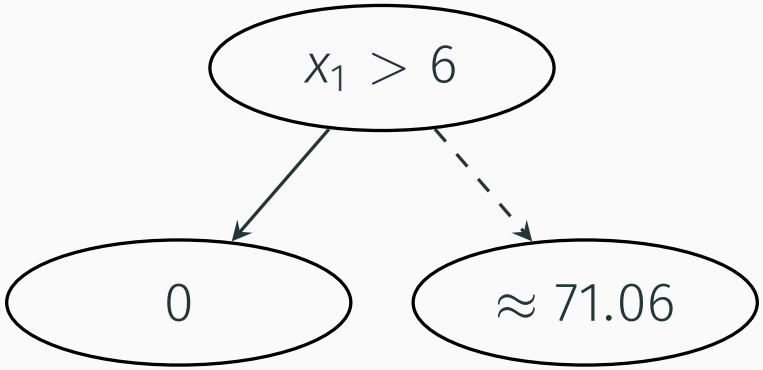
Context Trees: construction example

	x_1							
	a	a		c	c			d
	a	a		c		c	d	
	a	a	a	c	c			d
	a	a	a		c		d	d
x_2	b	b	b	c	c	c	d	
	b		b				d	d
		b	b	c	c	c		d
	b	b		c		c		d

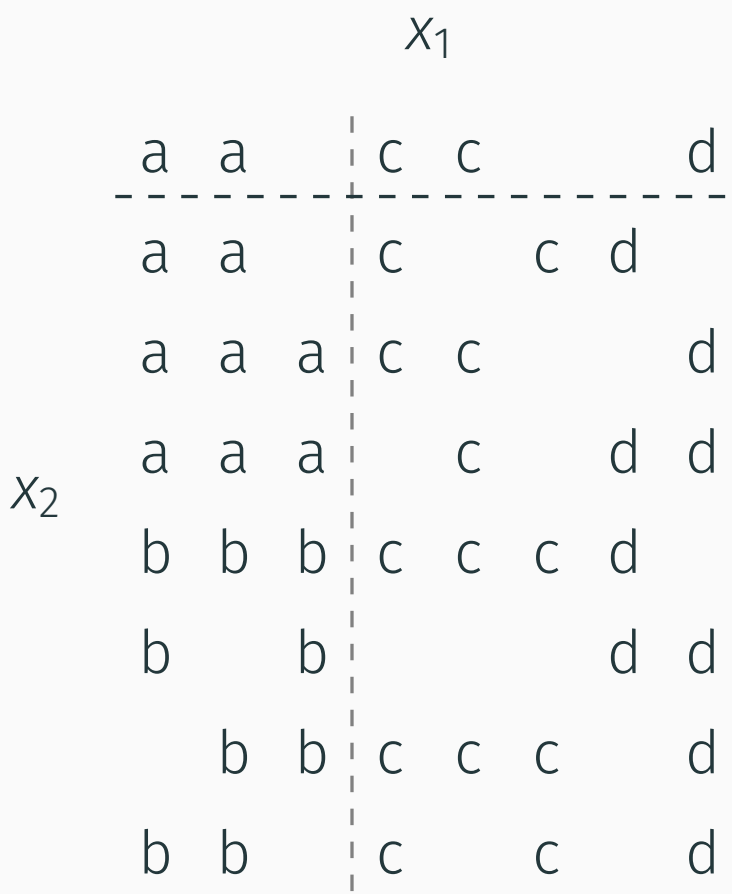


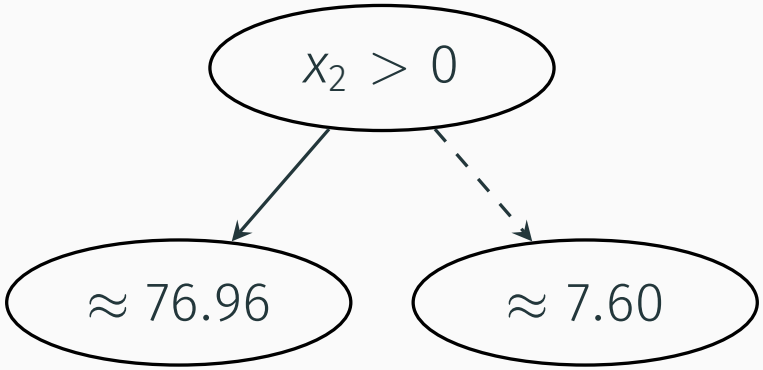
Context Trees: construction example

x_2 \ x_1								
	a	a		c	c			d
	a	a		c		c	d	
	a	a	a	c	c			d
	a	a	a		c		d	d
	b	b	b	c	c	c	d	
	b		b				d	d
		b	b	c	c	c		d
	b	b		c		c		d

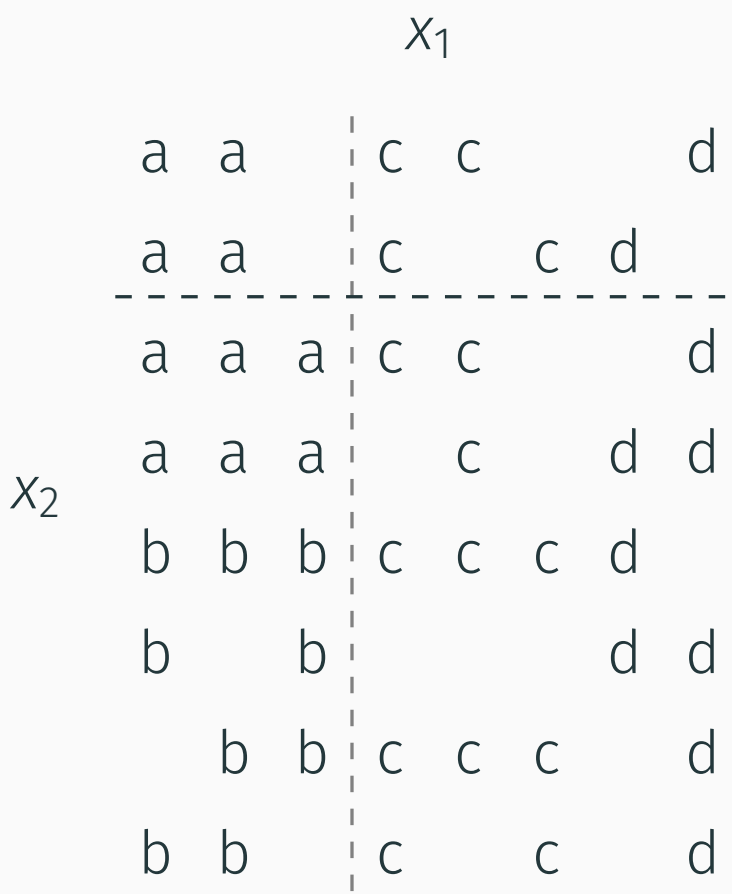


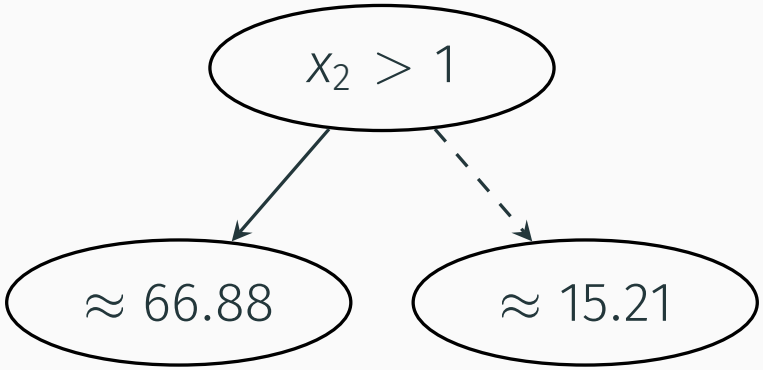
Context Trees: construction example



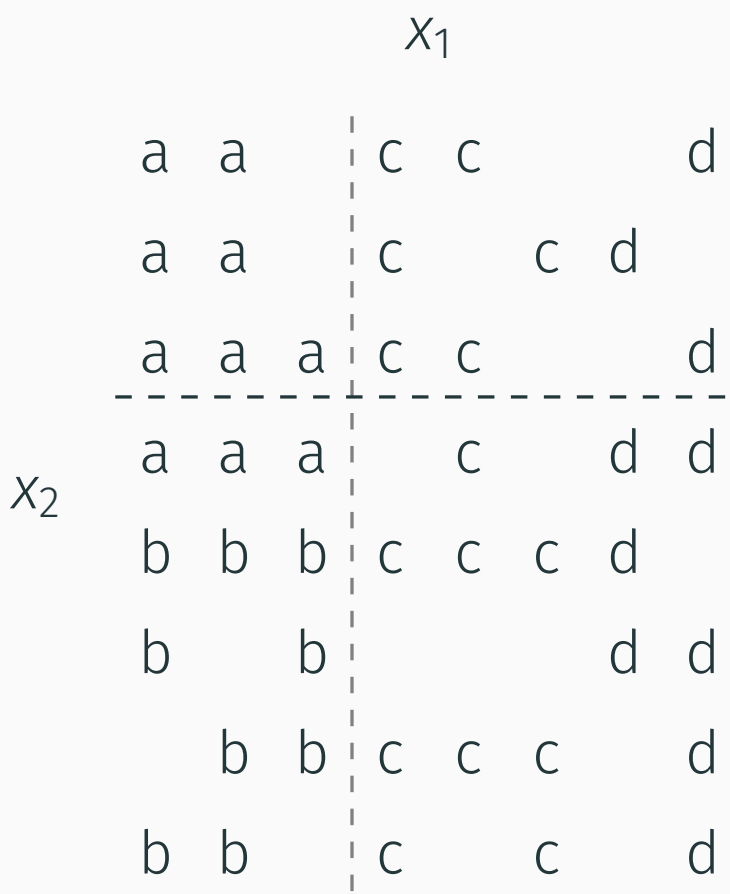


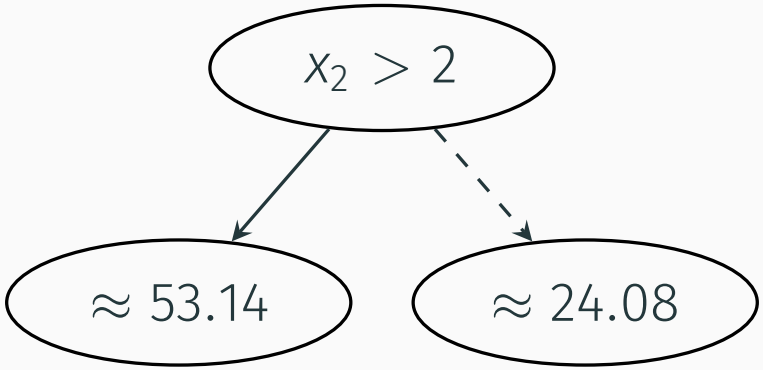
Context Trees: construction example



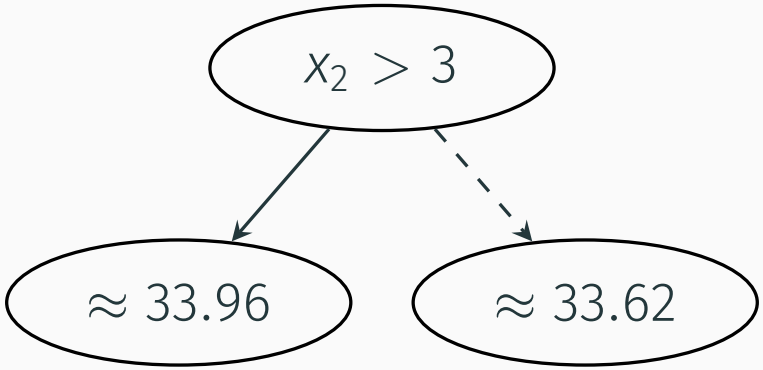


Context Trees: construction example

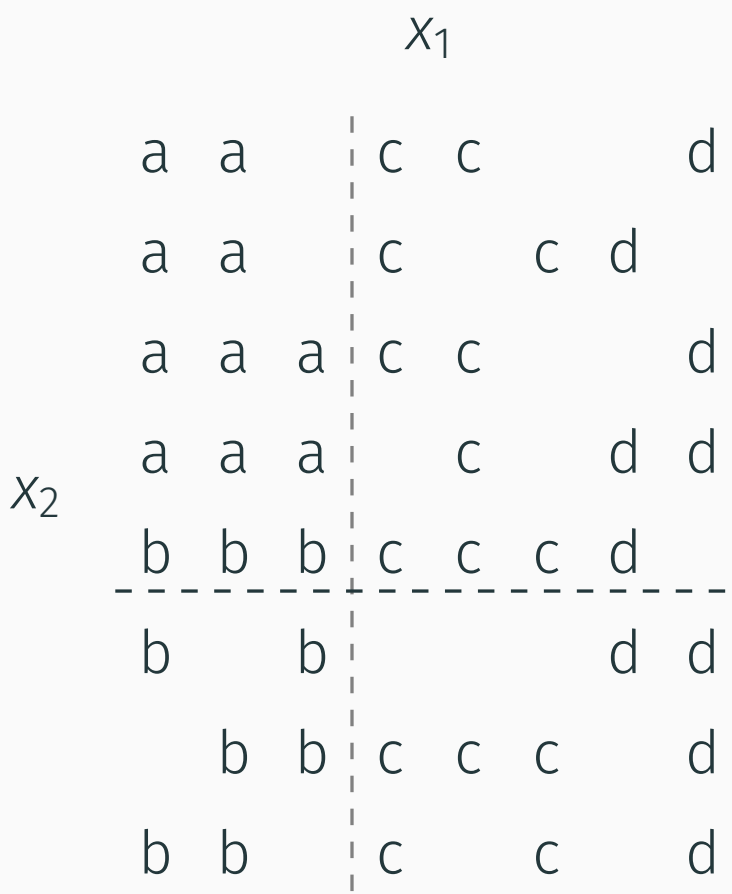


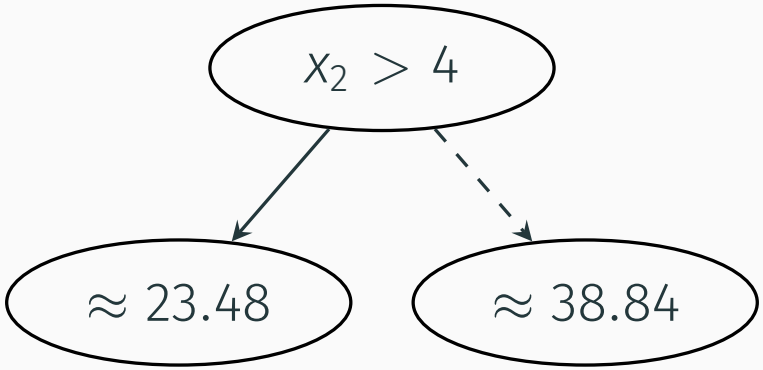


Context Trees: construction example

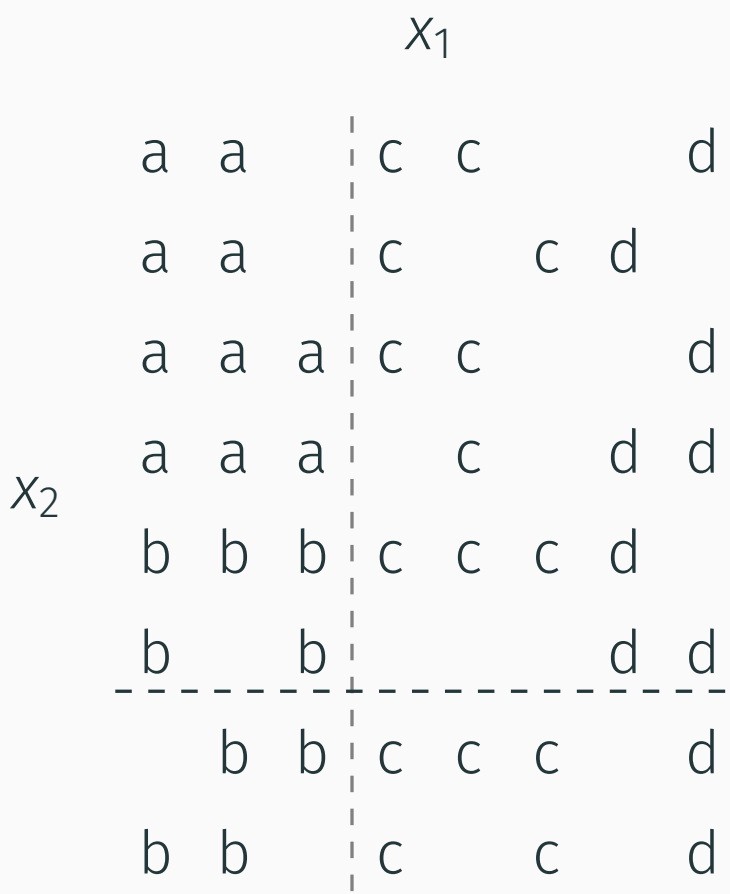


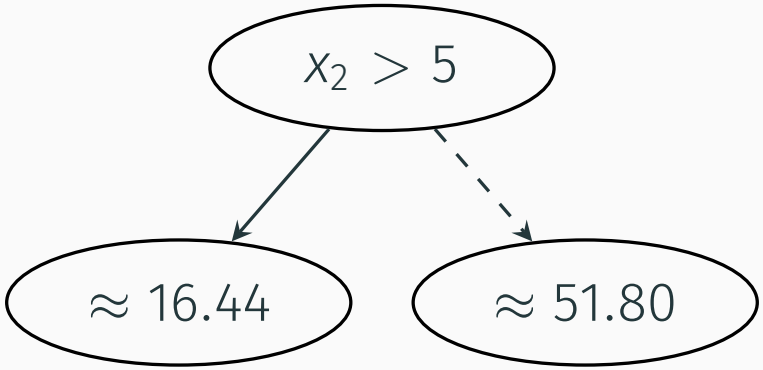
Context Trees: construction example



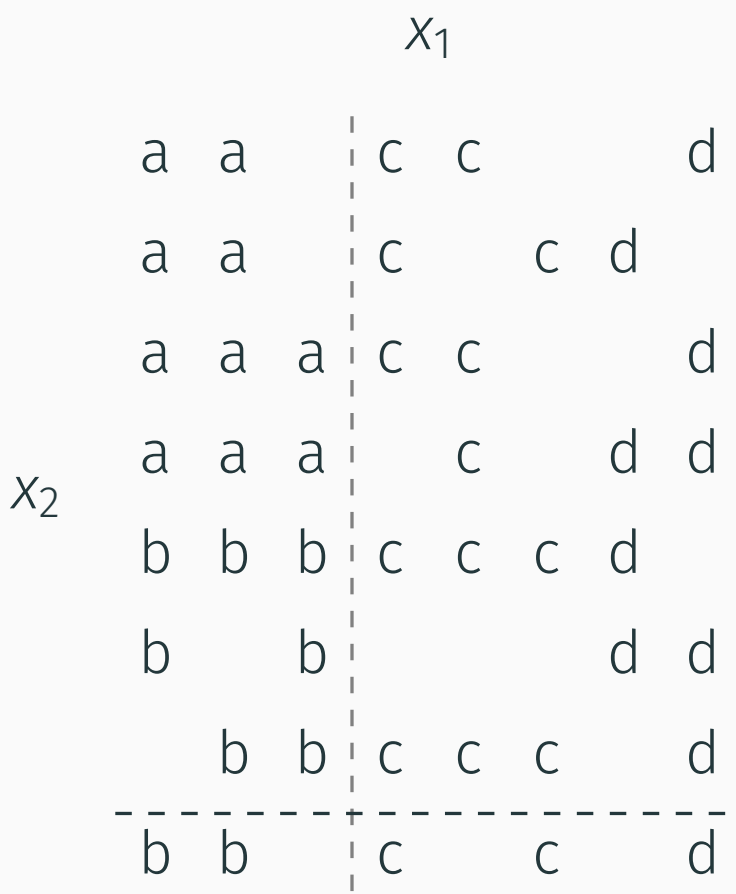


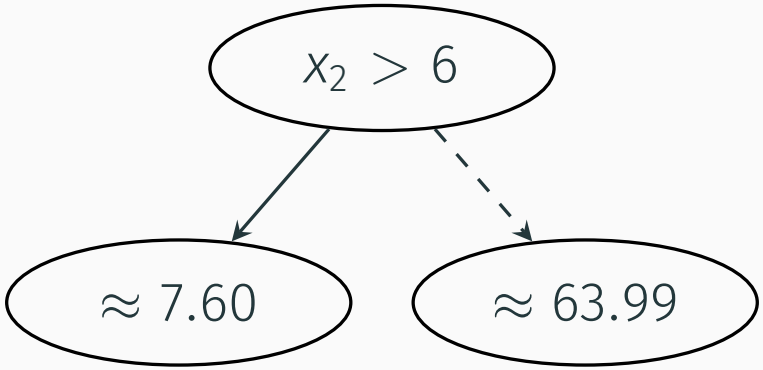
Context Trees: construction example





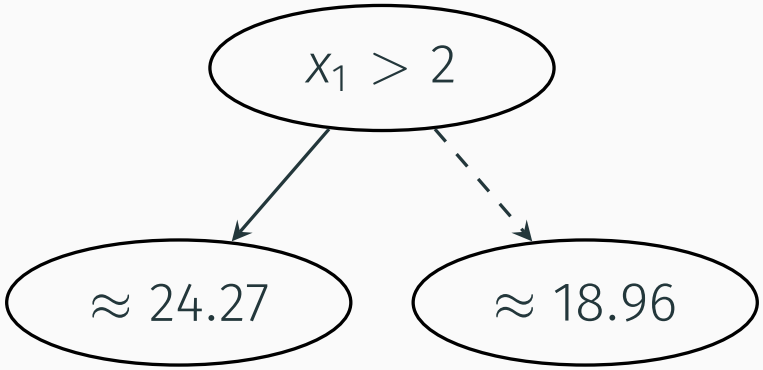
Context Trees: construction example



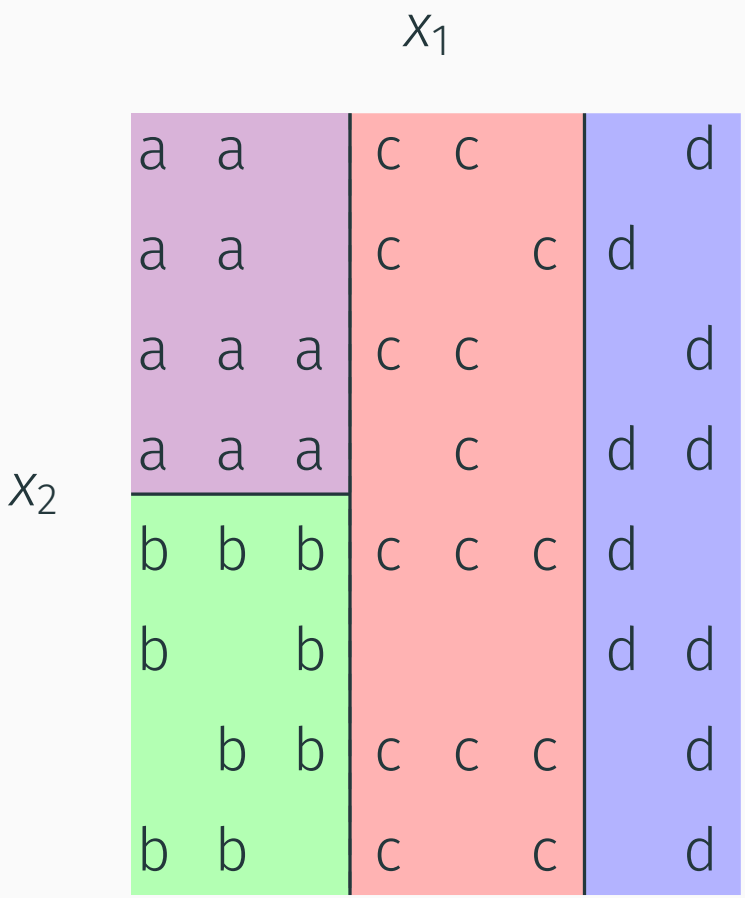


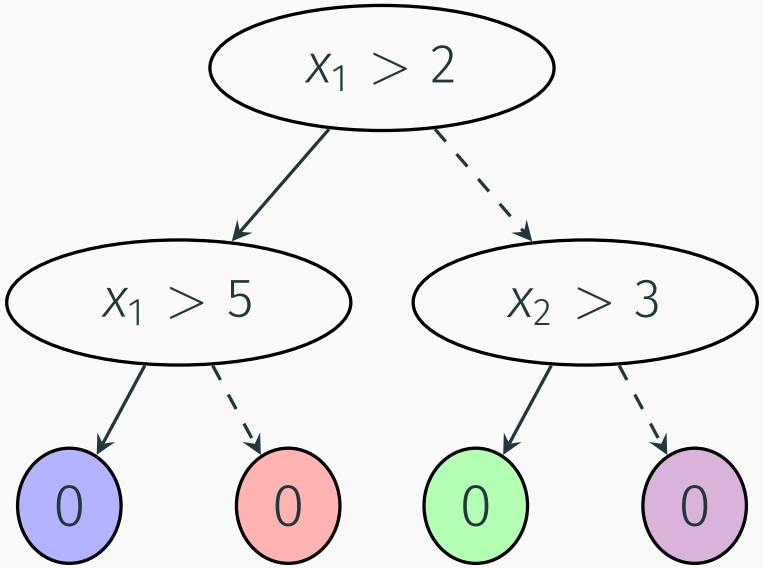
Context Trees: construction example

		x_1						
		a	a		c	c		d
		a	a		c		c	d
		a	a	a	c	c		d
		a	a	a		c		d
x_2		b	b	b	c	c	c	d
		b		b				d
			b	b	c	c	c	d
		b	b		c		c	d



Context Trees: construction example





Context Trees: application to graph compression

We can use context trees as a context model for Zuckerli

name	Zuckerli	tree
cnr-2000-hc	1.89	1.57
cnr-2000-nat-hc	2.07	1.58
in-2004-hc	1.33	1.14
in-2004-nat-hc	1.56	1.22
bn-jung-hc	2.07	1.97
uk-2002-hc	1.37	1.26
uk-2002-nat-hc	1.52	1.31
tw-2010-hc	12.10	11.42
tw-2010-nat-hc	13.63	12.67
pp-miner-hc	3.48	3.42
sk-2005-hc	1.26	1.06
sk-2005-nat-hc	1.97	1.49

Zuckerli: improved reference selection

To choose a reference for v (inspired by shingle ordering[3]):

- Compute, for each $u < v$, $|N(u) \cap N(v)|$
- Pick the top W nodes and estimate cost.

Faster heuristic:

- For each destination node, keep a list of the last 2 nodes that had an incoming edge towards it.
- $|N(u) \cap N(v)| \approx$ number of occurrences of u in $N(v)$'s lists.

[3] Flavio Chierichetti et al. "On compressing social networks". In: *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2009, pp. 219–228.

name	Zuckerli	tree	tree+ref
cnr-2000-hc	1.89	1.57	1.53
cnr-2000-nat-hc	2.07	1.58	1.47
in-2004-hc	1.33	1.14	1.14
in-2004-nat-hc	1.56	1.22	1.12
bn-jung-hc	2.07	1.97	1.92
uk-2002-hc	1.37	1.26	1.22
uk-2002-nat-hc	1.52	1.31	1.21
tw-2010-hc	12.10	11.42	11.45
tw-2010-nat-hc	13.63	12.67	12.61
pp-miner-hc	3.48	3.42	3.55
sk-2005-hc	1.26	1.06	0.99
sk-2005-nat-hc	1.97	1.49	1.10

Theoretical results

Compression of graphs of bounded arboricity

An unlabeled graph of arboricity a can be compressed in polynomial time[4] using at most

$$(a-1)n \log n + O(n)$$

bits.

We can use $2n$ bits to store the first forest, and $n \log n$ bits for every other forest.

[4] Harold N Gabow and Herbert H Westermann. "Forests, frames, and games: algorithms for matroid sums and applications". In: *Algorithmica* 7.1-6 (1992), p. 465.

Uniform attachment graphs

A $UA(n, d)$ graph[5] is generated by adding a degree-d node to the graph until it has n nodes. Neighbours are picked u.a.r.

A $UA(n, d)$ unlabeled graph has an entropy of at least

$$(d-1)n \log n + O(n)$$

$UA(n, d)$ has arboricity at most d, so it can be compressed optimally in polynomial time.

[5] Albert-László Barabási and Réka Albert. "Emergence of scaling in random networks". In: *science* 286.5439 (1999), pp. 509–512.

Preferential attachment graphs (Barabási-Albert)

Same as UA(n, d), but neighbours are picked with probability proportional to their degrees[6].

A $BA(n, d)$ unlabeled graph has an entropy of at least

$$(d-1)n \log n + O(n \log \log n)$$

$BA(n, d)$ has arboricity at most d, so it can be compressed optimally in polynomial time. This generalizes a known result[7] for $d > 2$.

[6] Barabási and Albert, "Emergence of scaling in random networks".

[7] Tomasz Łuczak, Abram Magner, and Wojciech Szpankowski. "Asymmetry and structural information in preferential attachment graphs". In: *Random Structures & Algorithms* 55.3 (2019), pp. 696–718.

Conclusions, future work and changes for reviewer comments

Conclusions

- New compression techniques
- New compression scheme for graphs that is $\sim 25\%$ denser for web graphs
- New high-density scheme that is even denser ($\sim 20\%$)
- Optimal scheme for Barabási-Albert random graphs

Future work

- New algorithms for tree construction
- Faster decompression with tree-based context modeling
- Tree-based context modeling for i.e. text
- Graph permutation heuristics for Zuckerli
- Extension of Zuckerli to weighted graphs

Changes due to reviewers' comments

- Added graphs to the evaluation
- Added more details on Hybrid Integer Encoding implementation
- Added a comparison of Hybrid Integer Encoding to a more modern technique
- Added a comparison of Context Trees with Context Tree Weighting

Publications during the PhD

Related publications

Zuckerli: A New Compressed Representation for Graphs.

L. Versari, IM. Comsa, A. Conte, R. Grossi, IEEE Access, 2020

ISO/IEC DIS 18181-1: Information technology - JPEG XL Image Coding System - Part 1: Core coding system

JPEG XL next-generation image compression architecture and coding tools. J. Alakuijala, R. van Asseldonk, S. Boukortt, M. Bruse, IM. Comșa, M. Firsching, T. Fischbacher, E. Kliuchnikov, S. Gomez, R. Obryk, K. Potempa, A. Rhatushnyak, J. Sneyers, Z. Szabadka, L. Vandevenne, L. Versari, J. Wassenberg - Applications of Digital Image Processing XLII, 2019

Other publications - 2020

Sublinear-Space and Bounded-Delay Algorithms for Maximal Clique Enumeration in Graphs A. Conte, R. Grossi, A. Marino, L. Versari - Algorithmica, 2020

Benchmarking JPEG XL image compression. J. Alakuijala, S. Boukortt, T. Ebrahimi, E. Kliuchnikov, J. Sneyers, E. Upenik, L. Vandevenne, L. Versari, J. Wassenberg - Optics, Photonics and Digital Technologies for Imaging Applications VI, 2020

Temporal coding in spiking neural networks with alpha synaptic function. IM. Comsa, T. Fischbacher, K. Potempa, A. Gesmundo, L. Versari, J. Alakuijala - IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2020

Truly Scalable K-Truss and Max-Truss Algorithms for Community Detection in Graphs. A. Conte, D. De Sensi, R. Grossi, A. Marino, L. Versari - IEEE Access, 2020

Other publications - 2019

Listing maximal subgraphs satisfying strongly accessible properties.

A. Conte, R. Grossi, A. Marino, L. Versari. SIAM Journal on Discrete Mathematics 33, 2019

A fast discovery algorithm for large common connected induced subgraphs. A. Conte, R. Grossi, A. Marino, L. Tattini, L. Versari - Discrete Applied Mathematics 268, 2019

Proximity Search for Maximal Subgraph Enumeration A. Conte, A. Marino, R. Grossi, T. Uno, L. Versari - arXiv preprint arXiv:1912.13446, 2019

Other publications - 2018 (1)

Efficient algorithms for listing k disjoint st-paths in graphs. R. Grossi, A. Marino, L. Versari - Latin American Symposium on Theoretical Informatics, 2018

Tight Lower Bounds for the Number of Inclusion-Minimal st-Cuts. A. Conte, R. Grossi, A. Marino, R. Rizzi, T. Uno, L. Versari - International Workshop on Graph-Theoretic Concepts in Computer Science, 2018

Finding maximal common subgraphs via time-space efficient reverse search. A. Conte, R. Grossi, A. Marino, L. Versari - International Computing and Combinatorics Conference, 2018

Other publications - 2018 (2)

D2K: scalable community detection in massive networks via small-diameter k-plexes. A. Conte, T. De Matteis, D. De Sensi, R. Grossi, A. Marino, L. Versari - Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018

Listing subgraphs by Cartesian decomposition. A. Conte, R. Grossi, A. Marino, R. Rizzi, L. Versari - International Symposium on Mathematical Foundations of Computer Science, 2018

Discovering k-Trusses in Large-Scale Networks. A. Conte, D. De Sensi, R. Grossi, A. Marino, L. Versari. IEEE High Performance extreme Computing Conference (HPEC), 2018

Q & A