6/15/2026 CPM2026

LZBE: an LZ-style compressor supporting $O(\log n)$-time random access

Hiroki Shibata (Kyushu University)

Yuto Nakashima (Kyushu University)

Yutaro Yamaguchi (The University of Osaka)

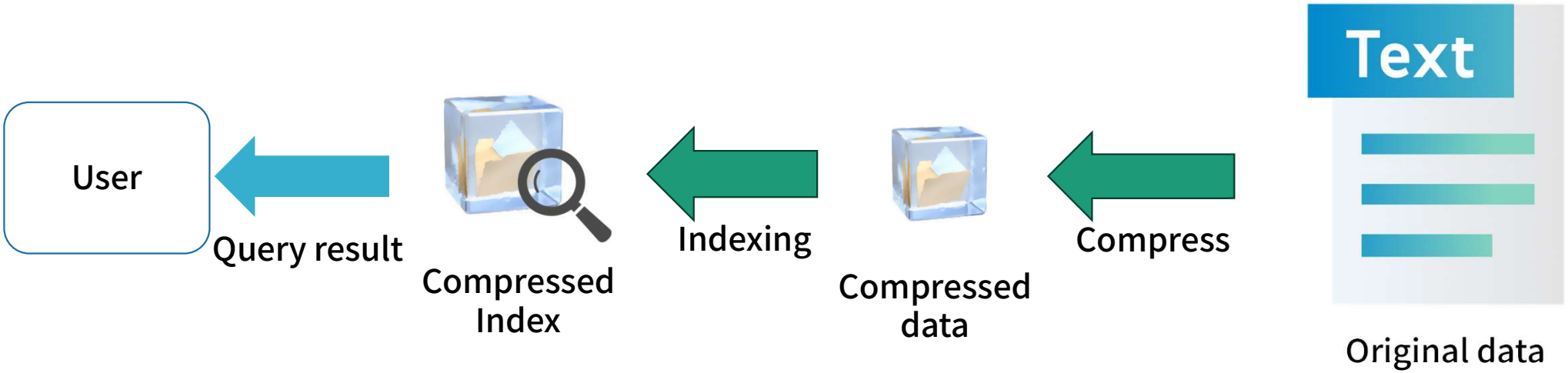
Shunsuke Inenaga (Kyushu University)

slides

Background: Compressed Data Processing

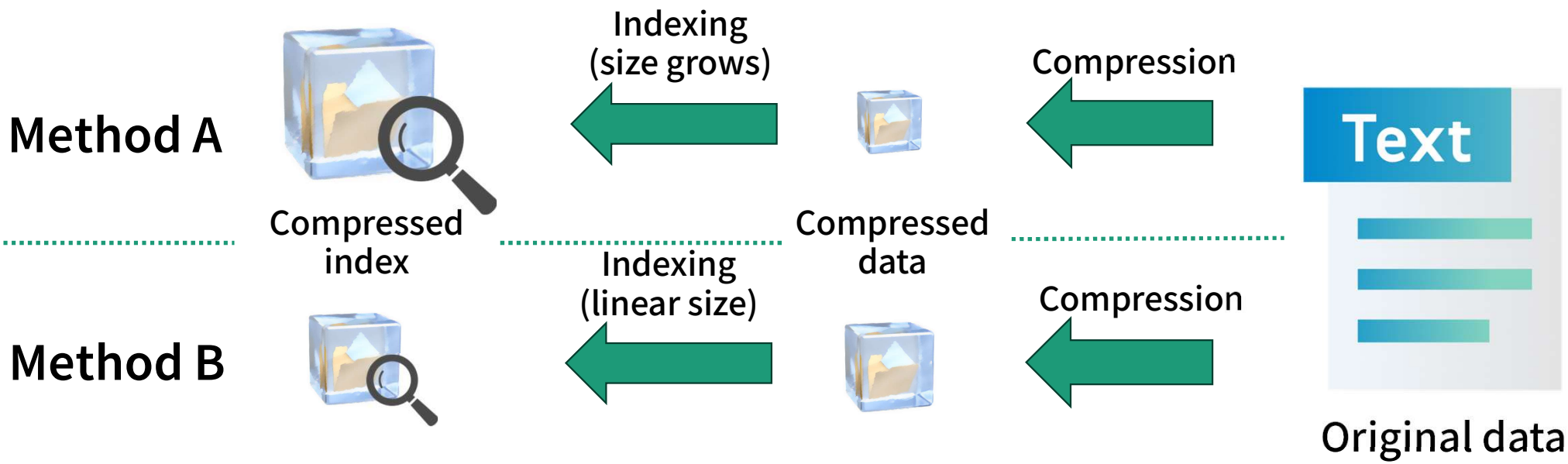
Two methods for compressed data processing (e.g. random access, pattern matching):

- Naive: Decompress all data for each query and process it using standard algorithms.
- Faster: Make a compressed index in advance and process queries using it.



Background: Compression/Query Trade-off

- **Compression performance** and **query capability** have a trade-off.
- Highly compressive schemes tend to resist linear-size indexing.



LZ Compression and and Grammar Compression

We focus on the relationship between LZ compression and grammar compression:

- Compression Performance: LZ compression is better (both in practical and in theoretical)
- Compressed Indexing:
  - Grammar Compression supports linear-space indexing and $O(\log n)$-time random access.
  - For LZ, no linear-space indexing have been proposed

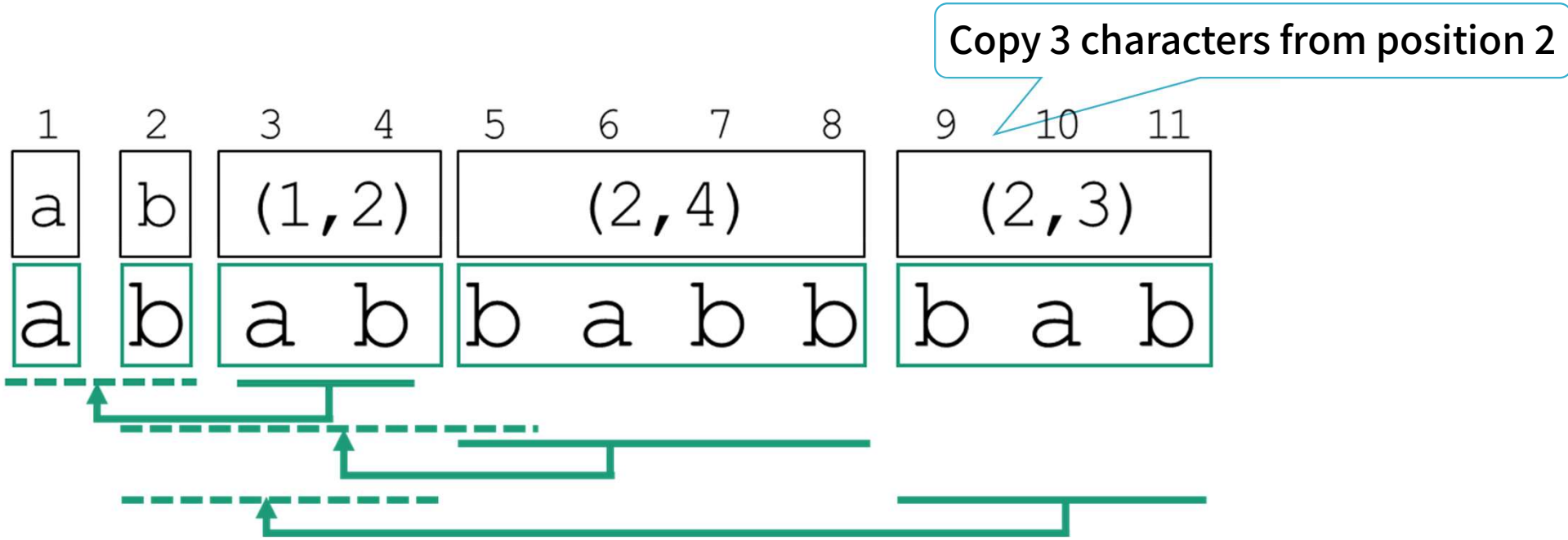
→ Goal: Combine LZ-style compression with grammar-like indexing.

n : input string length

Lempel-Ziv (LZ) Compression [Ziv and Lempel, 1977]

Represents data via a sequence of factors (LZ factors):

1. **A single literal character**
2. **A copy of a previous substring**



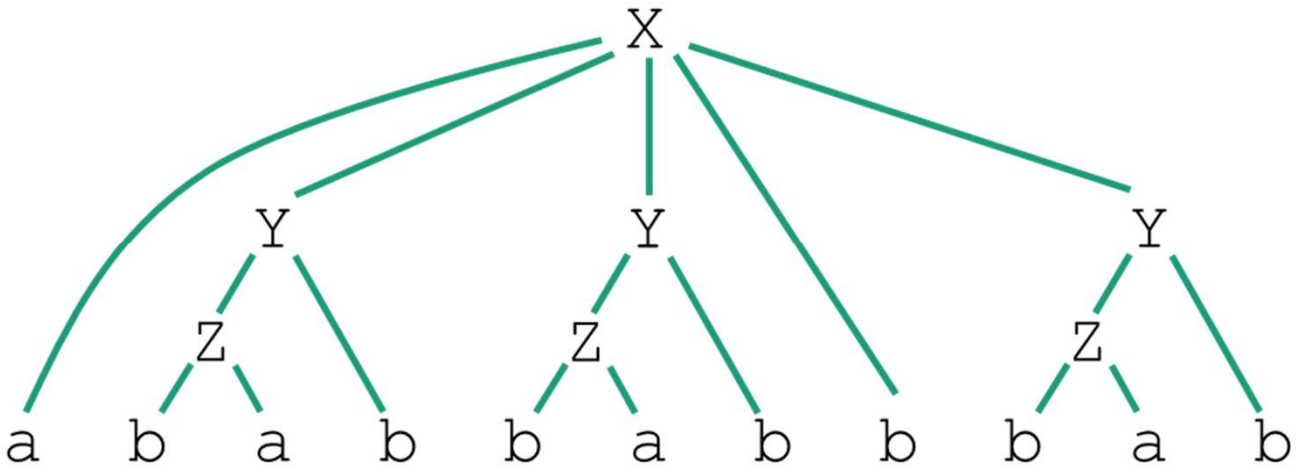
Grammar Compression [Kieffer and Yang, 2000]

Represents data using a Context-Free Grammar (CFG) with:

1. **A set of terminals Σ, and a set of non-terminals $\mathcal{X}$**
2. **Production rules $R = \{X_i \rightarrow expr_i \mid X_i \in \mathcal{X}\}$**
3. **Start symbol $S \in \mathcal{X}$**

$\Sigma : \{a, b\}$
$\mathcal{X} : \{X, Y, Z\}$
$S : X$

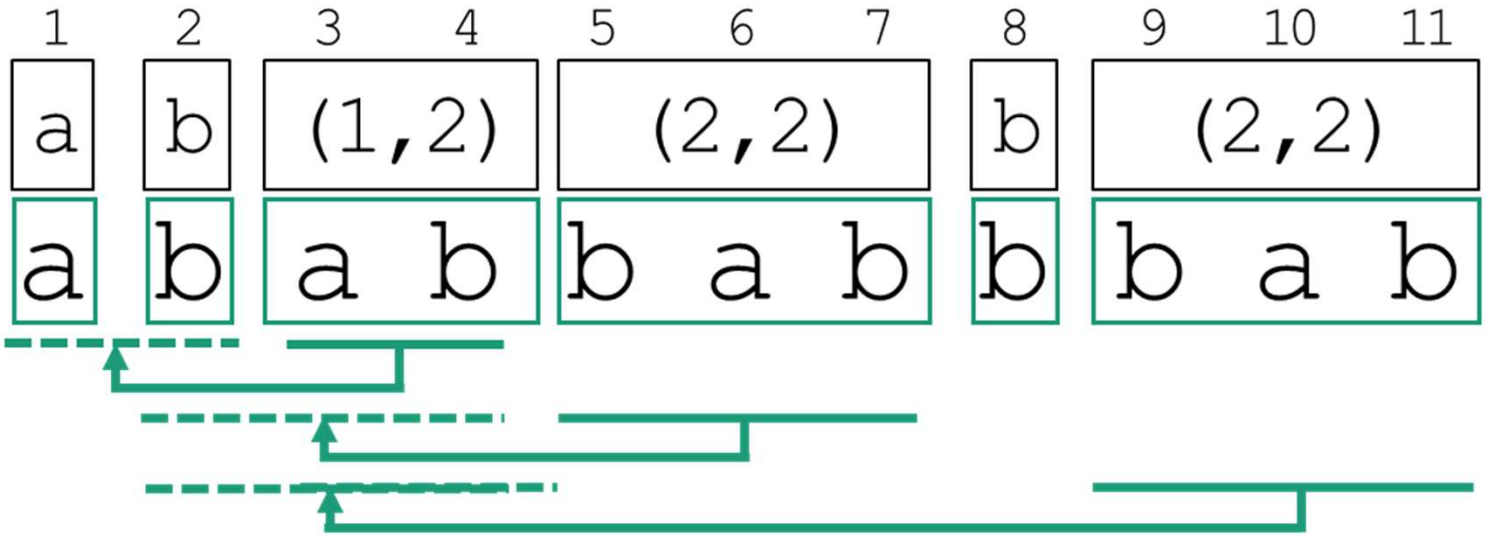
R
$X \rightarrow aYYbY$
$Y \rightarrow Zb$
$Z \rightarrow ba$



LZBE Compression

LZBE: A restricted LZ factorization where each copy factor refers to consecutive previous factors.

- **Both the beginning and end of the factor source are aligned with factor boundaries**



Our Results & Focus of This Talk

1. **Relationship between other compression methods**
2. **$O(\log n)$-time random access index for LZBE**
3. **Linear-time computation of greedy LZBE**
4. **A lower bound between greedy/optimal LZBE**

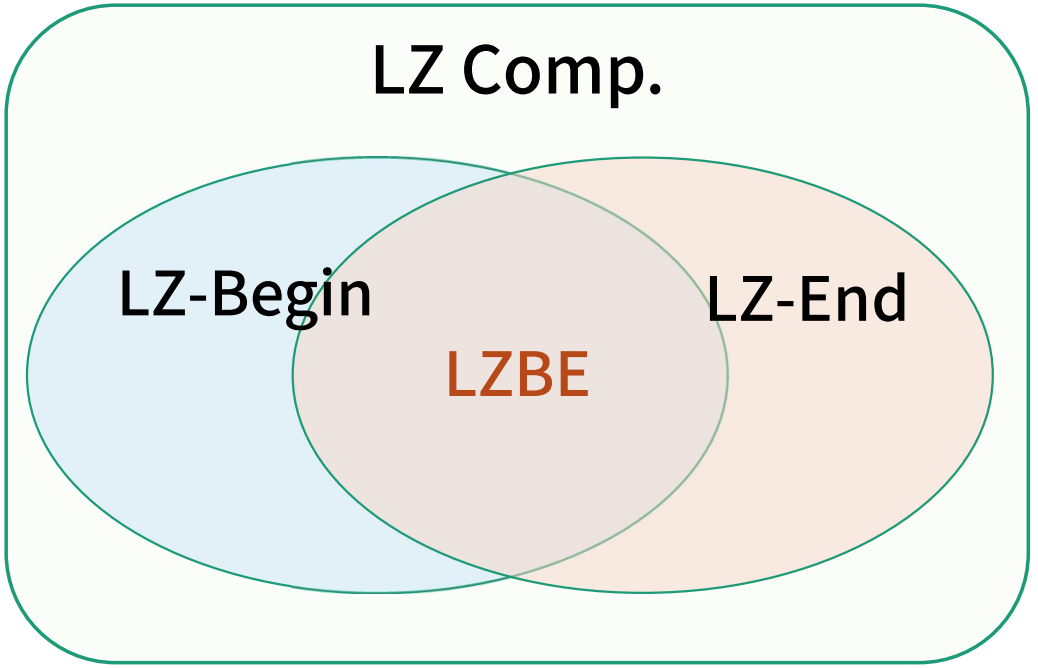
etc.

n : input string length

Relationships Between LZBE and Other LZ Variants

- Two restricted LZ variants: LZ-Begin and LZ-End [Kreft and Navarro, 2013]
- LZBE is a more restricted variant and supports $O(\log n)$ time random access using a linear-space index.

Method	Source Beginning Constraint	Source Ending Constraint	Linear-space Random Access
LZ Comp.			Difficult
LZ-Begin	✓		Not known
LZ-End		✓	$O(\log^3 n \log n)$
LZBE	✓	✓	$O(\log n)$

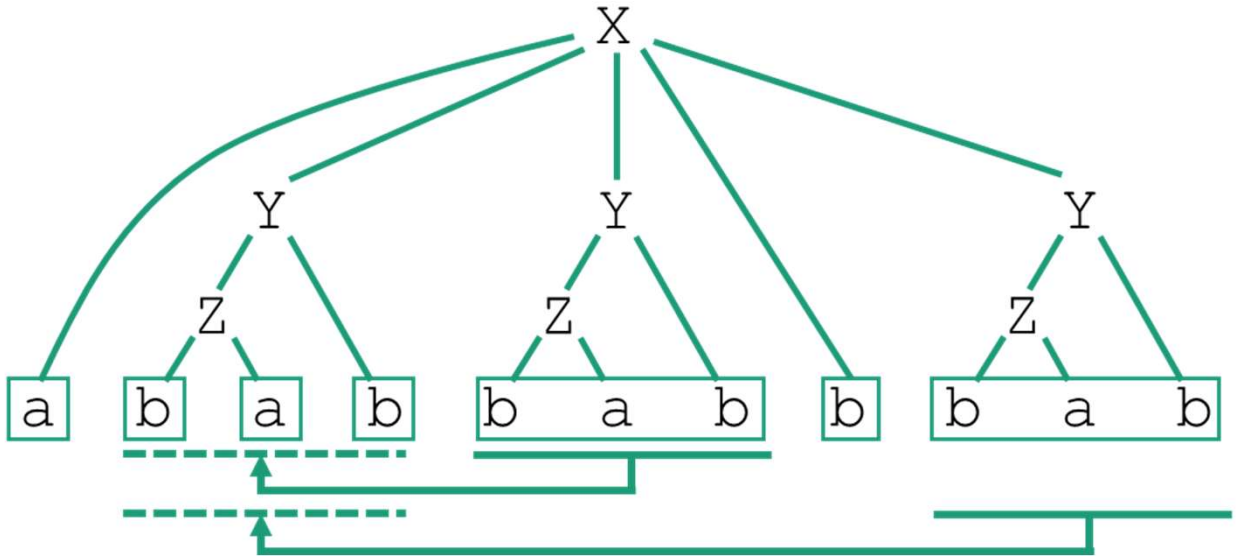


Size relationship: LZBE ≤ grammar

Method: Grammar Decomposition [Rytter, 2003]

Replace non-leftmost occurrences of a nonterminal symbol with a phrase pointing to the interval corresponding to the leftmost occurrence.

- **The resulting LZ size ≤ original grammar size.**
- **The resulting LZ factorization is an LZBE factorization.**

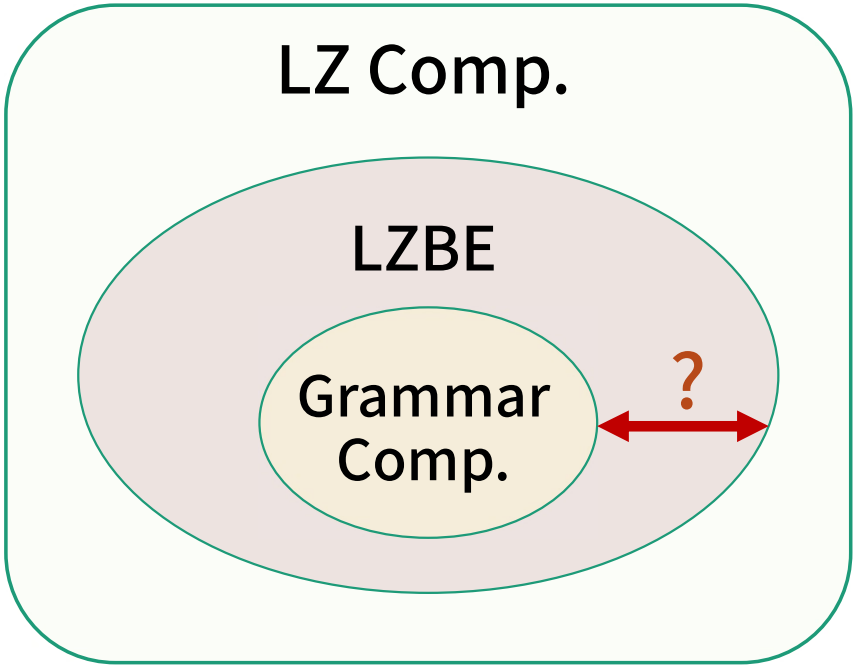


Finding a separation between LZBE and Grammar

We showed that LZBE size $\leq$ grammar size.

Question: Can LZBE be asymptotically smaller than grammar?

- We want to know whether such a family of strings exists.



A separation between LZBE and grammar

Theorem

For an arbitrarily large m, there exists a string T of length $\Theta(m^2)$ where:

- The size of greedy LZBE of T : $\Theta(m)$
- The size of the smallest grammar of T : $\Omega\big(m\alpha(m)\big)$

This theorem implies that LZBE is asymptotically more powerful than grammar compression.

- For any string: $|\text{the smallest LZBE}| \leq |\text{the smallest grammar}|$
- For a specific string: $|\text{the greedy LZBE}| \ll |\text{the smallest grammar}|$

$\alpha(x)$: the inverse Ackermann function

Proof outline: reduction from ORPP

Off-line Range Product Problem

- **Input**: a sequence $x_1, \ldots, x_m \in X^m$, ranges $[l_1, r_1], \ldots, [l_m, r_m]$
- **Output**: $q_i = \bigotimes_{j=l_i}^{r_i} x_j \ \ (1 \leq i \leq m)$

($\otimes$: a semigroup operation on X)

Known lower bound: **Requires $\Omega(m\alpha(m))$ semigroup operations for some inputs.** [Chazelle and Rosenberg, 1991]

Constructing a string from ORPP input

For an ORPP instance, $x_1, \ldots, x_m \in X^m, [l_1, r_1], \ldots, [l_m, r_m]$**, construct a string** T **as follows:**

$$T = x_1 \cdots x_m \$_1 Q_1 \$_2 Q_2 \$_3 \cdots \$_m Q_m \$_{m+1}$$

$$Q_i = x_{l_i} \cdots x_{r_i} \ (1 \leq i \leq m)$$

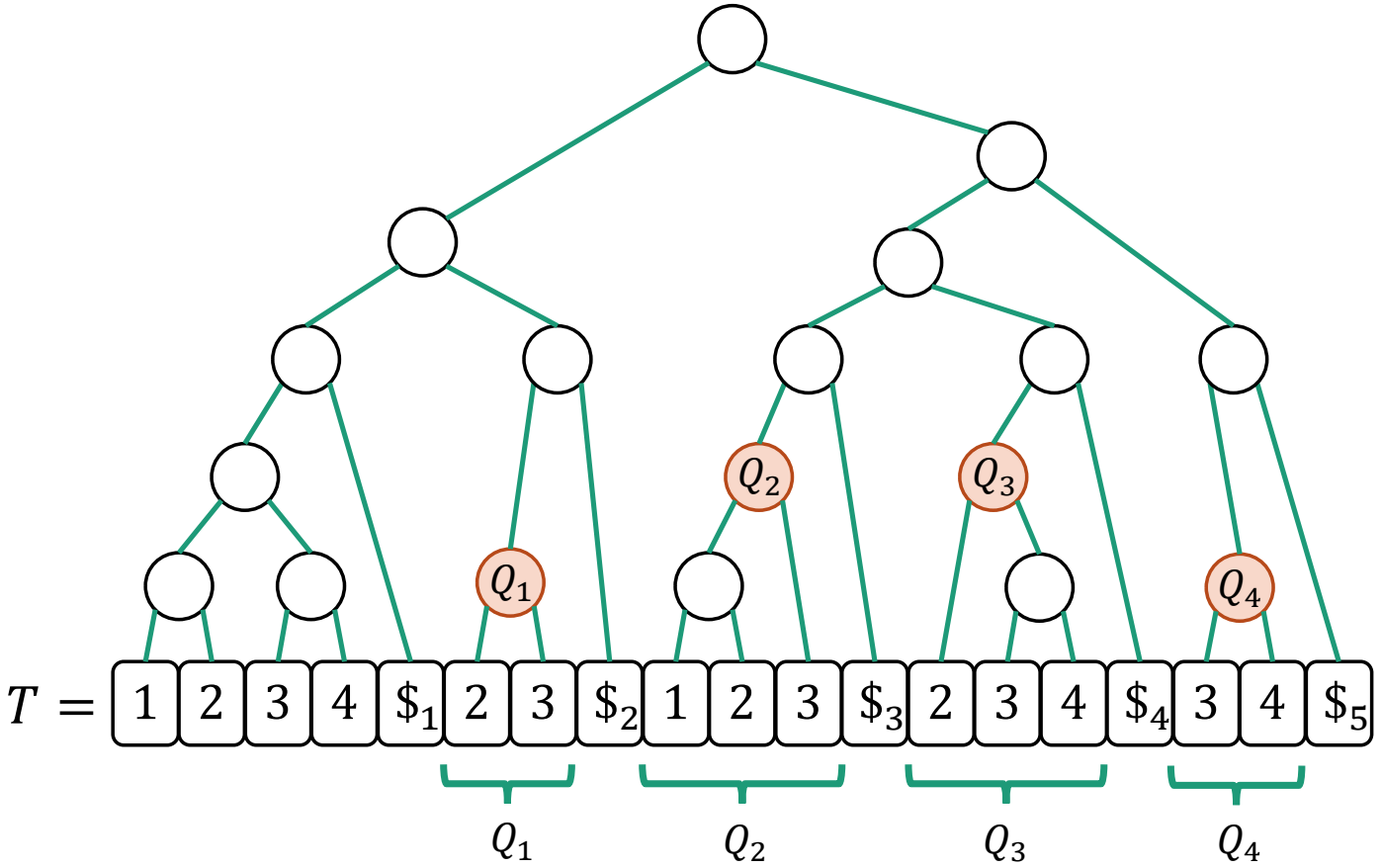
$\$_i$：区切り文字

Example: sequence is $(1,2,3,4)$ **and queries are** $[2,3], [1,3], [2,4], [3,4]$**:**

$$T = 1234 \ \$_1 \underbrace{23}_{Q_1} \ \$_2 \underbrace{123}_{Q_2} \ \$_3 \underbrace{234}_{Q_3} \ \$_4 \underbrace{34}_{Q_4} \ \$_5$$

Computing ORPP Answers via Grammar

- Compute semigroup values for all nonterminals bottom-up.
- Answer all queries using the corresponding nonterminals.



Smallest Grammar Size vs. LZBE Size

- **Grammar Size: An ORPP instance can be solved using $O(g)$ semigroup operations when the grammar size is g.**

  However, the problem has a lower bound $\Omega(m\alpha(m))$.

  → There exist inputs whose smallest grammar size is $\Omega(m\alpha(m))$.

- **LZBE size: The greedy factorization has size $O(m)$.**

$$T = \boxed{x_1} \cdots \boxed{x_m \$_1 Q_1 \$_2 Q_2 \$_3} \cdots \boxed{\$_m Q_m \$_{m+1}}$$

→ The sizes of the smallest grammar and the greedy LZBE factorization differ asymptotically.

Linear-space index for LZBE random access

- Similar to an existing random access index for grammar compression [Bille et al., 2015]
  - Combining heavy path decomposition and interval biased search trees
- A simple adaptation is impossible, because an LZBE factor may refer to many consecutive factors.
- We maintain a global IBST representing all factors, and precompute constant number of pointers for each source interval.

Conclusion

We proposed LZBE compression:

- Restricted LZ compression
- Supporting $O(\log n)$ time random access
- Better compression performance than grammar compression in theory