DCC IEEE

Presented at the 2020 IEEE Data Compression Conference
Snowbird, Utah, USA — March 24–27

Entropy Coders Based on the Splitting of Lexicographic Intervals

Danny Dubé
Danny.Dube@ift.ulaval.ca
/
Université Laval
Canada

Abstract

We propose a technique that performs entropy coding by splitting lexicographic intervals. We mention the main characteristics of our technique, where most of the characteristics definitely apply, by design, and the others are expected to apply, after empirical or theoretical demonstrations are provided. Our technique is (or, at least, should be):

- based on automata quite similar to Mealy machines;
- fast in encoding and decoding;
- able to achieve arbitrarily low redundancy;
- designed to require a small number of states;
- able to decode forwards (making it suitable for streaming);
- able to handle skewed probability distributions;
- intended for stationary memoryless sources;
- a kind of variable-to-fixed coding; and
- able to handle finite source alphabets of arbitrary sizes.

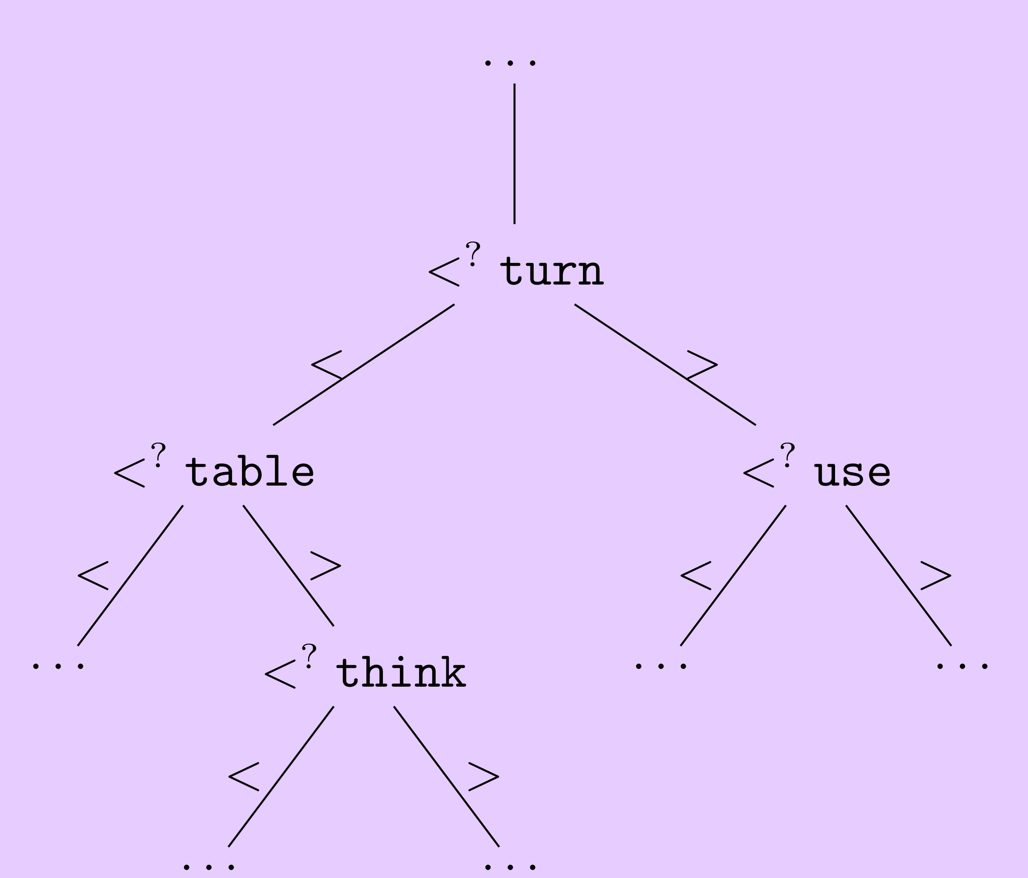
Analogy to a Word-Guessing Game

Inspiration for the coding technique:

> "*Is the secret word lexicographically smaller than w?*"

Translation into string-processing terms:

> "*Is the (infinite) input string lexicographically smaller than w?*"



Technical Tools

Input alphabet:

$$\Sigma \triangleq \{\mathtt{a}, \mathtt{b}, \ldots, \mathtt{z}\}$$

Output alphabet:

$$\mathbf{2} \triangleq \{\mathtt{0}, \mathtt{1}\}$$

Lexicographic bounds:

$$\mathcal{B} \triangleq \{\epsilon\} \cup \Sigma^* \cdot (\Sigma - \{\mathtt{a}\}) \cup \{\infty\}$$

Plain and split lexicographic intervals:

$$\mathcal{I} \triangleq \{[r,\, t] \mid r, t \in \mathcal{B} \text{ and } r < t\}$$
$$\widehat{\mathcal{I}} \triangleq \{[r \,\langle s\rangle\, t] \mid r, s, t \in \mathcal{B} \text{ and } r < s < t\}$$

Contents of intervals, by extension:

$$\mathrm{X}([r,\, t]) = \mathrm{X}([r\,\langle s\rangle\, t]) \triangleq \{\omega \in \Sigma^\infty \mid r < \omega < t\}$$

Conversion to plain intervals:

$$\mathrm{U}([r\,\langle s\rangle\, t]) \triangleq [r,\, t]$$

Conversion to split intervals:

$$\mathrm{S}_{\widehat{\mathcal{F}}}([r,\, t]) \triangleq \left\{[r\,\langle s\rangle\, t] \in \widehat{\mathcal{F}}\right\}$$

Trimming of lexicographic intervals:

$$\begin{array}{lcll}
\mathrm{T}([\epsilon, \mathtt{a}\cdot w]) & = & \mathrm{T}([\epsilon, w]) & \\
\mathrm{T}([\epsilon, \mathtt{b}]) & = & [\epsilon, \infty] & \\
\mathrm{T}([\epsilon, d\cdot w]) & = & [\epsilon, d\cdot w], & \text{if } (d = \mathtt{b} \text{ and } w \neq \epsilon) \text{ or } d > \mathtt{b} \\
\mathrm{T}([\epsilon, \infty]) & = & [\epsilon, \infty] & \\
\mathrm{T}([c\cdot v, c\cdot w]) & = & \mathrm{T}([v, w]) & \\
\mathrm{T}([c\cdot v, d]) & = & \mathrm{T}([v, \infty]), & \text{if } \mathrm{succ}(c) = d \\
\mathrm{T}([c\cdot v, d\cdot w]) & = & [c\cdot v, d\cdot w], & \text{if } (\mathrm{succ}(c) = d \text{ and } w \neq \epsilon) \text{ or } \mathrm{succ}(c) < d \\
\mathrm{T}([c\cdot v, \infty]) & = & [c\cdot v, \infty], & \text{if } c < \mathtt{z} \\
\mathrm{T}([\mathtt{z}\cdot v, \infty]) & = & \mathrm{T}([v, \infty]) &
\end{array}$$

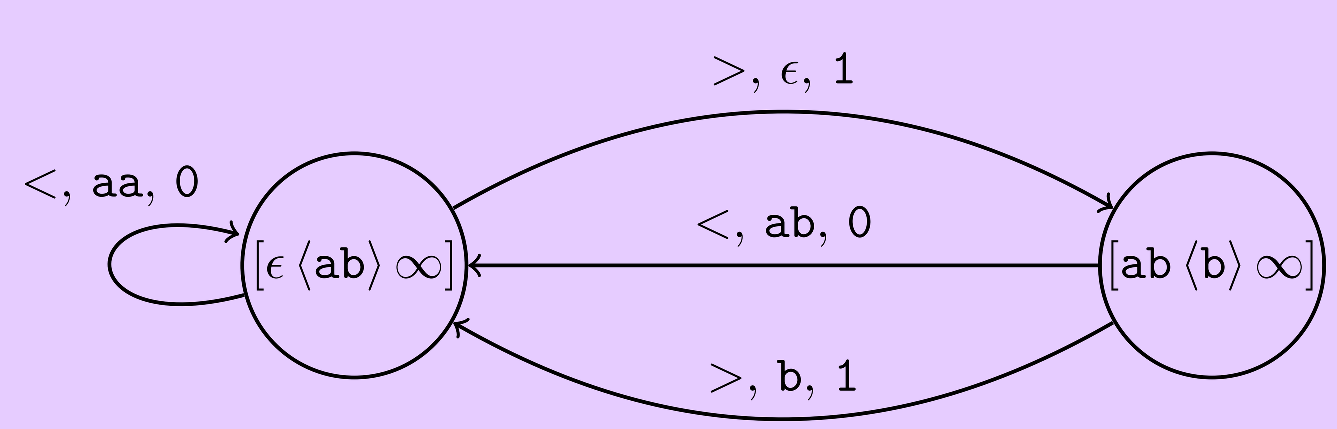
Effect of the trimming operation:

$$\mathrm{X}([r,\, t]) = \{w\} \cdot \mathrm{X}(\mathrm{T}([r,\, t])), \qquad \text{where } w \text{ is the LCP of } \mathrm{X}([r,\, t])$$

Coding rate of an automaton, in source symbols per output bit:

$$R(\widehat{\mathcal{Q}}) = \sum_{[r\,\langle s\rangle\, t] \in \widehat{\mathcal{Q}}} \begin{pmatrix} |\mathrm{LCP}(\mathrm{X}([r,\, s]))| \cdot \Pr(r < \Omega < s \mid r < \Omega < t) \\ +|\mathrm{LCP}(\mathrm{X}([s,\, t]))| \cdot \Pr(s < \Omega < t \mid r < \Omega < t) \end{pmatrix} \cdot \hat{p}([r\,\langle s\rangle\, t])$$

Example of an Automaton



Automaton: $\widehat{\mathcal{Q}} = \{[\epsilon\,\langle\mathtt{ab}\rangle\,\infty],\, [\mathtt{ab}\,\langle\mathtt{b}\rangle\,\infty]\}$
Probs.: $p(\mathtt{a}) = 0.7$ and $p(\mathtt{b}) = 0.3$
Coding rate: $R(\widehat{\mathcal{Q}}) = 1.126$ sym./bit

Definition of Automaton

An automaton is defined as a set of split intervals $\widehat{\mathcal{Q}} \subset \widehat{\mathcal{I}}$ with the following properties. Let $\mathcal{Q} = \{\mathrm{U}(I) \mid I \in \widehat{\mathcal{Q}}\}$.

Finiteness Set $\widehat{\mathcal{Q}}$ is finite.

Existence of a start state There exists a start state; i.e. $[\epsilon,\, \infty] \in \mathcal{Q}$.

Closure Transitions always lead to other states in $\widehat{\mathcal{Q}}$; i.e. for any $[r\,\langle s\rangle\, t] \in \widehat{\mathcal{Q}}$, we have both $\mathrm{T}([r,\, s]) \in \mathcal{Q}$ and $\mathrm{T}([s,\, t]) \in \mathcal{Q}$.

Determinism Given a specific knowledge about the input string, the automaton systematically decides to apply the same test on the input string; i.e. for any $[r,\, t] \in \mathcal{Q}$, there exists $s \in \mathcal{B}$ such that $\mathrm{S}_{\widehat{\mathcal{Q}}}([r,\, t]) = \{[r\,\langle s\rangle\, t]\}$.

References

[1] Jarek Duda. Asymmetric numeral systems: entropy coding combining speed of Huffman coding with compression rate of arithmetic coding, 2014. arXiv:1311.2540v2.

[2] Ryusei Fujita, Ken-ichi Iwata, and Hirosuke Yamamoto. An iterative algorithm to optimize the average performance of Markov chains with finite states. In *Proceedings of the IEEE International Symposium on Information Theory*, pages 1902–1906, Paris, France, July 2019.

[3] Michael Holcombe. *Algebraic Automata Theory.* Cambridge Studies in Advanced Mathematics. Cambridge University Press, 1982.

[4] D. A. Huffman. A method for the construction of minimum-redundancy codes. In *Proceedings of the Institute of Radio Engineers*, volume 40, pages 1098–1101, sep 1952.

[5] B. P. Tunstall. *Synthesis of Noiseless Compression Codes.* PhD thesis, Georgia Institute of Technology, 1967.

[6] J. S. Vitter. Design and analysis of dynamic Huffman codes. *Journal of the ACM*, 34(4):825–845, October 1987.

[7] Ian H. Witten, Radford M. Neal, and John G. Cleary. Arithmetic coding for data compression. *Communications of the ACM*, 30(6):520–540, 1987.