Data Compression Through Empirical Approximations of Kolmogorov Complexity

Bidipta Sarkari1,2, Sean Brandenburg2, Ari Trachtenberg2
Cupertino High School, 10100 Finch Ave. Cupertino, CA 95014;
Boston University Department of Electrical and Computer Engineering, 8 St. Mary’s St, Boston, MA 022152

Introduction

Kolmogorov Complexity: The shortest program that outputs the string.

File with integers from 1 to 1000: (2,893 bytes)
- C Kolmogorov Complexity: 35 bytes
- gzip: 1,850 bytes

This project studies Kolmogorov Complexity approximations as a form of data compression.

Empirical Results

<table>
<thead>
<tr>
<th></th>
<th>SUBLEQ, M</th>
<th>SUBLEQ, ALT</th>
<th>RSSB</th>
<th>MicroScript</th>
<th>FourV1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zip files</td>
<td>100.03%</td>
<td>99.10%</td>
<td>100.03%</td>
<td>99.14%</td>
<td>100.23%</td>
</tr>
<tr>
<td>Oblique</td>
<td>100.02%</td>
<td>0.74%</td>
<td>7.58%</td>
<td>0.10%</td>
<td>6.04%</td>
</tr>
<tr>
<td>Testing</td>
<td>100.02%</td>
<td>8.31%</td>
<td>10.02%</td>
<td>1.96%</td>
<td>10.02%</td>
</tr>
<tr>
<td>Increasing</td>
<td>100.02%</td>
<td>100.02%</td>
<td>100.02%</td>
<td>31.46%</td>
<td>100.02%</td>
</tr>
<tr>
<td>Calgary Corpus</td>
<td>100.01%</td>
<td>95.03%</td>
<td>98.49%</td>
<td>94.55%</td>
<td>98.63%</td>
</tr>
</tbody>
</table>

Compression ratio: compressed file size / plaintext size. Generated using SCC with machine inputs of size less than 5 bytes.

Short FourV1 Programs
- Print from j to j inclusive with skips of -j to 11 bytes
- Print bytes from j to j inclusive: 8 bytes
- Print byte c repeated j times: 10 bytes
- Fibonacci sequence mod 64F until j is chosen: 16 bytes

SUBLEQ and SUBLEQ, ALT

Universal Input: Any byte sequence is a program

Studied Groups:
- One Instruction Set Computers (OISC):
  - Turing-complete systems
  - Bytes are inputs to the only instruction.
- Code Golf Languages:
  - Languages for small source codes.

Procedure

Set up a database to map the machine’s outputs to inputs

The database can compress strings that the machine can decompress.

To Compress binary file f

Find most compressible substring s
Compress s into s’ and add delimiter
Loop while a compressible substring exists

To Decompress binary file f’

Find each s’ with delimiter
Decompress s’ to s with machine m

Machine Design

Criteria:
- Turing-Complete: Can simulate any Turing Machine

Universal Input: Any byte sequence is a program

Code Golf Languages:
- Languages for small source codes.

Analysis

The empirical results and theories highlight potential for data compression.

- Zip Files
  - SUBLEQ, ALT compressed by 1%
  - Could improve with larger databases
  - Not generally possible to compress zip files with statistical techniques.

- Sequence Compression
  - MicroScript compressed simple patterns very well.
  - Simple programs can express these long sequences.

- Asymptotic Behaviour
  - Most sequences must be Kolmogorov Random (thm 3)
  - With a large database, expected compression will be expected to be as good as gzip (thm 4)

- Machine Design
  - OISC designs like RSSB and SUBLEQ, M require long programs for simple sequences
  - More instructions could shorten programs

Theorems

Theorems adapted from Elements of Information Theory[2]

Although exact Kolmogorov Complexity depends on the computing model, there is a universal notion of complexity.

Theorem 1: If U is a universal computer, then for any computer A, KU(x) ≤ K(x) + cA

Where constant cA does not depend on x.

There are also bounds on the Kolmogorov Complexity of random strings because they can simply be “printed out” directly by the computing model.

Theorem 2: 1/2 ≤ K(x) ≤ 1/2 + log 1/2(a) + c

Asymptotically, expected Kolmogorov Complexity approaches entropy, the expected best statistical behaviour of total compression.

Theorem 4: Let stochastic process X be defined i.i.d. From a probability mass function. Then

E 1/2 K(X^n | n) → H(X)

Future Work

- Study alternate machines
- Machine Specialization
- Different database generation heuristics
- Hybrid compression algorithms with Kolmogorov compression and statistical methods.

References


Acknowledgements

I would like to thank my mentor, Professor Ari Trachtenberg, for his preliminary research on this topic and the support throughout the process. I would also like to thank Akshay Bhardwaj and the rest of SCC for helping me set up the project, reviewing my code, and stacking expertise throughout the summer. Finally, I would like to thank BU for providing me with resources for accessing research papers and for access to the SCC.