Practical Implementations of Arithmetic Coding

Paul G. Howard and *Jeffrey Scott Vitter*

Brown University
Department of Computer Science
Technical Report No. 92–18
Revised version, April 1992
(Formerly Technical Report No. CS–91–45)

Appears in *Image and Text Compression*,
James A. Storer, ed., Kluwer Academic Publishers, Norwell, MA, 1992, pages 85–112.

A shortened version appears in the proceedings of the
International Conference on Advances in Communication and Control (COMCON 3),
Victoria, British Columbia, Canada, October 16–18, 1991.

Practical Implementations of Arithmetic Coding[1]

Paul G. Howard[2] *Jeffrey Scott Vitter*[3]
Department of Computer Science
Brown University
Providence, R.I. 02912–1910

Abstract

We provide a tutorial on arithmetic coding, showing how it provides nearly optimal data compression and how it can be matched with almost any probabilistic model. We indicate the main disadvantage of arithmetic coding, its slowness, and give the basis of a fast, space-efficient, approximate arithmetic coder with only minimal loss of compression efficiency. Our coder is based on the replacement of arithmetic by table lookups coupled with a new deterministic probability estimation scheme.

Index terms: Data compression, arithmetic coding, adaptive modeling, analysis of algorithms, data structures, low precision arithmetic.

---

[1]A similar version of this paper appears in *Image and Text Compression*, James A. Storer, ed., Kluwer Academic Publishers, Norwell, MA, 1992, 85–112. A shortened version of this paper appears in the proceedings of the International Conference on Advances in Communication and Control (COMCON 3), Victoria, British Columbia, Canada, October 16–18, 1991.

[2]Support was provided in part by NASA Graduate Student Researchers Program grant NGT–50420 and by a National Science Foundation Presidential Young Investigators Award grant with matching funds from IBM. Additional support was provided by a Universities Space Research Association/CESDIS associate membership.

[3]Support was provided in part by National Science Foundation Presidential Young Investigator Award CCR–9047466 with matching funds from IBM, by NSF research grant CCR–9007851, by Army Research Office grant DAAL03–91–G–0035, and by the Office of Naval Research and the Defense Advanced Research Projects Agency under contract N00014–91–J–4052 ARPA Order No. 8225. Additional support was provided by a Universities Space Research Association/CESDIS associate membership.

1 Data Compression and Arithmetic Coding

Data can be compressed whenever some data symbols are more likely than others. Shannon [54] showed that for the best possible compression code (in the sense of minimum average code length), the output length contains a contribution of $-\lg p$ bits from the encoding of each symbol whose probability of occurrence is p. If we can provide an accurate model for the probability of occurrence of each possible symbol at every point in a file, we can use arithmetic coding to encode the symbols that actually occur; the number of bits used by arithmetic coding to encode a symbol with probability p is very nearly $-\lg p$, so the encoding is very nearly optimal for the given probability estimates.

In this paper we show by theorems and examples how arithmetic coding achieves its performance. We also point out some of the drawbacks of arithmetic coding in practice, and propose a unified compression system for overcoming them. We begin by attempting to clear up some of the false impressions commonly held about arithmetic coding; it offers some genuine benefits, but it is not the solution to all data compression problems.

The most important advantage of arithmetic coding is its flexibility: it can be used in conjunction with any model that can provide a sequence of event probabilities. This advantage is significant because large compression gains can be obtained only through the use of sophisticated models of the input data. Models used for arithmetic coding may be adaptive, and in fact a number of independent models may be used in succession in coding a single file. This great flexibility results from the sharp separation of the coder from the modeling process [47]. There is a cost associated with this flexibility: the interface between the model and the coder, while simple, places considerable time and space demands on the model's data structures, especially in the case of a multi-symbol input alphabet.

The other important advantage of arithmetic coding is its optimality. Arithmetic coding is optimal in theory and very nearly optimal in practice, in the sense of encoding using minimal average code length. This optimality is often less important than it might seem, since Huffman coding [25] is also very nearly optimal in most cases [8,9, 18,39]. When the probability of some single symbol is close to 1, however, arithmetic coding does give considerably better compression than other methods. The case of highly unbalanced probabilities occurs naturally in bilevel (black and white) image coding, and it can also arise in the decomposition of a multi-symbol alphabet into a sequence of binary choices.

The main disadvantage of arithmetic coding is that it tends to be slow. We shall see that the full precision form of arithmetic coding requires at least one multiplication per event and in some implementations up to two multiplications and two divisions per event. In addition, the model lookup and update operations are slow because of the input requirements of the coder. Both Huffman coding and Ziv-Lempel [59, 60] coding are faster because the model is represented directly in the data structures

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS

Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS

Sync your system to PACER to automate legal marketing.

WHAT WILL YOU BUILD? | **sales@docketalarm.com** | **1-866-77-FASTCASE**

fastcase®
Smarter legal research.