

## Effects of finite-precision arithmetic in enumerative coding

**Citation for published version (APA):**

Schouhamer Immink, K. A., & Janssen, A. J. E. M. (1997). Effects of finite-precision arithmetic in enumerative coding. In *Proceedings of the 1997 IEEE International Symposium on Information Theory, 29 June - 4 July 1997, Ulm, Germany* (pp. 141-141). Institute of Electrical and Electronics Engineers.  
<https://doi.org/10.1109/ISIT.1997.613056>

**DOI:**

[10.1109/ISIT.1997.613056](https://doi.org/10.1109/ISIT.1997.613056)

**Document status and date:**

Published: 01/01/1997

**Document Version:**

Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

**Please check the document version of this publication:**

- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

[Link to publication](#)

**General rights**

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:

[www.tue.nl/taverne](http://www.tue.nl/taverne)

**Take down policy**

If you believe that this document breaches copyright please contact us at:

[openaccess@tue.nl](mailto:openaccess@tue.nl)

providing details and we will investigate your claim.

# Effects of Finite-Precision Arithmetic in Enumerative Coding

Kees A. Schouhamer Immink and Augustus Janssen

Philips Research Laboratories, Prof. Holstlaan 6, 5656 AA Eindhoven, The Netherlands.

**Abstract** — The storage requirements of conventional enumerative schemes can be reduced by using floating point arithmetical operations instead of the conventional fixed point operations. The new enumeration scheme incurs a small coding loss. A simple relationship between storage requirements and coding loss is derived.

## I. INTRODUCTION

Translation using enumeration has the virtue that the complexity (weight storage) grows polynomially with the codeword length contrasting the complexity of direct look-up which grows exponentially. Here we use a floating point instead of a fixed point representation of the weights. The floating-point notation is convenient for representing numbers that differ many orders of magnitude. In this notation, each weight is represented by  $s$  bits, and as a result, the hardware required for storage grows linearly with the codeword length  $n$ . The penalty attached to the finite-precision representation of the weights is that it will entail a (small) loss in code rate. A quantitative trade-off between the precision of the number representation and concomitant code redundancy will be detailed. We will apply the theory to the enumeration of  $(dklr)$  sequences.

## II. ENUMERATION USING FLOATING-POINT ARITHMETIC

The hardware for implementing the enumeration algorithm comprises a (binary) adder, a subtracter, a comparator, and a look-up table of the pre-computed set of weights. The binary fixed-point representation of a single weight requires  $Rn$  bits per weight, where  $R$ ,  $0 < R < 1$ , is a constant. For the overall scheme, we need therefore storage proportional to  $Rn^2$ . We develop an enumeration method where the weights are specified in floating-point notation.

We employ a two-part radix-2 representation

$$I = (m, e)$$

to express the weight

$$I = m \times 2^e,$$

where  $I$ ,  $m$ , and  $e$  are non-negative integers. The two components  $m$  and  $e$  are usually called *mantissa* and *exponent* of the integer  $I$ , respectively. The translation of a weight into  $(m, e)$  is easily accomplished. It is assumed that the exponent of each weight is represented by  $e_p$  bits.

Let  $I$  be the short-hand notation of one of the weights  $N^0(i)$ . It is well known that the non-negative integer  $I$ ,  $I < 2^v$ , can be uniquely represented by a binary  $v$ -tuple  $\mathbf{x} = (x_{v-1}, \dots, x_0)$ , where

$$I = \sum_{i=0}^{v-1} x_i 2^i.$$

The binary  $v$ -tuple  $\mathbf{x}$  is called the binary fixed-point representation of  $I$ . Let

$$u = \lfloor \log_2 I \rfloor$$

be the position of the leading 'one' element of  $\mathbf{x}$ . Then the  $p$ -bit truncated decimal representation of  $I$ , denoted by  $[I]_p$ ,

$$[I]_p = \lfloor 2^{-(u+1-p)} I \rfloor 2^{u+1-p}, \quad (1)$$

can be represented in binary floating-point representation whose mantissa requires at most  $p$  non-zero bits. The enumeration algorithms can be employed directly by using the 'truncated' coefficients. The enumeration algorithm itself remains unchanged. The effect on the set of codewords will be that recursively a number of the highest ranking words of length  $i$  are discarded from the set of all lexicographically ordered sequences. Using finite precision of the weights representation (truncation) will result in coding loss as available words must be discarded. For  $(dk)$  sequences the loss in capacity resulting from the truncation of the weights can be described in terms of an autonomous finite-state machine. From the theory of feedback registers we know that the sequence of the mantissa will ultimately become (and remain) *periodic*.

A more heuristic analysis, where the truncation is modelled as a stochastic process, offers a simple rule of thumb of the capacity loss, namely

$$C(d, k) - \hat{C}(d, k) \approx 2^{-(p+2)}. \quad (2)$$

## III. CONCLUSIONS

We have introduced a scheme of enumerative coding using floating-point arithmetic. This scheme offers, for long codewords, a significant advantage in storage requirements. A simple relationship between coding efficiency versus encoding or decoding hardware (storage) has been derived.