THE HACKENBUSH NUMBER SYSTEM FOR COMPRESSION OF NUMBERICAL DATA

by

E. R. Berlekamp

Memorandum No. ERL-M360

18 September 1972

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

THE HACKENBUSH NUMBER SYSTEM FOR COMPRESSION OF NUMERICAL DATA

E. R. Berlekamp*

ABSTRACT

The Hackenbush number system is a method of representing numbers by expressing the integer part in unary and the fractional part in binary. The representation absorbs the sign bit and binary point into the string of binary digits which represents the number. Conversion between the Hackenbush representation and the more conventional fixed and floating point representations requires approximately the same small amount of computation as is required to convert between the fixed and floating representations. If the numbers to be represented are selected from a known bounded distribution which has tails that fall off exponentially or faster, the m-bit Hackenbush representation is typically more accurate than any floating point representation. For example, if the numbers are selected from the normal Gaussian distribution, the mean-square error of the m-bit Hackenbush representation is less than the mean-square error or any (arbitrarily complex) (m-1)-bit representation, and it is comparable to the mean square quantization error of the (m+n)-bit normalized floating point representation which has 1 bit of sign, n+1 bits of exponent, and (m-2) bits of characteristic.

*Departments of Mathematics and Electrical Engineering and Computer Sciences, and the Electronics Research Laboratory, University of California, Berkeley, California 94720.

Research sponsored by the U.S. Army Research-Office-Durham, Grant DA-ARO-D-31-124-72-G118.

1. INTRODUCTION

There are numerous occasions in which large arrays of numbers must be transmitted between the computer's core memory and some larger and slower storage device such as a disk or tape. For computational ease, each of the numbers usually occupies a full word of core memory, even though the data from which it originated may be of more limited accuracy. When the transfer of the array to the secondary storage device is a time-consuming process, there are strong incentives to pack several array numbers into a single computer word before these words are transferred to secondary storage. This packing achieves a faster transfer time at the expense of additional central processing work (conversion and packing).

The most commonly used compressed number formats are the fixed and floating point representations. For example, the numbers in the original array might be converted to an ad hoc 15-bit floating point representation, using 1 bit of sign, 4 bits of exponent, and 10 bits of characteristic in order to pack four of them into a 60-bit computer word. When the numbers to be compressed have a known distribution, then the fixed and floating point formats are all rather inefficient. For some distributions, there are systems of representing the numbers which allow one to attain more accuracy with 12 bits than any fixed or floating point representation that uses 15 bits.

The theoretical question of how many bits are required to represent numbers from a given distribution with a given accuracy has been studied by many authors. Shannon (1948, 1959) derived the fundamental limits. Elias (1970) introduced a new distortion measure which led to a number of further theoretical results. His paper also includes an extensive review of the work in this area. Algazi (1966), Goblick-Holsinger (1967) and others have investigated various "practical" quantization schemes. While several of these schemes attain accuracy which is close to the Shannon limit, they all require too much processing for the application described above. The conversion of a full-word (i.e., real) number to a shorter compressed (i.e., quantized) number in each of these schemes requires access to a rather large table or a modest amount of computation, at least comparable to the amount of computation required to evaluate an elementary function like exp or log. The speed ratio between the central processor and the input-output channel may not be large enough to make this much processing economical. On the other hand, the conversion between a standard full-word floating point number and its Hackenbush representation can be accomplished on some computers with only a very small amount of computation, roughly comparable to the amount of time required to fix a floating point number.

2. THE HACKENBUSH NUMBER SYSTEM

Hackenbush numbers are represented by finite or semi-infinite sequences of two letters: L and R.

Let X denote such a string and let $|X|$ denote the real number which X represents. Let $\bar{X}$ be the complement of X, obtained by replacing each L by R and vice versa. Let "X rounded up", denoted by $\uparrow X \uparrow$, be the number obtained by converting each L to 1, each R to 0, and then placing a binary point at the left and a terminal "rounding" one at the far right. For example, $\uparrow LLRLRR \uparrow = .1101001$; $\uparrow RRL \uparrow = .0011$.

The rules for converting Hackenbush numbers to real numbers are as follows:

1. $|RX| = - |L\bar{X}|$

2. $|LLX| = 1 + |LX|$

3. $|LRX| = \uparrow X \uparrow$

4. empty string = 0

As an example, we convert all Hackenbush strings of length 4 to diadic rationals (i.e., rational numbers whose denominators are powers of 2):

LLLL	=	4
LLLR	=	2 1/2
LLRL	=	1 3/4
LLRR	=	1 1/4
LRLL	=	7/8
LRLR	=	5/8
LRRL	=	3/8
LRRR	=	1/8
RLLL	=	-1/8
RLLR	=	-3/8
RLRL	=	-5/8
RLRR	=	-7/8
RRLL	=	-1 1/4
RRLR	=	-1 3/4
RRRL	=	-2 1/2
RRRR	=	-4

Every finite Hackenbush string represents a diadic rational, and every diadic rational may be uniquely represented as a finite Hackenbush string. The semi-infinite Hackenbush strings represent all real numbers as well as the transfinite numbers ω = LLLLL..., $-\omega$ = RRRR..., and the infinitesimals $1/\omega$ = LRRRR..., $-1/\omega$ = RLLLL... Every real number which is not a diadic rational has a unique representation as an infinite Hackenbush string, while any diadic rational, y, is arbitrarily close to the two infinite Hackenbush strings that represent $y + 1/\omega$ and $y - 1/\omega$. Thus 3/2 = LLR, $3/2 + 1/\omega$ = LLRLRRRRRRR..., $3/2 - 1/\omega$ = LLRRLLLLLL... On the other hand, the only Hackenbush string infinitesimally close to 1/3 is 1/3 = LRRLRLRLRLRL... Every real number is infinitesimally close to at least one Hackenbush string.

An m-bit Hackenbush representation of a real number, x, is defined as the first m bits of an infinite string which is infinitesimally close to x. Referring to the above list of 4-bit Hackenbush numbers, we see that the 4-bit representation of 1/3 = LRRLRLRL is LRRL = 3/8, which happens also to be the nearest 4-bit Hackenbush number. However, Π = LLLL RRRLRRLR... has the 4-bit representation of LLLL = 4, even though LLLR = 5/2 is closer. The 4-bit Hackenbush representation of $1 + 1/\omega$ is LLRR, while the 4-bit Hackenbush representation of $1-1/\omega$ is LRLL. Either of these two expressions may be taken as a 4-bit Hackenbush representation of 1.

Although the m-bit Hackenbush representation of y does not always yield the m-bit Hackenbush number closest to y, it has the advantages of flexibility and computational ease. One simply converts y to a very precise Hackenbush representation and then picks off the leftmost m bits. In a subsequent section we shall show that if y is a real number chosen from a known distribution (c.f., the normal distribution), then the accuracy of the m-bit Hackenbush representation is significantly better than a normallized floating point representation with (m-3) bits of characteristic, and only slightly worse than a floating point representation with (m-2) bits of characteristic, even though the floating point representations are allowed an arbitrarily large number of mantissa bits. The mantissa bits give the floating point numbers greater accuracy in representing very large or very small x, but these bits are usually wasted when x is taken from a known reasonable distribution.

3. THE GAME OF HACKENBUSH

Despite its usefullness in compressing numberical data, the Hackenbush number system was not invented for this purpose; it was discovered in the process of solving a certain two-person perfect-information game called Hackenbush. This game is played on a grounded graph. At each turn, a player removes a branch of the graph, and all branches and nodes which are no longer connected to the ground disappear. The game continues until some player is no longer able to make a legal move, at which point the game ends and the player unable to move loses. The name might be related to the fact that the graph often looks like a bush at which the players hack away.

In the original version of Hackenbush, which was invented by J. H. Conway and was first publicized in M. Gardner's column on mathematical games in the January 1972 issue of Scientific American, either player may take any branch of the graph. This version is called neutral Hackenbush. The number system to which this paper is devoted arises from a more complicated variation of the game, called left-right Hackenbush. In this version, branches of the original graph are colored. Some are colored Red, and some are colored Lavender. Each player is permitted to remove only his own color. The player called L (for left) can take only lavender branches, and the player called R (for right) can take only red branches.

The Hackenbush number system proves very useful for solving this game, especially when the graph is a tree. The role of this number system is most clearly visible in the case when the graph is merely a sum of strings, such as the graph of Fig. 1. In such a case, it can be proved that the game may be solved in the following manner. Each string is assigned the corresponding numerical value, and these numbers are added. If the resulting sum is positive, left can win no matter who moves first. If the resulting sum is negative, right can win no matter who moves first. If the resulting sum is zero, the game is so close that whoever moves second can win. The following calculation thus reveals the game of Fig. 1 to be a win for the second player.

$$\begin{aligned} LRL &= 3/4 \\ LLRRL &= 1-3/8 \\ RLLL &= -1/8 \\ RR &= \underline{-2} \\ \text{Total} &= 0 \end{aligned}$$

A forthcoming book by E. R. Berlekamp, J. H. Conway, and R. K. Guy will discuss further aspects of left-right Hackenbush and many other games, one of the most challenging of which is left-right-neutral Hackenbush. In this version of Hackenbush, some branches are colored L or R, but other branches may be taken by either player.

4. COMPARISON OF ACCURACY WITH FLOATING POINT REPRESENTATION

Let x be a random variable which assumes real values in the interval $a < x < b$ according to the distribution $p(x)$. Let the interval be partitioned into N subintervals, each of length $\delta = (b-a)/N$, and suppose that a quantizer approximates x by the midpoint of the subinterval within which x lies. Then the mean-square quantization error is given by

$$\overline{\ell^2} = \sum_{n=1}^{N} \int_{a+(n-1)\delta}^{a+n\delta} (x-(n-1/2)\delta)^2 \, p(x)dx$$

For small δ and smooth $p(x)$, this may be approximated by

$$\overline{\ell^2} \approx \frac{\delta^2}{12} \int_a^b p(x)dx$$

The m-bit Hackenbush quantizer partitions the interval [0,1] into 2^{m-2} subintervals; the interval [1,2], into 2^{m-3} subintervals; the interval [k,k+1] into 2^{m-k-2} subintervals, so that for large m, its total mean-square error is

$$\overline{\ell^2_{\text{Hack}}} \approx \sum_k \frac{1}{12(2^{m-k-2})^2} \int_k^{k+1} (p(x) + p(-x))dx \qquad (1)$$

On the other hand, the quantizer which converts x to a normalized floating point number with 1 bit of sign, (m-2) bits of characteristic, and and infinite number of mantissa bits partitions [1,2] into 2^{m-3} subinternals; [2,4], into 2^{m-4}; [4,8] into 2^{m-5},... while [1/2,1] is partitioned into 2^{m-2}, [1/4,1/2], into 2^{m-1},... giving a total mean square error of

$$\overline{\ell^2_{\text{Float}}} = \sum_j \frac{1}{12(2^{m-j-3})^2} \int_{2^j}^{2^{j+1}} (p(x) + p(-x))dx \qquad (2)$$

Notice that expressions (1) and (2) give equal weights to the subintervals $|1/2,1|$, $|1,2|$, and $|2,3|$, but that expression (2) gives a lower weight to the integrals over all other subregions. Hence $\overline{\ell^2_{Hack}} \geq \overline{\ell^2_{Float}}$, although for many distributions, the difference is not large. For example, if p(x) is the Gaussian distribution with mean zero and variance one, then $p(x) = \frac{\exp -x^2/2}{\sqrt{2\Pi}}$. Letting the sum in Eq. (2) run from $-\infty$ to ∞ gives $\overline{e^2_{float}} = 2.89570/4^m$, while Eq. (1), with k running from 0 to ∞, gives $\overline{e^2_{Hack}} = 3.52016/4^m$. According to the classical information theoretic limit of Shannon, the number of bits, m, which a quantizer must output in order to approximate the normalized Gaussian distribution with rms error $\overline{e^2}$ is given by $m \geq R(\overline{e^2})$, where the rate-distortion function $R(\overline{e^2}) = 1/2 \log \frac{1}{\ell^2}$. Hence, the optimum m-bit quantizer can not surpass $e^2_{opt} = 4^{-m}$, and it follows that the m-bit Hackenbush quantizer is better than any (m-1)-bit quantization for normal distribution.

While the standard deviation is the natural and most convenient unit for the Hackenbush quantizer, it turns out not to be theoretically optimum. If instead of encoding x, one Hackenbush quantizer encodes 1.03 x, the mean-square error is reduced by about one part in 1.3×10^{-4}. Since the multiplication becomes a significant fraction of the total time for the Hackenbush encoding algorithm, this refinement merits consideration only when the original distribution is unnormalized.

REFERENCES

1. V. R. Algazi, "Useful Approximations to Optimum Quantizers," IEEE Trans. Com. Tech., Com.-14: pp. 297-301, (1966).

2. P. Elias, "Bounds on Performance of Optimum Quantizers," IEEE Trans. Inform. Th., IT-16: pp. 172-184, (1970).

3. T. J. Goblick and J. L. Holsinger, "Analog Source Digitation: A Compression of Theory and Practice," IEEE Trans. Inform. Th., IT-13: pp. 323-366, (1967).

4. C. E. Shannon, "A mathematical Theory of Communication," Bell Sys. Tech. Jour., 37: pp. 379-623, (1948).

5. C. E. Shannon, "Coding Theorems for a Discrete Source with a Fidelity Criterion," IRE National Convention Record, pt. 4: pp. 142-163, (1959).

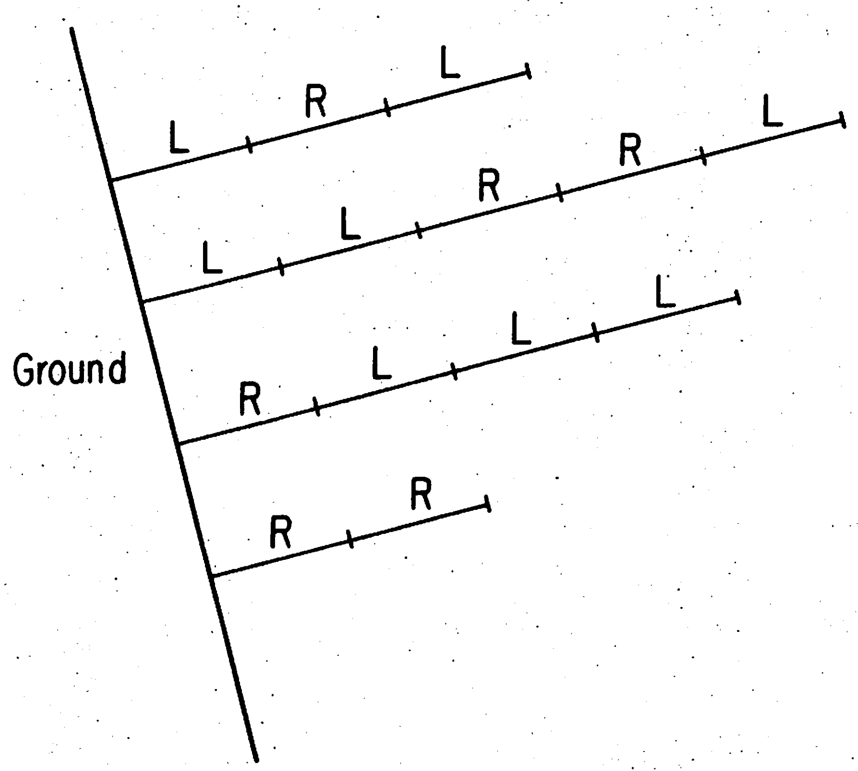

Fig. 1. A position in the game of left-right Hackenbush.