Extended Canonical Recoding

Nicholas Pippenger*
(nicholas@cs.ubc.ca)

Department of Computer Science
The University of British Columbia
Vancouver, British Columbia V6T 1Z4
CANADA

Abstract: Canonical recoding transforms a sequence of bits into a sequence of signed digits, preserving the numerical value of the sequence while reducing the number of non-zero digits. It is used to reduce the number of additions and subtractions when performing multiplication (or equivalently, the number of multiplications and divisions when performing exponentiation). Standard canonical recoding uses the digits 0 and ± 1. Any two non-zero digits are separated by at least one zero, so in the worst case $n/2 + O(1)$ non-zero digits are used to recode an n-bit sequence; in the average case $n/3 + O(1)$ non-zero digits are used. We introduce *extended canonical recoding*, which uses the digits 0, ± 1 and ± 3. Any two non-zero digits are separated by at least two zeroes, so at most $n/3 + O(1)$ non-zero digits are used in the worst case. We show that the scheme is uniquely determined by this condition, and that it is optimal among schemes using the same set of signed digits. Finally, we show that in the average case, extended canonical recoding uses $n/4 + O(1)$ non-zero digits.

* The work reported here was supported by an NSERC Research Grant and a Canada Research Chair.

1. Introduction

The idea of accelerating binary multiplication by recoding the multiplier was introduced by Booth [B1] in 1951. It was used in various forms when multiplication was performed by shifting and adding. The key idea is to use subtractions as well as additions, in order to reduce the total number of operations. It relies on the fact that a subtraction has the same (or nearly the same) cost as an addition. Thus, for example, multiplication by 7, which would normally be performed using three additions (corresponding to the three 1s in the binary expansion 111 of the multiplier), can be performed with one addition and one subtraction (corresponding to the two non-zero digits in the recoded version $100\overline{1}$ of the multiplier, where $\overline{1} = -1$ calls for a subtraction instead of an addition). (We should note that when we speak of the number of additions and subtractions, we are assuming that the partial product is initialized to zero, and that the adder is used for each non-zero digit in the recoded multiplier. An alternative is to use the first non-zero digit to initialize the partial product, and not to use the adder until a second non-zero digit is encountered. The effect of this improvement is to reduce the number of uses of the adder by exactly one, except in those cases in which all bits of the multiplier are 0s.) Interest in the recoding of multipliers abated with the advent of parallel multiplication algorithms that operate in constant time, independent of the value of the multiplier. Recoding techniques have enjoyed a renaissance, however, with the need in modern cryptography to multiply points on elliptic curves by large integers (see Morain and Olivos [M2]) and to perform exponentiation with very large exponents (see Brickell *et al.* [B2]). In the latter case, recoding is used to replace multiplications by a smaller number of multiplications and divisions. It is generally assumed in analysis that division has the same cost as addition; though this is not usually strictly true, it is justified if the reciprocal of the exponentiand is prepared in advance, which can be done with cost independent of the length of the exponent. In this paper we shall adhere to the original terminology, in which multiplication is performed by recoding the multiplier to call for subtractions as well as additions.

Our goal in this paper is to describe a new recoding technique, *extended canonical recoding*, in which the multiplier is recoded so as to call not only for additions and subtractions of the multiplicand, but also for additions and subtractions of three times the multiplicand. Thus, for example, multiplication by 21 would normally be performed using three additions (corresponding to the three 1s in the binary expansion 10101 of the multiplier), and this cannot be improved using 0s, 1s and $\overline{1}$s. It can be performed, however, with one addition and one subtraction of three times the multiplicand (corresponding to the two non-zero digits in the recoded version $300\overline{3}$ of the multiplier, where $\overline{3} = -3$ calls for a subtraction of three times the muliplicand). It will be assumed in analysis that addition or subtraction of three times the multiplicand costs the same as addition of the multiplicand itself; this is justified if the triple of the multiplicand is prepared in advance, which can be done with cost independent of the length of the multiplier.

The use of additions and subtractions of three times the multiplicand to accelerate multiplication has already been described by MacSorley [M1] (in the section "Uniform Shifts of Three"). Extended canonical recoding, however, enjoys certain optimality properties among schemes that recode the multiplier using the digits 0, 1, $\overline{1}$, 3 and $\overline{3}$. Indeed, extended canonical recoding is an extended version of *canonical recoding*, which has analogous optimality properties among schemes that recode the multiplier using the digits 0, 1 and $\overline{1}$. The first published descriptions of canonical recoding were given by Lehman [L1, L2], Tocher [T] and Reitwiesner [R], who all developed it indpendently during the 1950s. It was also discussed in the review by MacSorley [M1] (in the section "Multiplication Using Variable Length Shift").

In Section 2, we shall give a new presentation of standard canonical recoding. This will allow us to introduce, in a simple setting, new proof techniques that will be used in the following section. The main results of Section 2 are as follows. Each natural number has a unique representation, using the signed digits 0, 1 and $\overline{1}$, in which any two non-zero digits are separated by at least one zero. This representation, called the *canonical* representation, has the minimum possible number of non-zero digits among representations of the given natural number using the signed digits 0, 1 and $\overline{1}$. In the worst case, the recoding of an n-bit number uses $\lfloor (n+2)/2 \rfloor$ non-zero digits. In the average case, for a uniformly distributed n-bit number, $n/3+4/9+O(1/2^n)$ non-zero digits are used. This compares favorably with the technique of recoding pairs (see MacSorley [M1], in the section "Uniform Shifts of Two"), which also recodes using the digits 0, 1 and $\overline{1}$, and uses $3n/8+O(1)$ non-zero digits in the average case.

In Section 3, we shall we shall turn to extended canonical recoding. The main results of Section 3 are as follows. Each natural number has a unique representation, using the signed digits 0, 1, $\overline{1}$, 3 and $\overline{3}$, in which any two non-zero digits are separated by at least two zeroes. This representation, called the *extended canonical* representation, has the minimum possible number of non-zero digits among representations of the given natural number using the signed digits 0, 1, $\overline{1}$, 3 and $\overline{3}$. In the worst case, the recoding of an n-bit number uses $\lfloor (n+3)/3 \rfloor$ non-zero digits. In the average case, for a uniformly distributed n-bit number, $n/4 + 7/16 + O(1/2^{n/2})$ non-zero digits are used. This compares favorably with the technique of recoding triplets (see MacSorley [M1], in the section "Uniform Shifts of Three"), which also recodes using the digits 0, 1, $\overline{1}$, 3 and $\overline{3}$, and uses $7n/24 + O(1)$ non-zero digits in the average case.

2. Standard Canonical Recoding

In this section we shall recount the theory of canonical recoding, giving proofs that will generalize to cover extended canonical recoding in the next section. The canonical recoding of a number is most easily determined by working from the least significant bit to the most significant bit. Automata of various kinds will play a prominent role in our arguments, and we shall adhere to the convention that automata process their input strings from left to right. These circumstances lead us to employ the following unusual convention: when an automaton processes the bits of a number from least significant to most significant, we shall express the number as a string with the least significant bit at the left and the most significant bit at the right; this is of course precisely the reverse of the usual convention. On the less frequent occasions on which bits of a number are being processed from most significant to least significant, we shall employ the usual convention that the number is expressed as a string with the most significant bit at the left and the least significant bit at the right.

Let $x = x_1 \cdots x_n$ be a string over the alphabet $\{0, 1\}$ representing an integer

$$R(x) = \sum_{1 \le k \le n} x_k \, 2^{k-1},$$

with the least significant bit at the left. We adopt the convention that the empty string ε represents the integer zero, and note that the number represented by a string is unchanged if *trailing* 0s are appended (to the right end of the string). A string $y = y_1 \cdots y_n$ over the alphabet $\{0, 1, \overline{1}\}$ will be taken to represent the integer

$$S(y) = \sum_{1 \le k \le n} y_k \, 2^{k-1}$$

in the analogous way, where $\overline{1} = -1$. Note that $S(\cdots)$ is an extension of $R(\cdots)$: if $x \in \{0, 1\}^*$ contains only 0s and 1s, then $S(x) = R(x)$. The same convention concerning the empty string and observation about trailing 0s applies to these representations. A string y will be called *canonical* if any two non-zero digits (elements of $\{1, \overline{1}\}$) are separated by at least one 0.

Theorem 2.1: For every non-negative integer N, there is a canonical representation $y \in \{0, 1, \overline{1}\}^*$, which is unique up to trailing 0s.

Proof: To show the existence of a canonical representation, we apply the automaton M_1 described in Table 1 to the string $x00$, where $x \in \{0,1\}^*$ is a binary representation of N, to produce a string $y \in \{0,1,\overline{1}\}^*$.

p	ξ	q	η
$\rightarrow 0$	0	0	0
0	10	0	10
0	11	1	$\overline{1}0$
1	1	1	0
1	01	1	$\overline{1}0$
1	00	0	10

Table 1. Read-ahead automaton M_1 transcribing input in $\{0,1\}^*$ to canonically recoded output. Columns give old state p, input symbols ξ, new state q and output symbols η.

Automaton M_1 has two states, 0 and 1, with the initial state 0 being indicated by an arrow in the table. It is a *read-ahead* automaton, meaning that associated with each of its states is a set of transitions corresponding to the input sequences in a maximal prefix code. (A *prefix code* is a set of strings such that no string in the set is a prefix of any other string in the set. A prefix code (over a given alphabet) is *maximal* if no proper superset of strings (over the given alphabet) is a prefix code.) For example, when the automaton is in state 0, the input transitions correspond to the sequences in the maximal prefix code $\{0,10,11\}$. The automaton will read as many input symbols as necessary to determine which transition to follow. Thus when in state 0, if the next input symbol is 0, the automaton stays in state 0 and produces the output symbol 0. If the next input symbol is 1, however, the automaton must read another input symbol to determine whether to stay in state 0 and produce the output symbols 10 (if this additional input symbol is 0), or to go to state 1 and produce the output symbols $\overline{1}0$ (if the additional input symbol is 1).

We shall assume that enough trailing 0s are present in the input string to resolve any ambiguity about which transition to take. We shall also assume that enough trailing 0s are present so that the automaton finishes in state 0; these trailing 0s "flush out" the automaton, and may produce one additional non-zero output symbol. It is easy to see that two additional 0s suffice to accomplish these objectives in all cases.

Automaton M_1 can also be described by the following verbal explanation of its transition function. (This verbal explanation gives some insight into the construction of the transition table, and its style will serve us in situations in which presenting the transition table would be awkward.) If the old state p has the same parity as the next input symbol, then process this input symbol ξ by going to the new state $\bigl(p + R(\xi)\bigr) \,/\, 2$ and producing the output symbol $\eta = 0$. Otherwise, let $\xi \in \{0,1\}^2$ be the next two input symbols. The integer $p + R(\xi)$ is odd and in the interval $[1,3]$. Thus it can be expressed uniquely as $p + R(\xi) = 4q + S(\eta)$, where $q \in \{0,1\}$ and $\eta \in \{10, \overline{1}0\}$. Process these input symbols ξ by going to the new state q and producing the output symbols η.

The output string produced by the automaton is always canonical, since every non-zero output symbol is immediately followed by a 0. It remains to verify that the number $S(y)$ represented by the output string y is the same as the number $N = R(x) = R(x00)$ represented by the input string $x00$. To do this, we prove the following claim by induction on the length of the string v: if the automaton starts in state p, processes the input string $v \in \{0,1\}^*$, finishes in state q and produces the output string $w \in \{0,1,\overline{1}\}^*$, then

$$R(v) + p = S(wq).$$

The basis of the induction is the case $v = \varepsilon$, for which $w = \varepsilon$ and $q = p$, and the claim reduces to $p = S(p)$ for $p \in \{0,1\}$. For the inductive step, suppose that the first transition processes the input symbols ξ, where $v = \xi v'$, takes the automaton to state p', and produces the output symbols η, where $w = \eta w'$. Then by inductive hypothesis we have

$$R(v') + p' = S(w'q).$$

The inductive step is established in two cases, corresponding to the two clauses in the verbal explanation.

If p has the same parity as the next symbol of the input, so that ξ is a single symbol, then

$$R(\xi) + p = 2p' \text{ and } S(\eta) = 0 \tag{2.1}$$

This yields the desired conclusion,

$$R(v) + p = R(\xi) + 2R(v') + p = 2R(v') + 2p' = 2S(w'q) = S(\eta) + 2S(w'q) = S(wq),$$

where the middle equality is the inductive hypothesis, the first and last equalities are properties of $R(\cdots)$ and $S(\cdots)$, and the remaining two equalities follow from (2.1).

Otherwise, ξ comprises two symbols, so that

$$R(\xi) + p = S(\eta) + 4p'. \tag{2.2}$$

This yields the desired conclusion,

$$R(v) + p = R(\xi) + 4R(v') + p = 4R(v') + S(\eta) + 4p' = S(\eta) + 4S(w'q) = S(wq),$$

where the third equality is the inductive hypothesis, the first and last equalities are properties of $R(\cdots)$ and $S(\cdots)$, and the second equality follows from (2.2). This completes the inductive step, and thus the proof of the claim.

The automaton starts in state 0. Since the automaton is applied to the input string $x00$, the two trailing 0s ensure that the automaton finishes in state 0. Thus, using the claim we have

$$N = R(x) = R(x00) = S(y0) = S(y).$$

This completes the proof of the existence of the canonical representation.

To prove the uniqueness, up to trailing 0s, of the canonical representation of the natural number N, we use induction on N. The basis of the induction is the case $N = 0$, for which we must prove that the unique representation is ε, possibly followed by trailing 0s. Suppose, to obtain a contradiction, that $S(y) = 0$, where y contains at least one non-zero digit, and the smallest possible number of leading 0s. If $y = 0z$, then $S(z) = 0$, where z contains at least one non-zero digit and fewer leading 0s than y, a contradiction. Thus $y = uz$, where $u \in \{1, \overline{1}\}$, and $S(y) = S(u) + 2S(z)$. But $S(u)$ is odd and $2S(z)$ is even, so $S(y)$ is odd, again a contradiction. This completes the proof of the basis of the induction.

Assume then that $N \geq 1$. If N is even, then any representation of N must begin with 0. This leading 0 must be followed by a canonical representation of $N/2 < N$, which is unique, up to trailing 0s, by inductive hypothesis. If N is odd, then any representation of N must begin with $u \in \{1, \overline{1}\}$, and in a canonical representation this leading u must be followed by a 0. If $N \equiv 1 \pmod 4$, then we must have $u = 1$ (since any string beginning $\overline{1}0$ represents a number congruent to 3 modulo 4). This leading 10 must then be followed by a canonical representation of $(N-1)/4 < N$, which is unique, up to trailing 0s, by inductive hypothesis. If, however, $N \equiv 3 \pmod 4$, then we must have $u = \overline{1}$ (since any string beginning 10 represents a number congruent to 1 modulo 4). This leading $\overline{1}0$ must then be followed by a canonical representation of $(N+1)/4 < N$, which is unique, up to trailing 0s, by inductive hypothesis. This completes the proof of the uniqueness of the canonical representation, and thus of the theorem. □

The purpose of recoding is to reduce the number of non-zero digits in the recoded multiplier. The optimality of canonical recoding with respect to this objective is established by the following theorem.

Theorem 2.2: For every natural number N, the canonical representation of N has the smallest number of non-zero digits among the strings $y \in \{0, 1, \overline{1}\}^*$ such that $S(y) = N$.

Proof: We consider a read-ahead automaton M_2, which is an augmented version of M_1. The automaton M_2 has three states, 0, 1 and $\overline{1}$, with 0 being the initial state. The automaton M_2 accepts input strings in $\{0, 1, \overline{1}\}^*$ rather than merely $\{0, 1\}^*$. It is obtained from M_1 by adding the new state, $\overline{1}$, which is the mirror image of 1, and adding appropriate transitions to accomodate the input symbol $\overline{1}$. We shall give only a verbal description of the transitions. (A table analogous to Table 1 would have 17 rows.)

If the old state p has the same parity as the next input symbol, then process this input symbol ξ by going to the new state $\big(p + S(\xi)\big) \,/\, 2$ and producing the output symbol $\eta = 0$. Otherwise, let $\xi \in \{0, 1, \overline{1}\}^2$ be the next two input symbols. The integer $p + S(\xi)$ is odd and in the interval $[-3, 3]$. Thus it can be expressed uniquely as $p + S(\xi) = 4q + S(\eta)$, where $q \in \{0, 1, \overline{1}\}$ and $\eta \in \{10, \overline{1}0\}$. Process these input symbols ξ by going to the new state q and producing the output symbols η.

The output string produced by this automaton is again always canonical. In particular, a canonical input string will be transcribed unchanged, with the automaton remaining in initial state. And again we can prove the following claim by induction on the length of the string v: if the automaton starts in state p, processes the input string $v \in \{0, 1, \overline{1}\}^*$, finishes in state q and produces the output string $w \in \{0, 1, \overline{1}\}^*$, then

$$S(v) + p = S(wq).$$

The proof differs from that for M_1 only by the replacement of $R(\cdots)$ by $S(\cdots)$.

For any string $v \in \{0, 1, \overline{1}\}^*$, let $I(v)$ denote the number of non-zero digits in v. If $p \in \{0, 1, \overline{1}\}$ is a state of M_2, we define $J(p)$ by

$$J(p) = \begin{cases} 0, & \text{if } p = 0; \\ 1, & \text{if } p \neq 0. \end{cases}$$

We can then prove the following claim by induction on the length of the string v: if the automaton starts in state p, processes the input string $v \in \{0, 1, \overline{1}\}^*$, finishes in state q and produces the output string $w \in \{0, 1, \overline{1}\}^*$, then

$$I(w) + J(q) \leq I(v) + J(p).$$

Now let $y \in \{0, 1, \overline{1}\}^*$ be any string such that $S(y) = N$. Let M_2 process the input string $y00$, producing the output string $z \in \{0, 1, \overline{1}\}^*$. The automaton starts in state $p = 0$, and because of the two trailing 0s, finishes in state $q = 0$. Thus

$$S(z) = S(zq) = S(y00) + p = S(y) = N,$$

so z is a canonical recoding of N. But we also have

$$I(z) = I(z) + J(q) \leq I(y00) + J(p) = I(y).$$

This completes the proof of the theorem. □

Having seen that canonical recoding minimizes the number of non-zero digits in the recoding of any natural number, let us now consider what this minimum number of non-zero digits is in the worst case and the average case.

Theorem 2.3: For $n \geq 1$, the maximum, over all n-bit natural numbers N (that is, numbers such that $0 \leq N \leq 2^n - 1$), of the number of non-zero digits in the canonical recoding of N, is $\lfloor (n+2)/2 \rfloor$.

Proof: Let $x \in \{0, 1\}^n$ maximize the number of non-zero digits in the canonical recoding of $N = R(x)$. The canonical recoding is obtained by feeding the string $x00$ into the automaton M_1. This produces $n+2$ output symbols, and the construction of M_1 ensures that it finishes in state 0 and any non-zero digit in the output is immediately followed by a 0. This implies that there can be at most $\lfloor (n+2)/2 \rfloor$ non-zero digits in the output. This upper bound is attained for $N = R\big((10)^{(n-1)/2}1\big)$ for $n \geq 1$ odd and for $N = R\big((10)^{(n-2)/2}11\big)$ for $n \geq 2$ even. This completes the proof of the theorem. □

We now turn to the average number of non-zero digits.

Theorem 2.4: The expected number of non-zero digits in the canonical recoding of a uniformly distributed n-bit natural number N (that is, with all 2^n numbers N such that $0 \leq N \leq 2^n - 1$ being equally likely), is

$$\frac{n}{3} + \frac{4}{9} + O\left(\frac{1}{2^n}\right).$$

Proof: It will be convenient to consider yet another automaton, the automaton M_3 described in Table 2.

Automaton M_3 is a *write-behind* automaton, meaning that it reads exactly one input symbol per transition, but that it sometimes omits writing an output symbol during a transition, compensating by writing two output symbols during the following transition.

It is obtained from M_1 by interpolating a new state, $1/2$, between the states 0 and 1. The transitions into this state eliminate the need to read ahead, but they create the need to write behind. It is easy to see that any input sequence that takes M_1 from state $p \in \{0, 1\}$ to state $q \in \{0, 1\}$ does the same to M_3, and produces the same output sequence. (Because the writing of automaton M_3 is never more than one symbol behind the reading of the input, it is possible to construct an automaton that reads and writes one symbol per step, but with a delay of one step. This automaton uses the first input symbol to determine the initial state, then reads the second input symbol and writes the first output symbol in the first step, and so forth.)

p	ξ	q	η
$\rightarrow 0$	0	0	0
0	1	$1/2$	ε
$1/2$	0	0	10
$1/2$	1	1	$\bar{1}0$
1	0	$1/2$	ε
1	1	1	0

Table 2. Write-behind automaton M_3 transcribing input in $\{0, 1\}^*$ to canonically recoded output. Columns give old state p, input symbols ξ, new state q and output symbols η.

To obtain the expected number of non-zero digits in the canonical recoding of a uniformly distributed n-bit natural number, we consider feeding n independent unbiased random bits into M_3, followed by two 0s, and count the expected number of non-zero output symbols. Inspection of Table 2 shows that non-zero output symbols are produced during and only during transitions out of state $1/2$. It will be convenient to count them by counting the number of visits to state $1/2$. (The two trailing 0s ensure that the automaton finishes in state 0, so that transitions out of state $1/2$ are in one-to-one correspondence with visits to state $1/2$.) This focus on visits to the state $1/2$ allows us to ignore the actual outputs produced by the automaton.

Let us first consider the visits to state $1/2$ while processing the first n random bits. For $p \in \{0, 1/2, 1\}$, let P_p^k denote the probability that feeding k independent unbiased bits into M_3 takes it to state p. From the initial conditions $P^0_0 = 1$ and $P^0_{1/2} = P^0_1 = 0$, together with the recurrence relations

$$\begin{aligned}
P^{k+1}_0 &= \frac{1}{2}P^k_0 + \frac{1}{2}P^k_{1/2} \\
P^{k+1}_{1/2} &= \frac{1}{2}P^k_0 \qquad\qquad + \frac{1}{2}P^k_1 \\
P^{k+1}_1 &= \qquad\quad \frac{1}{2}P^k_{1/2} + \frac{1}{2}P^k_1,
\end{aligned}$$

we obtain that the generating functions $\Phi_p(X) = \sum_{k\geq 0} P^k_p\, X^k$ are given by

$$\begin{aligned}
\Phi_0(X) &= \frac{4-2X-X^2}{(1-X)(2-X)(2+X)} = \frac{1}{3}\frac{1}{1-X} + \frac{1}{6}\frac{1}{1+X/2} + \frac{1}{2}\frac{1}{1-X/2} \\
\Phi_{1/2}(X) &= \frac{X}{(1-X)(2+X)} = \frac{1}{3}\frac{1}{1-X} + \frac{1}{3}\frac{1}{1+X/2} \\
\Phi_1(X) &= \frac{X^2}{(1-X)(2-X)(2+X)} = \frac{1}{3}\frac{1}{1-X} + \frac{1}{6}\frac{1}{1+X/2} - \frac{1}{2}\frac{1}{1-X/2},
\end{aligned}$$

and thus that the probabilities are given by

$$\begin{aligned}
P^k_0 &= \frac{1}{3} + \frac{1}{6}\frac{(-1)^k}{2^k} + \frac{1}{2}\frac{1}{2^k} \\
P^k_{1/2} &= \frac{1}{3} - \frac{1}{3}\frac{(-1)^k}{2^k} \\
P^k_1 &= \frac{1}{3} + \frac{1}{6}\frac{(-1)^k}{2^k} - \frac{1}{2}\frac{1}{2^k}.
\end{aligned}$$

Thus the expected number of visits to the state $1/2$ due to the first n random bits is

$$\begin{aligned}
\sum_{1\leq k\leq n} P^k_{1/2} &= \sum_{1\leq k\leq n} \left(\frac{1}{3} - \frac{1}{3}\frac{(-1)^k}{2^k}\right) \\
&= \frac{n}{3} + \frac{1}{9} + O\left(\frac{1}{2^k}\right).
\end{aligned}$$

Now let us consider the visits to state $1/2$ due to the two trailing 0s. If the n random bits take the automaton to state 0, then there are no further visits to state $1/2$. If the n random bits take the automaton to state $1/2$, then again there are no further visits to state $1/2$ (and that visit to state $1/2$ has already been accounted for in the analysis of the

n random bits). If the n random bits take the automaton to state 1, then there is one further visit to state 1/2. Thus the expected number of visits due to the two trailing 0s is

$$\begin{aligned} P_1^n &= \frac{1}{3} + \frac{1}{6}\frac{(-1)^k}{2^k} - \frac{1}{2}\frac{1}{2^k} \\ &= \frac{1}{3} + O\left(\frac{1}{2^k}\right). \end{aligned}$$

Combining these contributions completes the proof of the theorem. □

The canonical recoding cannot be produced (with any bounded delay) by a finite automaton processing the bits from most significant to least significant, as can be seen by considering the canonical recodings of the integers $R\big((01)^{t+1}\big)$ and $R\big(11(01)^t\big)$. The recoding can be done in this direction, however, by a technique called "on-the-fly conversion", described by Frougny [F]. Since this technique works from the most significant bit to the least significant bit, we shall represent numbers as strings in the usual way, with the most significant bit at the left and the least significant bit at the right. The technique uses sequential machine that has a register, capable of holding a string, for each state of the finite automaton; for canonical recoding we shall call these registers R_0, $R_{1/2}$ and R_1. We initialize these registers as follows:

$$\begin{aligned} R_0 &\leftarrow 00 \\ R_{1/2} &\leftarrow 001 \\ R_1 &\leftarrow 01. \end{aligned}$$

(These initial assignments have the same effect as starting with empty registers and processing two leading 0s as described below.) We then process each bit of the multiplier in turn, from most significant to least significant. For each multiplier bit 0, we execute the following assignments (either simultaeously or in the order indicated):

$$\begin{aligned} R_1 &\leftarrow R_{1/2} \\ R_{1/2} &\leftarrow R_0 \cdot 01 \\ R_0 &\leftarrow R_0 \cdot 0. \end{aligned}$$

(Here the operator "$\cdot$" denotes concatenation.) For each multiplier bit 1, we execute the following assignments (again either simultaeously or in the order indicated):

$$\begin{aligned} R_0 &\leftarrow R_{1/2} \\ R_{1/2} &\leftarrow R_1 \cdot 0\overline{1} \\ R_1 &\leftarrow R_1 \cdot 0. \end{aligned}$$

After all the bits of the input have been processed, the recoded multiplier is held in the register R_0. The correctness of this algorithm is established by proving the following claim by induction on the number of input bits: after processing $n \geq 0$ input bits $x_n \cdots x_1$, register R_p contains a string $y_{n+2} \cdots y_1$, where $y_1 \cdots y_{n+2}$ is the string that would be produced by automaton M_1, started in state p and processing the string $x_1 \cdots x_n 00$ as input.

3. Extended Canonical Recoding

In this section we shall develop extended canonical recoding. Like the canonical recoding, the extended canonical recoding of a number is most easily determined by working from the least significant bit to the most significant bit. We shall therefore revert to our convention of representing numbers with the least significant bit at the left when they are being processed by automata working from the least significant to the most sigificant bit.

A string $y = y_1 \cdots y_n$ over the alphabet $\{0, 1, \overline{1}, 3, \overline{3}\}$ will be taken to represent the integer

$$E(y) = \sum_{1 \leq k \leq n} y_k \, 2^{k-1},$$

where $\overline{3} = -3$. Note that $E(\cdots)$ is an extension of $S(\cdots)$: if $x \in \{0, 1, \overline{1}\}^*$ contains only 0s, 1s and $\overline{1}$s, then $E(x) = S(x)$. The usual convention concerning the empty string and observation about trailing 0s applies to these representations. A string y will be called an *extended canonical* string if any two non-zero digits (elements of $\{1, \overline{1}, 3, \overline{3}\}$) are separated by at least two 0s.

Theorem 3.1: For every non-negative integer N, there is an extended canonical representation $y \in \{0, 1, \overline{1}, 3, \overline{3}\}^*$, which is unique up to trailing 0s.

Proof: To show the existence of an extended canonical representation, we apply the automaton M_4 described in Table 3 to the string $x000$, where $x \in \{0, 1\}^*$ is a binary representation of N, to produce a string $y \in \{0, 1, \overline{1}, 3, \overline{3}\}^*$.

p	ξ	q	η
$\to 0$	0	0	0
0	100	0	100
0	101	1	$\bar{3}00$
0	110	0	300
0	111	1	$\bar{1}00$
1	1	1	0
1	011	1	$\bar{1}00$
1	010	0	300
1	001	1	$\bar{3}00$
1	000	0	100

Table 3. Read-ahead automaton M_4 transcribing input in $\{0,1\}^*$ to extended canonically recoded output. Columns give old state p, input symbols ξ, new state q and output symbols η.

Automaton M_4 has two states, 0 and 1, with 0 being the initial state. Like M_1, it is a read-ahead automaton, but with the difference that it must sometimes read *two* additional input symbols to determine the appropriate transition.

We shall again assume that enough trailing 0s are present in the input string to resolve any ambiguity about which transition to take, and to return the automaton to state 0. It is easy to see that three additional 0s suffice to accomplish these objectives in all cases.

Automaton M_4 can also be described by the following verbal explanation of its transition function. If the old state p has the same parity as the next input symbol, then process this input symbol ξ by going to the new state $\bigl(p + R(\xi)\bigr) \,/\, 2$ and producing the output symbol $\eta = 0$. Otherwise, let $\xi \in \{0,1\}^3$ be the next three input symbols. The integer $p + R(\xi)$ is odd and in the interval $[1,7]$. Thus it can be expressed uniquely as $p + R(\xi) = 8q + E(\eta)$, where $q \in \{0,1\}$ and $\eta \in \{100, \bar{1}00, 300, \bar{3}00\}$. Process these input symbols ξ by going to the new state q and producing the output symbols η.

The output string produced by the automaton is always an extended canonical string, since every non-zero output symbol is immediately followed by two consecutive 0s. It remains to verify that the number $E(y)$ represented by the output string y is the same as the number $N = R(x) = R(x000)$ represented by the input string $x000$. To do this, we prove the following claim by induction on the length of the string v: if the automaton starts in state p, processes the input string $v \in \{0,1\}^*$, finishes in state q and produces the output string $w \in \{0, 1, \overline{1}, 3, \overline{3}\}^*$, then

$$R(v) + p = E(wq).$$

The basis of the induction is the case $v = \varepsilon$, for which $w = \varepsilon$ and $q = p$, and the claim reduces to $p = E(p)$ for $p \in \{0, 1\}$. For the inductive step, suppose that the first transition processes the input symbols ξ, where $v = \xi v'$, takes the automaton to state p', and produces the output symbols η, where $w = \eta w'$. Then by inductive hypothesis we have

$$R(v') + p' = E(w'q).$$

The inductive step is established in two cases, corresponding to the two clauses in the verbal explanation.

If p has the same parity as the next symbol of the input, so that ξ is a single symbol, then

$$R(\xi) + p = 2p' \text{ and } E(\eta) = 0 \tag{3.1}$$

This yields the desired conclusion,

$$R(v) + p = R(\xi) + 2R(v') + p = 2R(v') + 2p' = 2E(w'q) = E(\eta) + 2E(w'q) = E(wq),$$

where the middle equality is the inductive hypothesis, the first and last equalities are properties of $R(\cdots)$ and $E(\cdots)$, and the remaining two equalities follow from (3.1).

Otherwise, ξ comprises three symbols, so that

$$R(\xi) + p = E(\eta) + 8p'. \tag{3.2}$$

This yields the desired conclusion,

$$R(v) + p = R(\xi) + 8R(v') + p = 8R(v') + E(\eta) + 8p' = E(\eta) + 4E(w'q) = E(wq),$$

where the third equality is the inductive hypothesis, the first and last equalities are properties of $R(\cdots)$ and $E(\cdots)$, and the second equality follows from (3.2). This completes the inductive step, and thus the proof of the claim.

The automaton starts in state 0. Since the automaton is applied to the input string $x000$, the three trailing 0s ensure that the automaton finishes in state 0. Thus, using the claim we have

$$N = R(x) = R(x000) = E(y0) = E(y).$$

This completes the proof of the existence of the extended canonical representation.

To prove the uniqueness, up to trailing 0s, of the extended canonical representation of the natural number N, we use induction on N. The basis of the induction is the case $N = 0$, for which we must prove that the unique representation is ε, possibly followed by trailing 0s. Suppose, to obtain a contradiction, that $E(y) = 0$, where y contains at least one non-zero digit, and the smallest possible number of leading 0s. If $y = 0z$, then $E(z) = 0$, where z contains at least one non-zero digit and fewer leading 0s than y, a contradiction. Thus $y = uz$, where $u \in \{1, \overline{1}, 3, \overline{3}\}$, and $E(y) = E(u) + 2E(z)$. But $E(u)$ is odd and $2E(z)$ is even, so $E(y)$ is odd, again a contradiction. This completes the proof of the basis of the induction.

Assume then that $N \geq 1$. If N is even, then any representation of N must begin with 0. This leading 0 must be followed by an extended canonical representation of $N/2 < N$, which is unique, up to trailing 0s, by inductive hypothesis. If N is odd, then any representation of N must begin with $u \in \{1, \overline{1}, 3, \overline{3}\}$, and in an extended canonical representation this leading u must be followed by two 0s. If $N \equiv 1 \pmod{8}$, then we must have $u = 1$ (since any string beginning $\overline{1}00$, 300 or $\overline{3}00$ represents a number congruent to 7, 3 or 5 modulo 8). This leading 100 must then be followed by an extended canonical representation of $(N-1)/8 < N$, which is unique, up to trailing 0s, by inductive hypothesis. If $N \equiv 3 \pmod{8}$, then we must have $u = 3$ (since any string beginning 100, $\overline{1}00$ or $\overline{3}00$ represents a number congruent to 1, 7 or 5 modulo 8). This leading 300 must then be followed by an extended canonical representation of $(N-3)/8 < N$, which is unique, up to trailing 0s, by inductive hypothesis. If $N \equiv 5 \pmod{8}$, then we must have $u = \overline{3}$ (since any string beginning 100, $\overline{1}00$ or 300 represents a number congruent to 1, 7 or 3 modulo 8). This leading $\overline{3}00$ must then be followed by an extended canonical representation of $(N+3)/8 < N$, which is unique, up to trailing 0s, by inductive hypothesis. Finally, if $N \equiv 7 \pmod{8}$, then we must have $u = \overline{1}$ (since any string beginning 100, 300 or $\overline{3}00$ represents a number congruent to 1, 3 or 5 modulo 8). This leading $\overline{1}00$ must then be followed by an extended canonical representation of $(N+1)/8 < N$, which is unique, up to trailing 0s, by inductive hypothesis. This completes the proof of the uniqueness of the exztended canonical representation, and thus of the theorem. $\Box$

The optimality of extended canonical recoding, among schemes usig the same set of digits, is established by the following theorem.

Theorem 3.2: For every natural number N, the extended canonical representation of N has the smallest number of non-zero digits among the strings $y \in \{0, 1, \overline{1}, 3, \overline{3}\}^*$ such that $E(y) = N$.

Proof: We consider a read-ahead automaton M_5, which is an augmented version of M_4. The automaton M_5 has seven states, 0, 1, $\overline{1}$, 2, $\overline{2}$, 3 and $\overline{3}$, with 0 being the initial state. The automaton M_5 accepts input strings in $\{0,1,\overline{1},3,\overline{3}\}^*$ rather than merely $\{0,1\}^*$. It is obtained from M_4 by adding five new states, $\overline{1}$, 2, $\overline{2}$, 3 and $\overline{3}$, and adding appropriate transitions to accomodate the input symbols $\overline{1}$, 3 and $\overline{3}$. We shall give only a verbal description of the transitions. (A table analogous to Table 3 would have 419 rows.)

If the old state p has the same parity as the next input symbol, then process this input symbol ξ by going to the new state $\bigl(p+E(\xi)\bigr)\ /\ 2$ and producing the output symbol $\eta = 0$. Otherwise, let $\xi \in \{0,1,\overline{1},3\overline{3}\}^3$ be the next three input symbols. The integer $p+E(\xi)$ is odd and in the interval $[-23,23]$. Thus it can be expressed uniquely as $p+E(\xi) = 8q+E(\eta)$, where $q \in \{0,1,\overline{1},2,\overline{2},3,\overline{3}\}$ and $\eta \in \{100,\overline{1}00,300,\overline{3}00\}$. Process these input symbols ξ by going to the new state q and producing the output symbols η.

The output string produced by this automaton is again always an extended canonical string. In particular, an extended canonical input string will be transcribed unchanged, with the automaton remaining in initial state. And again we can prove the following claim by induction on the length of the string v: if the automaton starts in state p, processes the input string $v \in \{0,1,\overline{1},3,\overline{3}\}^*$, finishes in state q and produces the output string $w \in \{0,1,\overline{1},3,\overline{3}\}^*$, then

$$E(v) + p = E(wq).$$

The proof differs from that for M_4 only by the replacement of $R(\cdots)$ by $E(\cdots)$.

For any string $v \in \{0,1,\overline{1},3,\overline{3}\}^*$, let $I(v)$ denote the number of non-zero digits in v. If $p \in \{0,1,\overline{1},2,\overline{2},3,\overline{3}\}$ is a state of M_5, we define $J(p)$ by

$$J(p) = \begin{cases} 0, & \text{if } p = 0; \\ 1, & \text{if } p \neq 0. \end{cases}$$

We can then prove the following claim by induction on the length of the string v: if the automaton starts in state p, processes the input string $v \in \{0,1,\overline{1},3,\overline{3}\}^*$, finishes in state q and produces the output string $w \in \{0,1,\overline{1},3,\overline{3}\}^*$, then

$$I(w) + J(q) \leq I(v) + J(p).$$

Now let $y \in \{0,1,\overline{1},3,\overline{3}\}^*$ be any string such that $E(y) = N$. Let M_5 process the input string $y000$, producing the output string $z \in \{0,1,\overline{1},3,\overline{3}\}^*$. The automaton starts in state $p = 0$, and because of the three trailing 0s, finishes in state $q = 0$. Thus

$$E(z) = E(zq) = E(y000) + p = E(y) = N,$$

so z is an extended canonical recoding of N. But we also have

$$I(z) = I(z) + J(q) \leq I(y000) + J(p) = I(y).$$

This completes the proof of the theorem. □

Having seen that extended canonical recoding minimizes the number of non-zero digits in the recoding of any natural number, let us now consider what this minimum number of non-zero digits is in the worst case and the average case.

Theorem 3.3: For $n \geq 1$, the maximum, over all n-bit natural numbers N (that is, numbers such that $0 \leq N \leq 2^n - 1$), of the number of non-zero digits in the extended canonical recoding of N, is $\lfloor (n+3)/3 \rfloor$.

Proof: Let $x \in \{0,1\}^n$ maximize the number of non-zero digits in the extended canonical recoding of $N = R(x)$. The extended canonical recoding is obtained by feeding the string $x000$ into the automaton M_4. This produces $n+3$ output symbols, and the construction of M_4 ensures that it finishes in state 0 and that each non-zero digit is immediately followed by two consecutive 0s. This implies that the number of non-zero digits is at most $\lfloor (n+3)/3 \rfloor$. This upper bound is attained for $N = R\big((100)^{n-1}1\big)$ for $n \geq 1$ congruent to 1 modulo 3, for $N = R\big((100)^{n-2}11\big)$ for $n \geq 2$ congruent to 2 modulo 3 and for $N = R\big((100)^{n-2}11\big)$ for $n \geq 3$ congruent to 0 modulo 3. This completes the proof of the theorem. □

We now turn to the average number of non-zero digits.

Theorem 3.4: The expected number of non-zero digits in the extended canonical recoding of a uniformly distributed n-bit natural number N (that is, with all 2^n numbers N such that $0 \leq N \leq 2^n - 1$ being equally likely), is

$$\frac{n}{4} + \frac{7}{16} + O\left(\frac{1}{2^{n/2}}\right).$$

Proof: It will be convenient to consider yet another automaton, the automaton M_6 described in Table 4.

p	ξ	q	η
$\to 0$	0	0	0
0	1	1/2	ε
1/4	0	0	100
1/4	1	1	$\bar{3}00$
1/2	0	1/4	ε
1/2	1	3/4	ε
3/4	0	0	300
3/4	1	1	$\bar{1}00$
1	0	1/2	ε
1	1	1	0

Table 4. Write-behind automaton M_6 transcribing input in $\{0,1\}^*$ to extended canonically recoded output. Columns give old state p, input symbols ξ, new state q and output symbols η.

Automaton M_6 is a write-behind automaton; it sometimes omits writing output symbols during two consecutive transitions, compensating by writing three output symbols during the following transition. It is obtained from M_4 by interpolating three new states, 1/4, 1/2 and 3/4, between the states 0 and 1. The transitions into these states eliminate the need to read ahead, but they create the need to write behind. It is easy to see that any input sequence that takes M_4 from state $p \in \{0,1\}$ to state $q \in \{0,1\}$ does the same to M_6, and produces the same output sequence. (Because the writing of automaton M_6 is never more than two symbols behind the reading of the input, it is possible to construct an automaton that reads and writes one symbol per step, but with a delay of two steps. This automaton uses the first two input symbols to determine the initial state, then reads the third input symbol and writes the first output symbol in the first step, and so forth.)

To obtain the expected number of non-zero digits in the extended canonical recoding of a uniformly distributed n-bit natural number, we consider feeding n independent unbiased random bits into M_6, followed by three 0s, and count the expected number of non-zero output symbols. Inspection of Table 4 shows that non-zero output symbols are produced during and only during transitions out of states 1/4 and 3/4. It will be convenient to count them by counting the number of visits to these states. (The three trailing 0s ensure that the automaton finishes in state 0, so that transitions out of states 1/4 and 3/4 are in one-to-one correspondence with visits to these states.) Further inspection shows that visits to states 1/4 and 3/4 are in one-to-one correspondence with the visits to state 1/2 that immediately precede them. Thus we shall focus on visits to the state $1/2$, and ignore the actual outputs produced by the automaton.

Let us first consider the visits to state $1/2$ while processing the first n random bits. For $p \in \{0, 1/4, 1/2, 3/4, 1\}$, let P_p^k denote the probability that feeding k independent unbiased bits into M_6 takes it to state p. We begin by observing that for all $k \geq 0$ we have $Q_{1/4}^k = Q_{3/4}^k$. Let us write $Q_{*/4}^k$ for this common value.

From the initial conditions $Q_0^0 = 1$ and $Q_{1/2}^0 = Q_{*/4}^0 = Q_1^0 = 0$, together with the recurrence relations

$$\begin{aligned}
Q_0^{k+1} &= \frac{1}{2}Q_0^k && + \quad Q_{*/4}^k \\
Q_{1/2}^{k+1} &= \frac{1}{2}Q_0^k && \qquad\qquad + \frac{1}{2}Q_1^k \\
Q_{*/4}^{k+1} &= \qquad \frac{1}{2}Q_{1/2}^k && \\
Q_1^{k+1} &= && \quad Q_{*/4}^k + \frac{1}{2}Q_1^k,
\end{aligned}$$

we obtain that the generating functions $\Psi_p(X) = \sum_{k \geq 0} Q_p^k\, X^k$ are given by

$$\begin{aligned}
\Psi_0(X) &= \frac{4 - 2X - X^3}{(1-X)(2-X)(2+X+X^2)} = \frac{1}{4}\frac{1}{1-X} + \frac{1}{4}\frac{1+X/2}{1+X/2+X^2/2} + \frac{1}{2}\frac{1}{1-X/2} \\
\Psi_{1/2}(X) &= \frac{2X - X^2}{(1-X)(2-X)(2+X+X^2)} = \frac{1}{4}\frac{1}{1-X} - \frac{1}{4}\frac{1-X/2}{1+X/2+X^2/2} \\
\Psi_{*/4}(X) &= \frac{X^2 - X^3/2}{(1-X)(2-X)(2+X+X^2)} = \frac{1}{8}\frac{1}{1-X} - \frac{1}{8}\frac{1+3X/2}{1+X/2+X^2/2} \\
\Psi_1(X) &= \frac{X^3}{(1-X)(2-X)(2+X+X^2)} = \frac{1}{4}\frac{1}{1-X} + \frac{1}{4}\frac{1+X/2}{1+X/2+X^2/2} - \frac{1}{2}\frac{1}{1-X/2}.
\end{aligned}$$

Expressions for the probabilities are complicated, owing to the complex factors $1-\alpha X$ and $1-\beta X$, where $\alpha = (-1+i\sqrt{7})/4$ and $\beta = (-1-i\sqrt{7})/4$, of the denominator $1+X/2+X^2/2$. But since $|\alpha| = |\beta| = 1/\sqrt{2}$, we have from the partial-fraction expansion of $\Psi_{1/2}(X)$ that

$$Q_{1/2}^k = \frac{1}{4} + O\left(\frac{1}{2^{k/2}}\right).$$

Thus the expected number of visits to the state $1/2$ due to the first n random bits is

$$\sum_{1 \leq k \leq n} Q_{1/2}^k = \frac{n}{4} + \frac{3}{16} + O\left(\frac{1}{2^{n/2}}\right),$$

where the constant term $3/16$ is obtained by evaluating $\Psi_{1/2}(X) - (1/4)\big(1/(1-x)\big)$ at $X = 1$.

Now let us consider the visits to state $1/2$ due to the three trailing 0s. If the n random bits take the automaton to state 0, $1/4$ or $3/4$, then there are no further visits to state $1/2$. If the n random bits take the automaton to state $1/2$, then again there are no further visits to state $1/2$ (and that visit to $1/2$ has already been accounted for in the analysis of the n random bits). If the n random bits take the automaton to state 1, then there is one further visit to state $1/2$. Thus the expected number of visits due to the three trailing 0s is

$$Q_1^n = \frac{1}{4} + O\left(\frac{1}{2^{n/2}}\right),$$

from the partial-fraction expansion $\Psi_1(X)$. Combining these contributions completes the proof of the theorem. $\square$

The extended canonical recoding cannot be produced (with any bounded delay) by a finite automaton processing the bits from most significant to least significant, as can be seen by considering the extended canonical recodings of the integers $R\big((001)^{t+1}\big)$ and $R\big(111(01)^t\big)$. The recoding can be done in this direction, however, using "on-the-fly conversion". Since this technique processes the input from most significant bit to least significant bit, we shall again return to representing numbers with the most significant bit at the left. We shall use five registers: R_0, $R_{1/4}$, $R_{1/2}$, $R_{3/4}$ and R_1. We initialize these registers as follows:

$$\begin{aligned} R_0 &\leftarrow 000 \\ R_{1/4} &\leftarrow 00001 \\ R_{1/2} &\leftarrow 0001 \\ R_{3/4} &\leftarrow 00003 \\ R_1 &\leftarrow 001. \end{aligned}$$

(These initial assignments have the same effect as starting with empty registers and processing three leading 0s as described below.) We then process each bit of the multiplier in turn, from most significant to least significant. For each multiplier bit 0, we execute the following assignments (either simultaeously or in the order indicated):

$$\begin{aligned} R_1 &\leftarrow R_{1/2} \\ R_{3/4} &\leftarrow R_0 \cdot 003 \\ R_{1/2} &\leftarrow R_{1/4} \\ R_{1/4} &\leftarrow R_0 \cdot 001 \\ R_0 &\leftarrow R_0 \cdot 0. \end{aligned}$$

For each multiplier bit 1, we execute the following assignments (again either simultaeously or in the order indicated):

$$\begin{aligned} R_0 &\leftarrow R_{1/2} \\ R_{1/4} &\leftarrow R_1 \cdot 00\overline{3} \\ R_{1/2} &\leftarrow R_{3/4} \\ R_{3/4} &\leftarrow R_1 \cdot 00\overline{1} \\ R_1 &\leftarrow R_1 \cdot 0. \end{aligned}$$

After all the bits of the input have been processed, the recoded multiplier is held in the register R_0. The correctness of this algorithm is established by proving the following claim by induction on the number of input bits: after processing $n \geq 0$ input bits $x_n \cdots x_1$, register R_p contains a string $y_{n+3} \cdots y_1$, where $y_1 \cdots y_{n+3}$ is the string that would be produced by automaton M_4, started in state p and processing the string $x_1 \cdots x_n 000$ as input.

4. Conclusion

We have described extended canonical recoding, which uses the signed digits 3 and $\overline{3}$ to reduce the number of non-zero digits below that used by standard canonical recoding. Most of the known results concerning canonical recoding have analogues for extended canonical recoding. It is possible to extend canonical recoding still further, increasing the number of 0s between non-zero digits to three or more. This increase, however, requires using a set of signed digits whose size grows exponentially with the minimum distance between non-zero digits. Standard and extended canonical recoding appear to be particularly favorable cases in the trade-off between the size of the digit set and the average- or worst-case number of non-zero digits.

5. References

[B1] A. D. Booth, "A Signed Binary Multiplication Technique", *Quart. J. Mech. Appl. Math.*, 4 (1951) 236–240.

[B2] E. F. Brickell, D. M. Gordon, K. S. McCurley and D. B. Wilson, "Fast Exponentiation with Precomputation", *Proc. Eurocrypt*, (1992) 200–207.

[F] C. Frougny, "On-the-Fly Algorithms and Sequential Machines", *IEEE Trans. on Computers*, 49 (2000) 859–863.

[L1] M. Lehman, "High-Speed Digital Multiplication", *IRE Trans. on Electronic Computers*, 6 (1957) 204–205.

[L2] M. Lehman, "Short-Cut Multiplication and Division in Automatic Binary Digital Computers", *Proc. IEE*, 105 B (1958) 496–504.

[M1] O. L. MacSorley, "High-Speed Arithmetic in Binary Computers", *Proc. IRE*, 49 (1961) 67–91.

[M2] F. Morain and J. Olivos, "Speeding Up the Computations on an Elliptic Curve Using Addition-Subtraction Chains", *Theoretical Informatics and Applications*, 24 (1990) 531–543.

[R] G. W. Reitwiesner, "Binary Arithmetic", *Advances in Computers*, 1 (1960) 232–308.

[T] K. D. Tocher, "Techniques of Multiplication and Division for Automatic Binary Computers", *Quart. J. Mech. Appl. Math.*, 11 (1958) 364–384.