A Primer on Balanced Binary Representations

Jeffrey Shallit*
Department of Computer Science
University of Waterloo
Waterloo, Ontario, Canada N2L 3G1
shallit@graceland.uwaterloo.ca

July 1992 - Revised 1993

Abstract

We discuss balanced binary representations.

1 Introduction and Definitions

Every non-negative integer n can be represented essentially uniquely in base 2, as follows:

$$n = \sum_{0 \le i \le j} e_i 2^i$$

where $e_i \in \{0, 1\}$ and $e_j \neq 0$ for $n \neq 0$. We consider the consequences of enlarging the digit set to $\{-1, 0, 1\}$. We call such an expansion a *signed-digit* expansion.

One immediate consequence is that every integer, positive, negative, or zero, can be represented using the digits $\{-1, 0, 1\}$. In fact,

Theorem 1.1 *Every nonzero integer has an infinite number of signed-digit expansions.*

Proof. We prove this for positive integers n, the proof for negative integers being essentially identical. Write the ordinary base-2 representation of $n-1$ as $(n-1)_2 = e_j e_{j-1} \cdots e_0$. Choose any $k > j$, and consider the representation of 1 as $1 \ \overbrace{-1 \ -1 \ \cdots \ -1}^{k}$. Now add these two representations, term by term. The result is a representation of n using only the digits $1, 0$, and -1. ■

*Research supported in part by a grant from NSERC.

We now restrict our attention to a particular type of signed-digit expansion.

Theorem 1.2 *Every integer has a signed-digit representation containing no two adjacent nonzero digits.*

Proof. It suffices to prove the result for non-negative integers. We use induction on n. Clearly the result is true for $n = 0$. Now, if n is even, take a representation of $n/2$ and concatenate 0. If $n \equiv 1 \pmod 4$, take a representation of $(n-1)/4$ and concatenate 01. If $n \equiv -1 \pmod 4$, take a representation of $(n+1)/4$ and concatenate $0\ -1$. ■

Theorem 1.3 *Every nonzero integer has exactly one representation containing no two adjacent nonzero digits and no leading zeroes.*

Proof. Suppose $n = \sum_{0 \le i \le j} e_i 2^i = \sum_{0 \le i \le j} f_i 2^i$ an integer with at least two distinct representations. Without loss of generality we may assume $n > 0$ and n is the least such integer. Consider both of these expansions modulo 2. If $e_0 \equiv 0 \pmod 2$, then $f_0 \equiv 0 \pmod 2$. Hence, by dropping the least significant bit, we get two expansion for $n/2 < n$, a contradiction.

Similarly, by considering these expansions modulo 4, we find that either (i) $e_0 = f_0 = 1$ and $e_1 = f_1 = 0$, or (ii) $e_0 = f_0 = -1$ and $e_1 = f_1 = 0$. In the former case, $(n-1)/4$ has two distinct representations, and in the latter $(n+1)/4$ has two distinct representations. ■

We call such a representation the *balanced binary representation.*

We define the *weight* of a signed-digit representation to be the number of nonzero digits.

Theorem 1.4 *Balanced binary representation minimizes the weight over all signed-digit representations.*

Of course, there can be several signed-digit representations achieving the minimum weight, such as $1\ 0\ -1$ and 11 for 3.

Theorem 1.5 *There are* $t_n = \frac{2^n - (-1)^n}{3}$ *distinct representations of length* n.

Proof. Any representation of length n must either end in 0 or 1 or -1. In the former case, the representation consists of a valid representation of length $n-1$ concatenated with 0. In the latter case, the representation consists of a valid representation of length $n-2$ concatenated with either 01 or $0\ -1$. Thus $t_n = t_{n-1} + 2t_{n-2}$. Also $t_1 = 1$ and $t_2 = 1$, which gives the result. ■

2 Algorithms

The following algorithm computes the balanced binary representation for a non-negative integer n.

```
BBR(n)
if (n = 0) then
          return(ε)
     else
          determine e such that 2^e ≤ n < 2^(e+1)
          if (3n > 2^(e+2)) then
                    return(2^(e+1), -BBR(2^(e+1) − n)
               else
                    return (2^e, BBR(n − 2^e))
```

The following algorithm computes an alternative signed-digit representation that also has minimal weight:

```
BBR2(n)
if (n = 0) then
          return(ε)
     else
          determine e such that 2^e ≤ n < 2^(e+1)
          if (2^(e+1) − n ≤ n − 2^e) then
                    return(2^(e+1), -BBR2(2^(e+1) − n)
               else
                    return (2^e, BBR2(n − 2^e))
```

Note the outputs are different: BBR(11) gives $16-4-1$, while BBR2(11) gives $8+4-1$. Both representations are of weight 3.

3 Transducers

We can convert from ordinary binary representation to balanced binary using the following finite-state transducer:

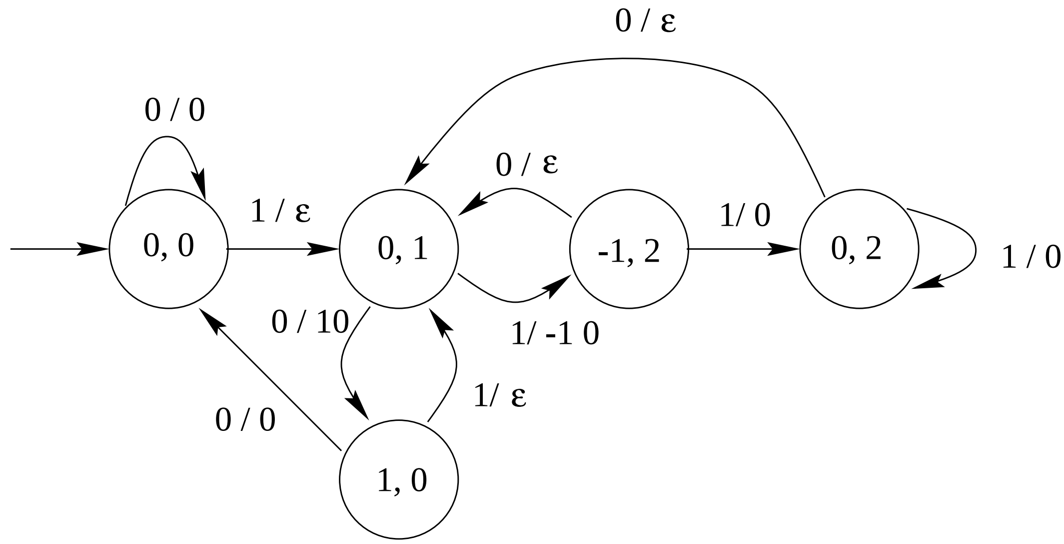

Figure 1: Transducer converting ordinary binary to balanced binary

The input is given starting with the least significant digit and the output has the same order. The input may need two additional zeroes at the end to achieve the complete output.

For example, on input 010111100 the output is 010−100010.

On the other hand, it is easy to see that no finite-state transducer can convert an arbitrary signed-digit binary expansion to ordinary binary. For example, if we take the most significant digit first, then if the input is $10000 \cdots 0$, the transducer cannot output any correct output until seeing the next digit. If it is 1, the output should be $10000 \cdots 01$. But if it is -1, the output should be $01111 \cdots 11$. Thus there is arbitrarily long delay, and no finite-state transducer will work.

However, we can convert from signed-digit binary to ordinary binary using a *pushdown transducer*. (For more about pushdown transducers, see [6, 7, 9, 8, 14].) Suppose we read the input starting with the most significant digit, followed by an endmarker. On input 1, for each following 0 you see, push a counter onto the stack until a 1 or -1 is seen. If you see a 1, output a 1 followed by a 0 for each counter on the stack (popping stack as you output). If you see a -1, output a 0 and then a 1 for each counter on the stack (popping stack as you output). Finally, there is an endmarker, which is treated like a 1.

4 k-automatic and k-regular sequences

It follows from the transducer in Section 3 that a sequence $(s_n)_{n \geq 0}$ is 2-automatic using an automaton processing the ordinary base-2 representation of n iff it is 2-automatic using an automaton processing the balanced binary representation of n.

Suppose we define $s(n)$ to be the sum of the digits in the balanced binary expansion of

n. Then we have, for $n \geq 0$,

$$\begin{aligned} s(2n) &= s(n); \\ s(4n+1) &= s(n)+1; \\ s(4n+3) &= s(n+1)-1. \end{aligned}$$

It follows from this that

$$\begin{aligned} s(8n+1) &= s(4n+1); \\ s(8n+3) &= s(n)+s(2n+1)-s(4n+1); \\ s(8n+5) &= -s(n)+s(2n+1)+s(4n+1); \\ s(8n+7) &= s(4n+3); \end{aligned}$$

and hence s is 2-regular.

Suppose we define $w(n)$ to be the weight (number of non-zero terms) in the balanced binary expansion of n. Then following the argument in Theorems 1.2 and 1.3 we find, for $n \geq 0$, that

$$\begin{aligned} w(2n) &= w(n); \\ w(4n+1) &= w(n)+1; \\ w(4n+3) &= w(n+1)+1. \end{aligned}$$

It follows that

$$\begin{aligned} w(8n+1) &= w(4n+1); \\ w(8n+3) &= -w(n)+w(2n+1)+w(4n+1); \\ w(8n+5) &= w(8n+3); \\ w(8n+7) &= w(4n+3); \end{aligned}$$

and so $(w(n))_{n \geq 0}$ is a 2-regular sequence in the sense of Allouche and Shallit [1].

The sequence $w(n)$ has the following expansion as a sum of pattern sequences:

$$w(n) = a_1(n) - \sum_{i \geq 0} a_{11(01)^i 1}(n).$$

Here $a_P(n)$ denotes the number of occurrences of the pattern P in the (ordinary) binary representation of n.

Note added January 1994: The sequence $(w(n))_{n \geq 0}$ also appears in a paper of Weitzman [20].

Theorem 4.1 *Suppose we define* $t(n) := \sum_{0 \leq k < 2^n} (w(n) - s_2(n))$*, where* $s_2(n)$ *counts the sum of the digits in the (ordinary) binary representation of* n*. Then* $t(n) = \frac{1}{6} n 2^n - \frac{4}{9} 2^n + -\frac{1}{18}(-1)^n + \frac{1}{2}$.

5 Previous work

Booth [3] discussed the use of binary numbers with both positive and negative digits, as did Avizienis [2] and Takagi & Yajima [18].

There are evident links between ordinary binary representation and addition chains. In the same way, there are links between signed-digit representation and addition/subtraction chains. See, for example, [17, 19, 4] and [11, Solution to Exercise 4.6.3.30, p. 638].

Reitwiesner [16] and Jedwab & Mitchell [10] proved that balanced binary represention gives a minimum weight representation.

Morain & Olivos [15], Eğecioğlu & Koç [5], and Koblitz [12] independently gave an application of balanced binary representation to speeding up computations on an elliptic curve. Koyama & Tsuruoka [13] discussed a signed-digit representation in which the average run-length of the blocks of zeroes is increased, while still retaining the minimum weight.

References

[1] J.-P. Allouche and J. O. Shallit. The ring of k-regular sequences. *Theoret. Comput. Sci.* **98** (1992), 163–197.

[2] A. Avizienis. Signed-digit number representations for fast parallel arithmetic. *IRE Trans. Electron. Comput.* **10** (1961), 389–400.

[3] A. D. Booth. A signed binary multiplication technique. *Quart. J. Mech. Appl. Math.* **4** (1951), 236–240.

[4] J. Bos and M. Coster. Addition chain heuristics. In G. Brassard, editor, *Advances in Cryptology—CRYPTO '89 Proceedings*, Vol. 435 of *Lecture Notes in Computer Science*, pages 400–407. Springer-Verlag, 1990.

[5] O. Eğecioğlu and C. K. Koç. Fast modular exponentiation. In *Proc. 1990 Bilkent Int'l. Conf. New Trends in Communication, Control, and Signal Processing*, Vol. 1, pages 188–194. Elsevier, 1990.

[6] J. Evey. Application of pushdown store machines. In *Proc. 1963 Fall Joint Computer Conference*, pages 215–227. AFIPS Press, 1963.

[7] P. C. Fischer. On computability by certain classes of restricted Turing machines. In *Proc. 4th Ann. IEEE Symp. Switching Circuit Theory and Logical Design*, pages 23–32, 1963.

[8] S. Ginsburg and S. A. Greibach. Mappings which preserve context-sensitive languages. *Inform. Control* **9** (1966), 563–582.

[9] S. Ginsburg and G. F. Rose. Preservation of languages by transducers. *Inform. Control* **9** (1966), 153–176.

[10] J. Jedwab and C. J. Mitchell. Minimum weight modified signed-digit representations and fast exponentiation. *Electronics Letters* **25** (1989), 1171–1172.

[11] D. E. Knuth. *The Art of Computer Programming. Volume 2: Seminumerical Algorithms.* Addison-Wesley, 1981. 2nd edition.

[12] N. Koblitz. CM-Curves with good cryptographic properties. In J. Feigenbaum, editor, *Advances in Cryptology—CRYPTO '91 Proceedings*, Vol. 576 of *Lecture Notes in Computer Science*, pages 279–287. Springer-Verlag, 1991.

[13] K. Koyama and Y. Tsuruoka. Speeding up elliptic cryptosystems by using a signed binary window method. In E. F. Brickell, editor, *Advances in Cryptology—CRYPTO '92 Proceedings*, Vol. 740 of *Lecture Notes in Computer Science*, pages 345–357. Springer-Verlag, 1993.

[14] P. M. Lewis, II and R. E. Stearns. Syntax directed transduction. *J. Assoc. Comput. Mach.* **15** (1968), 465–488.

[15] F. Morain and J. Olivos. Speeding up the computations on an elliptic curve using addition-subtraction chains. *RAIRO Inform. Théor. App.* **24** (1990), 531–544.

[16] G. W. Reitwiesner. *Binary arithmetic*, Vol. 1 of *Advances in Computers*, pages 231–308. Academic Press, 1960.

[17] A. Schönhage. A lower bound for the length of addition chains. *Theoret. Comput. Sci.* **1** (1975), 1–12.

[18] N. Takagi and S. Yajima. On-line error-detectable high-speed multiplier using redundant binary representation and three-rail logic. *IEEE Trans. Comput.* **36** (1987), 1310–1317.

[19] H. Volger. Some results on addition/subtraction chains. *Inform. Process. Lett.* **20** (1985), 155–160.

[20] A. Weitzman. Transformation of parallel programs guided by micro-analysis. In B. Salvy, editor, *Algorithms Seminar, 1992–1993*, pages 155–159. Institut National de Recherche en Informatique et en Automatique, France, December 1993. Rapport de Recherche, No. 2130.