High-order-bit First Conversion for Signed-Digit Representations

H. T. Kung
Harvard University
Cambridge, MA 02138
kung@harvard.edu

***Abstract*—This paper presents a high-order-bit first conversion (HFC) conversion scheme that converts conventional binary representations of numbers into their signed-digit representations (SDRs). HFC is novel in that it converts higher-order bits before lower-order bits. As such, HFC allows immediate use of significant SDR digits as soon as their values become available during conversion without waiting for the remaining digits to be converted. The properties of SDR, such as carry-free addition/subtraction and reduced-digit computation, can then be readily applied to these converted digits. We describe an application of HFC in deep learning accelerators.**

***Keywords*—Computer arithmetic; signed-digit representation; quantization; deep neural network (DNN); DNN accelerator**

I. Introduction

In signed-digit representations (SDRs), we represent numbers in positive digits, negative digits, and zero [1]. This paper focuses on radix-2 SDR, using digits in $\{-1, 0, 1\}$, as opposed to only $\{0, 1\}$ in a conventional binary representation. For example, in SDR we represent $30 = 2^4 + 2^3 + 2^2 + 2^1$ as $1000\bar{1}0$ where $\bar{1}$ denotes -1.

We know from the literature that SDRs offer several advantages in performing arithmetic. For example, in SDRs, we can perform carry-free addition or subtraction of numbers [1]. In integer multiplication, if we represent the multiplier in SDR, we can reduce the number of steps in accumulating shifted versions of the multiplicand [2]. SDRs have found their applications in various areas, including optical computing [3], cryptographic computation [4], and digit-pipelined arithmetic [5].

In the rest of this paper, we motivate the use of SDRs in the context of accelerating deep neural network (DNN) computations and then argue that for on-the-fly quantization of SDRs, we will need high-order-bit first conversion (HFC) for SDRs. In the final section, we present our proposed HFC scheme. To provide a contrast, we also describe a low-order-bit first conversion (LFC) scheme. While still using a simple finite state machine, HFC is more sophisticated than LFC, by requiring a 2-bit instead of 1-bit look ahead in converting each bit.

II. SDRs and Deep-learning Acceleration

A. SDRs and Bit-level Sparsity

In the interest of accelerating deep learning computation, SDRs have received increased attention in recent years. We know that an SDR can have at most 50% (roughly) of its digits to be nonzeros [4]. Thus one could use SDRs to exploit bit-level sparsity in reducing the computation cost of arithmetic operations [2], [6].

It turns out that, for DNN workloads, SDR can realize increased computation saving. For example, when we apply SDRs to the 8-bit activation values of ResNet-18, almost all the resulting values have no more than three nonzero digits (see Figure 8 in [7]).

B. SDRs and Quantization

Quantization is a widely used technique in reducing computation costs for DNNs. For instance, we may quantize 32-bit floating-point numbers (FP32) into 8-bit integers (INT8) for a 4x saving in memory access and a comparable saving in computation itself.

Suppose that we adopt the conventional uniform quantization. Then an 8-bit quantized integer DNN, as opposed to a 32-bit floating-point DNN, can allow a minimum impact on the classification accuracy (less than 1% in accuracy loss). However, further quantization with uniform quantization below 8 bits would have a severe accuracy impact (see, e.g., [7]).

Recent work has shown that if by using SDRs and applying term quantization as described in [7], we can further quantize the DNN to use on average only 2 or 3 nonzero digits per weight or activation value, with a minimum impact on classification accuracy.

C. The Need for High-order-bit First Conversion for SDRs

Typically we derive SDRs from conventional binary representations of numbers, which can be activation values in DNNs or values capturing real-world analog signals. To allow on-the-fly quantization of a SDR while it is being converted from a binary representation, we need to perform high-order-bit first conversion.

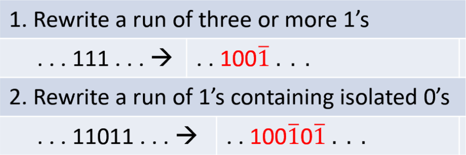
III. One-pass Conversion for SDRs

We convert a given binary representation to its SDC in one scan pass. Before presenting high-order-bit first conversion

DISTRIBUTION STATEMENT A. Approved for public release: distribution unlimited. This research is supported in part by AFRL under award number FA8750-18-1-0112, and DARPA under UCLA award number 0160GXA278 and MIT award number S5181.

(HFC) in Section III-B, we first describe a simpler low-order-bit first conversion (LFC) in Section III-A. It is instructive to compare the two schemes. Both schemes will yield SDRs with the minimum number of nonzero digits. Note that HFC needs to use 2-bit look ahead (LA), while LFC uses just 1-bit LA.

A key in these schemes is identification of a run of three or more 1's. For the schemes to achieve the minimum number of nonzero digits, we extend the notion of run to allow the presence of isolated 0's, where each isolated 0 is surrounded by two 1's on each side, such as 11011. The schemes implement the two rewrite rules in Figure 1 on runs identified.



1. Rewrite a run of three or more 1's	
. . . 111 . . . →	. . 100$\bar{1}$ . . .
2. Rewrite a run of 1's containing isolated 0's	
. . . 11011 . . . →	. . 100$\bar{1}$0$\bar{1}$ . . .

Fig. 1: Two rewrite rules used in the conversion schemes.

During conversion, when converting a given bit, the present state is either IN-A-RUN or NOT-IN-A-RUN, with the latter being the initial state. We assume that for the given binary representation that is to be converted to SDR, the most significant bit (MSB) is 0.

A. Low-order-bit First Conversion (LFC)

The LFC scans the given binary representation from right to left. As illustrated in Figure 2, LFC converts an input bit to an output digit, where the output digit is a function of both the input bit and the next bit (one lookahead bit) based on the finite state machine in Figure 3.

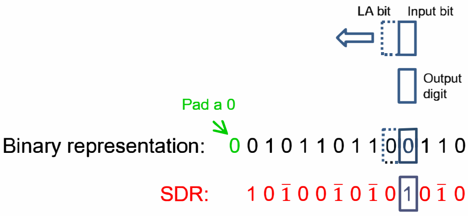
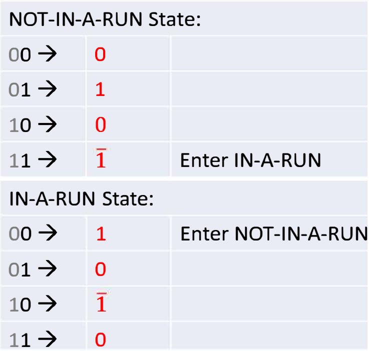

LA bit
Input bit
Output digit
Pad a 0
Binary representation: 0 0 1 0 1 1 0 1 1 0 0 1 1 0
SDR: 1 0 $\bar{1}$ 0 0 $\bar{1}$ 0 $\bar{1}$ 0 1 0 $\bar{1}$ 0

Fig. 2: Illustration of the low-order-bit first conversion (LFC) with 1-bit look ahead.



NOT-IN-A-RUN State:		
00 →	0	
01 →	1	
10 →	0	
11 →	$\bar{1}$	Enter IN-A-RUN
IN-A-RUN State:		
00 →	1	Enter NOT-IN-A-RUN
01 →	0	
10 →	$\bar{1}$	
11 →	0	

Fig. 3: Finite state machine for the low-order-bit first conversion (LFC) with 1-bit look ahead.

B. High-order-bit First Conversion (HFC)

The HFC scans the given binary representation from left to right. As illustrated in Figure 4, HFC converts an input bit to an output digit, where the output digit is a function of both the input bit and the next two bits (2 lookahead bits) based on the finite state machine in Figure 5.

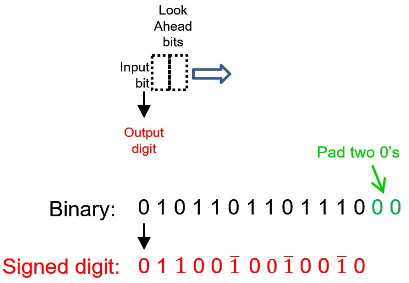
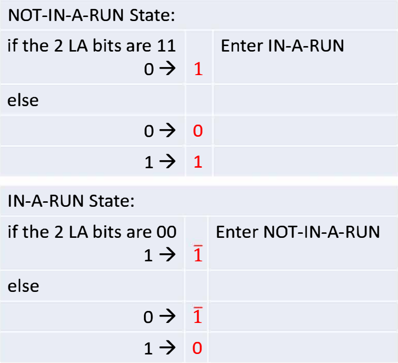

Look Ahead bits
Input bit
Output digit
Pad two 0's
Binary: 0 1 0 1 1 0 1 1 0 1 1 1 0 0 0
Signed digit: 0 1 1 0 0 $\bar{1}$ 0 0 $\bar{1}$ 0 0 $\bar{1}$ 0

Fig. 4: Illustration of the high-order-bit first conversion (HFC) with 2-bit look ahead.



NOT-IN-A-RUN State:		
if the 2 LA bits are 11 0 →	1	Enter IN-A-RUN
else		
0 →	0	
1 →	1	
IN-A-RUN State:		
if the 2 LA bits are 00 1 →	$\bar{1}$	Enter NOT-IN-A-RUN
else		
0 →	$\bar{1}$	
1 →	0	

Fig. 5: Finite state machine for the high-order-bit first conversion (HFC) with 2-bit look ahead.

References

[1] A. Avizienis, "Signed-digit numbe representations for fast parallel arithmetic," *IRE Transactions on electronic computers*, no. 3, pp. 389–400, 1961.

[2] A. D. Booth, "A signed binary multiplication technique," *The Quarterly Journal of Mechanics and Applied Mathematics*, vol. 4, no. 2, pp. 236–240, 1951.

[3] B. L. Drake, R. P. Bocker, M. E. Lasher, R. H. Patterson, and W. J. Miceli, "Photonic computing using the modified signed-digit number representation," *Optical Engineering*, vol. 25, no. 1, p. 250138, 1986.

[4] J. Jedwab and C. J. Mitchell, "Minimum weight modified signed-digit representations and fast exponentiation," *Electronics Letters*, vol. 25, no. 17, pp. 1171–1172, 1989.

[5] M. J. Irwin and R. M. Owens, "Digit-pipelined arnthmetic as illustrated by the paste-up system: A tutorial," *Computer*, no. 4, pp. 61–73, 1987.

[6] S. Sharify, A. D. Lascorz, M. Mahmoud, M. Nikolic, K. Siu, D. M. Stuart, Z. Poulos, and A. Moshovos, "Laconic deep learning inference acceleration," in *Proceedings of the 46th International Symposium on Computer Architecture*. ACM, 2019, pp. 304–317.

[7] H. T. Kung, B. McDanel, and S. Q. Zhang, "Term revealing: Furthering quantization at run time on quantized dnns," *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2020.