概率计算及混合概率计算

李洪革,陈宇昊,吴俊毅,宋印杰,朱新宇
(北京航空航天大学电子信息工程学院,北京市 100191)

摘 要: 非位置概率数的计算机制已经成为边缘计算片上系统的新范式. 本文介绍了概率计算(stochastic computing)的起源、发展和目前国内外的研究现状. 针对传统概率计算存在诸如计算时延长、脉冲串信息携带效率低等问题,本文提出了二进制数-概率脉冲串混合编码的混合概率数概念,并从数的表示机理上阐释了二进制数、概率数和混合概率数的数理关系,进而揭示了混合概率计算所具备的低时延、高算力和高能效比的计算特点. 本文基于40 nm CMOS工艺设计混合概率深度神经网络,该神经网络芯片在内核面积仅0.73 mm × 0.73 mm的条件下,设计4 544个乘累加(MAC)单元. 在时钟频率400 MHz条件下,总功率为102.3 mW,其中动态功耗仅97 μW. 通过ASIC芯片的实验测试表明,混合概率计算作为一种全新的颠覆性计算范式,与其他确定性、可扩展和全并行等概率计算方案相比,其能效比分别提高了50倍、2.5倍和3.26倍.

关键词: 概率数;概率计算;混合概率数;混合概率计算;深度神经网络;能效;算力
基金项目: 国家自然科学基金(No.62071019)
中图分类号: TP391 **文献标识码:** A **文章编号:** 0372-2112(2024)02-0428-13
电子学报URL:http://www.ejournal.org.cn **DOI:**10.12263/DZXB.20230222

Stochastic Computing and Hybrid Stochastic Computing

LI Hong-ge, CHEN Yu-hao, WU Jun-yi, SONG Yin-jie, ZHU Xin-yu
(*College of Electronic Information Engineering*, *Beihang University*, *Beijing* 100191, *China*)

Abstract: The calculation principle of non-positional stochastic number (SN) is a promising technique for realizing high-performance computing owing to its extremely low hardware cost. This paper introduces detailly the origin, development process and the domestic and foreign development present situation. However, a disadvantage of stochastic bitstream is that the computing latency, and information-carrying efficiency and so on. We presented a hybrid stochastic computing (HSC) based on a hybrid bitstream to solve these problems, which achieves a lower hardware cost, better accuracy, and faster speed. The HSC neural networks is fabricated by 40 nm low-power CMOS process, with a core area of 0.73 mm × 0.73 mm, power of 102.3 mW and clock of 400 MHz, which has 4 544 multiply and accumulation (MAC). The proposed Hybrid stochastic computing is tested by FPGA and ASIC. Compared with other stochastic computing method, the method proposed gains 50×, 2.5×,and 3.26× energy efficiency than other methods of traditional stochastic computing.

Key words: stochastic number; stochastic computing; hybrid stochastic number; hybrid stochastic computing; deep neural network; energy efficiency; computing performance
Foundation Item(s): National Natural Science Foundation of China (No.62071019)

1 概率计算简介

1.1 概率计算的起源

十六世纪,德国数学家莱布尼兹发明了二进制并进行了系统性深入研究,使其被广泛应用于今天的电子信息、计算机等各种计算领域. 1956年,冯·诺伊曼发表了“概率逻辑与不可靠单元综合成可靠有机体”的论文[1],该论文首次引入统计计算(probabilistic computing)的概念.

20世纪60年代,英国标准电信实验室的研究员盖恩斯(Gaines)公开发表了概率计算学术论文“stochastic computing”[2],并进行了相关的数学描述和讨论. 1969年,盖恩斯在其论文“stochastic computing system”[3]中对概率计算进行了全面系统的阐述. 与此同时,来自美国伊利诺伊大学的研究人员波佩尔鲍姆和阿夫索于1966年也发表了论文“noise-computer”,并介绍了在图像处理

收稿日期:2023-03-09;修回日期:2023-06-30;责任编辑:孙瑶

领域的科研成果. 1967年,该团队发表了“stochastic computing elements and systems”的研究成果[4],讨论了概率计算的基本方法及其硬件电路的实现策略. 本文中的“stochastic computing”也被国内翻译成随机计算.

20世纪90年代初,日本东北大学的Kondo和Sawada在IEEE神经网络会刊上发表了“functional abilities of a stochastic logic neural network”[5],全面系统地介绍了概率计算神经网络的性质和实现方法等研究成果,为概率计算构造神经网络奠定了基础. 2013年,Alaghi和Hayes刊文写道:概率计算的低资源消耗、误差容错和概率特性将会在某些应用中与传统二进制计算形成竞争[6,7].

近些年来,概率计算在世界范围内产生了一系列重要研究成果. 如2016年,Canals等人[8]提出了一种新的可扩展概率逻辑(Extended Stochastic Logic,ESL)编码方法,扩展了概率脉冲串的表示范围. 2017年,Liu等人[9]首次引入低差异序列,提高了概率计算的计算精度. 2016年,Jenson和Riedel提出了概率计算的确定性计算的方法[10],实现完全准确的概率计算,从而使“stochastic computing”摆脱了“随机”而迈入“确定性”的计算. 国内方面,北京大学王源团队提出一种全概率计算架构,采用28-nm工艺实现了198.9-TOPS/W的能效比且最大限度地提高计算容错能力[11,12]. 上海交通大学钱炜慷团队在国外、国内先后研究了利用传统概率计算进行伯恩斯坦多项式运算[13,14]. 南京大学王中风团队设计了同时具有二进制和概率计算双模式的神经网络加速器[15]. 传统概率计算经过十余年的研究和发展已经从数据转换、基础算术运算、脉冲编码甚至复杂函数和系统网络等各个方面都获得了显著的成果,本文将一些主要成果进行汇总,如表1所示.

表1　概率计算研究

分类	解决对象	具体问题	实施方法
基本单元	SC-BN转换	随机数发生器(SNG)	LFSR+比较器[4],循环移位共享LFSR+比较器[16],随机位翻转的线交换技术共享LFSR[17],加权二进制生成器WBG[18],线性排列WBG[19],综合WBG的向上计数器[20],拟内存方法[21],奇偶线对称交换[22],基于MUX与FSM的SNG设计[23]
		计数器	计数器及基于ADDIE的计算[24],并行累加计数器APC[25]
	算术运算	加减运算	多路选择器加法器[4],计数器加法器[26]
		乘法运算	与门/同或门乘法器[4],截位乘法器[27]
		乘累加运算	并行乘累加器(MAC)[27],差分乘累加器[28]
		平方运算	乘法电路的平方器[4]
		除法运算	基本除法器[4],概率相关性除法器[22],JK触发器的除法器[29]
基本属性	脉冲编码	概率(随机)编码	整型概率计算[30],扩展概率逻辑[8],独立符号位法[31],Halton序列编码[32],Sobol序列编码[33],随机量化编码[34],奇偶校验解码器[35]等
		混合概率编码	二进制-概率数混合编码[36,37]
	计算容错	高容错性	高性能容错计算[4,6,11,12,38]
复杂函数计算	函数运算	激活函数f(x)	FSM构造的激活函数[24],Btanh函数电路[27],ReLU函数电路[39],可配置激活函数电路[40]
	复杂函数	其他	伽马校正电路[20],Sobol边缘检测[20],伯恩斯坦多项式[13]
神经网络系统	概率神经网络	网络前向与反向传播	APC前向网络[41],MUX的前向及反向传播网络[42],ESL构造的反向传播[43]
		池化、归一化、正则化	级联MUX平均池化电路[44],最大池化及tanh电路最大池化[45],直接最大池化电路[46],归一化电路[47],正则化电路[16]
	概率神经网络优化	计算效率 计算时延 数据带宽	基于内积电路的卷积层电路[48],温度计式编码ASC[49],非易失性域墙存储器[50],自旋器件概率计算[51]

随着人工智能边缘计算的迅猛发展,边缘侧对芯片算力、能效比的要求愈来愈高. 随着传统二进制数的并行计算,其芯片算力和功耗的矛盾变得日益尖锐. 为了应对算力-功耗间不可调和的尖锐矛盾,学术界、产业界已经把视野逐渐转移到非二进制数的计算机制,如模拟域的计算. 伪模拟域的概率计算与二进制计算具有相同的逻辑器件又有模拟域计算的特点,因此,成为世界科技界广泛关注的焦点. 概率计算从提出至今已经在各个应用领域全面展开,其中包括LDPC译码器[6,7]、数字滤波[52,53]、图像处理[53~56]、神经网络[57~59]等多种数字计算领域,并取得了较高算力、极低功耗的高能效比的应用效果.

目前,随着深度学习的高速发展,一批科研团队也着眼于利用概率计算的低成本硬件优势,构造性能更优的AI芯片[15,23,60~62]. 此外,由于非确定的概率计算引入了随机性噪声,而噪声又可以用于解决过拟合问题,从

而提高了深度学习的推理精度[62]. 由于这些优势,概率计算在AI加速器芯片领域的应用成为新的研究热点,并催生许多新的研究问题,可以概括为以下几个方向.

(1)概率数表示,即概率数的表示机制和编码方法,主要包括概率数表示机制、随机数发生器及比较电路等优化. 其主要研究目标包括扩展概率脉冲的数值表示范围、计算精度的影响、提高编码效率等.

(2)算术运算电路,包括加减法、乘除法等,主要研究目标是提升算术单元的计算速度与精度,减少硬件开销,实现更高的计算能效比.

(3)复杂函数的计算,如伽马矫正、伯恩斯坦多项式等,可应用于图像处理算法,也可用于实现神经网络的激活函数等功能.

(4)各种复杂系统的实现,如神经网络及深度神经网络、图像处理算法、LDPC编译码以及加解密算法等.

总而言之,概率计算作为一种将概率统计数的表示应用于算术运算电路系统已展现出与传统二进制数计算完全不同表示方法、计算机制和性能优势. 两者却又同样采用CMOS逻辑门实现逻辑电路运算的物理属性,并在许多领域得到了广泛的实验验证或应用.

1.2 概率数表示

众所周知,二进制数是逻辑电路或计算机算术中一种最常用的数的表示系统,而二进制补码是该系统的编码协议. 在数的表示和计算过程中,用非二进制的一串高低电平脉冲中高电平“1”出现的概率来表示的数就是所谓的概率计算中使用的“概率数”. 概率数及其计算机制可以完美地实现传统二进制逻辑电路的算术运算,如加、减法和乘、除法等运算电路等[4~6,24]. 该计算机制不仅缩减了各种传统算术电路的硬件开销,而且可以获得计算精度、计算时延可调节的高能效比的计算优势.

在概率计算中,使用由0和1组成的脉冲串来表示概率数,概率数的取值范围为[0,1]. 理论上,脉冲串中高电平的概率值(p)等于无限长脉冲中“1”出现的频率($\hat{p}$),实际是使用有限长脉冲中“1”出现频率来近似代替. 这种脉冲被称作概率脉冲串(Stochastic Bitstream, SB),概率脉冲串(或称脉冲串)分为单极性(Unipolar)和双极性(Bipolar)两种表示方法. 以单极性的一组脉冲串长为16(1101 1111 1101 1111)的概率脉冲串为例:

$$\hat{p}=\frac{\sum 脉冲1}{\sum 脉冲总数}=\frac{14}{16} \tag{1}$$

用A表示一个概率脉冲串序列,A_j表示序列的第j位,在单极性表示中,概率值(p)就近似等于其中高电平或者比特“1”在全序列长度(L)中出现的频率($\hat{p}$),即

$$\hat{p}=\hat{p}(A=1)=\frac{\sum_{j=0}^{L-1}A_j}{L} \tag{2}$$

这里设A序列的总长度是L个时钟周期,由于$L\geqslant\sum_{j=0}^{L-1}A_j$,概率脉冲串所能表示的范围是[0,1]. 从概率统计角度,A中“1”出现的频率($\hat{p}$)的期望值即为A中“1”出现的概率(p),即

$$E(\hat{p})=\lim_{L\to\infty}\frac{\sum_{j=0}^{L-1}A_j}{L}=p \tag{3}$$

单极性概率脉冲串是进行一组伯努利试验(在同样的条件下重复地、相互独立地进行的一种随机试验),将二进制数与随机数序列依次进行比较,并按比较结果决定输出脉冲串的概率数;若被比较的二进制数大于随机数,则为高电平“1”,否则为“0”. 将该试验重复L次,即与该随机数发生器的L个随机数进行比较,得到长度为L的单极性的概率脉冲串;假设输入数据为$x\in[0,2^n-1]\cap Z$,随机数$R\in Z$在$[0,2^m-1]$上均匀分布(即电路中的m-bit随机数发生器,且$m\geqslant n$),可知$P\{x>R\}=x/2^m$. 这表示脉冲为1的概率会与被比较数x成正比,因此,完整携带了数据x的信息,实现了单极性的概率串,基本电路如图1所示.

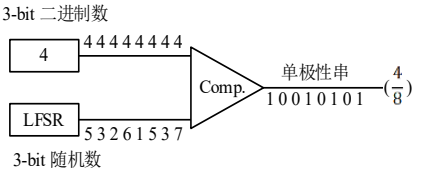


图1 单极性概率串发生器(取n=m=3)

2 混合概率计算及电路

虽然概率计算相比于二进制运算大幅度缩减了计算的硬件资源,但脉冲式概率数需要总长度L个时钟周期的脉冲串才能表示,这导致了计算时延比二进制数膨胀L倍. 另外,传统概率计算的乘法必须保证独立随机数序列以满足其独立性要求. 为了解决上述传统概率计算的固有问题,本文首次提出了混合概率数,以及混合概率计算机制.

2.1 混合概率数

概率脉冲串的每个高低电平都可以认为是一个离散的随机变量X生成的,其概率分布为$P(X=1)=p$, $P(X=0)=1-p$,其中p是概率值. 根据期望的定义,X的期望为

$$E[X]=1\cdot p+0\cdot(1-p)=p \tag{4}$$

在传统概率数中X仅取1或0,所以产生的是单比

特的概率脉冲串. 也就是,概率脉冲所代表的数值取决于生成脉冲序列的离散随机变量的期望值. 而且,离散的随机变量不必被限制在0或1,可以有更大的取值范围,因此,概率数也可以扩展为使用多比特概率脉冲,而不只是单比特概率脉冲串[36,37].

根据上述讨论,构造一种新的概率数表示方式,即混合概率脉冲串. 它使用多个并行的单比特概率脉冲串按照二进制权重组成混合概率脉冲串来表示数. 由此,多比特并行的混合脉冲串组合为一个二进制数序列,其期望值(E)就是其所表示的数值. 理论上,二进制数序列所代表的期望值(E)应等于无限长序列的平均值,但实际应用中则与传统概率计算相同,使用有限长序列的平均值($\hat{p}$)作为期望的估计值. 混合概率数的数学表示定义如下:

$$E[A]=\hat{p}=\frac{\sum_{j=0}^{L-1}\left(\sum_{i=0}^{m-1}A_i2^i\right)_j}{L} \tag{5}$$

混合脉冲串中,脉冲长度用L表示,A_i表示单比特脉冲串的数,m表示混合脉冲的并行度. 根据式(5),一个混合概率脉冲数可以表示为$\begin{cases}a_1=1101 & 1111 & 1101 & 0011\\ a_0=1011 & 0100 & 1001 & 1111\end{cases}$,

对于混合串 $p=a_1\times2^1+a_0\times2^0=$ 3213 2322 3203 1133,因此,

$$\hat{p}=\frac{\sum_{j=0}^{L-1}\left(\sum_{i=0}^{m-1}A_i2^i\right)_j}{L}=\frac{3+2+1+3+\cdots1+1+3+3}{16}=\frac{34}{16}.$$

对于单比特概率脉冲串,混合概率数的并行度$m=1$,其脉冲长度为L的$\hat{p}=\left(\sum_{j=0}^{L-1}A_j\right)\Big/L$,因此,传统概率脉冲串也是并行度$m=1$的混合脉冲串的特例;对于二进制数,即式(5)中取$L=1$且$m=n$($n$是比特位宽),该式表示基-2数的$p=\sum_{i=0}^{m-1}A_i\cdot2^i$表示,即二进制数是脉冲串长度$L=1$的混合概率脉冲串的另一种特例,如图2所示. 据此提出,单比特脉冲概率数和二进制数分别是混合概率数的两种特例. 任何二进制数的计算电路或系统,原则上也都适用混合概率数脉冲串形式的运算,被称作混合概率计算. 混合概率脉冲使用多比特脉冲串并行来表示,且并行的比特串之间遵循二进制数的位置数规则. 因此,多比特脉冲序列所需携带的信息量,或者说数值表示精度得到大幅提升. 反之,对于相同精度表示时脉冲串的长度将大幅缩短,有效解决了传统单比特概率串时延过大的问题.

综上可知,二进制数、单比特概率数和混合概率数具有相关性和一致性. 也可以认为,二进制数是单位脉冲长度的压缩编码数;而概率脉冲数是一种脉冲长度为L的非压缩编码数,混合概率脉冲数则是基于上述两者之间的压缩编码数. 若比较概率脉冲数、混合概率数以及二进制数,三种数据格式的信息携带能力,二进制最高,概率脉冲串最低,混合概率数介于两者之间,随着并行度m的提高,混合概率数的信息携带效率相比概率脉冲串将得到指数级的增长,并逐渐逼近二进制.

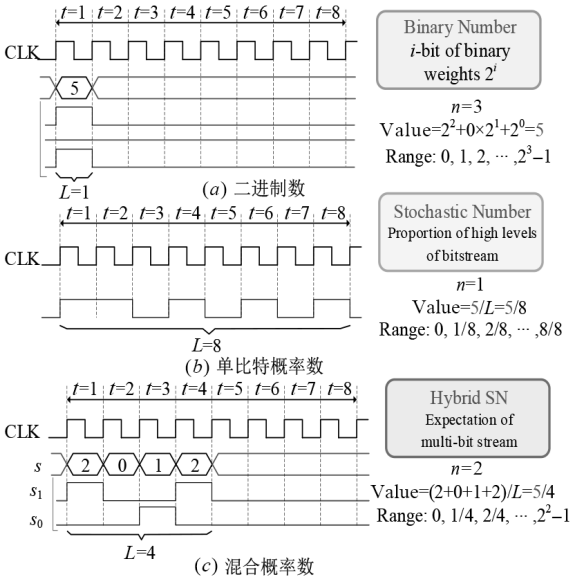


图2　不同数的表示

2.2　混合概率数电路

对于m-bit混合概率数的一般构造方式,可分为4个步骤.

步骤1　数据拆分:将原始二进制数按基-2位置数关系分解为m组位数,若设待编码数据为x,分解后数值为$y_1,y_2,\cdots,y_m$,则它们的关系可表述为$x=\sum_{i=0}^{m-1}y_i\cdot2^i$.

步骤2　并行概率脉冲编码:将$y_1,y_2,\cdots,y_m$分别与同一概率分布的随机数序列进行比较,根据比较结果并行地产生m组脉冲. 用A_i表示由y_i编码所得概率脉冲,A_i的电平值仍为0或1,$i=1,2,\cdots,m$. 若随机数在$0\sim2^n-1$上均匀分布,则A_i所表示的数值为$p_i=y_i/2^n$.

步骤3　加权组合:m组脉冲序列以基-2的权重组合后的混合串序列用s表示$\hat{E}=\left(\sum_{j=0}^{L-1}\left(\sum_{i=0}^{m-1}A_i2^i\right)_j\right)\Big/L$.

步骤4　混合概率脉冲的期望,对于混合概率串$s=\sum_{i=0}^{m-1}2^{i-1}s_i$,它的期望值为$E[S]=\sum_{i=0}^{m-1}2^{i-1}p_i=\sum_{i=0}^{m-1}\left(2^{i-1}y_i/2^n\right)=x/2^n$.

将6-bit二进制数编码为3-bit混合串的具体实例如图3所示. 为便于描述,用$x_0\sim x_5$分别表示待编码二进

制数x的最低位到最高位. 首先,将原始二进制数拆分为3个二进制数y_2,y_1,y_0. 根据混合概率数定义,拆分后的3个数应满足$x=y_2 2^2+y_1 2^1+y_0 2^0$. 拆分的目的是3个概率脉冲也满足基-2的位置数加权关系. 其次,根据拆分后二进制数位宽的最大值,选择了4-bit的随机数序列,用于y_0~y_2的概率脉冲编码. 最后,将所产生的3个概率脉冲串需按照二进制数的权重组合,从而得到携带二进制数信息的3-bit混合概率数,且不需要完成加权和的转换运算. 综上所述,混合脉冲数电路的硬件资源只包含一个RNG和3个比较电路. 因此,混合概率数编码的硬件开销比单比特概率脉冲电路在相同计算强度下具有更大优势.

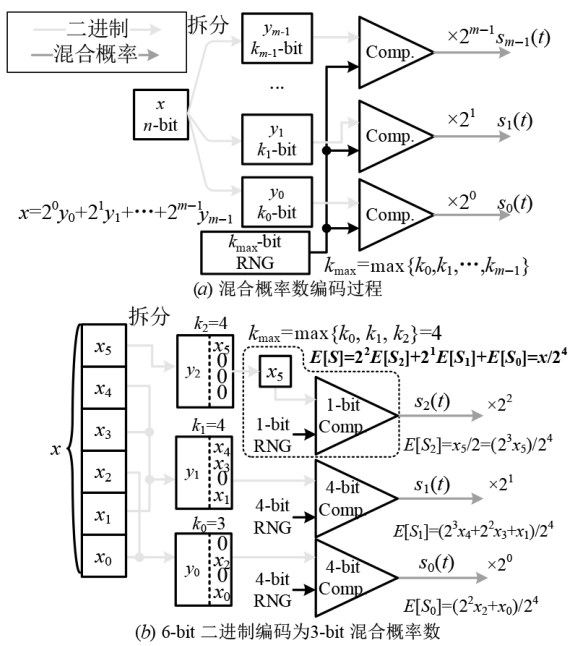


图3 混合概率数的编码电路

因此,所得混合串的期望值与原数据之间仅相差$1/2^n$的常系数,与传统概率计算类似,该系数的大小取决于使用随机数序列的取值范围. 图3(a)是完成上述操作的编码电路的一般结构,一个需要注意的细节是,使用的随机数发生器(RNG,n-bit RNG可生成在0~2^n-1范围上均匀分布的随机数序列)的位宽应不小于任何一个拆分后的数值y_i,这源于概率脉冲数的要求,如不满足该条件,所得概率脉冲的概率值会产生偏差. 另一方面,每个概率脉冲可以分别使用不同的RNG来生成,但要求它们的取值范围是相同的,也就是RNG的位宽需相同.

3 混合概率算术单元

3.1 乘法

混合概率的乘法运算的实现方法:令2个混合脉冲串处于同一时刻t的值逐一对应相乘,乘积结果按时间依次输出并组合成新的脉冲串,其数学表达式如下:

$$s_c(t)=s_a(t)s_b(t) \tag{6}$$

根据概率统计的理论,如果2个输入混合串间相互独立,则混合串输入与输出的期望值关系如下:

$$E\left[S_c\right]=E\left[S_a S_b\right]=E\left[S_a\right]E\left[S_b\right] \tag{7}$$

混合概率计算具体实例(11×16=176)如图4所示. 在传统确定性概率计算方法中,用概率脉冲准确地表示数值16或11,需要脉冲长$L=16$的概率脉冲串,而为了获得更精确的16与11脉冲串的乘积结果,则需要长为16×16=256的脉冲序列作为乘法输出.

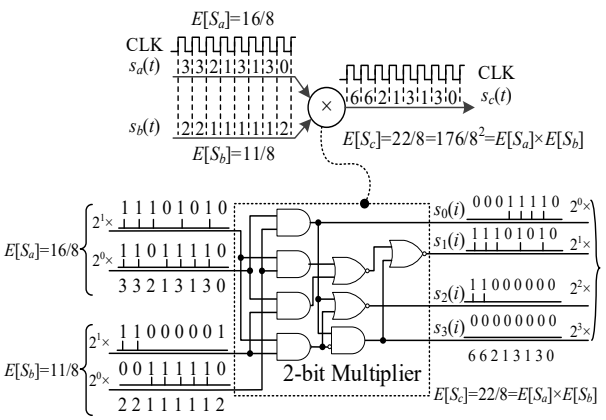


图4 混合概率的乘法运算

在实现相同计算精度的前提下,混合串则能够指数级地缩短脉冲串长度. 图4中展示为2-bit混合串,根据式(5),对于脉冲串长度$L=5$的2-bit混合串,即可准确表示输入数据16和11,脉冲串长度仅为单比特概率脉冲串的31.25%. 而为了得到足够精确的运算结果,脉冲串长度最多仅需要$5^2=25$,传统概率脉冲串所需序列长度是其10.24倍,图4中展示当长度仅为8就可以得到其精确的结果.

尽管混合串的运算带来了速度上的提升,但这一提升是以增加一定的硬件开销为代价. 对于2-bit混合串的乘法电路,由传统概率乘法的1个与门增加到8个逻辑门,从能效比角度考虑,性能提升并不算很明显. 为了有效提升能效,可以考虑另一种乘法方式,即m-bit混合串与概率脉冲的乘法. 如前所述,概率脉冲串是并行度为$m=1$的混合串,因此,两者的乘法依然只需保证独立性即可实现. 其优势在于,(1-bit)×(m-bit)的乘法器只需要m个逻辑与门即可,硬件开销随着m的提高仅仅线性增长,而脉冲串长度则是随m的提高,呈指数级缩短,因此随着m的提升,该方式可以获得指数级的能效提升.

例如计算二进制数16×16,把原始输入数16编码为概率脉冲串,长度$L=16$;另一个数16编码为2-bit混合串,长度$L=5$. 乘积混合串的长度最多仅需16 ×5=80,硬件开销仅为两个与门. 若使用传统概率脉冲串所

需序列长度 $L=16^2=256$,是混合概率的3.2倍. 因此,所提出混合概率计算的能效相比概率计算的乘法提升了60%,其计算电路如图5所示,由于序列长度较大时不便于绘制,这里依然取一种理想情况,脉冲串长度仅为8就得到了精确计算结果.

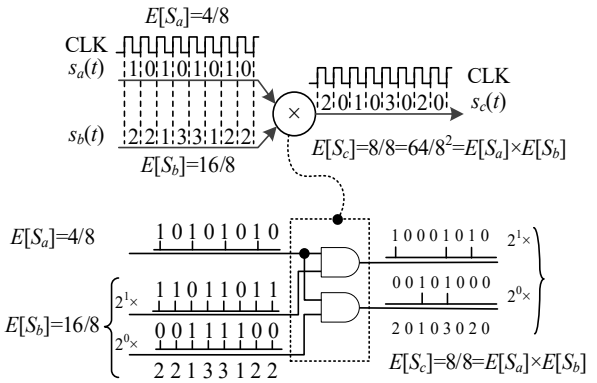


图5　混合概率脉冲串的高能效乘法

3.2　加减法

混合概率脉冲串的加法和减法运算的实现方法:令2个混合串处于同一时刻 t 的值逐一对应相加或相减,然后将运算结果作为该时刻的混合串序列输出,其数学表达式描述如下:

$$s_c(t)=s_a(t)\pm s_b(t) \tag{8}$$

根据概率统计的理论,输入与输出混合概率数的期望值存在如下关系:

$$E\left[S_c\right]=E\left[S_a\pm S_b\right]=E\left[S_a\right]\pm E\left[S_b\right] \tag{9}$$

混合串加法与减法的具体实例如图6所示. 混合串的加减法不需要考虑饱和的问题,因为其取值范围可通过增加并行度 m 来扩展,这使得加法的实现变得十分简单. 另一方面,与混合串乘法不同的是,加减法运算的成立不要求输入混合串满足独立性条件,并且不会产生计算误差.

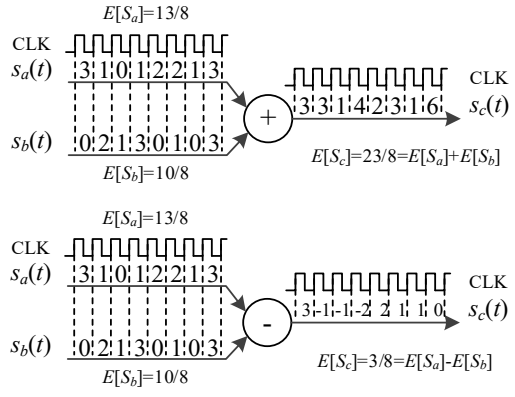


图6　混合概率脉冲串的加减法

3.3　除法

混合串的除法与概率脉冲串的除法原理基本一致,采用的同样是逐次逼近反馈结构,将其转变成乘法的逆运算. 反馈结构的功能是动态地调整商混合串的输出值大小,使得商混合串与除数混合串的乘积的期望等于被除数脉冲所代表的概率值,具体算法如下:

(1)用RNG随机地产生商混合串的第一个值;

(2)将商混合串与除数混合串相乘;

(3)用计数器累加被除数混合串,同时累减乘积混合串,计算乘积混合串与被除数混合串期望的差:

$$\mathrm{Num}_{\mathrm{count}}=\sum_{t=1}^{L}\left[s_a(t)-s_c(t)s_b(t)\right] \tag{10}$$

(4)根据计数器的当前数值决定商混合串的输出值,若计数器为正(表示被除数的期望值更大),则输出值等于RNG的值,使得商的期望值变大,进而使得乘积混合串的期望值变大,若计数器不为正(表示乘积的期望值更大)则输出0,避免差距进一步扩大.

重复进行第(2)~(4)步,直至将被除数和除数脉冲全部处理完毕,即完成了混合串的除法运算. 除法运算的电路结构及信号波形示例如图7所示.

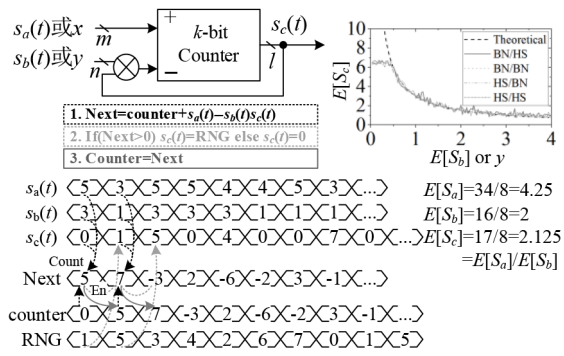


图7　混合串的除法运算

当 $\mathrm{Num}_{\mathrm{count}}$ 趋近于零时,表示 $\sum_{t=1}^{L}\left[s_a(t)-s_c(t)s_b(t)\right]=0$,于是可得

$$E\left[S_a\right]=\lim_{L\to\infty}\frac{\sum_{t=1}^{L}s_a(t)}{L}=\lim_{L\to\infty}\frac{\sum_{t=1}^{L}s_b(t)s_c(t)}{L}=E\left[S_bS_c\right] \tag{11}$$

若商混合串与除数混合串相互独立,则有 $E\left[S_bS_c\right]=E\left[S_b\right]E\left[S_c\right]$,如此就实现了除法运算. 这里的商混合串之所以取值为0或者RNG,正是为了保证商混合串与除数混合串间存在独立性,防止乘积混合串的期望值出现较大误差,进而影响除法运算的精度. 由于除法的实现是基于乘法的,若将除数混合串替换为二进制数,则在精度上同样能获得提升,因为避免了除数混合串与商混合串间独立性的影响.

4 混合概率计算容错性

概率脉冲串的主要优点是其容错能力高[11,13,38]，混合概率脉冲串同样具备这一特点. 先从数学角度解释概率脉冲以及混合概率计算的容错能力.

在噪声的环境中，概率脉冲串的每一个比特都将有一定的概率发生翻转，这被称作位翻转率(bit flip rate)，用a表示. 由于噪声的均值为0，即便当噪声的幅度极大时，也最多只有约50%的情况会导致位翻转. 因此，概率计算中位翻转率的取值范围是[0，0.5]，如图8所示.

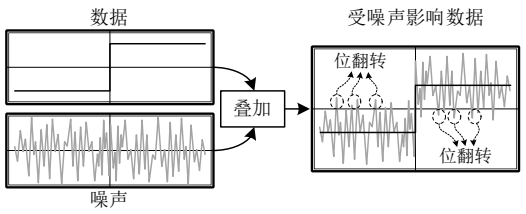


图8 噪声导致位翻转的示意图

假设$x(t)$为离散随机变量X生成的概率脉冲，$X=1$的概率$p\in[0,0.5]$，$X=0$的概率为$(1-p)$，$E[X]=1\cdot p+0\cdot(1-p)=p$. X_{noise}表示受噪声影响的X，受噪声影响可能发生的4种情况列举如下：

(1) $X=1, X_{\text{noise}}=1: p_1=p(1-a)$;

(2) $X=1, X_{\text{noise}}=0: p_2=pa$;

(3) $X=0, X_{\text{noise}}=1: p_3=(1-p)a$;

(4) $X=0, X_{\text{noise}}=0: p_3=(1-p)(1-a)$.

由此，可得X_{noise}的期望为

$$E[X_{\text{noise}}]=p_1+p_3+0\cdot(p_2+p_4)=(1-2a)E[X]+a \tag{12}$$

式(12)表明$E[X]$和$E[X_{\text{noise}}]$之间存在一个与位翻转率a有关的确定的线性函数关系. 因此，概率脉冲受噪声影响后的期望值与之前是一种线性映射，依然完整保留原始数据信息，正是这一特点使得概率脉冲具有极高的容错能力. 只有当位翻转率恰等于50%时，才会导致原始数据的信息丢失.

由于m-bit混合概率计算是由并行的m组概率脉冲串组成，式(12)在混合概率计算中同样成立. 设m组概率脉冲的期望$E[S_i]=p_i(i=1,2,\cdots,m)$，则混合概率串的期望值为

$$E[S]=\sum_{i=0}^{m-1}(2^{i-1}E[S_i])=\sum_{i=0}^{m-1}(2^{i-1}p_i) \tag{13}$$

受噪声影响后，各概率脉冲串的期望值变为

$$E[S_{i,\text{noise}}]=(1-2a)p_i+a \tag{14}$$

于是，受噪声影响的混合概率脉冲串的期望值为

$$\begin{aligned}E[S_{\text{noise}}]&=\sum_{i=0}^{m-1}(2^{i-1}E[S_{i,\text{noise}}])\\&=(1-2a)\sum_{i=0}^{m-1}(2^{i-1}p_i)+a(2^{m-1}+2^{m-2}+\cdots+2^0)\\&=(1-2a)E[S]+a(2^m-1)\end{aligned} \tag{15}$$

故位翻转错误也只会导致混合脉冲串的期望值产生线性变化. 综合上述数学推导可知，混合概率脉冲与概率脉冲都能够在受到噪声影响后，依然保留原始信息. 由于噪声产生的影响是线性的，因此噪声前与噪声后的数据是一一映射的，噪声前不同的数据，噪声后依然不同，且大小关系不变. 这一特性在图像处理中优势极为突出，虽然像素值产生了线性偏移，但依然能够肉眼分辨原始图像内容. 此外，如果能够估算翻转率的大小，则能够从其中恢复出原始数据.

5 混合概率计算实现

5.1 复杂函数的计算方法

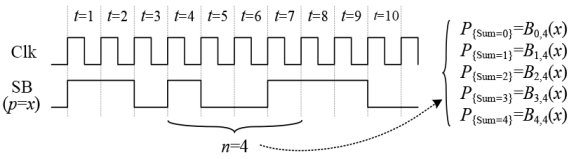
伯恩斯坦多项式(bernstein polynomial)是一种复杂函数拟合计算方法[63]，它是伯恩斯坦基多项式的线性组合，n阶伯恩斯坦多项式的表达式如下：

$$B(x)=\sum_{j=0}^{n}b_jB_{j,n}(x)=\sum_{j=0}^{n}b_j\binom{n}{j}x^j(1-x)^{n-j} \tag{16}$$

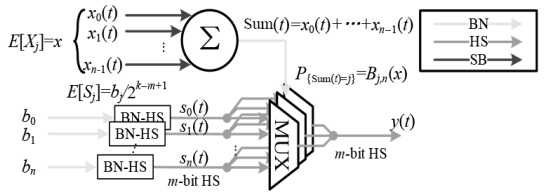
其中，$B_{j,n}(x)=\binom{n}{j}x^j(1-x)^{n-j}$为伯恩斯坦基多项式；$b_j$称作伯恩斯坦系数. 基多项式的函数运算，若使用二进制逻辑实现，运算复杂度较高，但在概率计算中却能够得到极大简化，这一化简不同于四则运算的低成本电路，而是源于概率脉冲序列中“1”数量的概率分布函数与伯恩斯坦基多项式一致. 考虑一个单极性概率脉冲(SB)，其生成“1”的概率为x，从任意时刻t起，取一段长度为n的脉冲序列$\text{SB}(t),\text{SB}(t+1),\cdots,\text{SB}(t+n-1)$，由于$n$长的序列中“1”的总数Sum，满足二项分布，于是有$P_{\{\text{Sum}=j\}}=\binom{n}{j}x^j(1-x)^{n-j}=B_{j,n}(x), j=0,1,\cdots,n$. 因此，重复多次统计概率值为$x$的概率脉冲中任意不同的$n$长序列中“1”的总数，即可获得基多项式的函数运算结果，如图9(a)所示.

利用上述方式，使用概率脉冲可以低成本地实现伯恩斯坦基多项式的计算，进而实现函数拟合功能[14]. 首先，并行生成n组概率值均为x的概率脉冲$x_0(t)$~$x_{n-1}(t)$，各时钟周期t下的n个并行的脉冲相加得到$\text{Sum}(t)$(n组脉冲的加权均为1，并非组合为混合串)；其次，利用“饱和加法”的思想，将$\text{Sum}(t)$作为选择信号，将n个伯恩斯坦系数编码为概率脉冲，并作为选择器

(MUX)输入;最后,经由 Sum(t)选择而产生的概率脉冲 $y(t)$即为基多项式与系数的乘累加运算结果,其运算电路如图9(b)所示. 此外,若将伯恩斯坦系数编码为混合串时,编码效率与精度会得到提升,此时输出$y(t)$也转变为混合串格式. 由于编码效率与精度得到提升,在执行Sum(t)与伯恩斯坦系数的乘累加运算时,混合串在准确性和延迟方面的优势会带来运算效率的指数级提升.



(a) 利用“1”数量的概率分布求伯恩斯坦基多项式



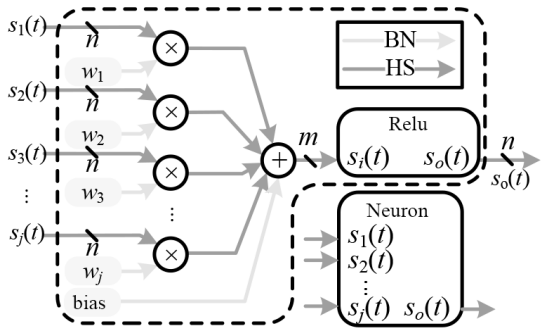
(b) 伯恩斯坦多项式运算电路

图9　基于混合概率数的伯恩斯坦多项式运算方法

此外,利用另一种方式实现混合概率数复杂函数的计算. 考虑以混合概率脉冲串输出的部分和与目标函数值的差值,以反馈的方式决定下一时刻的混合串输出,采用逐次逼近的思想,使得输出混合串逐渐接近目标函数值. 这里,对于混合串的部分和的函数计算可采用查找表(lookup table)的方式实现任意的函数拟合功能. 当然,若函数复杂度较低,也可以替换为具体的函数计算电路,例如神经网络中常见的Relu函数,则可以用比较器和数据选择器来替换查找表,还可以结合截位计算的思想,灵活地调整输出混合串的位宽,这种针对复杂函数的计算方法被应用到神经网络加速器的设计中,并展现出较高的能量效率.

5.2　混合概率神经网络加速器

在传统概率计算中,数据必须在概率脉冲和二进制之间反复转换,以避免相关性对计算的影响. 这导致电路难以实现深度神经网络的计算,并且数据转换还占用大量处理时间. 本文提出的混合概率计算解决了该问题,一方面,改进的混合概率计算将二进制视为各时钟周期数值相同的常数混合串,使得二进制数得以直接参与混合概率计算;另一方面,常数混合串参与的乘法运算不再受期望值乘法运算规则对于输入数据互不相关性的约束,也就不再需要频繁地在二进制数-概率数之间转换以消除相关性的影响,节省了转换的时间和大量的硬件开销. 基于混合概率计算设计的神经元电路如图10所示,其中,深度学习的*Relu*函数由带有截位功能的逐次逼近法实现. 权值和偏置数据以二进制编码直接参与计算,输入和输出均为n-bit的混合概率脉冲.



$E[S_o]=2^{-(n-m)}\mathrm{Relu}\{w_1\cdot E[S_1]+w_2\cdot E[S_2]+\ldots+w_j\cdot E[S_j]+\mathrm{bias}\}$

图10　混合概率计算的神经元结构

利用输入和输出混合串具有相同的位宽,可以实现任意个神经元的级联,便于网络结构的搭建,且可以形成深度学习的流水结构. 利用图10所示的神经元结构,搭建出一个5层的深度学习的神经网络,实现非线性可分的双月模型的二分类任务,其深度神经网络的结构如图11所示. 使用神经网络训练框架(Pytorch)进行网络参数训练;其次对网络的权重和偏置,包括输入x和y,由Matlab实现8比特二进制数量化以便送入神经网络进行推理计算.

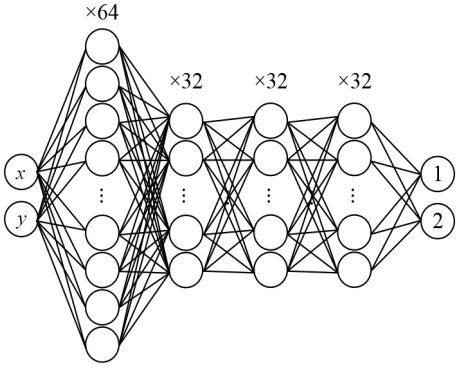


图11　5层全连接概率神经网络结构

该二分类任务的分类边界在图12中用虚线表示. 而使用不同脉冲串长度(L)的混合概率计算所得分类边界则被绘制为不同颜色的实线. 当$L=2^6$时,混合概率计算结果和原网络结果之间的差异仍然很小;当L减小到2^5时,两者所得分类边界开始出现较大差异,但基于混合概率计算所得到的边界依然可以很好地区分这两组点集.

该款基于混合概率计算实现的MLP神经网络芯片,采用TSMC 40 nm工艺完成流片,芯片的显微照片如图13(a)所示,其中计算核心的工作电压为1.1 V,内核面积为0.73 mm × 0.73 mm,包含4 544个乘法器和加

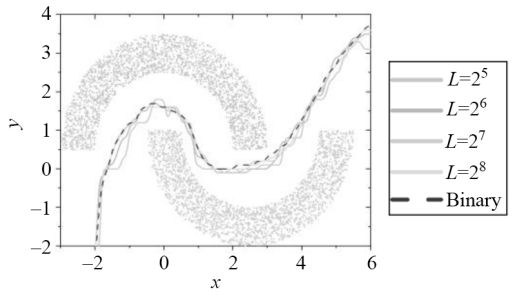
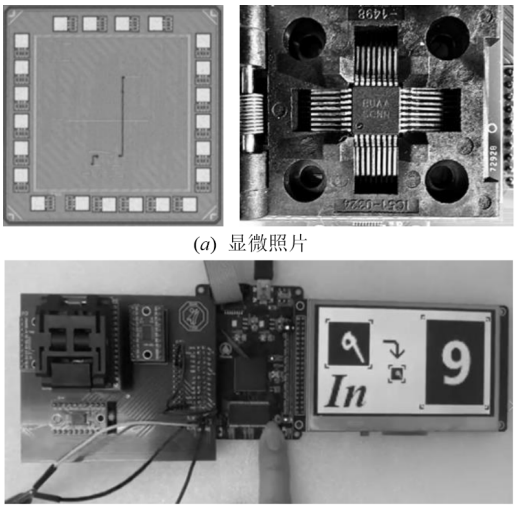


图12 混合概率计算实现的神经网络分类结果



(*a*) 显微照片

(*b*) 演示系统

图13 混合概率计算机制神经网络芯片

法器. 该混合概率计算芯片的推理系统如图13(*b*)所示,系统由MNIST数据集在Pytorch框架下训练,推理芯片能准确快速识别出0~9手写体数字. 所研制的混合概率脉冲芯片在400 MHz时钟频率下功耗为102.3 mW,而其中动态计算功耗仅为0.097 mW,网络参数存储的静态功耗为102.25 mW,静态功耗占99.9%. 高静态功耗是由于全连接神经网络的权重寄存器数量大而导致的较高的能量消耗.

混合概率芯片的能量效率方面,该芯片每消耗一焦耳能量可执行3.553×10^{13}次混合计算操作(乘法与加法各算一次操作),即35.53T SOPS/W(Stochastic Operation Per Second,SOPS);若脉冲序列长度为2^5,该芯片每消耗一焦耳能量可执行1.11×10^{12}次二进制运算操作,即1.1T OPS/W(Operation Per Second,OPS). 其能量效率可达传统概率脉冲串的50倍以上,与二进制数的神经网络的能量效率基本相当,具体各方面的参数指标的数据对比可参见表2.

表2 概率计算、二进制和混合概率计算对比

	传统概率数						二进制数			混合概率数	
	2019[64]	2021[65]	2022[66]	2022[67]	2017[62]	2022[15]	2017[68]	2020[69]	2022[70]	本文方法	
技术工艺	ASIC 45 nm	ASIC 45 nm	FPGA StratV	ASIC 45 nm	ASIC 65 nm	ASIC 28 nm	FPGA GX1150	ASIC 16 nm	FPGA GX1150	FPGA KCU116	ASIC 40 nm
输入数据位宽	8-bit	7-bit	8-bit	8-bit	8-bit	8-bit	16-bit	8-bit	8-bit	8-bit	8-bit
核心技术	确定性方法	可扩展概率逻辑	径向基函数概率逻辑	全并行概率计算	概率与二进制数双用网络	概率与二进制数双用网络	数据调度算法	共地参考多芯片组	全栈加速	概率-二进制混合数	概率-二进制混合数
MAC数	25	234 752	—	—	—	256	—	1 024 /cycle	—	4 544	4 544
功耗/mW	1.28	208	5 900	651	33.17	6.09	37 460	4 160	19 100	1 592	102.3
时钟频率/MHz	1 136	200	50	200	模拟电路	500	385	161-2001	200	200	400
概率数生成时延	2048	1 024	—	512	8	1.04 (均值)	—	—	—	32	32
网络结构	LeNet-5	MLP	RBF-NN	LeNet-5	LeNet-5	VGG16	VGG	ResNet-50	Re11et-50	MLP	
算力/GOPS	0.028	91.7	38.55	220	—	246	1 790	4 098	1 519	56.8	113.6
能效/(GOPS/W)	22	440.9	6.53	340	1 039	40.39	47.8	985.1	79.5	35.7	1 109.8

对比表2展示了二进制数、传统概率数和混合概率数等三类计算的实验结果,实验方法分别包括FPGA和ASIC两种手段. 神经网络结构分别包括LeNet,VGG,ResNet,MLP和RBF-NN等多种形式,其核心技术包括算法层面的确定性方法、ESL、数据调度算法和混合概率数等,网络结构则包括全并行概率计算、概率与二进制数双用网络等方法. 受限于传统概率计算的大时延的制约,传统概率神经网络的算力、能效比都不理想,而采用二进制数的网络算力较高,但能效比一般. 利用本文提出的混合概率数的神经网络算力体现了较好的结果,受限芯片面积MAC数量和MLP网络结构制约了算力的上限;另一方面,本方法的芯片能效比在静态功耗占99.9%条件下已达1.1 TOPS/W,如仅计算峰值动态功耗的能效比则可达1 170 TOPS/W. 后期在改进网络模型、电路结构和工艺的条件下会有更高性能的跃升.

6　总结

随着大数据、人工智能对大算力、高能效比的高度依赖,后摩尔时代迫切需要新计算机制的边端芯片支撑高能效比算力的需求. 本文介绍了目前概率计算的研究现状和发展趋势,并基于传统概率计算的研究基础,首次提出了混合概率数的数的表示机制以及混合概率计算,阐述了二进制数、概率数和混合概率数的相互关系. 在混合概率计算中,三种数可以在混合概率计算中同时使用,避免了传统概率计算不同数互相转换的资源开销. 由于二进制数在混合概率计算中是混合概率数的特例——常数,且二进制数和两类概率数之间计算时无独立性的要求,从而保证了更高的计算效率. 此外,讨论了混合概率计算的噪声容错问题,分析了混合概率数的高噪声容错效果. 最后,在基础算术单元、复杂计算函数和概率脉冲神经网络等多个领域进行电路和系统级设计实现. 其中的混合概率神经网络采用40 nm CMOS工艺制造,其能量效率比是传统概率计算的50倍以上. 实验结果表明,所提出新的混合概率计算方法比传统概率计算性能更好,避免了长计算时延、低计算精度和较高复杂度等主要物理缺陷. 实验表明,在充分发挥混合概率数的计算优势下,未来的边缘计算或类脑计算中将会展现出独特的技术性能和应用前景.

参考文献

[1] VON NEUMANN J. Probabilistic Logics and the synthesis of reliable organisms from unreliable components[M]//Automata Studies (AM-34). Princeton: Princeton University Press, 1956: 43-98.

[2] GAINES B R. Stochastic computing[C]//Proceedings of the Spring Joint Computer Conference on - AFIPS'67. New York: ACM, 1967: 149-156.

[3] POPPELBAUM W J, AFUSO C, ESCH J W. Stochastic computing elements and systems[C]//Proceedings of the Fall Joint Computer Conference. New York: ACM, 1967: 635-644.

[4] GAINES B R. Stochastic computing systems[M]//TOU JT. Advances in Information Systems Science. Boston, MA: Springer, 1969: 37-172.

[5] KONDO Y, SAWADA Y. Functional abilities of a stochastic logic neural network[J]. IEEE Transactions on Neural Networks, 1992, 3(3): 434-443.

[6] ALAGHI A, HAYES J P. Survey of stochastic computing [J]. ACM Transactions on Embedded Computing Systems, 2013, 12(2S): 92.

[7] HAYES J P. Introduction to stochastic computing and its challenges[C]//2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC). Piscataway: IEEE, 2015: 1-3.

[8] CANALS V, MORRO A, OLIVER A, et al. A new stochastic computing methodology for efficient neural network implementation[J]. IEEE Transactions on Neural Networks and Learning Systems, 2016, 27(3): 551-564.

[9] LIU S T, HAN J. Energy efficient stochastic computing with Sobol sequences[C]//Design, Automation & Test in Europe Conference & Exhibition (DATE). Piscataway: IEEE, 2017: 650-653.

[10] JENSON D, RIEDEL M. A deterministic approach to stochastic computation[C]//2016 IEEE/ACM International Conference on Computer-Aided Design (ICCAD). Piscataway: IEEE, 2016: 1-8.

[11] ZHANG Z D, WANG R S, ZHANG Z, et al. Circuit reliability comparison between stochastic computing and binary computing[J]. IEEE Transactions on Circuits and Systems Ⅱ: Express Briefs, 2020, 67(12): 3342-3346.

[12] HU Y X, ZHANG Y W, WANG R S, et al. A 28-nm 198.9-TOPS/W fault-tolerant stochastic computing neural network processor[J]. IEEE Solid-State Circuits Letters, 2022, 5: 198-201.

[13] QIAN W K, LI X, RIEDEL M D, et al. An architecture for fault-tolerant computation with stochastic logic[J]. IEEE Transactions on Computers, 2011, 60(1): 93-105.

[14] QIAN W K, RIEDEL M D, ROSENBERG I. Uniform approximation and Bernstein polynomials with coefficients in the unit interval[J]. European Journal of Combinatorics, 2011, 32(3): 448-463.

[15] CHEN Z Y, MA Y F, WANG Z F. Hybrid stochastic-binary computing for low-latency and high-precision infer-

ence of CNNs[J]. IEEE Transactions on Circuits and Systems I: Regular Papers, 2022, 69(7): 2707-2720.

[16] ICHIHARA H, ISHII S, SUNAMORI D, et al. Compact and accurate stochastic circuits with shared random number sources[C]//2014 IEEE 32nd International Conference on Computer Design (ICCD). Piscataway: IEEE, 2014: 361-366.

[17] XIE Y, LIAO S Y, YUAN B, et al. Fully-parallel area-efficient deep neural network design using stochastic computing[J]. IEEE Transactions on Circuits and Systems Ⅱ: Express Briefs, 2017, 64(12): 1382-1386.

[18] YANG M, LI B Z, LILJA D J, et al. Towards theoretical cost limit of stochastic number generators for stochastic computing[C]//2018 IEEE Computer Society Annual Symposium on VLSI (ISVLSI). Piscataway: IEEE, 2018: 154-159.

[19] AHMAD SALEHI S. Low-cost stochastic number generators for stochastic computing[J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2020, 28 (4): 992-1001.

[20] CHEN T H, TING P S, HAYES J P. Achieving progressive precision in stochastic computing[C]//2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP). Piscataway: IEEE, 2017: 1320-1324.

[21] KIM K, LEE J, CHOI K. An energy-efficient random number generator for stochastic circuits[C]//2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC). Piscataway: IEEE, 2016: 256-261.

[22] CHEN T H, HAYES J P. Design of division circuits for stochastic computing[C]//2016 IEEE Computer Society Annual Symposium on VLSI (ISVLSI). Piscataway: IEEE, 2016: 116-121.

[23] SIM H, NGUYEN D, LEE J, et al. Scalable stochastic-computing accelerator for convolutional neural networks [C]//2017 22nd Asia and South Pacific Design Automation Conference (ASP-DAC). Piscataway: IEEE, 2017: 696-701.

[24] BROWN B D, CARD H C. Stochastic neural computation. I. Computational elements[J]. IEEE Transactions on Computers, 2001, 50(9): 891-905.

[25] TING P S, HAYES J P. Stochastic logic realization of matrix operations[C]//2014 17th Euromicro Conference on Digital System Design. Piscataway: IEEE, 2014: 356-364.

[26] SIM H, LEE J. A new stochastic computing multiplier with application to deep convolutional neural networks [C]//2017 54th ACM/EDAC/IEEE Design Automation Conference (DAC). Piscataway: IEEE, 2017: 1-6.

[27] KIM K, KIM J, YU J, et al. Dynamic energy-accuracy trade-off using stochastic computing in deep neural networks[C]//2016 53nd ACM/EDAC/IEEE Design Automation Conference (DAC). Piscataway: IEEE, 2016: 1-6.

[28] HOJABR R, GIVAKI K, TAYARANIAN S R, et al. SkippyNN: an embedded stochastic-computing accelerator for convolutional neural networks[C]//2019 56th ACM/IEEE Design Automation Conference (DAC). Piscataway: IEEE, 2019: 1-6.

[29] CHU S I. New divider design for stochastic computing[J]. IEEE Transactions on Circuits and Systems Ⅱ: Express Briefs, 2020, 67(1): 147-151.

[30] ARDAKANI A, LEDUC-PRIMEAU F, ONIZAWA N, et al. VLSI implementation of deep neural network using integral stochastic computing[J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2017, 25 (10): 2688-2699.

[31] ZHAKATAYEV A, LEE S, SIM H, et al. Sign-magnitude SC: Getting 10X accuracy for free in stochastic computing for deep neural networks[C]//2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC). Piscataway: IEEE, 2018: 1-6.

[32] ALAGHI A, HAYES J P. Fast and accurate computation using stochastic circuits[C]//2014 Design, Automation & Test in Europe Conference & Exhibition (DATE). Piscataway: IEEE, 2014: 1-4.

[33] LIU S T, HAN J. Toward energy-efficient stochastic circuits using parallel sobol sequences[J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2018, 26(7): 1326-1339.

[34] LI B Z, NAJAFI M H, YUAN B, et al. Quantized neural networks with new stochastic multipliers[C]//2018 19th International Symposium on Quality Electronic Design (ISQED). Piscataway: IEEE, 2018: 376-382.

[35] GROSS W J, GAUDET V C, MILNER A. Stochastic implementation of LDPC decoders[C]//Conference Record of the Thirty-Ninth Asilomar Conference onSignals, Systems and Computers. Piscataway: IEEE, 2006: 713-717.

[36] CHEN Y H, LI H G. Stochastic computing using amplitude and frequency encoding[J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2022, 30 (5): 656-660.

[37] LI H G, CHEN Y H. Hybrid logic computing of binary and stochastic[J]. IEEE Embedded Systems Letters, 2022, 14(4): 171-174.

[38] LIU Y D, LIU L B, LOMBARDI F, et al. An energy-efficient and noise-tolerant recurrent neural network using stochastic computing[J]. IEEE Transactions on Very

Large Scale Integration (VLSI) Systems, 2019, 27(9): 2213-2221.

[39] LI Z, LI J, REN A, et al. HEIF: Highly efficient stochastic computing-based inference framework for deep neural networks[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2019, 38(8): 1543-1556.

[40] LIU Y D, WANG Y Z, LOMBARDI F, et al. An energy-efficient stochastic computational deep belief network [C]//2018 Design, Automation & Test in Europe Conference & Exhibition (DATE). Piscataway: IEEE, 2018: 1175-1178.

[41] LI J, REN A, LI Z, et al. Towards acceleration of deep convolutional neural networks using stochastic computing [C]//2017 22nd Asia and South Pacific Design Automation Conference (ASP-DAC). Piscataway: IEEE, 2017: 115-120.

[42] BROWN B D, CARD H C. Stochastic neural computation Ⅱ Soft competitive learning[J]. IEEE Transactions on Computers, 2001, 50(9): 906-920.

[43] LIU Y D, LIU S T, WANG Y Z, et al. A stochastic computational multi-layer perceptron with backward propagation[J]. IEEE Transactions on Computers, 2018, 67(9): 1273-1286.

[44] REN A, LI Z, WANG Y Z, et al. Designing reconfigurable large-scale deep learning systems using stochastic computing[C]//2016 IEEE International Conference on Rebooting Computing (ICRC). Piscataway: IEEE, 2016: 1-7.

[45] REN A, LI J, LI Z, et al. SC-DCNN: Highly-scalable deep convolutional neural network using stochastic computing[C]//Proceedings of the Twenty-Second International Conference on Architectural Support for Programming Languages and Operating Systems. New York: ACM Press, 2017: 405-418.

[46] YU J, KIM K, LEE J, et al. Accurate and efficient stochastic computing hardware for convolutional neural networks[C]//2017 IEEE International Conference on Computer Design (ICCD). Piscataway: IEEE, 2017: 105-112.

[47] LI J, YUAN Z H, LI Z, et al. Normalization and dropout for stochastic computing-based deep convolutional neural networks[J]. Integration, the VLSI Journal, 2019, 65: 395-403.

[48] SIMARD P Y, STEINKRAUS D, PLATT J C. Best practices for convolutional neural networks applied to visual document analysis[C]//2003 Proceedings of Seventh International Conference on Document Analysis and Recognition. Piscataway: IEEE, 2003: 958-963.

[49] ZHANG Y W, ZHANG X Y, SONG J H, et al. Parallel convolutional neural network (CNN) accelerators based on stochastic computing[C]//2019 IEEE International Workshop on Signal Processing Systems (SiPS). Piscataway: IEEE, 2019: 19-24.

[50] MA X L, ZHANG Y P, YUAN G, et al. An area and energy efficient design of domain-wall memory-based deep convolutional neural networks using stochastic computing[C]//2018 19th International Symposium on Quality Electronic Design (ISQED). Piscataway: IEEE, 2018: 314-321.

[51] JIA X T, YANG J L, DAI P C, et al. SPINBIS: Spintronics-based Bayesian inference system with stochastic computing[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2020, 39(4): 789-802.

[52] ALAGHI A, HAYES J. Dimension reduction in statistical simulation of digital circuits[C]//Proceedings of the Symposium on Theory of Modeling & Simulation: DEVS Integrative M&S Symposium. New York: ACM, 2015: 1-8.

[53] ICHIHARA H, SUGINO T, ISHII S, et al. Compact and accurate digital filters based on stochastic computing[J]. IEEE Transactions on Emerging Topics in Computing, 2019, 7(1): 31-43.

[54] NAJAFI M H, LILJA D J. High quality down-sampling for deterministic approaches to stochastic computing[J]. IEEE Transactions on Emerging Topics in Computing, 2021, 9(1): 7-14.

[55] LI P, LILJA D J, QIAN W K, et al. Computation on stochastic bit streams digital image processing case studies [J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2014, 22(3): 449-462.

[56] EL-DERHALLI H, LE BEUX S, TAHAR S. Design space exploration of stochastic computing architectures implemented using integrated optics[J]. IEEE Transactions on Emerging Topics in Computing, 2021, 9(4): 2158-2169.

[57] MIN S J, LEE E W, CHAE S I. A study on the stochastic computation using the ratio of one pulses and zero pulses [C]//1994 IEEE International Symposium on Circuits and Systems (ISCAS). Piscataway: IEEE, 1994: 471-474.

[58] SATO S, NEMOTO K, AKIMOTO S, et al. Implementation of a new neurochip using stochastic logic[J]. IEEE Transactions on Neural Networks, 2003, 14(5): 1122-1127.

[59] LI H G, HAYAKAWA Y, SATO S, et al. Hardware implementation of an inverse function delayed neural network using stochastic logic[J]. IEICE - Transactions on Information and Systems, 2006, E89-D(9): 2572-2578.

[60] FAIX M, LAURENT R, BESSIÈRE P, et al. Design of stochastic machines dedicated to approximate Bayesian inferences[J]. IEEE Transactions on Emerging Topics in Computing, 2019, 7(1): 60-66.
[61] LI W J, XU N Y, WANG R S, et al. Efficient compression methods for wire-spread-based stochastic computing deep neural networks[J]. IEEE Transactions on Circuits and Systems Ⅱ: Express Briefs, 2022, 69(11): 4538-4542.
[62] LEE V T, ALAGHI A, HAYES J P, et al. Energy-efficient hybrid stochastic-binary neural networks for near-sensor computing[C]//Design, Automation & Test in Europe Conference & Exhibition (DATE). Piscataway: IEEE, 2017: 13-18.
[63] LORENTZ G G. Bernstein Polynomials[M]. Second edition. Providence: merican Mathematical Society, 1986.
[64] FARAJI S R, HASSAN NAJAFI M, LI B Z, et al. Energy-efficient convolutional neural networks with deterministic bit-stream processing[C]//2019 Design, Automation & Test in Europe Conference & Exhibition (DATE). Piscataway: IEEE, 2019: 1757-1762.
[65] LIU S S, TANG X C, NIKNIA F, et al. Stochastic dividers for low latency neural networks[J]. IEEE Transactions on Circuits and Systems I: Regular Papers, 2021, 68(10): 4102-4115.
[66] MORÁN A, PARRILLA L, ROCA M, et al. Digital implementation of radial basis function neural networks based on stochastic computing[J]. IEEE Journal on Emerging and Selected Topics in Circuits and Systems, 2023, 13(1): 257-269.
[67] FRASSER C F, LINARES-SERRANO P, DE LOS DE RIOS I D, et al. Fully parallel stochastic computing hardware implementation of convolutional neural networks for edge computing applications[J]. IEEE Transactions on Neural Networks and Learning Systems, 2023, 34(12): 10408-10418.
[68] ZHANG J L, LI J. Improving the performance of OpenCL-based FPGA accelerator for convolutional neural network[C]//Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. New York: ACM, 2017: 25-34.
[69] ZIMMER B, VENKATESAN R, SHAO Y S, et al. A 0.32-128 TOPS, scalable multi-chip-module-based deep neural network inference accelerator with ground-referenced signaling in 16 nm[J]. IEEE Journal of Solid-State Circuits, 2020, 55(4): 920-932.
[70] LIU S L, FAN H X, FERIANC M, et al. Toward full-stack acceleration of deep convolutional neural networks on FPGAs[J]. IEEE Transactions on Neural Networks and Learning Systems, 2022, 33(8): 3974-3987.

作者简介

李洪革 男. 2005年获得日本东北大学信息科学研究院的工学博士学位. 2006年至2008年,在日本东北大学生物智能机器人系担任助理教授. 现任北京航空航天大学电子信息工程学院教授、博导. 主要研究方向为智能(概率)计算、显示驱动/触控芯片、芯片安全及电磁干扰等实时混合系统的信号处理及片上设计领域的交叉. 已经发表期刊和国际会议等学术论文100余篇,已授权专利15项.
E-mail: honggeli@buaa.edu.cn

陈宇昊 男. 2019年获得北京航空航天大学电子信息工程学院的学士学位. 现为北京航空航天大学电子信息工程学院博士研究生. 主要研究方向为概率计算、神经形态计算.
E-mail: zy1902608@buaa.edu.cn

吴俊毅 男. 2023年获得北京航空航天大学高等理工学院的学士学位. 现为北京航空航天大学电子信息工程学院硕士研究生. 主要研究方向为低功耗数字电路设计、概率计算.
E-mail: zy2302410@buaa.edu.cn

宋印杰 男. 2021年获得北京航空航天大学高等理工学院的学士学位. 现为北京航空航天大学电子信息工程学院博士研究生. 主要研究方向包括数字集成电路、密码芯片设计.
E-mail: yinjiesong@buaa.edu.cn

朱新宇 男. 2020年获得合肥工业大学微电子学院的硕士学位. 现为北京航空航天大学电子信息工程学院博士研究生. 主要研究方向为异构多核处理器SoC设计、神经网络专用芯片.
E-mail: zxyuser@buaa.edu.cn