

引用格式:熊海翡,贺光辉. 基于随机计算的大规模 MIMO 检测算法研究与硬件实现[J]. 微电子学与计算机,2020,37(7):36-41. [XIONG H F, HE G H. Design and implementation of stochastic computing based massive MIMO detector[J]. Microelectronics & Computer, 2020, 37(7): 36-41.]

基于随机计算的大规模 MIMO 检测算法研究与硬件实现

熊海翡, 贺光辉

(上海交通大学 电子信息与电气工程学院, 上海 200240)

摘 要: 为了减少大规模 MIMO 检测算法的复杂度以适应第五代移动通信系统的要求, 本文提出了一种基于随机计算的低复杂度线性检测算法. 随机计算把传统二进制数转化为一串 01 序列, 使得复杂的计算电路能通过简单的门逻辑实现, 从而大幅度的降低硬件资源消耗. 通过采用基于二段分解的随机计算矩阵乘法器, 检测算法的计算消耗大大降低. 此外, 我们通过 Vivado HLS 实现了基于随机计算的预处理共轭梯度算法. 仿真结果表明, 该算法在 128×8 规模的大规模 MIMO 系统, 误比特率为 10^{-6} 时, 和最优检测性能误差小于 0.2 dB; 而 FPGA 结果表示, 基于随机计算的检测算法不需要采用任何 DSP, 同时能节省 20.7% 的寄存器消耗.

关键词: 大规模 MIMO; 检测算法; 随机计算; Vivado HLS

中图分类号: TN929.1

文献标识码: A

文章编号: 1000-7180(2020)07-0036-06

Design and implementation of stochastic computing based massive MIMO detector

XIONG Hai-fei, HE Guang-hui

(School of Electronic Information and Electrical Engineering,
Shanghai Jiao Tong University, Shanghai 200240, China)

Abstract: To decrease the hardware consumption of massive MIMO systems and satisfy the requirement of the fifth-generation wireless communication system, in this paper a low-complexity linear detection algorithm based on the stochastic computing (SC) is proposed. SC transforms the traditional binary number to a sequence consisting of 0 and 1 so that the complicated computation circuit can be realized by simple gate logic, thus saves hardware cost. By adopting the proposed SC-based matrix multiplier, the hardware cost is reduced significantly. Moreover, we realized the SC-based pre-condition conjugate gradient detection algorithm through Vivado HLS. The software simulation result illustrates that the proposed SC algorithm has only 0.2 dB performance degradation compared with optimal detection at $\text{BER}=10^{-6}$ for 128×8 MIMO systems. While the FPGA results show our algorithm requires no DSP and reduced the cost of flipflops by 20.7%.

Key words: massive MIMO; detection algorithm; stochastic computing; high level synthesis

1 引言

大规模多输入多输出技术 (Massive Multiple Input Multiple Output, Massive MIMO) 由于有上

百根基站天线同时服务于数十的用户, 因此频谱效率大大提升, 已经成为了第五代移动通信系统 (5G) 的关键技术^[1]. 然而, 随着天线数越来越多, 大规模 MIMO 检测模块的计算复杂度已成为限制传输速

率的瓶颈,因此设计并实现一种复杂度低的检测器成为大规模 MIMO 研究的重点之一。

大规模 MIMO 检测算法可分为最优检测、次优检测和线性检测^[2]三种。目前,有很多近似 MMSE 算法用迭代的方法逼近检测结果以避免大规模矩阵求逆^[3-5]。这些迭代算法大多包含较多的矩阵向量乘法,随着大规模 MIMO 的规模增大,运算复杂度仍然较高。而随着摩尔定律即将走到尽头,急需新颖的替代计算的方法。

随机计算(Stochastic Computing, SC)则可以解决计算复杂度高的问题^[6]。SC 通过把传统二进制数转化为一串只包含 0 和 1 的序列,从而将原本复杂的计算单元转化为简单的门电路实现。通过采用 SC,有可能从根本上降低大规模 MIMO 检测的复杂度。然而 SC 的精度较低,为了将用在大规模 MIMO 检测中,许多单元需要得到重新设计。

本文设计了高精度 SC 的矩阵乘法模块并运用在 PCG 中。通过使用 Vivado HLS,我们得到了优化后电路的硬件消耗。仿真结果表明,在 128×8 的大规模 MIMO 系统中,PCGSC 能达到和 MMSE 相近的性能,同时能在不增加其他资源的情况下不使用 DSP。

2 大规模 MIMO 检测

2.1 系统模型

考虑一个有 B 根接收天线的基站同时服务于 U 个用户($U \ll B$, 如 $U = 8, B = 128$) 的大规模 MIMO 系统。发射端 U 个用户将原始比特流经调制、映射到平均能量为 E_s 的星座点上,然后通过无线信道传输到基站天线。接收信号可以表示为

$$y = Hx + n \quad (1)$$

式中,复向量 $x = [x_1 \ x_2 \ \cdots \ x_U]^T$ 为能量归一化的发射向量,复向量 $y = [y_1 \ y_2 \ \cdots \ y_B]^T$ 为接收向量;瑞利衰落信道 $H \in C_{B \times U}$ 的每个元素为独立同分布的标准正态分布变量。噪声 $n = [n_1 \ n_2 \ \cdots \ n_B]^T$ 为方差为 N_0 的加性高斯白噪声(additive white Gaussian noise, AWGN)。

2.2 MMSE 检测

大规模 MIMO 系统的 MMSE 检测能接近最优解,其给出(1)的解如下:

$$\tilde{x} = (H^H H + \sigma^2 I)^{-1} H^H y = A^{-1} b \quad (2)$$

式中, $\tilde{x}$ 是估计的向量输出; $b = H^H y$ 是匹配滤波器的输出; $A = H^H H + \sigma^2 I$ 是 MMSE 滤波矩阵,也被

称为 Gram 矩阵; $\sigma^2 = N_0 / E_s$ 是信噪比。为避免运算复杂度高的矩阵求逆,MMSE 检测多写成以下线性方程组的形式:

$$A \tilde{x} = b \quad (3)$$

在大规模 MIMO 中,式(3)的解可以写为:

$$\tilde{x} = \operatorname{argmin} \|b - A \tilde{x}\| \quad (4)$$

然后通过采用 NS、GS 等迭代的算法求解线性方程组的近似解。

2.3 预处理共轭梯度法

以预处理共轭梯度法(PCG)为例介绍迭代法的过程^[7]。PCG 通过拆分 Gram 矩阵并采取等效增广的变形避免 Gram 矩阵的求解,从而减小 CG 算法的复杂度;同时采用初值估计和预处理的方法来加速算法的收敛。其完整算法流程如下:

算法 1: PCG 算法

输入: $H_{m \times n}$, $y_{m \times 1}$

初值估计:

$$G_{ii} = \sum_{j=1}^n \bar{H}_{ji} \bar{H}_{ji}, D_v^i = 1 / (G_{ii} + \sigma^2),$$

$$b = H^H y, x_1 = D_v^i b$$

预处理:

$$r_1 = y - H x_1, s_1 = H^H r_1, p_1 = z_1 = D_v s_1$$

等效增广:

for $i = 1:k$

$$w = H p_i$$

$$\mu = s_i^H z_i / w^H w$$

$$x_{i+1} = x_i + \mu p_i$$

$$r_{i+1} = r_i - \mu w_i$$

$$s_{i+1} = H^H r_{i+1} - \sigma^2 x_{i+1}$$

$$z_{i+1} = D_v s_{i+1}$$

$$v = s_{i+1}^H z_{i+1} / s_{i+1}^H z_{i+1}$$

$$p_{i+1} = z_{i+1} + v p_i$$

endfor

输出: $x = x_{i+1}$

通过采用 PCG 算法,CG 的性能得到了提升,同时在迭代次数较小时($k \leq 3$)复杂度更低。

3 随机计算及其单元优化

3.1 SC 原理

SC 将 0 到 1 之间的二进制数转化为一串只包含 0 和 1 的序列,序列中 1 所占的比例即为数的大小,但是序列的长度和 1,0 的位置都是不确定的。例如,序列 $\{1,0,1,0\}, \{1,1,0,0\}, \{1,0,1,0,1,0,1,0\}$ 均可以表示数 0.5。应用 SC 后,二进制数下很复杂的运算可以由简单的电路门来实现。例如,表示数 a 和 b 的 SC 序列 A 和 B,逐比特通过与门时,根据与门性质,输出的序列 C 表示的数 c 即为 a 乘 b 。SC 的另一个优点是容错率高。在长度为 n 的随机序列中,

单比特翻转所带来的误差仅为 $1/n$;而在二进制数中如果错误发生在高位则误差很大.当然,SC 也有其局限性:一是计算精度不高;二是序列的长度随着表示数精度的提升指数性增长^[6].

SC 系统一般包括三个部分:序列生成单元(Binary to Stochastic, B2S),序列计算单元和序列转化单元(Stochastic to Binary, S2B).B2S 把数转化为 SC 序列,一般而言需要一个随机数生成器(例如线性反馈移位寄存器)和一个比较器.S2B 将 SC 序列转化为数,一般可以通过计数器实现.序列计算单元在 SC 域下通过简单的门电路完成各种计算,例如 SC 乘法器仅需要一个与门实现,而带缩放的 SC 加法器可以用多路复用器(MUX)实现,具体单元见图 1^[6].

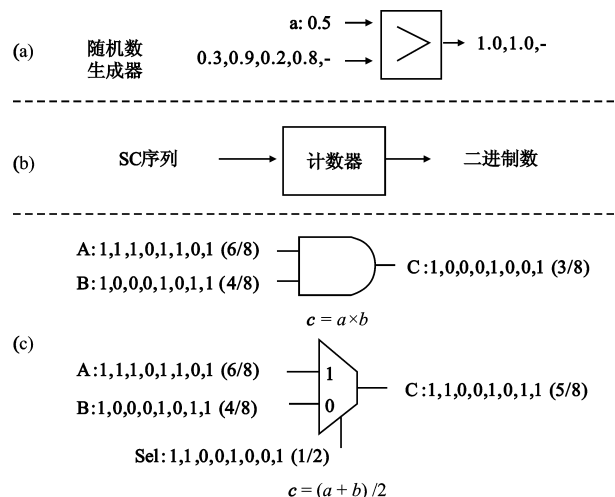


图 1 随机计算单元

3.2 SC 优化方法

一般而言, k -bit 的定点数需要 $2k$ 长度序列来表示,因此 SC 延迟较大.例如,要表示一个 10 位定点数 s ,需要 1 024 位长度的 SC 序列 S .但是若把 s 拆分为 2 个 5 位定点数,则只需要 2 个 32 位序列,能大大节省空间并减少延迟.基于上面的思想,我们可以采用二段分解来降低 SC 序列长度^[8].

一个 $2k$ 比特的定点数 x 可以表示为

$$x = \sum_{i=0}^{2k-1} a_i 2^i = m \cdot 2^k + n$$

式中,

$$m = a_{2k-1}2^{k-1} + \dots + a_k$$

$$n = a_{k-1}2^{k-1} + \dots + a_0$$

这里, m 和 n 为 k 比特的定点数,且范围仍在 $(0, 1)$.利用 2×2^k 的序列来表示原先需要 2^{2k} 长的 SC 序列,能大大缩短计算的延迟.

使用二段分解计算 SC 加法时,只需将高位序列和低位序列对应相加即可;计算乘法时,需要计算两个高低位序列共四种组合的 SC 乘法,可以视精度要求舍弃部分高阶小量来降低复杂度^[8].

3.3 基于 SC 的矩阵乘法

由于 PCG 算法中大部分计算复杂度集中于矩阵向量乘,本小节致力于设计基于 SC 的矩阵乘法.根据矩阵乘法的定义,如果每次单独调用 SC 乘法或加法器,则需要大量 B2S 和 S2B 单元,因此启发在 SC 域完成所有操作.

因为 SC 的加法采用 MUX 时有较大的精度损失,采用加法器直接对序列相加.原因有以下三点:(1)尽管使用加法器会把 SC 序列转化为整数序列,但是由于执行完加法后会立刻执行 S2B,因此并不影响接下来的计算;(2)使用加法器能提高准确度,从而减少序列长度;(3)单比特加法器的硬件消耗非常小.对于 SC 乘法我们则采用传统的与门结构.此外,为了避免中间的转化操作,我们在乘法结束后统一执行加法.在采用二段分解后的 SC 矩阵乘法如下所示:

算法 2: SC 矩阵乘法

输入: $X_{m \times n}, Y_{n \times 1}$

将 Y 转化为 SC 序列并存储于 $C_{n \times 2^k}$ 和 $D_{n \times 2^k}$ 中

for $i = 1:m$

将 X 的第 i 行转化为 SC 序列并存储于 $A_{n \times 2^k}$ 和 $B_{n \times 2^k}$ 中

$E = A \odot C$

$F = A \odot D$

$G = B \odot C$

对 E, F, G 按行求和得到结果 $e_{n \times 1}, f_{n \times 1}, g_{n \times 1}$

$z = e \cdot 2^k + f + g$

$Z(i) = \text{sum}(z) / 2^{2k}$

endfor

输出: $Z_{m \times 1}$

注意到采用二段分解计算乘法时舍弃了部分计算结果来减少复杂度,而 3.4 小节的仿真结果表明其性能损失可以忽略不计.

3.4 基于 SC 的 PCG 检测算法

将 SC 运用于 PCG 算法以降低硬件成本.考虑到硬件实现的实现,替换了所有矩阵向量乘和向量点乘,而简单的向量加法和除法则保持在二进制域.这是因为 SC 的除法设计困难且精度不高,而 SC 的加法可能存在溢出且二进制加法本身计算简单,所以选择替换传统计算复杂但是 SC 简单的乘法来最大限度的利用 SC 和二进制计算的优点.

为验证算法的准确性,在大规模 MIMO 平台上仿真了 PCG SC、PCG 和其他一些经典算法的误码率(Bit Error Rate, BER)-信噪比(Signal Noise Rate, SNR)曲线.我们采用了 128×8 的大规模

MIMO 系统,调制方式为 64-QAM,信道编码方式为 [133 171] 多项式编码的 1/2 卷积码. 图 2 给出了 BER 曲线,其中 FP 表示定点数结果,精度为 12 (4 位整数部分和 8 位小数部分);对应的,SC 序列长度为 212 (使用二段分解后长度为 64).

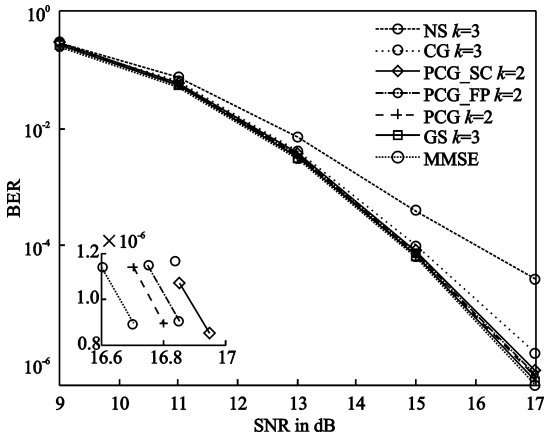


图 2 算法 BER 分析

可以发现,PCG 算法迭代次数 $k=2$ 时的性能好于 CG 算法 3 次迭代的性能,而 PCG SC 和 PCG 的性能一致,并且都能接近 MMSE 的性能. 由此可见,SC 的使用并不会带来性能损失.

4 系统设计及性能仿真

4.1 VLSI 架构设计

Vivado HLS 是赛灵思 (Xilinx) 公司开发的高层次综合工具. 通过使用 HLS,用户能直接用 C/C++ 等高级语言对现场可编程门阵列 (FPGA) 进行编程而不需要手动编写 RTL,从而可以加速 IP 创建. HLS 设计流程主要包括 C 仿真, C 综合, C-RTL 协同仿真和 IP 封装等. 为快速高效地实现并评估相关算法,采用 Vivado HLS 进行系统设计^[9].

根据 3.4 小节的仿真结果,在系统中使用 12 位精度的定点数 (4 位整数部分和 8 位小数部分) 以降低硬件成本. 为了算法 2 更利于硬件实现,提出以下两种优化手段.

(1) 矩阵乘法优化

PCG 和 PCG SC 中都包含大量的矩阵乘法,因此第一步对矩阵向量乘法进行优化. 以 PCG 中的 $C_{128 \times 1} = A_{128 \times 8} B_{8 \times 1}$ 的计算为例,根据矩阵乘法的定义实现了相关 C++ 代码: 计算 $C_{128 \times 1}$ 需要两层循环,且内层循环为 $c_i += \sum_{k=1}^8 a_{ik} \times b_k$.

在 HLS 中,未优化结构的时序如图 3(a) 所示. 由于乘法是串行处理的,全部计算完成需要 $(24+2)$

$\times 128 = 3\,328$ 个周期,延迟太长,因此对内层循环进行流水线处理. 通过添加指令 pipeline 后,内层循环的耗时得到了一定减少,如图 3(b) 所示. 但是由于乘累加结构的数据依赖性,内层循环无法完全流水线化,延迟仍然过长. 因此,通过重写代码和展开循环 (指令: unroll), 内层循环变成如图 3(c) 所示的加法树结构. 此时内层循环仅需 7 个周期完成,其中 4 个来自读取,3 个来自乘法、加法和存储;通过对矩阵进行分区 (指令: array_partition), 可以将获取数据的等待时间进一步减少为 1. 此使内层循环没有数据依赖性,因此对外侧循环执行流水线 (指令: pipeline); 同时考虑到双端口存储器每个周期可以读两个数据,将结构并行化为两个部分,如图 3(d) 所示. 现在的完成整个矩阵乘法的延迟仅为 67 个周期.

同样,对于向量乘法器 (算法 1 中的第 3 行,第 5 行和第 10 行) 也使用加法器树、展开和流水线方法,可以在一个周期内完成计算.

在 PCGSC 中,算法 1 中的所有矩阵向量乘法均被 SC 替换,执行向后变换的计数器 (算法 2 中的第 9 行) 也使用加法器树. 考虑到 SC 中的所有元素均为 0 和 1,将它们存储为布尔类型以节省空间. 在 PCGSC 中,由于硬件成本较低,算法 2 中的结构都采用展开指令 (unroll) 来减少延迟.

(2) 流水线优化

通过上述结构替换,PCG 的延迟为 595 个周期. 为进一步优化时序,我们需要对整个函数进行流水线处理.

此时,PCG 数据通路中的关键路径是矩阵乘法模块,例如 $C_{128 \times 1} = A_{128 \times 8} B_{8 \times 1}$,需要 67 个周期才能完成. 因此为添加流水线,将每个阶段的延迟尽可能地平衡到 67 个周期. 考虑到 HLS 中每个步骤的延迟,最终得到了如图 4 所示的一个 9 级流水线结构 (指令: dataflow). 图 4 中列出了每个阶段的延迟和功能. 经过 9 级流水线处理后,我们可以每 71 个周期获得输出的结果,因此大大提高了吞吐率.

PCG SC 的流水线几乎与 PCG 的流水线相同. 但是有 11 个阶段,因为向量乘法 $C_{1 \times 1} = A_{1 \times 128} B_{128 \times 1}$ 需要 67 个周期. 为平衡结构,需要将 PCG 中的第 7 阶段和第 9 阶段分开.

4.2 硬件实现结果

通过 HLS 仿真,得到了算法在 Xilinx Virtex-7 XC7VX690T FPGA 上的硬件资源消耗.

表 1 给出 PCG 和 PCG SC 的资源消耗,包括

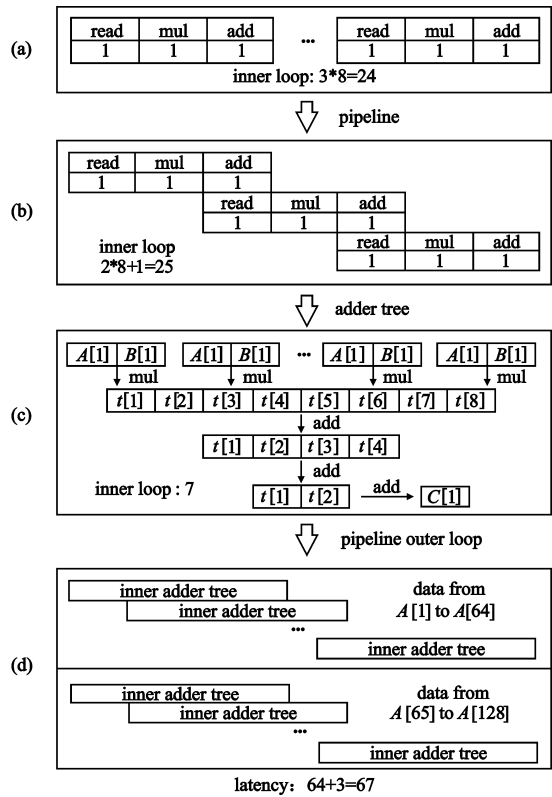


图3 HLS对矩阵向量乘法的优化

SLICE, LUT, FF, DSP48 和 BRAM18 等. 显然, 就 DSP48 而言, 由于 PCG SC 未使用 DSP, 其性能远优于传统 PCG. 从该表中, 还可以轻松了解算法中每个部分的硬件消耗: 初始部分会花费大量硬件资源, 约占整个算法的三分之一; 与第一次迭代相比, 第二次迭代消耗的硬件资源更少, 原因是在第二次迭代中不再需要进行下一次迭代的计算.

表1 PCG 和 PCG SC 各部分 FPGA 资源消耗

		初始化	迭代 1	迭代 2	总数
PCG	Slice 数	5 922	4 921	2 257	13 705
	LUT 数	13 746	12 538	5 276	34 433
	FF 数	16 407	9 081	4 831	41 672
	DSP 数	402	268	153	838
	BRAM 数	32	0	0	32
PCG SC	Slice 数	5 655	4 699	2 145	13 088
	LUT 数	16 984	15 491	5 276	42 545
	FF 数	13 009	7 201	5 969	33 043
SC	DSP 数	0	0	0	0
	BRAM 数	16	0	0	16

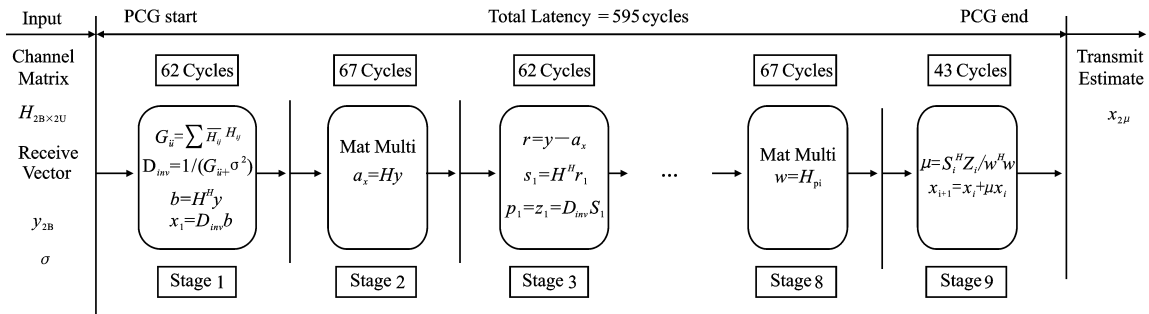


图4 PCG 算法九级流水线硬件结构

另外, 提出的算法将与其他最近提出的大规模 MIMO 检测器进行比较, 包括基于 CG^[4] 和 NS^[5] 的大规模 MIMO 检测器, 结果见表 2. 其中, CG^[4] 并不包括 Gram 矩阵的求解, 因此其资源消耗较少. 就 DSP48 的数量而言, PCG 和 PCG SC 远优于所有其他设计. 关于 PCG 和 PCG SC 的对比. 采用 SC 后, PCG 的 FF 减少了 20.7%, DSP48 减少了 100.0%, BRAM18 减少了 50.0%. 同时, 从 3.4 小节的仿真结果可以看出两种算法的 BER 性能相似. 因此, PCGSC 算法在硬件成本上具有更好的性能.

表2 128×8 大规模 MIMO 检测硬件消耗比较

	PCG	PCG SC	CG ^[4]	NS ^[5]
调制阶数	64-QAM	64-QAM	64-QAM	64-QAM
包含 Gram	是	是	否	是
迭代次数	2	2	3	3
Slice 数	13 705	13 088	1 094	48 244
LUT 数	34 433	42 545	3 324	148 797
FF 数	41 672	33 043	3 878	161 934
DSP 数	838	0	33	1 016
BRAM 数	32	16	1	16
时钟/MHz	142	142	412	317
延迟周期	639	782	951	196
吞吐/Mbps	100	100	20	621

值得注意的是,作为一种近似计算技术,SC 适用但不局限于 PCG 算法.通过替换矩阵乘法等复杂度高的计算,任何算法均通过可以采用 SC 来降低硬件消耗.

5 结束语

随机计算是一种近似计算技术,通过把二进制数转为 01 序列来简化计算单元.本文通过采用 SC,大大降低了大规模 MIMO 系统检测的复杂度.考虑到线性检测的迭代算法中包含大量复杂度高的矩阵乘法,本文提出的随机计算矩阵乘法模块,能够在不对中间结果进行 B2S 和 S2B 的前提下完成计算,从而大大节省面积消耗;同时通过采用二段分解并舍弃低级小量,能在不影响计算精度的前提下大幅减少随机计算序列的长度.通过使用 Vivado HLS,实现基于随机计算的预处理共轭梯度检测算法.在添加合适的 HLS 指令进行优化后,电路的吞吐率得到了提升.仿真结果表明基于随机计算的算法和最优算法相比几乎没有 BER 损失,同时不需要任何 DSP 单元.

参考文献:

- [1] ANDREWS J G, BUZZI S, CHOI W, et al. What will 5G be? [J]. IEEE Journal on selected areas in communications, 2014, 32(6): 1065-1082.
- [2] 崔超,贺光辉.一种 8×8 MIMO 系统的近最优检测算法[J]. 信息技术,2019,(3):5-9
- [3] WU M, YIN B, WANG G, et al. Large scale MIMO detection for 3GPP LTE: Algorithms and FPGA im-

plementations [J]. IEEE J. Sel. Topics Signal Process, 2014, 8(5):916-929.

- [4] WU Z, ZHANG C, XUE Y, et al. Efficient architecture for soft-output massive MIMO detection with gauss-seidel method[C]//Proc. IEEE Int. Conf. Circuits Syst. (ISCAS). Canada, Montreal, 2016: 1886-1889.
- [5] YIN B, WU M, CAVALLARO J R, et al. VLSI design of largescale soft-output MIMO detection using conjugate gradients[C]//Proc. IEEE Int. Symp. Circuits Syst. (ISCAS). Portugal, Lishbon, 2015: 1498-1501.
- [6] BROWN B D, CARD H C. Stochastic neural computation. I. Computational elements[J]. IEEE Transactions on Computers, 2001, 50(9): 891-905.
- [7] 秦小波,大规模 MIMO 中预处理算法的研究[D]. 上海:上海交通大学,2016.
- [8] CHEN J, HU J, ZHOU J. Hardware and Energy-Efficient Stochastic LU Decomposition Scheme for MIMO Receivers[J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2016, 24(4): 391-1401.
- [9] 张浩盛,贺光辉,李龙,等.基于 HLS 实现的小步长电磁暂态仿真系统[J]. 信息技术,2017(10):38-42.

作者简介:

熊海翡 男,(1994-),硕士研究生.研究方向为随机计算及大规模 MIMO 检测的算法研究与硬件实现.

E-mail: xhf_1994@163. com.

贺光辉 男,(1980-),博士,副研究员.研究方向为通信系统 VLSI 电路设计和随机计算设计等.