近 DRAM 加速矩阵乘法

Aman Sinha*, 会员，IEEE 和 Bo-Cheng Lai，会员，IEEE
电子研究所，国立阳明交通大学

台湾新竹 Email :
amansinha.sw@gmail.com,
bclai@nycu.edu.tw

2024 IEEE 第 17 届嵌入式多核/众核片上系统（MCSoC）symposium | 979-8-3315-3047-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/MCSoC64144.2024.00048

摘要—通用矩阵乘法（GeMM）是机器学习、数据科学、计算机图形学和科学模拟等领域的一项基本计算操作。然而，由于内存访问带宽不足、缓存效率低、硬件利用率低和功耗巨大等挑战，其在冯·诺依曼计算机（如 CPU 和 GPU）上的性能受到瓶颈。在各类大数据管道中，GeMM 常与复杂的顺序分析一起出现，导致 GPU 性能急剧下降。此外，现代 Nvidia GPU 中提供的 GeMM 优化张量核心无法扩展到非 GeMM 任务。近年来，人们探索了各种近内存计算（NMC）架构，以缓解 GeMM 等分析的数据密集特性。本研究通过在堆叠式 DRAM 平台上使用由简单互联处理核心组成的集群，评估了 NMC 在 GeMM 中的性能潜力。该架构在保持与高端 Nvidia GPU 相当的高效率的同时，功耗更低，且能高度扩展到各种非 GeMM 逻辑复杂的计算任务。

索引词—深度神经网络，通用矩阵乘法，矩阵乘法，近存计算，RISC-V，堆叠存储

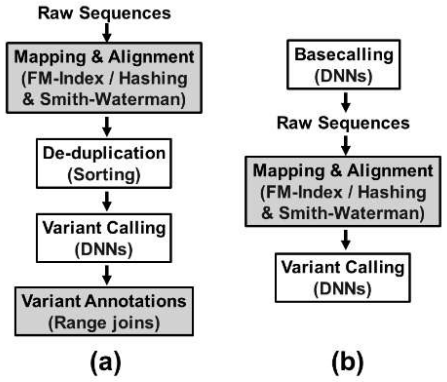

图 1. 涉及顺序复杂执行和数据并行 GeMMs 的生物信息学分析流程。(a) 使用 DeepVariant [6]进行短读序列分析流程，(b) Oxford Nanopore 长读分析流程[5]。

红色块表示非 GeMM 复杂数据密集型分析。

I. I

矩阵乘法在各种高性能计算（HPC）和人工智能应用中构成核心计算[1]，主要应用于线性代数和深度神经网络（DNNs）。通用矩阵乘法（GeMM）定义为操作 D = α AB + β C，其中矩阵 A、B 和 C 的维度分别为 m × k、n × k 和 m × n，α和β是标量值。GeMM 计算需要矩阵 B 的转置，这通常通过各种技术逻辑上完成，而不是物理上移动内存中的数据[2]。

当前 GeMMs（广义矩阵乘法）的技术现状。GeMMs 的广泛应用推动了 GPU 设计空间的多种架构演进，其中张量核心（TCs）[2]是 Nvidia 大规模并行 GPU 的最新发展。除了显而易见的密集计算外，GeMM 还涉及大量数据访问，通常严重受制于缓存效率低下的内存访问[3]。高带宽内存（HBM）是近期缓解 GPU 内存访问挑战的趋势。增强数据重用的通用机制利用更快的内存层级作为缓冲区

GPU 整体内存层次结构中的洗牌操作，包括全局内存、L2 和 L1 缓存、共享内存和寄存器。CUTLASS [4]通过利用丰富的核心本地寄存器作为 GeMM 输入和输出的数据缓冲空间，在现代 Nvidia Tensor 核心上提供高性能 GeMM 的访问。然而，GPU 的深层次内存结构仍然在通过内存访问延迟隐藏实现吞吐量和数据移动的能量效率之间存在权衡。此外，GPU 执行复杂顺序逻辑的架构限制，为涉及数据并行和顺序计算流水线的大数据分析的效益成本权衡带来了另一个主要挑战。图 1 描绘了生物信息学中的这种情况，其中分析流水线[5]、[6]涉及复杂的数据密集型并行和顺序执行模式。

GPU 在处理数据绑定工作负载方面的局限性催生了各种近内存计算（NMC）技术[7]，这些技术将执行单元放置在靠近内存设备的位置，以克服冯·诺依曼计算机中有限的访问通道，同时提供廉价的扩展性[8]–[11]。然而，为了充分发挥 NMC 架构的性能能力，有必要最大化 NMC 设备的宿主机无关执行。这要求采用此类流水线工作负载（图 1）明确

本研究由台湾科技部支持[MOST 111-2221-E-A49-092-MY3]。

*通讯作者

2771-3075/24/$31.00 ©2024 IEEE DOI
10.1109/MCSoC64144.2024.00048

在 NMC 设备上。目前对这类设计的有限探索是这项工作的主要动机。例如，在图 1a 中，虽然映射与对齐之前已经在 NMC 平台上得到加速，但其他步骤尚未利用相同的性能优势[12][13]。请注意，NMC 与处理内存（PIM）不同，后者利用内存电路进行位并行计算[14]，这可能影响标准内存操作。这项工作关注 NMC 而非 PIM[15]。

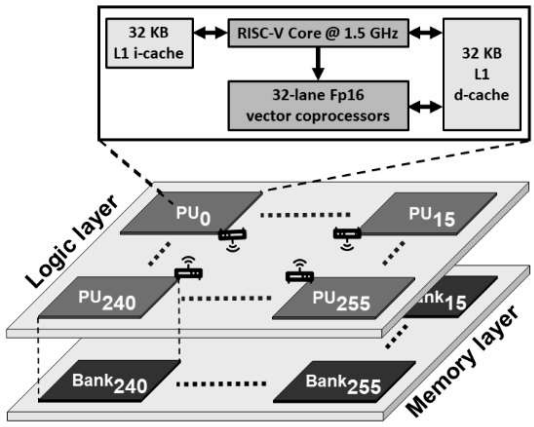

图 2. 整体 SEDRAM PU 系统组织，源自[9]。所有 PU 在逻辑层通过网络芯片（NoC）以网状拓扑连接，使用单周期延迟的先进路由器[9]。

近 DRAM 计算在堆叠存储平台上对 GeMMs 的潜力。虽然 HBM 一直是减少数据带宽挑战的主要内存技术，但它已被证明面临高接口能耗[16], [17]和扩展性不灵活[9]等挑战。堆叠嵌入式 DRAM（SEDRAM）通过用混合键合[16], [17]替换硅通孔（TSV）来解决这些挑战，从而实现接口功耗降低 6.67 倍。我们之前的工作[9]已广泛探索了 SEDRAM 平台，用于使用 FMIndex 数据结构进行数据密集型字符串匹配任务。这项工作扩展并评估了其在 GeMM 任务中的性能潜力。

表 I

PP、APower V100 比较

GPU 5120 PU SEDRAM . SEDRAM PU

估计为 40，V100 12[18].

	SEDRAM 处理器		
	1 个处理器	5120 个处理器	V100
峰值性能（TFLOPs）0.048		245.76	125
面积（mm）	0.161	822.3	815
功耗（W）	0.0479	245.45	300

表 I 总结了基于 SEDRAM 的处理器单元（PU）在 1500 MHz 下运行 32 路矢量协处理器进行半精度浮点（FP16）计算时的峰值理论性能。与 V100 GPU 上的 640 个张量核心[18]相比采用 5120 个 PU 的 SEDRAM 组织可以在几乎降低 20%的功耗下实现双倍吞吐量。此外，SEDRAM PU 在更优的技术节点上预计能提高面积效率。

II. N-DRAM AGMM

图 2 展示了基于我们先前工作[9]推导出的 SEDRAM PU 的整体组织结构，同时去除了原始设计中不必要的组件。此外，矢量协处理器已配置为用于半精度浮点（FP16）运算，而非原始的 64 位设计。然而，由于矢量通道数量相同，整体面积和功耗特性将保持不变。

基于片上网络（NoC）的缓存高效矩阵转置 如前所述（第一部分），最先进的 CUTLASS 利用 GPU 上丰富的寄存器来执行分块 GEMM[2]，而不是物理转置矩阵 B 以最大化单指令多数据（SIMD）计算。然而，SEDRAM PU 缺乏这样的丰富寄存器。因此，它依赖于矩阵 B 的优化转置来利用向量协处理器的 SIMD 功能。请注意，我们的设计与 CUTLASS 类似，也执行分块 GeMM，我们显式的矩阵转置操作是唯一的算法差异。

图 3 展示了通过 NoC 传输实现的每条缓存行部分矩阵转置机制，这些缓存行是通过 DRAM 组读取填充的。原始输入矩阵（图 3a 中的 A 和 B）由主机 CPU（图 3b）分布到 PU 集群中。假设每条缓存行能够容纳两个矩阵单元，而每行 DRAM 则假设存储两个这样的缓存行数据。图 3c-3e 中高亮的单元显示了产生矩阵 B 部分转置的洗牌操作。此外，每次洗牌操作后都通过矢量协处理器计算执行部分 GeMM，用于 Fp16 乘积和归约求和。整体设计最大限度地利用了为矩阵 B 激活的每行 DRAM 缓冲区。通过每个 PU 上的单条自定义指令实现 NoC 传输，而实际传输与矢量协处理器上的 Fp16 计算重叠。可以推断，NoC 性能是整体设计高吞吐量的关键组成部分。

III. EE

我们进行了实验来评估基于堆叠内存的 Near-DRAM 加速 GeMMs 的性能潜力，并将它们与高效的基线 Nvidia CUTLASS[4]库在配备 640 个 Tensor 核心[2]、[18]的 V100 GPU 上执行的性能进行比较。模拟了由 256 个 SEDRAM PU 组成的 SEDRAM 模块，每个 PU 拥有 256 MB 内存组，并将性能扩展到 20 个这样的 SEDRAM 模块，以便与 V100 GPU 的大芯片尺寸[18]进行公平比较。使用具有 32 路 FP16 矢量协处理器的 RISC-V 核心的每个 PU，通过精确周期的 Gem5 进行了模拟。

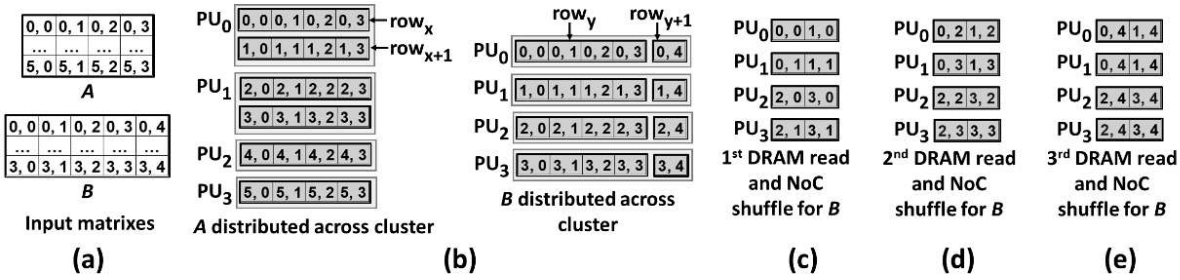

图 3. 四个 SEDRAM PU 集群上的 NoC 中心 GeMM 设计示意图。(a) 尺寸分别为 6×4 和 4×5 的原输入矩阵 A 和 B，(b) 输入矩阵在集群内存模块上的分布。假设缓存行可以容纳两个数据项，而每个 DRAM 行可以容纳四个这样的数据项。(i, j)表示位于第 i 行第 j 列的矩阵单元值。(c) 首先进行 DRAM 读取，然后跨 PU 进行数据洗牌，以及向量协处理器执行乘积和约简和操作。(d) 第二次，(e) 第三次，DRAM 读取、基于 NoC 的数据洗牌以及乘积和约简和操作，以最大化缓存行和 DRAM 行缓冲区的利用率。

[19] 集成了 Ramulator [20] 的模拟器，类似于我们之前的工作 [9], [10]。然而，Gem5 的 MinorCPU 配置的运行频率升级到了 1500 MHz。向量协处理器执行使用自定义指令进行模拟。相比之下，NoC 流量模式使用 Garnet [21] 在网格拓扑中单独模拟，并与 PU 执行结果集成，假设 NoC 传输与 PU 执行重叠。PU 集群的整体模拟方法与之前的工作 [9] 相同。

这项工作对最近开发的堆叠嵌入式 DRAM 的近内存计算在 GeMM 任务上的性能进行了初步评估。这种简单的组织实现了与高端现代 GPU 相当的性能特征，同时提供了更大的扩展性。我们的未来工作将进一步优化设计，并评估相关生物信息学工作负载端到端流程执行的性能潜力。

A 这项工作得到了台湾科技部的支持[MOST 111-2221-E-A49-092-MY3]。

表 II

与 CUTLASS-GMM 的性能比较

NV100 GPU

矩阵维度	V100 本工作	运行时间 (ms) 加速比	
64 X 64 X 64	1.405	0.512	2.74 倍
128 X 128 X 128	1.762	1.024	1.72 倍
256 X 256 X 256	2.483	2.048	1.21 倍
512 X 512 X 512	4.042	8.192	0.49x
1024 X 1024 X 1024	13.026	24.57	0.53x
2048 X 2048 X 2048	70.876	114.688	0.62x
4096 X 4096 X 4096	516.854	589.824	0.87x

如表 II 所示，对于较小的 GeMM 工作负载，SEDRAM PU 组织实现了高达 2.74 倍的性能提升，而对于一些较大的工作负载，性能则下降了一半。需要注意的是，小型和中型 GeMM 工作负载在各种应用中广泛使用，例如生物信息学[22]，因此表 II 中的结果具有有效性。此外，随着 SEDRAM PU 技术节点的改进，PU 的数量将增加，从而进一步提升其性能。最后，SEDRAM PU 组织的极高可扩展性是另一个重要的权衡因素。

IV. C

通用矩阵乘法（GeMMs）是各种高性能计算工作负载的重要组成部分，它们在当今最先进的 GPU 上的执行面临着内存访问效率和扩展到完整分析流程的挑战

R

[1] E. Qin, A. Samajdar, H. Kwon, V. Nadella, S. Srinivasan, D. Das, B. Kaul, 和 T. Krishna, “Sigma: 一种用于 DNN 训练的稀疏和不规则 GEMM 加速器，具有灵活的互连，” 在 2020 年 IEEE
国际高性能计算机体系结构研讨会 (HPCA). IEEE, 2020, 第 58–70 页.

[2] S. Markidis, S. W. Der Chien, E. Laure, I. B. Peng, 和 J. S. Vetter, “Nvidia tensor core 可编程性、性能与精度，” 在
2018 IEEE 国际并行与分布式处理 symposium 工作坊 (IPDPSW). IEEE, 2018, pp. 522–531.

[3] 陈栋，金昊，郑磊，黄宇，姚鹏，桂晨，王启，刘浩，何浩，廖翔等，“一种适用于近 DRAM 处理-内存架构的通用卸载方法”，在 2022 年 IEEE 国际
并行与分布式处理 symposium (IPDPS). IEEE, 2022, pp. 246–257.

[4] T. Faingnaert, T. Besard, 和 B. De Sutter, “灵活高效的 GEMM”
kernels on gpus,” IEEE Transactions on Parallel and Distributed Systems, vol. 33, no. 9, pp. 2230–2248, 2021.

[5] Y. Wang, Y. Zhao, A. Bollas, Y. Wang, and K. F. Au, “Nanopore sequencing technology, bioinformatics and applications,” Nature biotechnology, vol. 39, no. 11, pp. 1348–1365, 2021. [6] Y.-L. Lin, P.-C. Chang, C. Hsu, M.-Z. Hung, Y.-H. Chien, W.-L. Hwu, F. Lai, and N.-C. Lee, “Comparison of gatk and deepvariant by trio sequencing,” Scientific Reports, vol. 12, no. 1, p. 1809, 2022. [7] G. Singh, L. Chelini, S. Corda, A. J. Awan, S. Stuijk, R. Jordans, H. Corporaal, and A.-J. Boonstra, “A review of near-memory computing architectures: Opportunities and challenges,” in 2018 21st Euromicro
数字系统设计会议 (DSD). IEEE, 2018, 第 608-617 页.

[8] A. Sinha, H.-C. Yang, P.-Y. Liu, Y.-S. Kuo, Y. Fang, T.-S. Chang, K.H. Li, 和 B.-C. Lai, “Dsim: 在近 DRAM 加速器上进行分布式序列匹配用于基因组组装，” IEEE Emerging & Selected Topics in Circuits and Systems 期刊
《电路与系统精选专题》，第 12 卷，第 2 期，第 486-499 页，2022 年。

[9] A. Sinha, Y. Fang, 和 B.-C. Lai, “Regal: 基于 lpddr4x 堆叠式 dram 的基因组分析可重编程引擎，” 在 2023 IEEE
国际电路与系统会议 (ISCAS). IEEE, 2023, pp. 1–5.

[10] A. Sinha, P.-Y. Liu, Y. Fang, J.-Y. Mai, and B.-C. Lai, “Grona: A
框架用于近内存加速器的 gather-and-reduce，”在 2023 年 IEEE 第 16 届嵌入式多核/众核系统芯片（MCSoC）国际 symposium。IEEE，2023 年，第 225-232 页。
[11] J. G´omez-Luna, I. El Hajj, I. Fernandez, C. Giannoula, G. F. Oliveira,
和 O. Mutlu，“基准测试以内存为中心的计算系统：分析真实的处理内存硬件，”在 2021 年第 12 届国际绿色和可持续计算会议（IGSC）。IEEE，2021 年，第 1-7 页。

[12] A. Sinha, B. C. Lai, 和 J. Y. Mai，“基于二进制索引的可扩展”
基因组数据上的范围连接，”IEEE/ACM 计算生物学与生物信息学汇刊，2023。

[13] K. Wang, M. Li, and H. Hakonarson, “Annovar: 功能注释
来自高通量测序数据的遗传变异，”核酸研究，2010。

[14] N. Verma, H. Jia, H. Valavi, Y. Tang, M. Ozatay, L.-Y. Chen, B. Zhang,
和 P. Deaville，“内存计算：进展与前景”，IEEE 固态电路杂志，第 11 卷，第 3 期，第 43-55 页，2019。
[15] C. Nie, C. Tang, J. Lin, H. Hu, C. Lv, T. Cao, W. Zhang, L. Jiang,
X. Liang, W. Qian 等，“Vspim：通过向量标量操作实现 SRAM 处理内存的 DNN 加速”，IEEE 计算机汇刊，2023。

[16] B. Fujun, J. Xiping, W. Song, Y. Bing, T. Jie, Z. Fengguo, W. Chunjuan,
W. Fan, L. Xiaodong, Y. Guoqing 等，《用于 LPDDR4/4x 的堆叠式嵌入式 DRAM 阵列：采用混合键合 3D 集成技术，逻辑-内存接口为 34GB/s/1GB 0.88 pJ/b》，在 2020 年 IEEE 国际电子器件会议 (IEDM)。IEEE, 2020, 第 6–6 页。

[17] D. Niu, S. Li, Y. Wang, W. Han, Z. Zhang, Y. Guan, T. Guan, F. Sun,
F. Xue, L. Duan 等，《184qps/w 64MB/mm² 3D 逻辑-DRAM 混合键合：采用近工艺内存引擎的推荐系统》，在 2022 年 IEEE 国际固态电路会议 (ISSCC)，第 65 卷。IEEE, 2022, 第 1–3 页。

[18] J. Peddie，《计算加速器和其他 GPU》，在《GPU 历史新进展》。Springer, 2023, 第 239–304 页。[19] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu,
J. Hestness, D. R. Hower, T. Krishna, S. Sardashti 等，《gem5 模拟器》，《ACM SIGARCH 计算机架构新闻》，第 39 卷，第 2 期，第 1-7 页，2011 年。

[20] Y. Kim, W. Yang, 和 O. Mutlu, “Ramulator: 一个快速且可扩展的 DRAM 模拟器，” IEEE 计算机架构快报, 卷. 15, 第. 1, 页.
45–49, 2015.
[21] N. Agarwal, T. Krishna, L.-S. Peh, 和 N. K. Jha, “Garnet: A detailed
在完整系统模拟器内部的片上网络模型，”在 2009 年 IEEE 系统与软件性能分析国际 symposium。IEEE，2009，第 33-42 页。

[22] Z. Xu, Y. Mai, D. Liu, W. He, X. Lin, C. Xu, L. Zhang, X. Meng,
J. Mafofo, W. A. Zaher 等，“Fast-bonito：一种基于深度学习的更快纳米孔测序碱基调用器，”生命科学中的人工智能，卷 1，第 100011 页，2021。