神威太湖之光加速计算在脑神经网络模拟中的应用

栗学磊[1),2)] 朱效民[3)] 魏彦杰[1)] 冯圣中[1),2)]

[1)](中国科学院深圳先进技术研究院高性能计算研究中心 广东 深圳 518055)

[2)](国家超级计算深圳中心(深圳云计算中心) 广东 深圳 518055)

[3)](高效能服务器和存储技术国家重点实验室 济南 250101)

摘 要 脑神经网络模拟是脑科学研究和理论验证的重要方法.为提高脑模拟速度,异构加速已开始应用于脑模拟.然而现有异构加速脑模拟软件均存在明显的访存性能和计算精度问题.为此,本文基于神威太湖之光研发了脑模拟软件 SWsnn,确保了随机访存多发生在高速缓存中.为避免主存访问的随机性,将频繁出现随机访存且数据量较小的神经元信息长时间停留在局部存储(LDM),同时将数据量很大的突触连接数据存储在主存,且尽可能连续访问主存.为避免可塑性导致的对突触连接的随机搜索,对脉冲时间依赖可塑性(STDP)算法采用需要前再更新的方法,以确保主存访问的连续性.为了提高脑模拟精度,设计环形缓冲和延迟传送联合应用方法,以支持高精度时间步长的脑模拟.在此基础上,对 SWsnn 进行向量化、访存隐藏等优化操作,计算性能进一步提高约 50%.SWsnn 对 10^4 神经元全连接网络实现了生物实时模拟,比同等规模浮点计算能力 GPU 上运行的 CARLsim 快 10 倍左右.

关键词 脉冲神经网络(SNN);脑模拟;SW26010;随机访存;环形缓冲;神威太湖之光

中图法分类号 TP391.9 **DOI 号** 10.11897/SP.J.1016.2020.01025

Application of Sunway TaihuLight Accelerating in Brain Neural Network Simulation

LI Xue-Lei[1),2)] ZHU Xiao-Min[3)] WEI Yan-Jie[1)] FENG Sheng-Zhong[1),2)]

[1)](*Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, Guangdong* 518055)

[2)](*National Supercomputing Center in Shenzhen, Shenzhen, Guangdong* 518055)

[3)](*State Key Laboratory of High-end Server & Storage Technology, Jinan* 250101)

Abstract Brain neural network simulation is an important approach for brain science research. Due to the requirement of current brain experiment condition, the real-time brain simulation is becoming more and more important, leading to the challenge of high performance simulating. To improve the performance of the brain simulation, heterogeneous architecture based algorithms and software have been developed, such as NeMo, CARLsim and GeNN. However these accelerating results are not obvious. There are two common problems for most brain simulation softwares based on heterogeneous architecture, i. e. memory access performance and simulation accuracy. Therefore, in this paper we develop a brain simulation software, named SWsnn, with high memory performance and accuracy using SW26010, the processors of Sunway TaihuLight supercomputer. SWsnn makes most of random access operations always occur in high-speed cache, and support high accuracy time step sizes. Since neurons can be represented by a small data structure and often are accessed randomly in general, we keep neuron data in the local dynamic

收稿日期:2019-09-27;在线出版日期:2020-02-20.本课题得到科技部重点研发计划(2018YFB0204403,2016YFB0201305)、国家自然科学基金(U1813203,61433012)、国家自然科学青年基金(41804128,31601028)、深圳市基础研究项目(JCYJ20180507182818013,GGFW2017073114031767,JCYJ20170413093358429)、中国博士后科学基金(2019M653132)、中国科学院重点实验室(2011DP173015)资助.**栗学磊**,博士,助理研究员,中国计算机学会(CCF)会员,主要研究方向为高性能计算与地球物理学.E-mail:xl.li@siat.ac.cn.**朱效民**,博士,中国计算机学会(CCF)会员,主要研究方向为并行计算与性能优化.**魏彦杰**(通信作者),博士,研究员,中国计算机学会(CCF)会员,主要研究领域为高性能计算、计算生物学.E-mail:yj.wei@siat.ac.cn.**冯圣中**(通信作者),博士,研究员,中国计算机学会(CCF)会员,主要研究领域为高性能计算、生物信息学.E-mail:sz.feng@siat.ac.cn.

memory (LDM) for long time. The large-scale synaptic connection data is stored in the main memory, and do the best to keep it accessed continuously. In order to avoid the random access of the synaptic connection due to the random search for the plasticity of synapses, we use the method of updating before needed of synapse weights. It means that we only update the weights of spiking-time dependent plasticity (STDP) algorithm before it is needed, and we only store the synaptic connection data on the side of pre-neurons. To improve the accuracy of the brain simulation, SWsnn support different time steps in the model by using ring buffer and delayed message passing simultaneously. Based on the methods above, we optimize SWsnn using different techniques, such as vectorization and access hiding, and the performance is further improved by about 50%. SWsnn can simulate biological real-time brain activity for full-connected networks with 10^4 neurons on a single SW26010. Compared with CARLsim running on a GPU with similar FLOPS to SW26010, SWsnn is about 10 times faster than CARLsim.

Keywords Spiking Neural Network (SNN); brain simulation; SW26010; memory random access; ring buffer; Sunway Taihulight

1 引 言

脑神经网络模拟(简称脑模拟)是脑科学研究方法的有效途径之一,是对脑科学理论验证的重要方法.脉冲神经网络(Spiking Neural Network,SNN)是现有脑神经网络模拟的主要方法,也是连接脑科学与人工智能的重要工具.当前,已有多套脉冲神经网络模拟软件获得开发,并且成熟应用于脑科学研究,比如 NEURON、NEST、GENESIS 等.脑神经网络非常复杂且规模巨大,人脑大约有 10^{10} 个神经元和 10^{14} 个突触组成.现有的脑模拟研究多用于小区块模拟,规模约 10^5 个神经元.随着大规模超算的快速发展,大规模多脑区脑模拟已经实现[1].然而,常规模拟软件面临明显计算慢的问题,模拟的墙钟耗时比生物真实时间高约 2～3 个数量级,远不能实现实时模拟[1-3].

SNN 实时模拟对适应真实环境交互,探索真实生物智能机制具有重要意义[4].为了提高模拟速度,异构设备(如 GPU、Xeon Phi 等)加速计算已经开始用于 SNN 模拟算法,比如 NeMo[5]、CARLsim[6]、GeNN[7]、HRLSim[8] 等.异构计算对脑模拟具有一定的加速效果,但是这些研究尚处于测试阶段,还面临多个问题有待研究.

现有的异构加速计算脑模拟软件多数有明显的访存性能和计算精度问题.由于神经元之间的连接具有随机性、非连续性和非局部性等特征,导致 SNN 算法实现存在较强的随机访存问题.对于单节点 CPU 脑模拟实现,随机访存是影响计算性能的主要因素,而对异构设备则影响更大.因此,如何解决随机访存问题,提高异构众核利用率,是实现 SNN 算法异构加速的关键问题.

有关计算精度问题,现有的多数开源异构加速 SNN 软件(如 NeMo、CARLsim)仅支持 1ms 量级的离散时间步长、脉冲计时精度、突触延时精度等,这些离散取值与常规 CPU 成熟软件(如 NEST)存在明显差异.NEST 支持 0.1ms 量级或更小量级的脑模拟离散时间步长.由于时间步长等离散设计对脑模拟误差影响很大[9],有必要设计高精度的 SNN 异构加速计算.

为解决异构加速计算面临的性能和精度问题,同时推广国产处理器应用范围,本文依托神威太湖之光超算,研发了基于申威加速器的脑模拟软件 SWsnn,详细研究了脑模拟面临的随机访存和计算精度问题,实现高精度高性能的脑模拟计算.

SWsnn 通过确保随机访存发生在高速缓存中,实现了大幅度性能提升.SWsnn 结合申威加速器的局部存储(LDM)和寄存器通信属性,将可能出现随机访存且数据量较小的神经元数据长时间停留在 LDM,将数据量很大的突触连接数据存储在主存,且尽可能避免非连续访问.为避免可塑性导致的对突触连接的随机搜索,将相关的突触连接信息只在前神经元一侧存储排序,并采用需要前再更新的方法实现脉冲时间依赖可塑性计算(Spiking-Time Dependent Plasticity,STDP),有效避免突触连接随机访存,同时支持 STDP 即时更新.为提高脑模拟

精度且节约 LDM 空间，设计了环形缓冲和延迟传送联合应用方法，以支持高精度 SNN 模拟.

本文首先分析 SNN 模拟在异构设备实现中面临的问题，并提出相应的解决方案，实现高精度高性能 SNN 模拟；第 2 节将简述 SNN 神经元和突触模型基本原理，以及申威加速器(SW26010)基本框架；第 3 节主要描述 SNN 实现流程，分析随机访存形成原因和解决方案，并设计数据存储及其合理性分析；第 4 节将进行 SWsnn 的数值实验，并对计算精度和计算性能加以分析；第 5 节将基于性能分析对 SWsnn 进行进一步性能优化；第 6 节对当前还面临的问题进行讨论；最后，第 7 节是本文的总结，并指出进一步的工作方向.

2　基本原理

2.1　Izhikevich 神经元模型

脑神经网络模型由大量神经元和突触组成，图 1 显示了神经元示意图[10]. 神经元由细胞体、轴突和树突组成. 前神经元(或源神经元)轴突连接到后神经元(或目标神经元)树突或细胞体可形成突触. 当源神经元膜电位受外部影响逐渐增加到一定阈值，将激发一个脉冲，通过突触上的多种单向传输的离子通道传到目标神经元，传输电流大小用突触权重衡量，传到所需的时间用突触延时表示. 一个神经元可连接约 10^4 个突触，因此突触数量远大于神经元数量. 下面部分将介绍相关的神经元和突触模型.

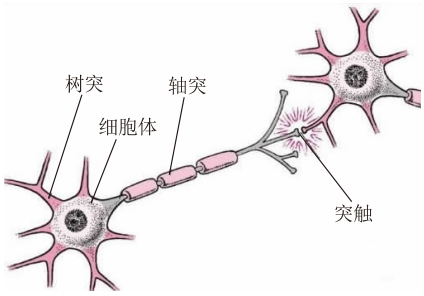


图 1　神经元细胞和突触[10]

当前已有许多神经元模型被提出[11]，其中具有代表性的模型有三种：Hodgkin-Huxley (H-H)模型、Leaky Integrate-Fire (LIF)模型、Izhikevich 模型. H-H 模型比较复杂，参数多且计算量大，可模拟多种类型神经元，是 NEURON 软件的主要支持模型. LIF 模型比较简单，计算量小，只可模拟简单神经元，是 NEST 的主要支持模型. Izhikevich 模型是较新的模型，是 H-H 和 LIF 的折衷. Izhikevich 模型较简单且可模拟多种神经元，也是新型 SNN 软件主流支持模型. 因此，本文选择 Izhikevich 模型用于模拟. Izhikevich 模型满足以下常微分方程组[12]

$$\begin{cases}\dfrac{dv}{dt}=0.04v^2+5v+140-u+I\\ \dfrac{du}{dt}=a(bv-u)\end{cases}\tag{1a}$$

$$\text{if } v\geq 30\,\text{mV},\ \text{reset}\begin{cases}v\leftarrow c\\ u\leftarrow u+d\end{cases}\tag{1b}$$

其中，v 为膜电位，u 为膜恢复变量，a、b、c、d 为四个模型参数. I 为与树突连接相关的输入电流，是许多突触电流的叠加. 通过调整参数，Izhikevich 可模拟 20 多种神经元行为，且模拟精度很高.

方程(1)常用的数值求解方法为 Euler 方法和 Runge-Kutta 方法. 我们同时开发两种数值算法，以方便与其他软件对比.

2.2　突触电流模型与可塑性模型

与突触相关的模型主要包括两部分，即突触电流模型和可塑性模型[13-14]，两者均与突触权重有关，前者描述突触权重的累加方法，后者描述突触权重的更新算法.

突触电流模型主要基于两种模式：电流模式(Current-Based，CUBA)和电导模式(Conductance-Based，COBA)[8]. 电流模式较简单，电流只与权重有关，是激发脉冲传送到目标神经元时的加权求和. 电导模式较复杂，电流除了与权重有关外，还与神经元膜电位和电导参数有关. 此处仅列出电流和电导模式的基本关系式

$$I_j^{\text{syn}}=\sum_{i=1}^{N}f_{ij}w_{ij}\tag{2}$$

其中，I_j^{syn} 表示第 j 个目标神经元的突触电流，w_{ij} 为第 i 个源神经元到第 j 个目标神经元的突触连接权重，f_{ij} 为电流因子. 突触电流模型的详细描述可参考文献[8].

突触可塑性是突触属性随时间的改变，是实现脑神经的学习和记忆的理论基础. 其中，脉冲时间依赖可塑性(Spike-Timing Dependent Plasticity，STDP)是常用的脑模拟突触可塑性模型，也是 SNN 常用的非监督学习方法[15]. STDP 模型认为突触权重的改变与前后神经元相对脉冲时间差有关，图 2 显示了统计获得的 60 对脉冲时间差与突触权重改变量的关系[13].

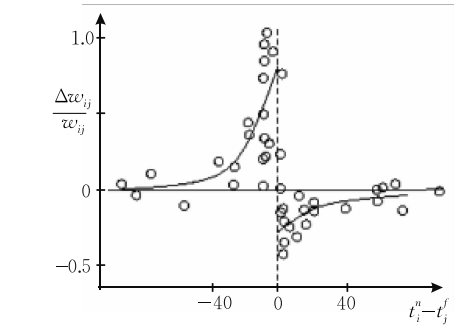


图 2 STDP 函数[13]

常规的权重改变量可以表示为[13]

$$\Delta w_{ij} = \sum_n \sum_f W(t_i^n - t_j^f) \tag{3}$$

其中，Δw_{ij} 为 w_{ij} 的改变量，t_i^n 表示前神经元 i 第 n 个脉冲传送到突触的时间，t_j^f 表示后神经元 j 第 f 个脉冲传送到突触的时间. 因此，t_i^n 和 t_j^f 是脉冲激发时间分别加上各自传到突触的延时 d_A 和 d_D. 突触延时满足关系 $d_{ij} = d_A + d_D$，其中 d_{ij} 为脉冲传送的总延时，如图 3 所示. $W(x)$ 为 STDP 函数，常用的 STDP 函数可表示为

$$\begin{aligned} W(x) &= A_+ \exp(-x/\tau_+), \text{ for } x>0 \\ W(x) &= -A_- \exp(x/\tau_-), \text{ for } x<0 \end{aligned} \tag{4}$$

其中，A_+ 和 A_- 与权重有关，τ_+ 和 τ_- 为时间常数，一般在 10 ms 量级.

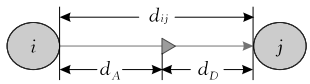


图 3 脉冲传送延时

2.3 SW26010 异构众核处理器基本架构

神威太湖之光超算系统采用 40 960 块 SW26010 异构众核处理器，通过计算插件板、计算超节点和计算机仓等模式进行系统扩展[16]. 本论文侧重开发单核组模拟软件，因此仅关注 SW26010 单核组架构与性能.

如图 4 所示，SW26010 众核芯片包括四个核组，每个核组由 1 个主核和 64 个从核组成. 每个核组连接 8 GB 主存，其中从核可利用 DMA 直接访问主存. 每个从核配置 64 KB 局部存储(LDM)，且 LDM 分配由程序员手动控制.

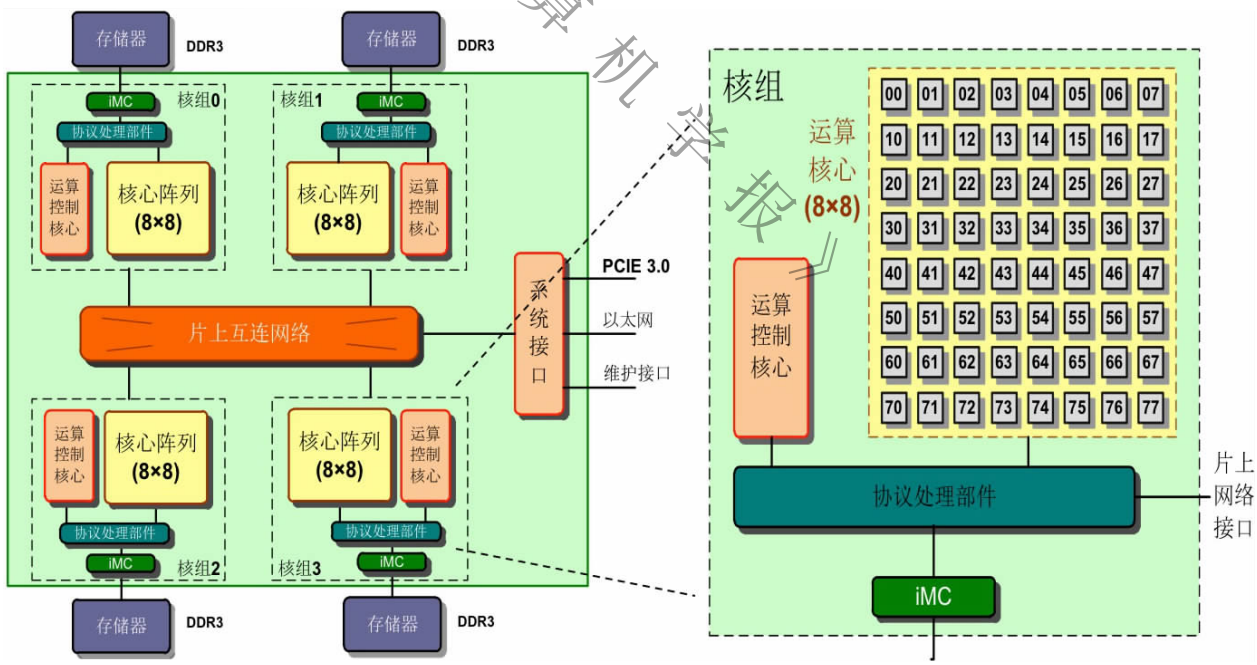


图 4 SW26010 芯片架构

SW26010 芯片主频为 1.45 GHz，每时钟周期主核完成 16 次浮点操作，从核完成 8 次浮点操作，浮点运算理论峰值分别为 23.2 GFlops 和 11.6 GFlops，单个核组总计算能力为 765.5 GFlops.

SW26010 在实现 SNN 计算中具有两点明显优势：(1) 局部存储(LDM). 每个从核有 64 KB LDM，64 个从核共有 4 MB LDM，且由编程人员手动控制. 这与常用的 GPU 相比，具有明显优势. 一般每个流多处理器(SM)具有共享存储 64 KB，且由许多流处理器(SP)和多个线程块(block)共享；(2) 寄存器通信. 支持从核之间的快速通信，相当于局部存储的共享. 每次寄存器通信，点对点实测耗时约 20 时钟周期，行/列广播实测耗时约 27 时钟周期，每次传输 256 位数据.

但是 SW26010 在实现 SNN 中也存在明显的访存问题. 从核访问主存通过 DMA 访存通道进行，访存延迟在 1000 时钟周期量级. DMA 由 64 从核共享通道，因此从核并行的 DMA 访存压力较大. 并且 DMA 访存粒度对带宽影响很大，只有访存粒度足够高时(约 1 KB)，才能获得较好的访存带宽. 另外，SW26010 当前可用的编程和编译环境有限，仅支持 C、C++、Fortran 语言.

3　SNN 随机访存与存储结构分析

3.1　SNN 模拟基本流程

图 5 显示了 SNN 模拟实现的基本流程，是多数 SNN 模拟软件共同使用的基本框架.

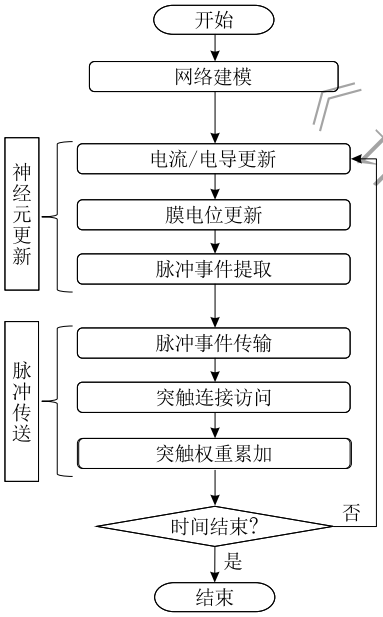


图 5　SNN 模拟流程图

SNN 模拟计算任务主要包括两部分：神经元更新(Neuron update)和脉冲传送(Spike delivering). 神经元更新以求解常微分方程(1)为主，为计算密集型. 各神经元之间的更新相互独立，因此计算量与神经元数量接近线性相关. 脉冲传送是 SNN 模拟的复杂部分，为访存密集型. 当某前神经元发生脉冲时，脉冲传送过程中需要对前神经元连接的突触信息进行访问和权重累加等操作. 突触连接数据的存储排序和计算算法直接影响访存性能.

SNN 模拟存储数据主要包括两类：神经元数据和突触连接数据. 神经元数据较少，一般在 MB 量级，与 LDM 空间为同等规模. 而突触连接数据规模很大，一般在 GB 量级，比神经元高三个数量级. 同时，在计算过程中随机访存主要出现在 MB 量级的神经元数据读写中，而主要的突触连接数据一般可实现连续访问.

基于以上特征分析，并结合 SW26010 处理器架构状况，我们设计合适的存储和计算安排. 即如果将神经元数据长时间保留在 LDM，将突触数据存储在主存，且避免其非连续访问，可以解决主要访存问题，大幅提升计算性能.

然而，SNN 的具体实现比较复杂，有多个问题需要考虑. 为了达到以上目的，下面对面临的访存问题和存储架构等主要问题进行详细分析.

3.2　随机访存形成原因分析

随机访存是当前影响现有 SNN 软件计算性能的主要因素，因此有必要对其进行详细分析.

导致 SNN 随机访存的原因主要有两种：(1) 连接的随机性和非局部性；(2) 突触可塑性(STDP)的双向相关性. 这也是导致 SNN 实现复杂的两种主要因素.

(1) 连接的随机性和非局部性

对于第一种原因，如图 6 所示，当某前神经元发生脉冲时，需要将其连接的大量突触信息传送给许多后神经元. 在此过程中，相关突触连接信息一般可进行连续访问，而相关的后神经元信息分布具有随机性和非局部性. 这时，处理器需要在 MB 量级空间内搜索这些神经元. 但是，对于已有的 CPU/GPU 软件，MB 量级空间的直接搜索存在严重的随机访存. 现有的 CPU 成熟软件 NEST 尚未进行随机访存问

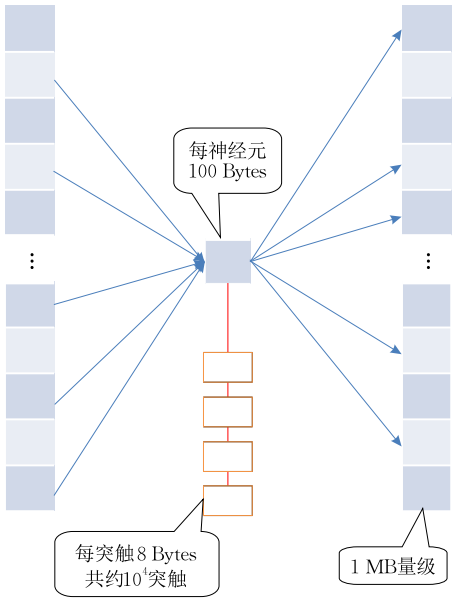


图 6　神经元与连接的存储情况

题的相关分析与优化，也存在明显的随机访存问题.而对 GPU 软件，随机访存是其明显弱点，还不能有效解决.

为了解决这个问题，并结合 SW26010 局部存储 LDM 属性，我们将频繁出现随机访存的神经元数据长时间停留在 LDM，使随机访存只出现在高速的 LDM.这样可有效解决由于该原因导致的随机访存问题.

(2)突触可塑性(STDP)的双向相关性

虽然脉冲的网络传输是单向的，但是可塑性(STDP)算法中的权重更新同时与前神经元和后神经元相关，即双向相关性.如图 6 所示，当某神经元发生脉冲时，需结合所连接的后神经元的脉冲序列更新后连接的突触权重，同时也需结合所连接的前神经元的脉冲序列更新前连接的突触权重.对于前者，可连续访问突触信息，并且可在脉冲传送过程中同时进行.而对于后者，需要对其他神经元的突触连接数组进行非连续性的搜索与访问，导致可塑性计算存在严重的随机访存问题.

现有的 STDP 权重更新有两种方式，即发生后即更新和需要前再更新.

对于发生后即更新，即当神经元激发脉冲后，就对相关的前后连接突触进行更新.该更新分为前神经元和后神经元的突触更新两部分.然而无论突触按前神经元还是按后神经元排序，两部分至少有一部分需要随机搜索和访存.该随机访存对计算性能影响很大.NeMo 和 CARLsim 只计算权重改变量，而不进行即时更新，以避免搜索的复杂性.这也导致两种软件均不支持 STDP 即时更新.

需要前再更新是 STDP 应用的另一种有效方法.NEST 利用该方法实现了权重即时更新[14].如图 6 所示，我们研究发现，在实际情况中，虽然后神经元激发脉冲后，就应该更新其前连接突触的权重，但是此权重值不会即刻用于计算，直到前神经元的脉冲再次激发.因此，可以等到前神经元脉冲再次激发时，即此权重值再次用于计算时，再将其进行更新.该方法避免了后神经元对突触的随机搜索.该方法的主要局限在于限制了轴突延时的取值范围，有关局限性详细信息可参考文献[14].

因此，本文对可塑性计算设计了需要前再更新的方法，以避免对主存突触信息的随机搜索，解决可塑性导致的随机访存问题.

基于以上分析与处理，本文可实现随机访存只发生在局部缓存 LDM 中，以明显提高 SNN 计算性能.

3.3 突触延时

突触延时的存在进一步加剧了神经元数据访问的随机性和非局部性.当神经元脉冲发生时，其所连接的突触信息并不能即刻传送至目标神经元，即存在突触延时.并且，不同突触的延时各不相同，延时分布具有随机性.突触延时的存在使 SNN 明显变得更加复杂.现有异构加速脑模拟软件对此的支持均不是很好.NeMo 和 CARLsim 只支持 1 ms 时间步长的延时和脉冲计时，计时精度均较低，明显影响模拟计算结果.HRLSim 只能支持几个特殊的延时值[8]，延时计时精度较低.

突触延时一般有两种处理方法：环形缓冲法和延迟传送法.

环形缓冲法较简单，以 NEST 为代表.该方法为各神经元配置一个环形缓冲区，缓冲区大小与时间步长和最大延时有关.当前神经元激发脉冲时，将其连接的突触信息传送到目标神经元，并依据突触延时值在缓冲区中向后偏移一定距离再累加电流信息.当偏移后的地址超出缓冲区，则依据缓冲区大小对累加地址进行一个周期的前移.然而该方法需对单个神经元配置较多的缓冲存储空间，而 LDM 存储空间有限，这就会导致一般情况下的 LDM 不足以长时间保存该缓冲区.因此，该方法不适用于 SW26010 的 LDM 存储.

延迟传送法是指在脉冲激发时，不是即刻发送到后神经元，而是推迟到后神经元需要时再发送.NeMo 和 CARLsim 均使用该方法处理突触延时.该方法需要将突触信息依照延时进行重新排序，以便判断哪些突触需要传送.该方法需要将脉冲信息保存到最大延时的时间才能释放，因此脉冲信息占据空间比环形缓冲法要大.然而，该方法要求延时和脉冲计时均以最小延时值为时间步长间隔，导致 SNN 算法计时精度只能在 1 ms 量级，这明显影响 SNN 模拟计算精度.

对于以上延时的分析，并结合 SW26010 实际情况，本文设计了环形脉冲和延迟传送的联合应用，以同时解决 LDM 空间不足和 SNN 计算精度过低问题.这样就能实现对高精度时间步长 SNN 模拟的支持.

3.4 数据存储结构设计

结合以上问题的具体分析以及解决方案设计，该部分基于环形缓冲和延迟传送联合应用方法，列出了本文设计的实际存储结构模式.

首先，以 10^4 个神经元全连接网络为例，分析两部分数据的空间需求情况. 神经元基本组成包括 v、u、a、b、c、d、I 这 7 个浮点型数据，占据 28 个字节. 对于复杂模型和高精度模拟，神经元还会增加电导属性、电流缓冲、脉冲信息等空间，一般约 100 字节. 这样，10^4 个神经元约需 1 MB 存储空间，明显小于单核组 LDM 空间总大小. 突触信息基本组成包括权重、延时、目标神经元编号，其中后两者可合并存储，因此每个突触最少需要 8 字节存储. 10^4 个神经元连接的突触数可达到 10^8 个，因此突触信息需要至少 800 MB 存储空间. 所以神经元和突触存储空间数据量相差约 3 个数量级.

因此，神经元数据空间很小，并且在脉冲信息传送计算中需要随机搜索访存. 为此，可将神经元数据长时间停留在 LDM. 神经元数据在 LDM 存储具有明显优势，可能出现的随机访存主要是神经元数据的电流累加缓冲区，将其停留在 LDM 即可确保主要随机访存发生在 LDM，而 LDM 数据的随机访存对计算性能影响较小.

而突触数据量明显较大，因此将其存储在主存空间，且尽可能减少非连续访问. 为实现环形缓冲和延迟传送的联合应用方法，提高计算精度同时节省 LDM 空间，我们将延时分为两部分存储：突触连接存储和缓冲区存储. 图 7 显示了突触连接数据的存储格式. 对于突触延时 d_{ij} 与最小延时 $d_{\min}$ 的比值分为两部分，其中整数部分通过延迟传送方法表述，如图 7(a)所示；小数部分通过环形缓冲算法表述，如图 7(b)所示. 图 7(a)为突触连接信息的三维列表排序情况(以 $d_{\min}=1$ ms 为例)，其中第一维为前神经元编号(pre-nid)，第二维为突触延时(整数部分)，第三维为突触信息(包含突触延时小数部分). 第三维的每个突触连接元素为一个 8 字节结构体，如图 7(b)所示，其中 4 个字节存储浮点型权重值，3 字节存储目标神经元编号(post-nid)，1 字节存储延时小数部分. 由此可知，小数部分可支持延时的最高精度为 1/256 ms.

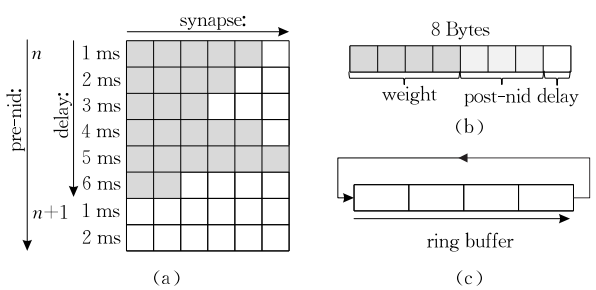


图 7　环形缓冲和延迟传送的联合应用

图 7(c)为环形缓冲区，每个神经元配置一个，用于统计突触电流. 该缓冲区长时间存储在 LDM，是脉冲传送过程中权重累加计算的主要访存变量. 环形缓冲区长度为 $d_{\min}/dt$，其中 dt 为时间步长. 本文支持的时间步长取值包括 $d_{\min}$、$1/2d_{\min}$、$1/4d_{\min}$、$1/8d_{\min}$、$1/16d_{\min}$，对应的环形缓冲区长度分别为 1、2、4、8、16.

环形缓冲和延迟传送的联合应用是对两种算法优点的发挥与缺点的折衷，适用于 SW26010 众核加速计算.

基于以上的计算和存储设计，我们开发了 SNN 异构加速软件 SWsnn，实现了有效的高性能高精度脑神经网络模拟计算. 图 8 显示了 SWsnn 脑网络模拟计算中脉冲事件分布情况的计算结果.

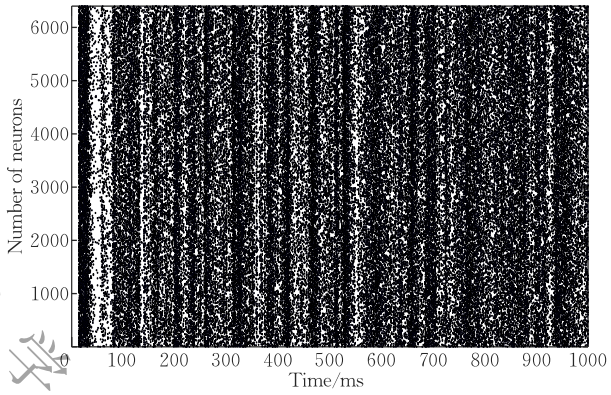


图 8　脉冲事件分布图

4　SWsnn 数值实验和精度/性能分析

4.1　不同软件数值实验结果对比

下面测试 SWsnn 计算结果，并与开源软件 CARLsim、GeNN 对比. SWsnn 和 CARLsim、GeNN 使用完全相同的模型参数和输入脉冲序列. 网络模型配置 10^4 神经元，其中兴奋性和抑制性神经元占比分别为 80% 和 20%. 每个神经元连接 10^4 个突触，即全连接. 由于网络规模偏大，现有的 Quadro K620 GPU 显存难以支持，因此该模型选择 CARLsim 和 GeNN 的 CPU 模式计算. 为方便对比，网络模型不使用随机设置和输入. 我们将神经元参数统一设定为 $a=0.02$，$b=0.2$，$c=-65$，$d=8.0$，膜电位和膜恢复分别使用 -65 和 -13 进行初始化，脉冲激发频率控制在 10 Hz 左右.

首先，对比 SWsnn 和 CARLsim 的计算结果. 由于 CARLsim 只支持 1 ms 时间步长，因此 SWsnn 和 CARLsim 均采用 1 ms 时间步长精度. 最小延时

和最大延时均设为 1 ms. 图 9 为两种计算结果对比，由图可知，两种软件计算结果完全一致.

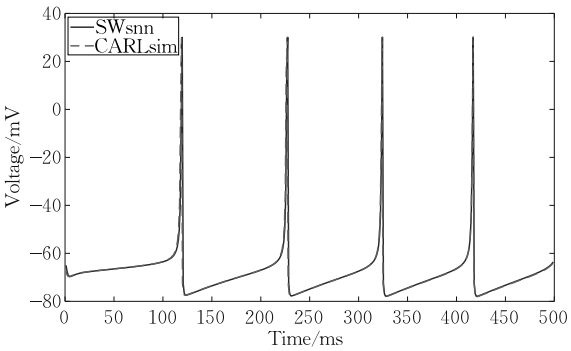


图 9 SWsnn 和 CARLsim 神经元膜电位对比

其次，对比 SWsnn 和 GeNN 的计算结果. GeNN 通过代码生成模式产生模拟源代码，通过检查源代码确保了神经元和突触模型参数完全一致. SWsnn 和 GeNN 均支持高精度时间步长，时间步长均采用 1/8 ms. 最小延时均设为 1 ms，最大延时均设为 10 ms. 图 10 为两种计算结果对比，由图可知，两种软件计算结果完全一致.

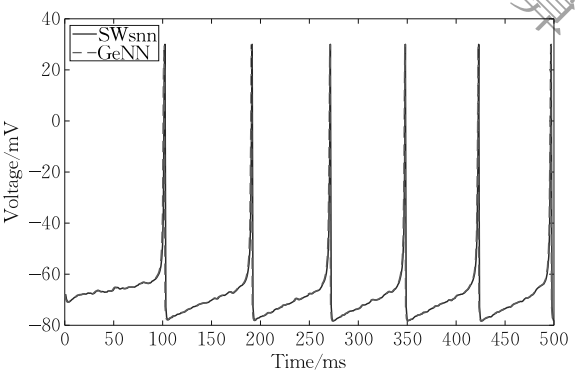


图 10 SWsnn 和 GeNN 神经元膜电位对比

本次实验 SWsnn 与两种软件设计的神经元更新算法和突触电流算法是一致的，但是数据结构、数据排序等相互之间有很大差别. 因此，这三种软件结果一致，但计算性能不同. 另外，图 10 比图 9 脉冲激发频率偏高，这与时间步长精度的差别有关.

4.2 SWsnn 计算精度分析

SNN 模拟的离散误差主要包括两部分：微分方程计算误差、脉冲时间传输误差[9]. 现有的多数 SNN 模拟精度研究主要针对前者[17-19]，而对后者研究较少[9]. 然而，脉冲时间传输误差对 SNN 模拟影响比微分方程计算误差明显更大，同时影响因素更多且复杂. 本文不过多讨论离散计算误差，仅比较 SWsnn 不同时间步长计算精度的差别.

图 11 显示了不同时间步长设置下的 SNN 模拟神经元膜电位分布情况. 由图可知，当时间步长足够小时(1/16 ms)，膜电位离散更新足够密集，微分方程计算和脉冲时间传输均比较精准，这时具有较高的模拟精度. 随着时间步长的逐步增大，微分方程计算误差逐步增大. 并且受脉冲和延时计时精度以及时间步长精度的影响，脉冲传输到目标神经元的时间逐步后移，导致脉冲时间传输误差也逐渐增大. 因此，时间步长的设置对 SNN 模拟精度影响明显. 同时，时间步长设置对脉冲激发频率也有一定影响，当时间步长设置较大时，平均脉冲激发频率偏低. 另外，当时间步长为 1 ms 时，模拟计算误差过大. 因此，1 ms 时间步长不适用于 SNN 模拟. 鉴于模拟精度与计算需求考虑，1/8 ms 或 1/4 ms 时间步长是折衷的有效设置.

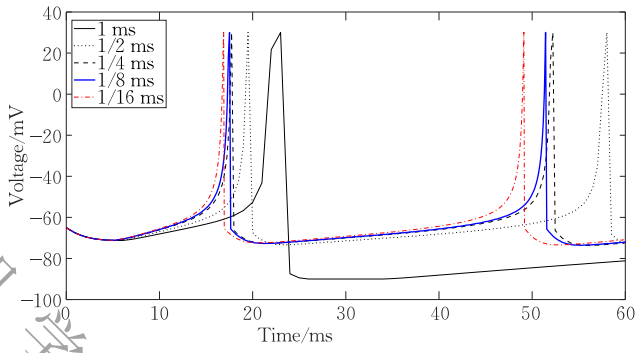


图 11 不同时间步长 SWsnn 模拟神经元膜电位

4.3 SWsnn 计算性能分析

SNN 模拟的神经元更新和脉冲传送两部分的计算框架有明显差别. 神经元更新为计算密集型，各神经元相互独立更新. 脉冲传送为访存密集型，算法复杂多变. 为方便性能分析，将脉冲传送分为两部分：DMA 访存、权重累加. 因此，神经元更新、DMA 访存、权重累加是 SNN 模拟的三个主要任务，也是需要重点分析与优化的主要部分.

对于确定规模的网络模型，影响 SWsnn 计算耗时的因素主要有三种：时间步长(time step)、激发频率(firing rate)、最大延时(max delay). 图 12 显示了三种影响因素对不同计算任务耗时的影响. 在默认情况下，三个因素设置为：时间步长 1/8 ms、激发频率 12 Hz、最大延时 10 ms. 另外，激发频率还受输入信息和计算精度影响，可能会有小的变动.

图 12(a)为计算耗时随时间步长的变化. 由此可知，时间步长与神经元更新耗时成反比，而对 DMA 访存和权重累加影响很小. 但是，由于其影响计算精度，导致其对激发频率有影响. 对于 1/16 ms、1/8 ms、

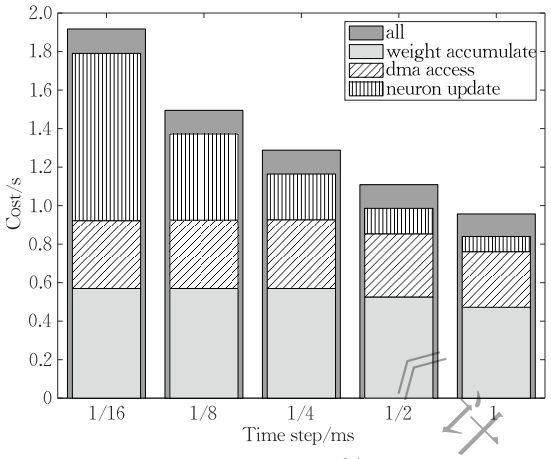
1/4 ms 情况，激发频率均为 12 Hz；对于 1/2 ms 和 1 ms，激发频率分别为 11 Hz 和 10 Hz. 因此，后两种情况的 DMA 访存和权重累加耗时有所降低.

图 12(b)为计算耗时随激发频率的变化. 由此可知，激发频率与 DMA 访存和权重累加耗时均线性相关，而对神经元更新影响很小.

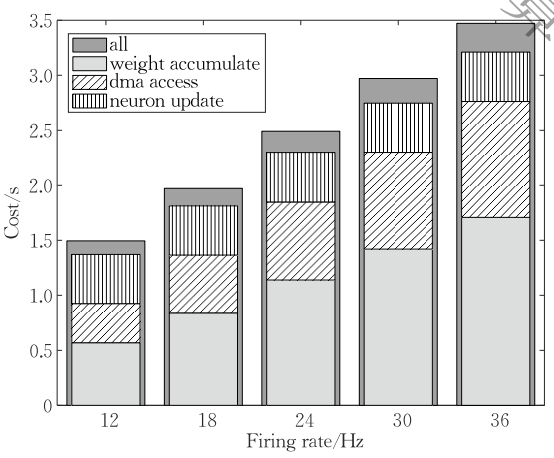
图 12(c)为计算耗时随最大延时的变化. 由此可知，最大延时主要影响 DMA 访存耗时，而对神经元更新和权重累加影响很小. 这是由于最大延时与 DMA 访存次数线性相关. 随着最大延时的增加，DMA 访存次数逐渐加大，而访存粒度逐渐减小，这对 DMA 访存带宽影响甚大.

另外，图 12 立方图的深灰色部分代表模拟计算的全部耗时，它比三个任务耗时总和多出的部分是其他辅助计算，如脉冲事件信息广播传输等.

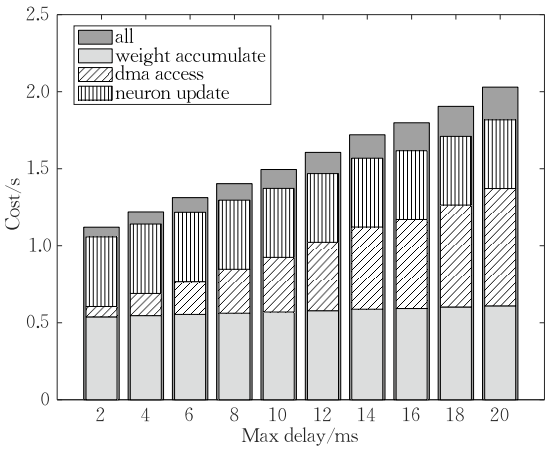
综上所述，时间步长主要影响神经元更新耗时，激发频率与 DMA 访存和权重累加耗时均线性相关，最大延时主要影响 DMA 访存耗时. 基于该性能分析，我们将对 SWsnn 进行性能优化.



(a) Firing rate 12 Hz; Max delay 10 ms



(b) Time step 1/8 ms; Max delay 10 ms



(c) Time step 1/8 ms; Firing rate 12 Hz

图 12　三种因素对三部分计算耗时的影响

5　SWsnn 性能优化

5.1　适用于 SWsnn 的主要优化方法

基于神经元更新、DMA 访存、权重累加的性能分析，同时结合 SW26010 处理器所支持的优化技术，我们对三部分进行各自的性能优化. SW26010 处理器支持常用的计算优化技术，然而，现用的 SW26010 编译器 sw5cc 自动优化不成熟，多数需要手动编程优化.

SW26010 常用的优化技术主要包括 SIMD 向量化、DMA 访存隐藏等[20-22]，也是 SWsnn 优化中主要采取的优化方法. SW26010 处理器支持 256 位 SIMD 向量化. SIMD 向量化操作指令主要包括两类：浮点向量化和定点向量化. 浮点向量化支持 4 个单精度或双精度浮点的并行运算操作，定点向量化支持 8 个 4 字节或 1 个 256 位定点运算. DMA 访存由从核控制，且为非阻塞访存. 通过重叠 DMA 访存传输和从核计算时间，可实现 DMA 访存隐藏.

(1) SIMD 向量化优化

神经元更新是计算密集型，对神经元信息按序更新，且期间神经元之间相互独立，没有依赖. 因此该计算易于并行与扩展，SIMD 向量化技术非常适用于神经元更新. 神经元更新优化主要利用浮点向量化进行，可实现 4 个神经元浮点数据向量化操作. 为方便向量化优化，我们对 SWsnn 神经元数据进行整理，减少结构体使用，确保可向量化变量的连续访问. 表 1 显示了 SIMD 向量化优化方法和加速情况. 由表 1 可知，SIMD 向量化在神经元更新的优化最明显，最高可加速约 3 倍. 在神经元更新阶段，由于存在电流/电导更新和脉冲事件提取等串行计算，导

致向量化未达到接近 4 倍的加速效果.

表 1 向量化优化方法与加速情况

	优化方法	加速情况
神经元更新	浮点向量化	2.0～3.0 倍
权重累加	定点向量化	1.3～1.4 倍

权重累加也可利用 SIMD 向量化优化,但是权重累加是访存密集型,对局部存储(LDM)的环形缓冲区进行随机搜索和访问,因此可向量化程度不够明显.我们主要将定点向量化用于权重累加的地址定位计算,实现加速不是特别明显.由表 1 可知,SIMD 向量化对权重累加有一定加速,平均加速约 1.38 倍.

(2) DMA 访存隐藏优化

DMA 访存隐藏的优化情况与可重叠的计算耗时有关,即权重累加计算耗时.在权重累加计算不做优化的情况下,可隐藏大部分 DMA 访存耗时.然而,权重累加进行向量化优化后,可隐藏的 DMA 访存耗时明显变小.即使如此,DMA 访存隐藏优化仍在一定程度上提高了整体性能,具有实际优化意义.

DMA 访存隐藏不够明显,最多可隐藏约 0.2 s 耗时.这主要有两点原因:(1) 多线程并行本身隐含访存隐藏效果,在某一线程访问期间,许多其他线程仍在计算;(2) DMA 访存次数过多,访存粒度较小.随着延时增大、神经元增多等,访存稀疏性会进一步加剧,这会导致 DMA 访存次数的增加和粒度的减小.

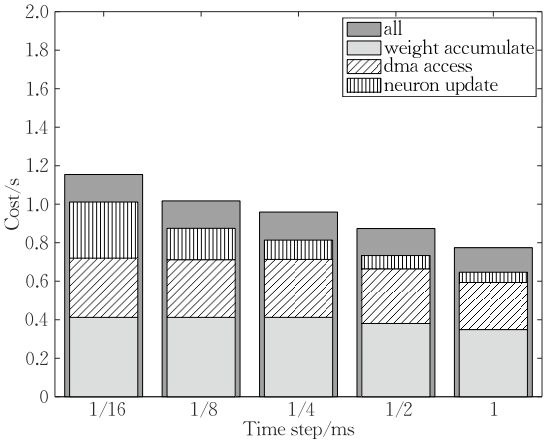
针对这个问题,SW26010 的主要优化方案是主核协助整理 DMA 访存数据,以增加访存粒度,减少访存次数.但是实验表明,该方法当前尚不适合 SWsnn 优化.在脉冲传送阶段,计算任务主要是权重累加,简单且快速.而该优化方法中主核需要对众核线程的计算数据承担整理连接任务,但是主核整理速度远不如众核并行计算速度,即使在主核使用向量化、循环展开等优化操作后.实验表明,主核整理数据速度比从核权重累加速度慢 8～10 倍,不适用于 SWsnn 优化.因此,DMA 访存仍是我们需要进一步研究的重点.

(3) 其他优化

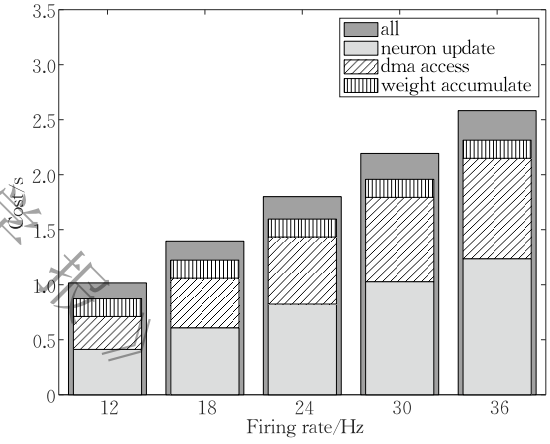
另外,我们还通过减少分支等操作进一步提升了加速效果.现有的 SNN 软件在计算热点(即权重累加)阶段均存在多项可优化分支语句.我们尽可能减少计算热点分支操作次数,计算速度比其他 SNN 软件要快得多.

5.2 优化后的性能分析

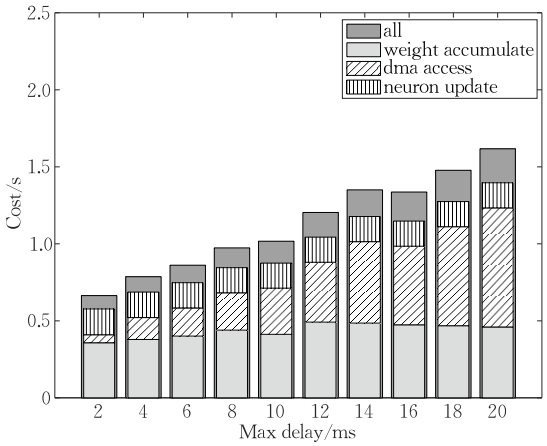
图 13 为优化操作后的各部分计算耗时情况,与图 12 的优化前耗时情况相对应.由图 13 可知,当前在 10^4 神经元全连接网络、时间步长 1/8 ms、最大延时 10 ms、激发频率 12 Hz 情况下,优化后的 SWsnn 可实现实时模拟.



(a) Firing rate 12 Hz; Max delay 10 ms



(b) Time step 1/8 ms; Max delay 10 ms



(c) Time step 1/8 ms; Firing rate 12 Hz

图 13 优化操作后三部分计算耗时情况

5.3 优化后的 SWsnn 大规模计算实验

下面,我们利用优化后的 SWsnn 进行几种的测

试与比较. 首先，对不同规模神经元网络进行测试. 如图 14 所示，该图显示的是每神经元均连接 10 000 突触，时间步长均为 1/8 ms，激发频率均为 12 Hz，最大突触延时均为 10 ms 的不同神经元规模网络模拟计算耗时，由此可见，随着神经元规模的增大，神经元更新、权重累加均接近线性分布. 而 DMA 访存耗时不具有线性属性. 这是由于受 LDM 存储空间限制，当神经元增至 40 000 以后，神经元数据不能再常驻在 LDM，需要进行更替，增加了 DMA 压力. 同时随着神经元的增多，突触连接的稀疏性也加强.

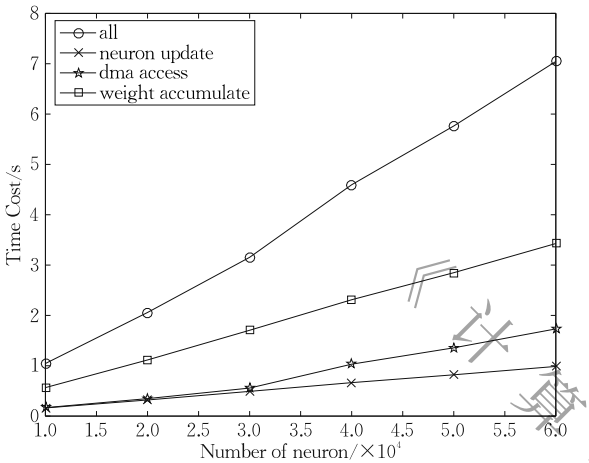


图 14　不同规模网络计算耗时(每神经元连接 10 000 突触)

受主存空间(8 GB)影响，当每神经元连接 10 000 突触时，SWsnn 最多可模拟 60 000 神经元规模网络. 为增加可模拟的神经元规模，我们对每神经元连接 5000 突触的网络进行了模拟，如图 15 所示. 由此

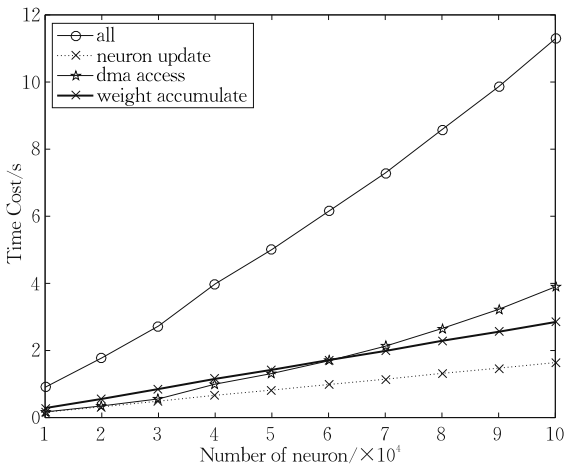


图 15　不同规模网络计算耗时(每神经元连接 5000 突触)

可知，神经元更新、权重累加仍具有线性耗时属性，而 DMA 访存耗时与神经元数量接近抛物关系. 并且规模较大时，DMA 访存耗时已超过计算耗时.

5.4　SWsnn 与其他 GPU 软件的性能对比

该部分将 SWsnn 与开源 GPU 软件 CARLsim 和 GeNN 分别进行计算性能对比.

首先，我们对 SWsnn 和开源的 CARLsim 进行计算性能对比. SWsnn 在 SW26010 的单核组上运行，浮点计算能力 765.5 GFlops；CARLsim 在 Quadro K620 GPU 上运行，单精度浮点计算能力 813 GFlops. SWsnn 和 CARLsim 均为单精度计算. 由于 K620 显存较小(2 GB)，本次比较的网络规模为 6000 神经元全连接网络. 表 2 和表 3 分别显示了最大延时为 1 ms 和 10 ms 情况下的耗时对比. 由此可见，SWsnn 比 CARLsim 计算速度快 10 倍左右. 如果考虑 SWsnn 在高精度情况下的激发频率偏高情况，则其实际计算速度比 CARLsim 快 10 倍以上.

表 2　最大延时为 1 ms 时 SWsnn 和 CARLsim 性能对比

Software	Time Step/ms	Firing Rate/Hz	Cost/s	Speedup
CARLsim	1	10	2.806691	1.0
SWsnn	1	10	0.262451	10.7
SWsnn	1/8	12	0.353534	7.9

表 3　最大延时为 10 ms 时 SWsnn 和 CARLsim 性能对比

Software	Time Step/ms	Firing Rate/Hz	Cost/s	Speedup
CARLsim	1	10	5.875452	1.0
SWsnn	1	10	0.436893	13.4
SWsnn	1/8	12	0.582934	10.1

然后，我们与 GeNN 软件性能进行简单对比. 除了与 CARLsim 对比外，我们还比较了近几年发布的 GPU 软件 GeNN. 文献[7]列出了 GeNN 针对两个 SNN 模型计算耗时信息，本文选择与我们规模相近的模型进行对比. 我们提取文献[7]计算性能较好的几种情况进行估测与对比，表 4 列出了其中可对比的相关信息. 由此可知，SWsnn 与 GeNN 的计算模型规模相近，但运行 GeNN 的 GPU 计算能力比 SW26010 单核组计算能力明显要高. 并且，SWsnn 模拟的激发频率明显高于 GeNN. 综合计算能力、计算规模等各方面影响的估测，对于权重累加计算速度，优化后的 SWsnn 比 GeNN 要高 4 倍以上，而对于总耗时，则约快 8 倍.

表 4　SWsnn 和 GeNN 性能对比

Software	Neurons	Synapses	GPU	Max GFLOPS	Time step/ms	Firing rate/Hz	Neuron update/s	Weight accumulate/s	All cost/s
GeNN	77169	0.3×10^9	1050 Ti	2100.0	0.1	<10	2.0	2.0	14.0
GeNN	77169	0.3×10^9	K40c	4290.0	0.1	<10	1.2	1.2	4.0
GeNN	77169	0.3×10^9	V100	14000.0	0.1	<10	0.4	0.4	1.8
SWsnn	40000	0.4×10^9	—	765.5	1/8	12	0.63	1.64	4.0

6 讨 论

本文以随机访存对计算性能的影响为主要研究路线，详细分析了 SNN 中随机访存问题的形成原因. 结合 SW26010 结构特征，设计了合适的计算和存储方法，确保随机访存只发生在高速的局部存储中，同时解决了异构面临的计算精度问题.

有关 SNN 面临的随机访存问题，据我们所知，已有 SNN 软件与文献未曾进行详细分析与优化，而随机访存是影响其计算性能的最主要因素之一. 因此，本文通过解决随机访存问题开发我们自己的高性能高精度软件 SWsnn 具有重要意义.

当前我们开发的 SWsnn 尚处于初始阶段，还存在几个主要问题正在进一步研究：

(1) DMA 访存问题. 与 DMA 访存对应的计算任务为权重累加，简单且快速，导致很难全部隐藏 DMA 访存耗时. 并且随着模拟规模的增大，访存稀疏性加强，致使 DMA 访存耗时超过计算任务耗时. 因此，我们正在搜寻适用的访存优化方法.

(2) 计算规模. 当前 SWsnn 最多支持 8GB 主存的单核组神经元模拟，模拟规模有限. 因此，我们正在进行分布式计算研究，以支持更大规模 SNN 模拟.

(3) 存储和计算均衡问题. SWsnn 初始版本以三维数组存储网络连接数据，这在实际应用中可能面临存储不均匀以及计算规模不均衡问题. 对此我们正在利用变尺度一维存储方式改进 SWsnn，并尽可能减少同步限制，以解决均衡问题.

(4) 向量化问题. 本文设计的方法仅确保了随机访存在 LDM 发生，但并未消除随机访存. 随机访存的存在依然导致向量化难以实现，难以进一步提高实际计算能力. 对此我们正在进行算法研究与探索，以进一步提高 SNN 计算性能.

7 结束语

本文开发了神威太湖之光单核组的高精度高性能脑模拟软件 SWsnn，解决 SNN 模拟的随机访存问题，支持 1 ms、1/2 ms、1/4 ms、1/8 ms、1/16 ms 时间步长，确保脑模拟计算精度. 本文的 SWsnn 比同等规模计算能力的 GPU 上运行的 CARLsim 快 10 倍，而时间步长精度要高 1 个数量级. 在此基础上，我们下一步将发展多核组和分布式并行的 SWsnn 版本，以充分发挥 SWsnn 和神威超算的应用价值.

参 考 文 献

[1] Jordan J, Ippen T, Helias M, et al. Extremely scalable spiking neuronal network simulation code: From laptops to exascale computers. Frontiers in Neuroinformatics, 2018, 12: Article 2

[2] Kunkel S, Schmidt M, Eppler J M, et al. Spiking network simulation code for petascale computers. Frontiers in Neuroinformatics, 2014, 8: Article 78

[3] Kunkel S, Potjans T C, Eppler J M, et al. Meeting the memory challenges of brain-scale network simulation. Frontiers in Neuroinformatics, 2012, 5: Article 35

[4] Igarashi J, Shouno O, Fukai T, et al. Real-time simulation of a spiking neural network model of the basal ganglia circuitry using general purpose computing on graphics processing units. Neural Networks, 2011, 24: 950-960

[5] Fidjeland A K, Roesch E B, Shanahan M P, et al. NeMo: A platform for neural modelling of spiking neurons using GPUs //Proceedings of the 20th IEEE International Conference on Application-specific Systems, Architectures and Processors. Washington, USA, 2009: 137-144

[6] Beyeler M, Carlson K D, Chou T S, et al. CARLsim3: A user-friendly and highly optimized library for the creation of neurobiologically detailed spiking neural networks//Proceedings of the IEEE 2015 International Joint Conference on Neural Networks. Killarney, Ireland, 2015: 21-29

[7] Knight J C, Nowotny T. GPUs outperform current HPC and neuromorphic solutions in terms of speed and energy when simulating a highly-connected cortical model. Frontiers in Neuroscience, 2018, 12: Article 941

[8] Minkovich K, Thibeault C M, O'Brien M J, et al. HRLSim: A high performance spiking neural network simulator for GPGPU clusters. IEEE Transactions on Neural Networks and Learning Systems, 2014, 25(2): 316-331

[9] Henker S, Partzsch J, Schüffny R. Accuracy evaluation of numerical methods used in state-of-the-art simulators for spiking neural networks. Journal of Computational Neuroscience, 2012, 32: 309-326

[10] Sun Jiu-Rong. Introduction to Brain Science. Beijing: Peking University Press, 2001: 29-42(in Chinese)
(孙久荣. 脑科学导论. 北京：北京大学出版社，2001：29-42)

[11] Xu Ling-Feng, Li Chuan-Dong, Chen Ling. Comparison and analysis for neuron models. Acta Physica Sinica, 2016, 65(24): 240701(in Chinese)
(徐泠风，李传东，陈玲. 神经元模型对比分析. 物理学报，2016，65(24)：240701)

[12] Izhikevich E M. Simple model of spiking neurons. IEEE Transactions on Neural Networks, 2003, 14(3): 1569-1572

[13] Sjöström J, Gerstner W. Spike-timing dependent plasticity. Scholarpedia, 2010, 5(2): 1362

[14] Morrison A, Aertsen A, Diesmann M. Spike-timing-dependent

plasticity in balanced random networks. Neural Computation, 2007, 19(6): 1437-1467

[15] Ghosh-Dastidar S, Adeli H. Spiking neural networks. International Journal of Neural Systems, 2009, 19(4): 295-308

[16] Fu H, Liao J, Yang J, et al. The Sunway TaihuLight supercomputer: System and applications. Science China Information Sciences, 2016, 59: 072001

[17] Shelley M J, Tao L. Efficient and accurate time-stepping Schemes for Integrate-and-Fire neuronal networks. Journal of Computational Neuroscience, 2001, 10(11): 111-119

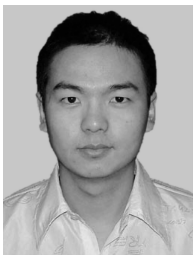
[18] Stewart R D, Bair W. Spiking neural network simulation: Numerical integration with the Parker-Sochacki method. Journal of Computational Neuroscience, 2009, 27(5): 115-133

[19] Valadez-Godínez S, Sossa H, Santiago-Montero R. The step size impact on the computational cost of spiking neuron simulation //Proceedings of the IEEE 2017 Computing Conference. London, UK, 2017: 722-728

[20] Dong W, Kang L, Quan Z, et al. Implementing molecular dynamics simulation on Sunway TaihuLight system//Proceedings of the IEEE 18th International Conference on High Performance Computing and Communications. Sydney, Australia, 2016: 443-450

[21] Duan X, Gao P, Zhang T, et al. Redesigning LAMMPS for peta-scale and hundred-billion-atom simulation on Sunway TaihuLight//Proceedings of the SC18: International Conference for High Performance Computing, Networking, Storage and Analysis. Dallas, USA, 2018: 111-119

[22] Fang J, Fu H, Zhao W, et al. swDNN: A library for accelerating deep learning applications on Sunway TaihuLight //Proceedings of the 2017 IEEE International Parallel and Distributed Processing Symposium. Orlando, USA, 2017: 615-624



LI Xue-Lei, Ph.D., assistant researcher. His research interests include high performance computing and geophysics.

ZHU Xiao-Min, Ph. D., researcher. His research interests include parallel computing and performance optimization.

WEI Yan-Jie, Ph. D., professor. His research interests include high performance computing and computational biology.

FENG Sheng-Zhong, Ph.D., professor, Ph. D. supervisor. His primary research interests include high performance computing and bioinformatics.

Background

Brain simulation is a major method for research of brain. Due to the requirement for brain science experiment, simulation performance becomes more and more important. Now some heterogeneous accelerating simulation softwares have been developed, most of which use GPU, such as NeMo, CARLsim and GeNN. However, these softwares cannot accelerate the simulation obviously. The random access is the major bottleneck problem, which makes the accelerating effect unsatisfactory.

In this paper, we provide suitable computing method based on Sunway TaihuLight. To overcome the access bottleneck, we set new storage and computing method for SNN to keep the random access phenomena always occur on high-speed cache, like local dynamic memory (LDM). This method makes the costs from random access become negligible.

Based on the method above, we have developed our simulation software, SWsnn, which is obviously faster than existing GPU softwares, such as CARLsim and GeNN.

This work was partly supported by the National Key Research and Development Program of China under Grant Nos. 2018YFB0204403 and 2016YFB0201305, the National Science Foundation of China under Grant Nos. U1813203 and 61433012, the National Natural Youth Science Foundation of China under Grant Nos. 41804128 and 31601028, the Shenzhen Basic Research Fund under Grant Nos. JCYJ20180507182818013, GGFW2017073114031767 and JCYJ20170413093358429, the China Postdoctoral Science Foundation under Grant No. 2019M653132, the Chinese Academy of Sciences Key Lab under Grant No. 2011DP173015. We would also like to thank the funding support by the Shenzhen Discipline Construction Project for Urban Computing and Data Intelligence, Youth Innovation Promotion Association, Chinese Academy of Sciences to Yanjie Wei.