•滑坡堰塞湖灾害机理与防控•

DOI:10.12454/j.jsuese.202300725

本刊网刊

一种改进GPU加速策略的物质点分析方法及其在滑坡模拟中的应用

王　斌[1,2],陈鹏林[1,2],王　頔[1],徐顺心[1,3],许子凯[1,4],吴进东[1,3]

(1.中国科学院 武汉岩土力学研究所 岩土力学与工程国家重点实验室,湖北 武汉 430071;2.中国科学院大学 工程科学学院,北京 100049;
3.武汉理工大学 资源与环境工程学院,湖北 武汉 430070;4.长安大学 公路学院,陕西 西安 710064)

摘　要:近年来物质点法发展成为岩土工程领域一种重要的大变形数值模拟方法,被广泛应用于滑坡、溃坝、隧道突水突泥等问题的研究。伴随着应用场景的规模化与复杂化,对于方法本身的精度要求和效率需求继而持续增加,导致其计算成本逐步上升,制约了物质点法进一步在大规模岩土工程问题中的应用。鉴于此,本文提出一种改进图形处理器(GPU)加速策略的物质点法,引入模块化编程思想,采用简洁的多组1维数组的方式进行数据存储结构和基于硬件层级的内存操作管理处理数据竞争,以提高物质点法的模拟效率,解决面向过程的GPU加速策略存在的扩展性问题,并形成高效且灵活的模拟构架。通过模拟铝棒坍塌试验和理想边坡失效过程,结果显示,基于改进GPU加速策略的物质点方法具有较好的并行性,较已有Taichi–GPU物质点法在性能上提升10%左右。最后,应用本文提出的物质点方法模拟再现新磨村滑坡的全过程,得出当物质点数目扩大2.5倍左右时,计算效率提升20倍左右。

关键词:GPU加速;物质点法;滑坡;大变形数值模拟
中图分类号:P642　　**文献标志码:**A　　**文章编号:**2096-3246(2025)04-0001-11

物质点法(MPM)的起源可以追溯至流体力学当中的particle-in-cell(PIC)方法[1],Sulsky等[2]随后通过引入拉格朗日格式来描述物质结构本身的变形,同时保留欧拉格式进行控制方程的求解,从而成功地将物质点法应用到了固体力学领域。欧拉格式的引入使物质点法得以避免传统拉格朗日方法中的网格畸变问题,能够实现大变形问题的模拟。因此,近年来物质点法得到了研究学者的广泛关注,并被成功应用在滑坡[3–5]、挡土墙[6]、边坡稳定性分析[7]、溃坝[8]、突水突泥[9]等岩土工程问题中。

相较于传统有限元法,物质点法需要遍历每个物质点与背景计算网格间的相对位置,从而进行网格信息与物质点信息间的相互映射。因此,相较于有限元的计算,会额外增加一部分计算时间。另外一方面,物质点方法存在一些典型问题,比如粒子跨越网格引起的计算误差等,研究人员可通过使用高阶型函数等形式进行处理[10–12],但这也会进一步增加计算时间。因此,高效模拟方法成为物质点法的重要研究课题之一。

在物质点法的映射、计算和求解过程中,单元间的计算彼此独立,因此十分适用于通过并行计算的方法进行加速,并基于此发展出基于中央处理器(CPU)并行[13]和图形处理器(GPU)并行[14]的加速策略。基于CPU的并行方法可通过超算集群调用大量内核进行并行计算,从而实现加速和大规模计算。该方法已经应用在超短脉冲激光辐射照金属[15]、高速撞击[13]等领域。然而,CPU并行存在计算成本较高、内核交互耗时等问题,限制了其进一步的发展。不同于CPU广泛的应用场景,GPU在硬件层面专为并行计算设计,拥有更多的核心、更低的线程调用成本。尽管GPU单个线程计算能力小于CPU,但其能够快速唤起几百甚至上千内核进行计算,从而大幅提高总体计算效率。近年来,GPU架构的快速发展使得通过使用线程、域分解或它们的某种组合并行化现有算法来加速基于物理的仿真成为可能。Xia等[16]使用的GPU加速平滑粒子流体动力学(SPH)模型,相较于串行CPU计算总体效率提高7～10倍。Zhang等[17]在岩土工程大变形问题中使用GPU加速显式光滑粒子有限元(SPFEM),相较于CPU串行计算,效率提高10倍左右。Dong等[14]将

收稿日期:2023－09－14　　**修回日期:**2023－11－20　　**网络出版日期:**2024－05－28
基金项目:国家自然科学基金资助项目(52379111;51979270);中国科学院率先行动“百人计划”项目
作者简介:王　斌(1989—),男,研究员,博士生导师. 研究方向:岩土计算力学与试验. E-mail: bwang@whrsm.ac.cn

GPU加速方法物质点法应用于岩土工程中，实现了近20倍的加速效果，但粒子映射网格过程发生的数据竞争采用CPU方式处理，会影响计算效率。Hu等[11]开发了Taichi高效并行语言，并应用了移动最小二乘物质点法（MLS–MPM），获得了较好的加速效果。

在物质点法模拟中，由于物质点移动影响，网格信息的数据结构通常是稀疏的，所以对数据结构的处理会大幅度影响内存使用和计算效率。Hoetzlein等[18]在GPU上扩展计算机图形学中最流行的OpenVDB稀疏存储方案，提出了高效内存架构的GVDB点，以支持动态拓扑变化。Setaluri[19]和Gao[20]等分别证明SPGrid是（MPM）中高效的数据结构。物质点信息通常是存储在结构体数组（array of struct，AoS）或数组结构体（structure of array，SoA）中。顺序线程访问顺序内存地址时，SoA促进物质点数据的合并内存访问。然而，为了保持这种高效的数据访问模式，物质点需要在每个时间步之后重新排序，造成SoA在序列化的收集和分散操作中效率较低。相反，AoS更容易映射到物质点的概念，并且由于单个粒子数据的局域性，在非合并内存访问模式的情况下表现良好。然而，这样的内存布局阻止了粒子数据的合并读写，从而在可能合并时显著抑制GPU和向量化CPU的性能。为综合SoA和AoS的优缺点，Wang等[21]以MPM为中心提出了数组–结构–数组的数据结构，使用AoSoA分层方式存储粒子数据，粒子重新组织在低级桶和高级桶中以保存内存访问和数据传输的效率。

尽管在物质点法方面已有相关研究构建了GPU加速的物质点法，并显著提高了模拟效率。然而，这些研究多注重模拟效率以及大规模计算能力，程序设计和模型均较为简单。随着MPM在各种实际问题中的应用，这些面向过程编程的GPU加速策略表现出灵活性差、开发成本高等局限性，制约了MPM理论的不断丰富和发展，也限制了MPM在各种实际问题中的广泛应用。因此，本文提出一种改进的GPU加速策略，避免了传统面向过程编程的GPU代码结构的局限性，构建了高效且灵活的物质点法程序结构，并将其应用于大规模滑坡模拟。

1 GPU加速物质点法

1.1 物质点法基本理论

物质点法通过求解质量守恒、动量守恒以及能量守恒方程获取物体的运动状态、变形过程等信息。在仅考虑固体运动过程的模拟中，质量方程在离散过程中自动满足，无须求解。体系的变形和运动过程需通过求解动量守恒方程来实现。当体系仅考虑运动和变形过程时，由于不存在热交换等能量转换，能量方程亦会自动满足。因此，固体物质点法求解的控制方程可以写为：

$$\rho \boldsymbol{a} = \nabla \boldsymbol{\sigma} + \rho \boldsymbol{b} \tag{1}$$

式中，ρ为材料密度，$\boldsymbol{a}$为加速度，$\boldsymbol{\sigma}$为应力张量，$\boldsymbol{b}$为单位质量上的体积力。式（1）可进一步写为弱形式，即：

$$\int_{\Omega} \rho \omega \boldsymbol{a} \mathrm{d}\Omega + \int_{\Omega} \boldsymbol{\sigma} \nabla \omega \mathrm{d}\Omega = \int_{\Omega} \rho \omega \boldsymbol{b} \mathrm{d}\Omega + \int_{\Gamma} \rho \boldsymbol{\tau} \omega \mathrm{d}\Gamma \tag{2}$$

式中，Ω和Γ分别代表求解区域及边界，ω为试函数，$\boldsymbol{\tau}$为边界上作用面力。物质点法求解时，需先将连续体离散为物质点，连续体的质量可近似表示为：

$$m_p = \int \rho(x,t)\, \delta(x - x_p)\, \mathrm{d}V_p \tag{3}$$

式中，m_p为物质点p的质量，$\boldsymbol{x}$表示该物质点所代表区域内的所有空间位置坐标，$\boldsymbol{x}_p$为物质点p的坐标，V_p为物质点表征的体积，δ为狄拉克函数。因此，连续介质的质量可以表征为：

$$\sum_p m_p = \sum_p \int \rho(\boldsymbol{x},t)\, \delta(\boldsymbol{x} - \boldsymbol{x}_p) \mathrm{d}V_p = \int \rho(\boldsymbol{x},t) \mathrm{d}\Omega \tag{4}$$

将式（4）代入式（2），则控制方程的积分形式转化为物质点上信息的求和：

$$\sum_p m_p \omega(\boldsymbol{x}) \boldsymbol{a}(\boldsymbol{x}) + \sum_p m_p \boldsymbol{\sigma}(\boldsymbol{x}) : \nabla \omega(\boldsymbol{x}) = \sum_p m_p \omega(\boldsymbol{x}) \boldsymbol{b}(\boldsymbol{x}) + \sum_p m_p \omega(\boldsymbol{x}) \boldsymbol{\tau}(\boldsymbol{x}) h^{-1} \tag{5}$$

式中，h为假定的边界层厚度。

在物质点法中，动量方程的求解发生在背景网格上。因此，需要将物质点信息向网格点进行映射。物质点与网格点间的信息映射可以表示为：

$$f(\boldsymbol{x}_p) = \sum_i N_i(\boldsymbol{x}_p) f(\boldsymbol{x}_i) \tag{6}$$

式中，$f(\cdot)$代表质量、动量等物理量，$N_i(\boldsymbol{x}_p)$为形函数，下标p和i分别指代物质点和网格节点。将式（6）代入式（5）可以获取动量方程的矩阵形式：

$$\boldsymbol{M}\boldsymbol{a} = \boldsymbol{F}_{\mathrm{ext}} - \boldsymbol{F}_{\mathrm{int}} \tag{7}$$

式中，$\boldsymbol{M}$为集中质量矩阵，$\boldsymbol{F}_{\mathrm{int}}$与$\boldsymbol{F}_{\mathrm{ext}}$分别表示作用在节点上的内力与外力。因而，动量方程可以通过式（7）进行求解。

1.2 物质点法计算流程

物质点法的计算流程主要由CPU控制，如图1所示。整个物质点法计算过程由多个时间步组成，物理时间等于时间步长乘总时间步。整个计算流程由CPU控制，大规模计算由GPU执行，分为物质点遍历和网格结点遍历。

图2为物质点法求解原理示意图。图2（a）表示重置背景网格并将物质点携带的物理信息映射至网格节点；图2（b）表示拉格朗日计算，在网格结点上求解

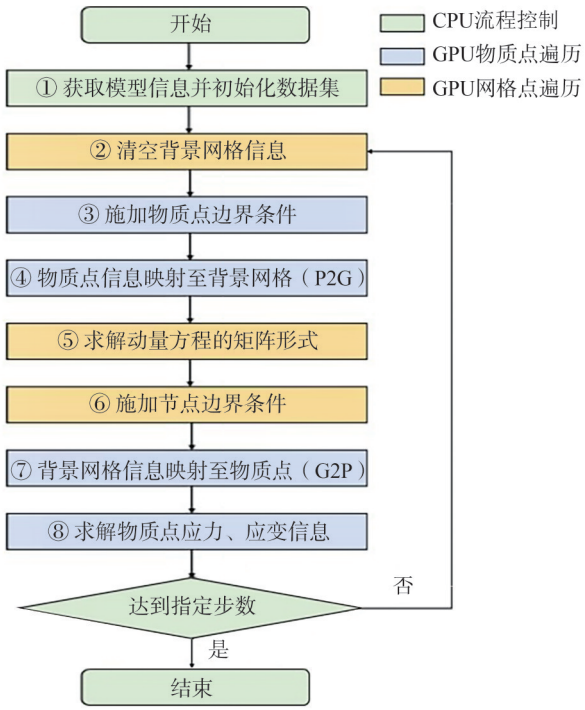


图1　USL物质点法的计算流程

Fig. 1　Calculation processes of USL material point method

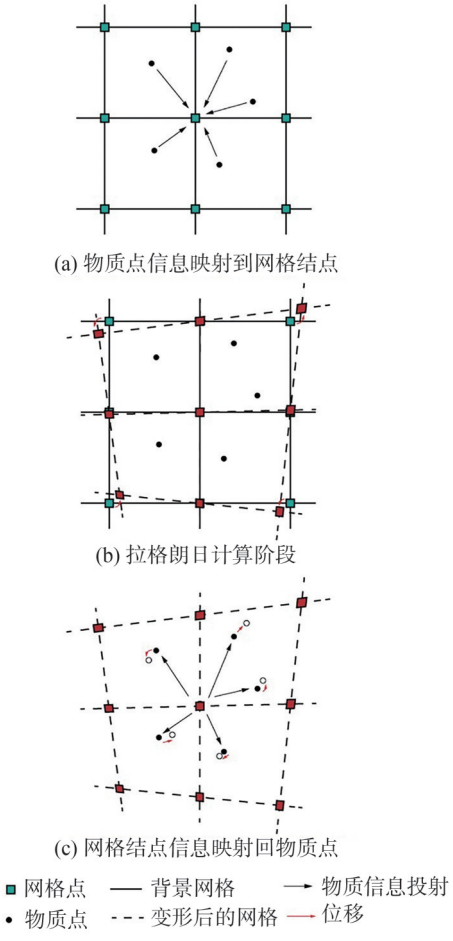


图2　物质点法求解原理示意图

Fig. 2　A schematic illustration of material point method

更新速度、加速度、位移等信息。图2(c)表示更新后的节点再通过形函数映射回物质点，然后更新物质点的物理量(速度，位置等)。图2(b)相当于有限元计算，图2(a)和(c)是物质点特有计算过程，这也使物质点法具备大变形计算能力。

1.3　GPU加速策略

在物质点法计算流程中，诸如粒子信息映射网格节点(P2G)、网格节点信息映射回物质点(G2P)、动量方程求解、应力更新等过程均发生在单元内部，而单元之间的计算彼此独立，因此十分适用于并行加速构架。以P2G过程为例，GPU加速物质点法并列策略如图3所示。对于每个需要进行投射的物质点分配为一个线程，线程内进行单个物质点的P2G映射过程。线程隶属于GPU块，进行计算时，块内的线程数目由程序模型数目及GPU计算性能决定，从而实现最优计算效率。同时，计算流程中，根据求解的对象不同，并行过程可以发生在物质点或网格点上(图1)。然而，上述并行策略虽在计算过程相互独立，但其在内存数据访问、算法扩展性等方面仍存在问题。本文针对上述问题进行了研究改进，改善了现有GPU加速物质点并行方法的计算效率和可扩展性。

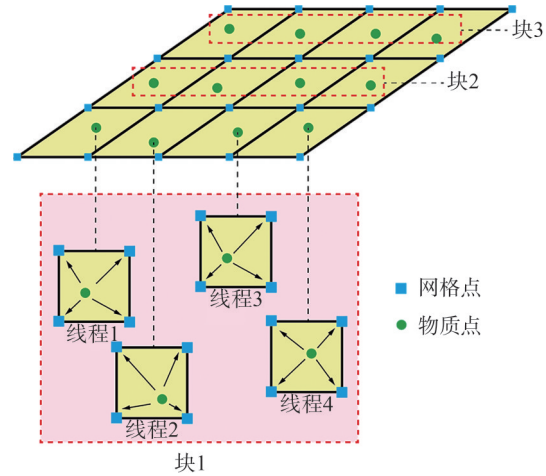


图3　GPU加速物质点法并行策略

Fig. 3　Parallel strategy of GPU-accelerated MPM

1）　模块化编程设计

由于基于GPU加速的编程为面向过程编程模型，在开启核函数后，各计算步骤均需固定编码为线性过程(图4(a))，传统的GPU开发中通常沿用这一策略[14,17]，导致程序编码灵活性差、重复性代码多、开发工作量大，不利于复杂程序功能的实现。如图1所示的编程过程，其②～⑧步均需编码在一个函数内，当使用其他如USF格式时，则需更换函数重新进行编码，同时，对于各类复杂设定的边界条件，则需增设大量逻辑判定，增加了代码的复杂度和工作量。为了避免

这一问题，本文采用了C++面向对象的程序架构，将计算流程拆分为多个核函数并置于各对象内，从而避免了单一面向过程的编程限制，使得每步流程可以实现封装、继承和多态等功能。从图4可知，物质点法计算流程中的①～⑧步被分别编码为独立的核函数，并封装在Data、Boundary、Cast等对象中，由Simulation对象进行管理执行。各对象间计算相互独立，当模拟格式改变时，由于相关GPU编码被内置至相关对象内，只需调整不同的Simulation对象对流程进行管理即可实现对GPU运行过程的管理。另一方面，采用该框架能够利用C++语言的继承、多态等特性，可以根据输入条件建立不同的子对象进行运算，避免流程中进行逻辑判断。例如，在进行第③步时，边界条件往往较为复杂，需要根据模型特性进行单独调整设定，基于面向过程的编码方式需将所有边界条件列出，并逐一判断是否对其执行。而在本文中，由于GPU代码被内置于相关对象内，仅需由Simulation唤起Boundary父对象的执行函数进行边界条件设置，具体执行内容由Boundary根据输入信息进行定义，这使得Simulation和Boundary相关代码编码能够互相独立，能够任意编写不同的边界条件而无须在Simulation的流程控制中进行调整。通过这一特性，本程序可以大量定义出固定、滑轮、反射、振动、自定义、周期性边界以及重力等多种边界条件，还可以定义多种本构方程，如弹性模型、弹塑性模型等，均通过相同的Simulation对象进行控制，提高了程序的灵活性。此外，具有相似功能的边界条件可以通过继承的方式复用部分代码，进一步减少代码开发量。

根据上述介绍可知，本文所提出的并行策略将多个操作步骤模块化，使程序转换为面向对象编程形式，大幅提高了代码的复用率以及程序的易用性。但同时，模块化也导致了GPU并行过程被拆分打断，核函数需多次启动，一定程度上会降低模拟效率。因此，本方法通过减少核函数启动时的数据规模以及合理的流程划分进行了效率优化。效率测试结果显示，这一负面影响并未显著影响程序的计算效率（详见第3节）。

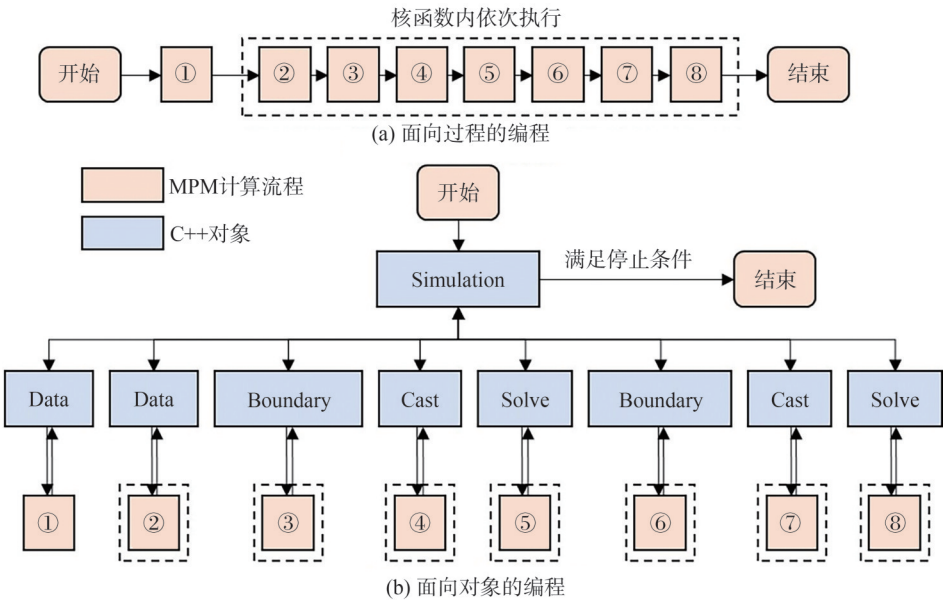


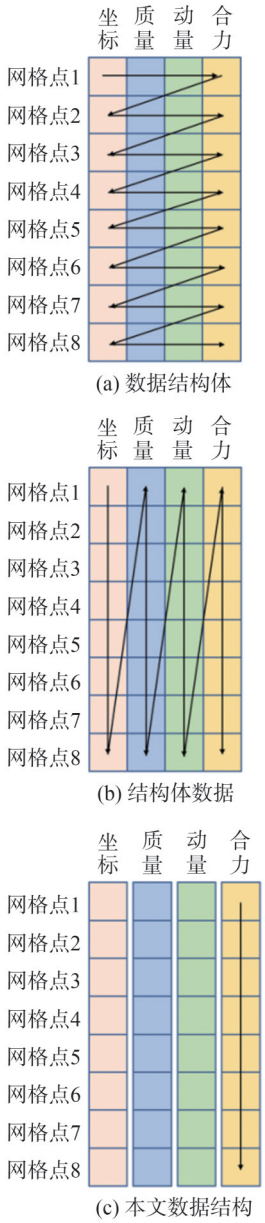
图4 模块化编程方法

Fig. 4 Modular programming approach

2） 数据存储结构

物质点法求解过程涉及大量单元数据信息的处理，因此，高效的数据访问对并行计算效率十分重要。在编程设计层面，数据结构是影响数据访问效率的主要因素。在面向对象编程的计算程序中，模型信息往往以结构体数组（AoS）的形式存储（图5(a)），即模型数据首先按单元（物质点或网格点）索引，每个单元内含有各类物理信息，如坐标、质量等。AoS编码方式十分直观，并适用于大多数编程语言。然而，在并行计算中，模型信息则以数组结构体（SoA）的形式存储（图5(b)）。这是由于在并行过程中，多个线程会同时执行相同的内存操作，即访问同字段的数据（如所有网格点的force）。SoA形式存储的数据使得同字段数据访问位于相邻的位置，减少了索引的计算消耗。在此基础之上，本文使用了更为简洁的多组1维数组的方式进行数据存储（图5(c)）。这一存储形式索引过程更为简洁（如图5箭头所示），且由于数据增添便捷，更易于模块化管理。例如，在传统GPU加速并行程序中，模型信息往往会组织为结构体，随后传入核函数进行并行计算，此时，对结构体的任何修改（如增加网格点法向

力信息）均会导致该核函数的适用对象发生变化，而在本文的数据结构中，相关信息被拆分为多项1维数组，不论数据结构如何复杂，仅需传入相关运算信息即可，使得核函数运算不依赖数据结构。



(a) 数据结构体

(b) 结构体数据

(c) 本文数据结构

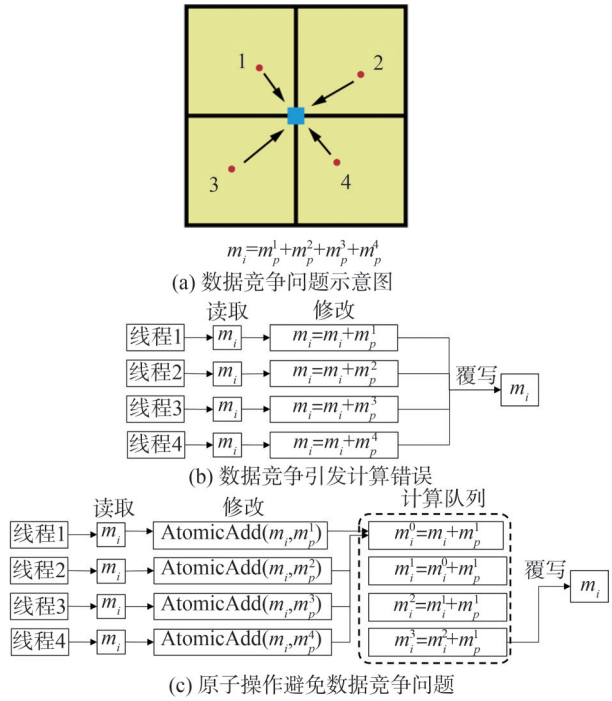
图5　GPU加速并行策略

Fig. 5　GPU-accelerated parallel strategy

然而，核函数的传入数据存在一定的限制，当数据种类较多时无法传入。这一问题可通过结合模块化编程设计来避免，即将模拟中的各个部分拆分，仅向核函数传入所需的部分数据信息。这样既可避免数据管理困难，同时也减少了冗余的数据索引，实现了高效模块化数据管理。

3）　数据竞争

物质点法模拟中，虽然单元间的计算过程彼此独立，但对于数据的访问和处理可能发生重叠，引发数据竞争问题，如图6所示。从图6可知，在P2G过程中，多个物质点可能向一个网格点投射质量信息。此时，该物质点的质量应为 $m_i=m_p^1+m_p^2+m_p^3+m_p^4$。然而，在多线程并行时，4个物质点可能同时读取网格点原始信息，分别进行累加计算后，覆写至网格点信息。此时，网格点的数据为4个线程中任意一个的累加结果（图6(b)）。这种争相获取—修改—覆写数据的过程称为数据竞争。显然，数据竞争会引起部分计算结果的丢失，从而导致错误的计算结果。数据竞争的处理方式主要有两种。一是采用合理的程序设计和预处理，避免这类情况的发生。如在P2G过程中，建立网格点关联的物质点列表，对网格点进行并行操作，逐一从相关联的物质获取映射信息[14]。然而，建立关联列表的过程可能会带来新的数据竞争和内存管理问题，同时增加了计算量。另一种解决方法则是通过线程管理，使用合理的锁、挂起等机制，将竞争过程转变为线性操作。本文采用了原子操作来避免数据竞争的问题，采取基于硬件层级的内存操作管理来避免数据竞争。如图6(c)所示，在计算过程中，使用原子求和（AtomicAdd）后，相同内存位置的操作会被合并，形成计算队列，队列依次执行并在结束后返回。这一方法无须建立复杂的映射关系，有利于提高计算效率。



(a) 数据竞争问题示意图

(b) 数据竞争引发计算错误

(c) 原子操作避免数据竞争问题

图6　数据竞争及解决方法

Fig. 6　Data race and solutions

2　基于铝棒坍塌模型的程序效率测试

2.1　模型选取

铝棒坍塌模型高0.1 m，长0.2 m，如图7所示。模

型，采用 16 384 个物质点对模型进行离散，网格间距为 1.562 5 mm，每个网格内 4 个物质点（图 7）。应力应变响应采用 Drucker–Prager 本构模型模拟，材料参数为[22]：弹性模量 0.84 MPa，泊松比 0.3，内聚力 0 MPa，内摩擦角 19.8°，剪胀角 0.001°，密度 2 650 kg/m^3。模型的底部 xy 平面为摩擦边界，xz 和 yz 平面为反射边界[23]。试验过程先对模型施加重力形成稳定重力场，然后取消右侧边界，模拟物理试验中撤掉挡板的过程。模拟时间步长 Δt 为 5.0×10^{-7} s，总共计算步数为 2.0×10^6 步，模拟物理时间为 1 s。

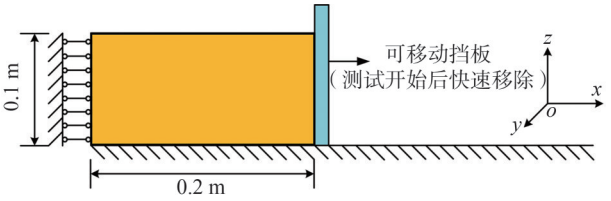


图 7 铝棒坍塌模型

Fig. 7 Model setup of aluminum rod collapse simulation

程序模拟结果与物理试验、无网格法对比结果如图 8 所示。图 8 中，红线代表无网格法表示的滑移线和堆积体表面轮廓线，蓝线表示物理试验坍塌后的滑移线和轮廓线。程序模拟的铝棒坍塌模型与物理实验、无网格数值方法的结果基本吻合，验证了程序在大变形弹塑性问题的适用性。

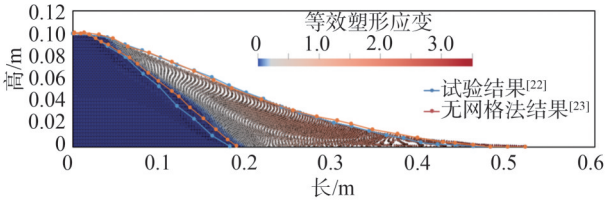


图 8 数值解与试验结果对比

Fig. 8 Comparison between numerical simulation and experimental results

2.2 效率测试

为了测试程序模拟的效率，综合考虑空间复杂度和时间复杂度，增加模型的物质点和网格数量以及计算时间，开展模拟并统计模拟所需要的时间及内存占用情况。模拟使用 GPU 型号为 NVIDIA GeForce RTX 3060ti。

不同规模的物质点计算效率对比如图 9 所示。从图 9 可以看出，不同尺度模型的模拟总时长及单点单步（每个物质点每步）计算时长。随着物质点数的增加模型计算时间呈线性增加，说明程序结构具有良好的并行特性，模拟耗时与模型计算规模线性相关。同时，模拟结果显示，本程序进行单步单点模拟所需的时间约为 1.62×10^{-8} s，最优并行效率发生在 100×10^4 物质点以上，这是由于并行计算存在一定的资源和数据分配开销，当物质点数目较少时，这类开销占比较大，导致了整体效率的下降。为进一步评估算法性能进行了时间复杂度的考虑，选用物质点数 1×10^6 的物理参数，物质点数量 1 048 576 个，但物理时间分别设置为 1、2、4、6、8、10 s，统计计算时间随物理模拟时间变化情况（图 9）。随时间复杂度增加，计算时间呈现更良好的线性，同样展现了程序结构良好的并行特性。模拟所需显存大小也随模型规模呈现线性关系，如图 10 所示，说明程序无明显冗余显存占用。

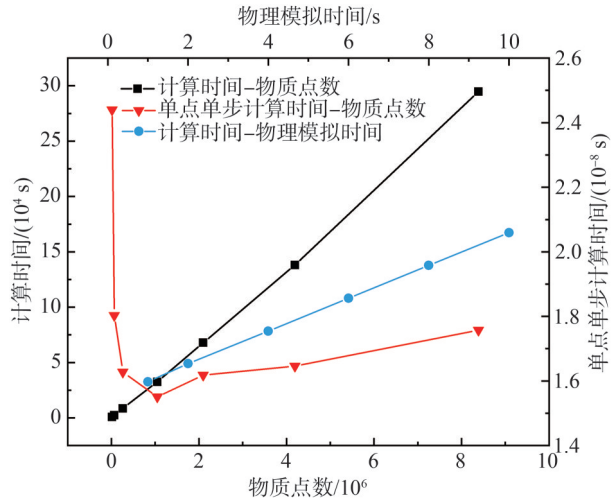


图 9 计算效率对比

Fig. 9 Computational efficiency comparison

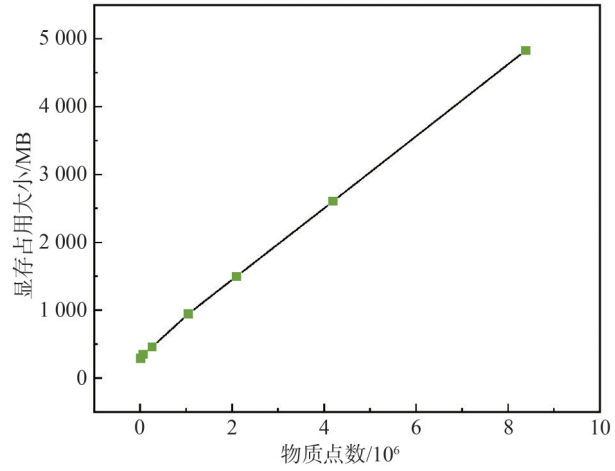


图 10 物质点与显存占用量关系

Fig. 10 Relationship between memory occupancy and number of material points

随后，对物质点计算过程中的各流程计算效率进行了统计分析，如图 11 所示。由图 11 可知，流程④（P2G 过程）用时占比最高，包含物质点信息映射至背景网格流程。该投射过程采用了部分原子操作以避免数据竞争，因而产生了较高的计算开销。用时占比位于第 2 位的是流程⑧，该流程为物质点应力应变求解过程，涉及计算较为复杂，耗时量较大，其余流程②、③、⑤、⑥、⑦达到最优并行效果，计算耗时较短。

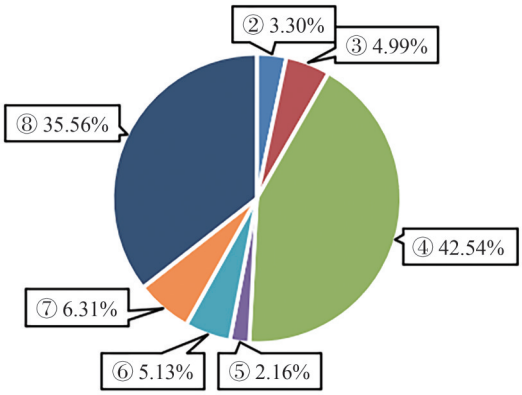


图 11　不同计算流程时间占比

Fig. 11　Proportion of computational time costed by each process

3　基于理想边坡模型的模拟对比测试

3.1　模型选取

理想边坡模型采用Drucker–Prager本构模型，如图12所示，材料参数[24]为：弹性模量70 MPa，泊松比0.3，内聚力1 kPa，内摩擦角20°，剪胀角0°，密度2 100 kg/m^3。模型的底部z方向为固定边界，x和y方向为反射边界。边坡模型采用正六面体八节点单元的3维模型，每个单元中有8个质点，共19 400个物质点。x方向111个网格结点，y方向2个网格结点，z方向36个网格结点，共13 200个网格结点。时间步长dt = 9.441 5×10^{-4} s，总计算步数为15 588步。

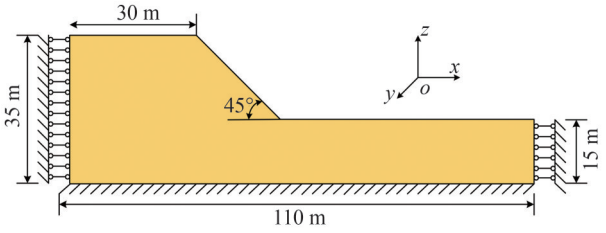


图 12　简化边坡模型几何图

Fig. 12　Simplify the slope model geometry

3.2　程序对比分析

边坡破坏后的最终导致塑性应变分布如图13所示。由图13可知，边坡在坡脚处发生了破坏，随后沿边坡内部向上形成贯通的塑性剪切带，最后导致边坡向下滑塌，边坡的最终位移量为12.761 m。边坡破坏形式和滑移量与Taichi–CPU[25]、Taichi–GPU[25]、MPM3D–F90[26]模拟结果基本吻合，其中，Taichi–CPU和Taichi–GPU完全一致，因此没有区分两种程序结果。4个程序的效率对比如表1所示。从表1可以看出：在相同的模拟参数下，本文GPU加速程序较Taichi–GPU模拟效率提高了9.6%；Taichi–CPU和MPM3D–F90均为CPU串行计算，计算效率相差不大，较其相比，本文的模拟方法计算效率分别提高了49倍和54倍。

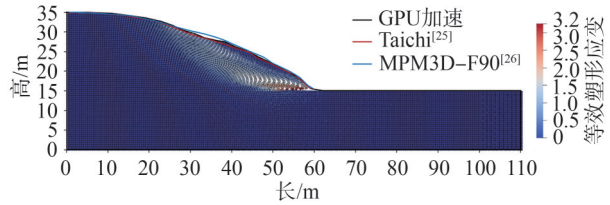


图 13　各程序失稳后的边坡轮廓对比及等效塑性应变分布

Fig. 13　Comparison of slope profile and plastic strain distribution after slope failure

表 1　程序计算效率对比

Tab. 1　Comparison of computing efficiency between different programs

算法	计算时长/s
GPU加速	14.16
Taichi–GPU[25]	15.67
MPM3D–F90[26]	702.66
Taichi–CPU[25]	776.80

4　新磨村滑坡分析

为验证改进GPU加速策略的物质点法模拟大规模滑坡工程的可行性和计算优势，针对茂县新磨村滑坡进行建模模拟。新磨村滑坡发生于四川省茂县新磨村，山体高约3 400 m，位于岷江一级支流松坪沟左岸。原始斜坡上陡下缓，滑源区坡度约为50°。由于受到地震影响，该区域山体存在大量裂隙，最终在降雨的诱发下，形成了2.87×10^6 m^{3}的滑坡。该滑坡造成了10人死亡，73人失踪，整个新磨村几乎被毁。

基于物质点法（MPM），构建了新磨村滑坡的3维模型，复现了滑坡的失稳运动全过程。基于滑坡前后地形，建立了新磨村滑坡MPM模型如图14所示，选取网格尺寸为10 m×10 m×10 m，建立了背景网格，共计7 626 000个节点。滑坡区域位于模型中央，使用46 893个物质点进行模拟。根据Zhao等[27]的研究，采用基于速度场的摩擦接触模型来模拟滑坡体与山体的相互作用过程。摩擦系数公式为：

$$\mu(v)=\mu_1+(\mu_0-\mu_1)\times\exp(-v^A/v_c) \tag{8}$$

摩擦参数分别设置分为峰值摩擦系数μ_1=0.70（整个接触面可达到最大的摩擦系数），稳态摩擦系数μ_0=0.23，临界速度v_c=4.0（高速衰减的临界速度），常数系数A=0.6，反映摩擦系数$\mu(v)$随速度(v)增加而减小的速率。同时，为了简化在模型计算中由于复杂本构积分可能造成的效率与收敛问题，模拟中采用了经典的Drucker–Prager模型，材料参数如下：摩擦角为30°，内聚力为0 MPa，杨氏模量为8.1 GPa，泊松比为0.18，密度为2 200 kg/m^3。滑坡在自重作用下发生滑

塌，重力加速度为9.8 m/s^2。

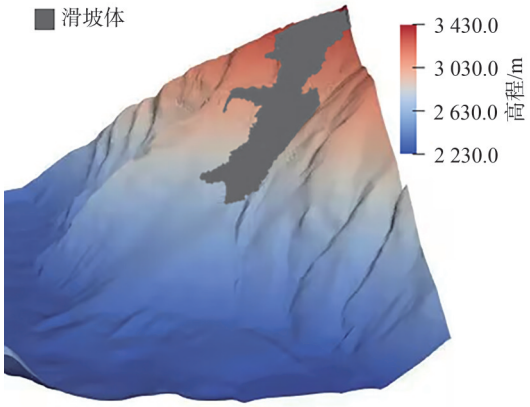


图14 新磨村滑坡MPM模型

Fig. 14 MPM model of Xinmo landslide

根据Fan等[28]描述，岩体深层早期存在的裂缝，在地震的作用下发生分离，形成滑源区滑坡体。在降雨的情况，滑源区滑坡体开始顺层滑动，在滑动带上撞击原始岩体，裹挟下部滑面上的块石，两者碰撞不断发生动量交换，随着时间推移岩体破碎解体成碎屑，形成碎屑流。碎屑流的重力势能在山脚处转为极大的动能，推挤山体下方岩体。碎屑流堆积体到达地势开阔的新磨村和河道附近，形成扇形堆积区。整个滑动过程分为3个阶段：崩滑阶段，在雨水冲击的作用下，岩石块体沿着前期地震产生裂缝向下运动；碎屑流阶段，岩石块体运动过程不断撞击滑坡面上的块石，在不断的能量交换下岩块变为碎屑岩；散落堆积阶段，高速运动的碎屑岩在地形开阔的坡脚处形成扇形堆积区。

新磨村滑坡失效过程数值模拟结果如图15所示，模拟滑坡总运动时间为100 s，与实际工程记录的滑坡持续时间基本吻合。滑坡体初始位置分为滑源区和变形区两个部分，图15(a)中红色区域速度较大的粒子是滑坡源区粒子，推动蓝色区域变形区粒子开始滑动。图15(b)原始滑坡体和滑道的岩体不断撞击，随着时间推移岩体破碎解体成碎屑，形成高速运动的碎屑流。图15(c)碎屑流到达坡脚，粒子速度v开始放缓，粒子呈扇形趋势逐渐向河道附近流动。图15(d)整个滑坡运动结束，堆积体呈扇形区域堆积在坡脚，滑坡结束后，滑面上仍残留滑坡碎屑流堆积体。

图16所示为新磨村滑坡堆积区域对比图，粒子代表MPM模拟结果。图16(a)中，蓝色粒子是MPM模拟结果，蓝色粒子堆积区域即MPM堆积区域。从16(a)可以看出，滑动面上残留有部分的碎屑块体，在河床附近形成了扇形堆积区，这与实际的堆积结果基本相符合[28]。此外，图16(b)将GPU加速物质点法与其他数值方法(MPM[27]、PFC[29]和MassMov2D[29])的堆积结

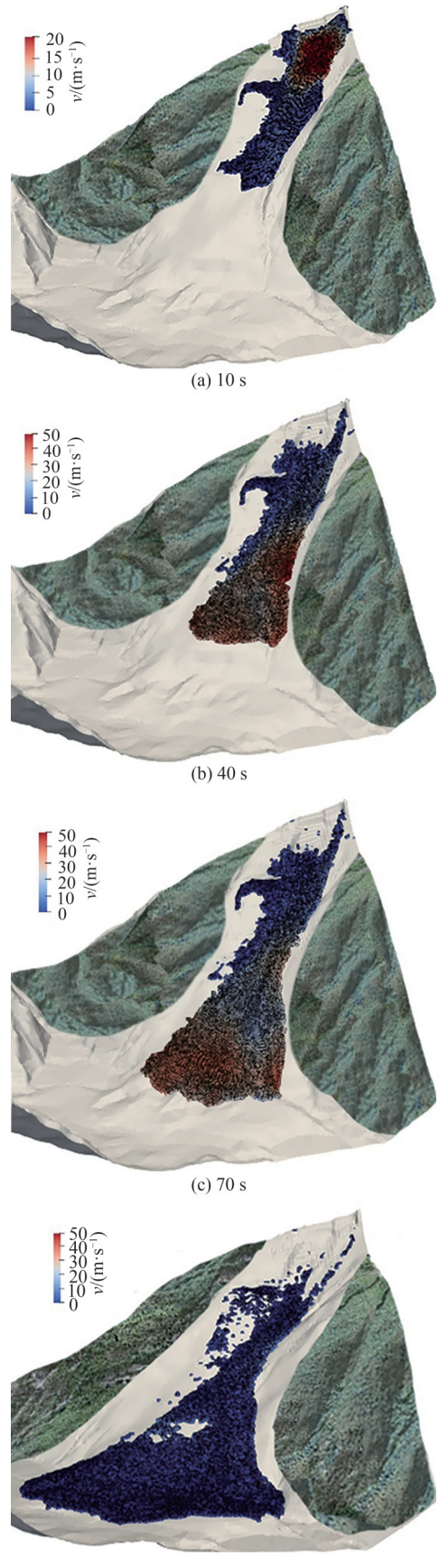
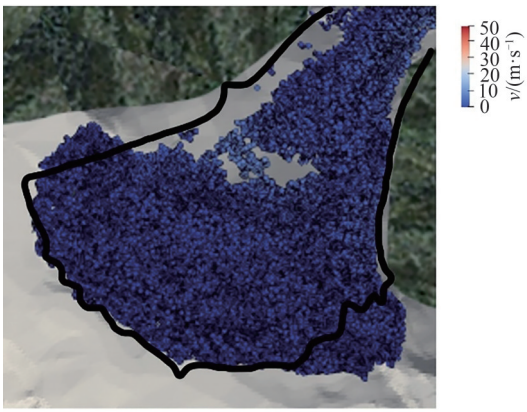


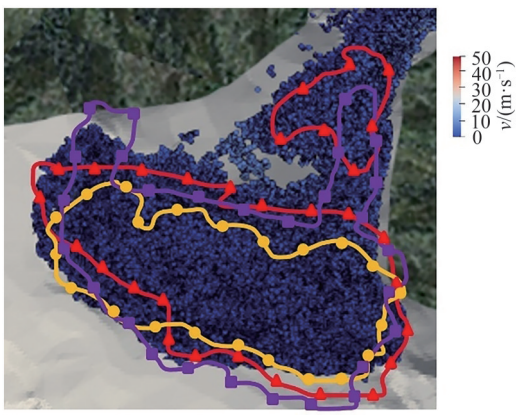
图15 新磨滑坡数值模拟失效过程

Fig. 15 Numerical simulation failure process of Xinmo landslide

果进行了对比。由图16(b)可见，4种方法模拟的河床堆积物结果大致相同。对于边坡松散沉积，本文模拟结果(GPU加速物质点法)与Zhao等[27]MPM的模拟结果基本一致，较PFC模拟结果堆积面积略大。对于边坡残留堆积，本文模拟结果与实际吻合度更高，较MPM与PFC模拟范围略大，这可能是由于本文使用了更精细的模型，能够捕捉坡面上残留的浅层堆积体。



(a) MPM堆积区域与实际区域对比



(b) MPM堆积区域与其他数值方法对比

图16　新磨村滑坡堆积区域对比

Fig. 16　Map of Xinmo landslide accumulation area

本次模拟使用的GPU型号为NVIDIA GeForce RTX 3060ti，滑坡模拟过程总耗时1 h左右，Zhao等[27]采用串行CPU程序模拟则花费20 h左右，计算时间提高了20倍左右，且物质点数目增加了2.7倍，计算效率对比如表2所示。对于自然边坡模拟，因自然条件变化，不同地区的材料参数变化巨大，常需要不断进行模拟率定参数范围。当使用GPU加速物质点法调试参数时，GPU算法的效率优势就凸显出来。因此，针对实际大型岩土工程问题的模拟，开展GPU并行策略下的物质点方法模拟具有较好的优越性。

表2　程序计算效率对比

Tab. 2　Program calculation efficiency comparison

算法	物质点数量	计算时间/s
CPU单线程	17 111	72 000.0
GPU并行计算	46 893	3 746.2

5　结　论

本文提出了一种改进GPU加速策略的物质点法，该方法在GPU并行基础上，使用模块化编程设计，选择多组1维数组的数据存储结构和原子操作数据竞争处理方法，在保证模拟效率的同时，提高了代码的灵活性，改善了算法的可扩展性。经过理想边坡的效率测试，本文的GPU加速程序较Taichi–GPU语言模拟效率提升约10%，较串行CPU程序提高了约50倍。最后，将文中所提出的改进GPU物质点方法应用到新磨村滑坡破坏全过程的模拟中，进一步说明了改进GPU策略的物质点方法在实际工程问题模拟上的可行性与优越性。

参考文献：

[1] Harlow F H. The particle-in-cell computing method for fluid dynamics[J]. Methods in Computational Physics, 1964, 3: 319–343.

[2] Sulsky D, Chen Z, Schreyer H L. A particle method for history-dependent materials[J]. Computer methods in applied mechanics and engineering, 1994, 118(1/2): 179–196.

[3] Andersen S, Andersen L. Modelling of landslides with the material-point method[J]. Computational Geosciences, 2010, 14: 137–147.

[4] Zhong Zuliang, He Kaiyuan, Song Yixiang, et al. Large-displacement landslides based on affine velocity matrix-improved material point method[J]. Chinese Journal of Geotechnical Engineering, 2022, 44(9): 1626–1634. [钟祖良，贺凯源，宋宜祥，等. 基于仿射速度矩阵改进物质点法的大位移滑坡研究[J]. 岩土工程学报，2022，44(9)：1626–1634.]

[5] Xie Yanfang, Li Xinpo, Zhao Shux, et al. MPM-based numerical analysis of the kinematic characteristics of Xinmo landslide in Maoxian County, Sichuan, China[J]. Mountain Research, 2018, 36(4): 589–597. [谢艳芳，李新坡，赵曙熙，等. 基于物质点法的新磨村滑坡动力特性分析[J]. 山地学报，2018，36(4)：589–597.]

[6] Więckowski Z, Youn S K, Yeon J H. A particle-in-cell solution to the silo discharging problem[J]. International journal for numerical methods in engineering, 1999, 45(9): 1203–1225.

[7] Wang Bin, Feng Xiating, Pan Pengzhi, et al. Slope failure

analysis using the material point method[J]. Chinese Journal of Rock Mechanics and Engineering,2017,36(9):2146–2155.[王斌,冯夏庭,潘鹏志,等.物质点法在边坡稳定性评价中的应用研究[J].岩石力学与工程学报, 2017,36(9):2146–2155.]

[8] Liang Dongfang, Zhao Xuanyu, Soga Kenichi.Simulation of overtopping and seepage induced dike failure using two-point MPM[J].Soils and Foundations,2020,60(4):978–988.

[9] Xie Xiaochuang, Ceccato Francesca, Zhou Mingliang, et al. Hydro-mechanical behaviour of soils during water-soil gushing in shield tunnels using MPM[J].Computers and Geotechnics,2022,145:104688.

[10] Sun Zheng.A study for the improved B-spline material point method[D].Zhejiang:Zhejiang University,2018.[孙政.B样条物质点法的算法改进研究[D].浙江:浙江大学, 2018.]

[11] Hu Yuanming, Fang Yu, Ge Ziheng, et al. A moving least squares material point method with displacement discontinuity and two-way rigid body coupling[J]. ACM Transactions on Graphics(TOG),2018,37(4):1–14.

[12] Yuan Weihai, Wang Haocheng Liu Kang, et al. Analysis of large deformation geotechnical problems using implicit generalized interpolation material point method[J]. Journal of Zhejiang University–SCIENCE A,2021,22(11):909–923.

[13] Huang Peng,Zhang Xiong,Ma Shang,et al.Shared memory OpenMP parallelization of explicit MPM and its application to hypervelocity impact[J].Computer Modeling in Engineering & Sciences,2008,38(2):119–148.

[14] Dong Y, Wang D, Randolph M F. A GPU parallel computing strategy for the material point method[J]. Computers and Geotechnics,2015,66:31–38.

[15] Lei Chunxia, Gan Yong. Study of the thickness rffects on copper films heated by the femtosecond laser[J]. Bulletin of Science and Technology,2018,34(8):6–9.[雷春霞,干湧.超短脉冲激光辐照金属的物质点法并行算法研究[J].科技通报,2018,34(8):6–9.]

[16] Xia Xilin,Liang Qiuhua.A GPU-accelerated smoothed particle hydrodynamics(SPH) model for the shallow water equations [J].Environmental modelling & software,2016,75:28–43.

[17] Zhang Wei,Zhong Zhihao,Peng Chong,et al.GPU-accelerated smoothed particle finite element method for large deformation analysis in geomechanics[J]. Computers and Geotechnics,2021,129:103856.

[18] Hoetzlein R K.GVDB: Raytracing sparse voxel database structures on the GPU[C]//Proceedings of High Performance Graphics,New York:Association for Computing Machinery(ACM),2016:109-117.

[19] Setaluri R, Aanjaneya M, Bauer S, et al. SPGrid: A sparse paged grid structure applied to adaptive smoke simulation [J].ACM Transactions on Graphics(TOG),2014,33(6):1–12.

[20] Gao Ming,Wang Xinlei,Wu Kui,et al.GPU optimization of material point methods[J]. ACM Transactions on Graphics (TOG),2018,37(6):1–12.

[21] Wang Xinlei,Qiu Yuxing,Slattery Stuart R,et al.A massively parallel and scalable multi-GPU material point method[J]. ACM Transactions on Graphics(TOG),2020,39(4):30:1–30.

[22] Bui H H, Fukagawa R, Sako K, et al. Lagrangian meshfree particles method(SPH) for large deformation and failure flows of geomaterial using elastic-plastic soil constitutive model[J]. International Journal for Numerical and Analytical Methods in Geomechanics,2008,32(12):1537–1570.

[23] Li Jianguo, Wang Bin, Jiang Quan, et al. Development of an adaptive CTM–RPIM method for modeling large deformation problems in geotechnical engineering[J]. Acta Geotechnica,2021:1–19.

[24] Huang Peng. Material point method for metal and soil impact dynamics problems[D]. Beijing: Tsinghua University, 2011.[黄鹏.金属及岩土冲击动力学问题的物质点法研究[D].北京:清华大学,2011.]

[25] Hu Yuanming, Li Tzumao, Anderson Luke, et al. Taichi: A language for high-performance computation on spatially sparse data structures[J]. ACM Transactions on Graphics(TOG), 2019,38(6):1–16.

[26] 张雄,廉艳平,刘岩,等.物质点法[M].北京:清华大学出版社,2013.

[27] Zhao Shuxi,He Siming,Li Xinpo,et al.The Xinmo rockslide-debris avalanche:An analysis based on the three-dimensional material point method[J].Engineering Geology,2021,287: 106109.

[28] Fan Xuanmei,Xu Qiang,Scaringi Gianvito,et al.Failure mechanism and kinematics of the deadly June 24th 2017 Xinmo landslide, Maoxian, Sichuan, China[J]. Landslides, 2017,14:2129–2146.

[29] Scaringi Gianvito,Fan Xuanmei,Xu Qiang,et al.Some considerations on the use of numerical methods to simulate past landslides and possible new failures:the case of the recent Xinmo landslide(Sichuan,China)[J].Landslides,2018,15:1359–1375.

An Improved GPU-accelerated MPM and Application in Landslide Modelling

WANG Bin[1,2], *CHEN Penglin*[1,2], *WANG Di*[1], *XU Shunxin*[1,3], *XU Zikai*[1,4], *WU Jindong*[1,3]

(1. State Key Laboratory of Geomechanics and Geotechnical Engineering Safety, Institute of Rock and Soil Mechanics, Chinese Academy of Sciences, Wuhan 430071, China;
2. School of Engineering Science, University of Chinese Academy of Science, Beijing 100049, China;
3. School of Resources and Environmental Engineering, Wuhan University of Technology, Wuhan 430070, China;
4. School of Highway, Chang'an University, Xi'an 710064, China)

Abstract: In recent years, the material point method (MPM) has become an important large-deformation numerical simulation method in geotechnical engineering and is widely utilized to study issues such as landslides, foundations, dam failures, and water-soil gushing in shield tunnels. As the scale and complexity of application scenarios increase, the accuracy requirements and efficiency needs for numerical methods also rise, resulting in higher computational costs that restrict the further application of MPM in large-scale geotechnical engineering problems. The simulation efficiency of MPM improves significantly by introducing parallel acceleration technology; however, specific issues, such as program architecture and extensibility, still limit their development in engineering applications. This study proposes an improved GPU acceleration strategy for MPM by incorporating modular programming concepts, efficient data structures, and data competition handling methods to establish a high-performance and easily extendable program architecture. The software's performance is evaluated by simulating aluminum rod collapse experiments and the slope failure process. The results indicated that the software achieves effective parallelization and demonstrates approximately a 10% improvement over the existing Taichi-GPU MPM. Finally, the proposed MPM method is applied to simulate the Xinmo landslide, and the computational efficiency improves by approximately 20 times when the number of material points increases by about 2.5 times.

Key words: GPU-accelerated; material point method; landslide; large deformation numerical simulation

（编辑　张　琼）

引用格式: Wang Bin,Chen Penglin,Wang Di,et al.An improved GPU-accelerated MPM and application in landslide modelling[J].Advanced Engineering Sciences,2025,57(4):1–11.[王斌,陈鹏林,王頔,等.一种改进GPU加速策略的物质点分析方法及其在滑坡模拟中的应用[J].工程科学与技术,2025,57(4):1–11.]