Vol. 29 高等学校化学学报 No. 12
2008 年 12 月 CHEMICAL JOURNAL OF CHINESE UNIVERSITIES 2425 ~ 2429

分子动力学模拟计算在通用图形处理芯片上的实现

宋国梁, 翁经纬, 李振华, 王文宁, 范康年
(复旦大学化学系, 上海 200433)

摘要 将在计算生物分子中广泛应用的 CHARMM 力场应用于 Windows computer cluster server(WCCS)环境下, 并实现了该力场及分子动力学模拟程序的通用显卡(GPU)并行计算. 对一些多肽链的动力学模拟结果显示, 与 CPU 计算相比, GPU 计算在计算速度上有巨大的提升. 与 64 位 Athlon 2.0G 相比, 在 NVIDIA GeForce 8800 GT 显卡上的动力学模拟速度提高了至少 10 倍, 而且这个效率比会随着模拟体系及每块尺寸的增大而增大. 模拟体系的增大使得 GPU 并行单元的计算空载相对减少, 块尺寸的增大使缓存区尺寸相对减少, 单块计算效率得以提高. 在测试样本中, 该效率比最高可达到 28 倍以上. 利用 GPU 计算还对一条含有 397 个原子的多肽链进行了分子动力学模拟, 给出了氢键分布随时间的变化结果.

关键词 分子动力学; 图形处理芯片; CHARMM; Windows computer cluster server
中图分类号 O641 **文献标识码** A **文章编号** 0251-0790(2008)12-2425-07

具有生物功能的分子通常含数千甚至上万个原子, 而对于大多数的生物分子间的反应、蛋白质的折叠、膜蛋白的打开和闭合等重要的动力学过程, 其时间尺度通常为微秒, 甚至毫秒级别. 对这些动力学过程进行模拟, 有时还需要进行数千甚至上万条轨迹的模拟以获得具有统计意义的结果. 要得到定性可靠甚至定量精确的结果, 所使用的势能函数必须具有足够的精度, 而势能函数越精确, 其表达形式通常会越复杂, 计算的速度就会越慢. 因此, 目前对生物大分子的动力学模拟由于受到计算速度的限制, 模拟的时间尺度通常为纳秒级别. 因此, 提高动力学模拟和势能函数的计算速度, 是进行可靠的有实用价值的动力学模拟的关键. 最近, 不再局限于 CPU 计算的新技术, 即基于图形处理芯片(Graphics processing units, GPU)的通用(General purpose)计算技术(GPGPU)已经取得重大突破, 这极大地提升了计算速度. 对比 CPU, GPU 最大的优势有三点: (1) 超强的浮点运算能力. GPU 的浮点运算速度是 CPU 的 4 倍以上, 目前最新的显卡上的 GPU 的浮点运算速度甚至达到了 CPU 的 100 倍左右. (2) GPU 具有超强的输入输出带宽. GPU 和显存之间的交换带宽是 CPU 和内存之间交换带宽的 10 倍以上. 由于受到 CPU 和内存之间的带宽限制, 目前的多核计算机的计算性能并不理想. 例如, 我们在一台 Intel 双 CPU 的 8 核计算机上测试了材料模拟软件 VASP 的并行效率时发现, 同时使用 8 核进行计算, 其速度慢于只使用 4 核的, 并行效率低于在单 CPU 计算机之间通过 Merynet 和 Infinite Band 通讯的并行效率. (3) GPU 性价比和能耗效率比都更高. 目前 GPU 的价格和 CPU 相当, 甚至更低. 同时 GPU 的能耗也仅比主流双核 CPU 处理器略高, 因此其性价比和能耗效率比都是非常优异的.

由于单块显卡上的 GPU 的运算速度已经远远超过目前最快的 CPU, 同一台 PC 计算机还可以配置多块显卡, 因此, 单台计算机的性能可以得到大幅度的提升. 目前, 计算机厂商如 NVIDIA, AMD 都在加速研发 GPGPU 项目, 以降低 GPU 计算的编程难度. NVDIA 已经为 GPGPU 高性能计算提供了 CUDA (Compute unified device architecture)计算平台, 并针对 GPU 的编程提供了 C 语言开发环境. AMD 也为 GPU 编程提供了 Peakstream 的 Linux 和 Windows 版开发工具. Microsoft 公司已经在其编程开发软件 Visual Studio 中提供了对 GPU 编程的支持. 常用的线性数学库 BLAS 已经应用到了 GPGPU 上(如

收稿日期: 2008-10-07.
基金项目: 国家自然科学基金(批准号: 20433020, 20673024, 20828003)资助.
联系人简介: 范康年, 男, 教授, 博士生导师, 从事量子化学理论和催化研究. E-mail: knfan@fudan.edu.cn

NVIDIA 的 CUBLAS 库). 目前, GPGPU 已在量子化学计算及分子动力学模拟等方面广泛应用[1~10].

本文首次将在生物大分子模拟中广泛使用的 CHARMM 力场在 Windows computer cluster server (WCCS)上实现 GPU 计算, 并且在 GPU 下开发出动力学模拟程序. 程序开发语言选择 Visual C++, 开发平台选用 Visual Studio 2005. GPU 程序的开发平台选用 CUDA.

1 CHARMM 力场的 GPU 实现

CHARMM 力场[11~14]是分子动力学中最经典的力场之一. CHARMM 力场的表达式如下:

$$U(\vec{R}) = \sum_{\text{bond}} K_b(b - b_0)^2 + \sum_{\text{angle}} K_\theta(\theta - \theta_0)^2 + \sum_{\text{dihedral}} K_\phi[1 + \cos(n\phi)] + \sum_{\text{UB}} K_{\text{UB}}(S - S_0)^2 + \sum_{\text{improper}} K_{\text{imp}}(\varphi - \varphi_0)^2 + \sum_{\text{nonbond}} \left[\left(\frac{A_{ij}}{r_{ij}}\right)^{12} - \left(\frac{B_{ij}}{r_{ij}}\right)^{6}\right] sw(r_{ij}^2, r_{\text{on}}^2, r_{\text{off}}^2) + \sum_{\text{nonbond}} \frac{q_i q_j}{4\pi\varepsilon_0 r_{ij}}$$

式中, b, θ, φ 是键长、键角和二面角, 第四和第五项是 UB(Urey-Bradly)力场和 IMPROPER 力场. 第六和第七项是范德华力和静电力. 其中 sw 函数是范德华力的开关函数.

由于 CHARMM 力场使用键长、键角和二面角的内坐标形式, 使基于这些内坐标的计算值与实验测量值之间的相对误差更小, 而且易于通过参数的调整实现更加精确的模拟. CHARMM 力场主要包含两部分, 前 5 项是基于关联(键长、键角和二面角等)的力场, 后两项是长程力场(静电和 VWD 等). 长程力场各粒子之间可以很好的拆分, 并行效率很高. 但关联力场用于 GPU 并行计算的主要困难在于, 基于关联的力场计算在串行化条件下仅需计算一次力场即可更新相关多个粒子的力的分量, 而在 GPU 并行化计算情况下, 这些修改必然导致内存修改冲突. 通常的解决方案有: 串行化(通过使用消息队列或者更新队列, 使所有更新按先入先出方式依次进行)和多次更新(每次仅更新一个粒子坐标. 重复多次计算)两种. 这两种方法各有利弊. 本文采取了一种折中方案, 即先进行一次并行力场计算, 然后再进行多次更新. 这样最耗时的力场计算仍然计算一次, 只是数据更新变成多次, 计算量并不增加太多, 仍然保持了总的并行效率.

2 分子动力学程序的 GPU 实现

分子动力学模拟是利用势能函数给出的体系总势能, 以及每个原子上所受的力, 从一个给定的分子初始状态(初始结构和每个原子的初始速度)出发, 通过差分方法求解牛顿运动方程, 模拟实际分子的运动过程. 因此一个动力学模拟程序一般由 4 部分组成: (1) 初始条件(结构和初始速度)的读入; (2) 给定结构的能量和原子所受力的计算; (3) 利用力和初始条件进行动力学模拟; (4) 分析结果和输出结果. 最关键的部分是第二部分和第三部分. 其中第二部分是计算中最耗时的部分. GPU 计算的关键在于把计算中最耗时、最适合于 GPU 的计算任务合理地分配给 GPU 执行.

分子动力学部分的程序比较容易应用于 GPU, 以原子为切分单位的计算单元即可以简单实现很高的并行度. 唯一的问题在于, 当分子很大(超过 1000 个原子)时, GPU 的共享内存(Shared memory)及反高速块内存(Block memory)都无法满足要求, 必须将大分子切分成中型区块才能够分块计算. 这必将引入键合的缓冲区原子和长程力的缓冲区原子, 从而使计算效率下降, 然而这些缓冲区原子不太多, 不会引起太大的问题. 通常, 取区块内原子的所有外区相关键合原子为键合原子的缓冲区, 取 VWD 半径内的所有原子为长程力缓冲区. 非缓冲区原子静电力处理可以通过将每区块聚合为一个电荷中心, 然后扣除缓冲区原子来实现.

3 CPU+GPU 及多节点混合编程的实现

由于 GPU 受显存和内部高速缓存的限制, 本身的扩展性不很好, 即使是 NVIDIA 最新的 GPU (GTX280)也只有 1G 显存和 240 个计算线程, 这使得更大规模的计算必须使用多节点并行的模式. 因此, 同时充分有效地将 CPU 的计算资源和 GPU 的计算能力用在更大的计算体系上, 是一个很有挑战

性的课题. 图 1 是 CPU + GPU 及多节点混合编程的示意图.

初步的实践表明，将并行计算任务中内存修改冲突较大的部分分配给 CPU，将并行度很高且计算繁重的部分分配给 GPU，会取得很好的效果. 其难点在于如何使两者协调工作，任何一部分的延迟或空闲都会明显导致整体计算性能的降低. 例如，在特定的应用层面，针对蛋白质的分子动力学模拟，精确地分配两者的计算时间使之在异步工作后基本能够同时完成分配任务并实现同步交换数据，可以使性能损失控制在可接受的范围内. 具体步骤为：将力场的计算和梯度的更新(将力场计算得到的力从内坐标转换为直角坐标)两个任务分别交给 GPU 和 CPU 完成，由于在程序上这两个部分是串行执行的，如果直接分配则会导致 GPU 和 CPU 相互等待，解决方案是将 GPU 计算任务分成 n 份，每完成其中一份即让 CPU 开始异步执行梯度更新，这样双方仅需要等待原来 $1/n$ 的时间，性能损失不大.

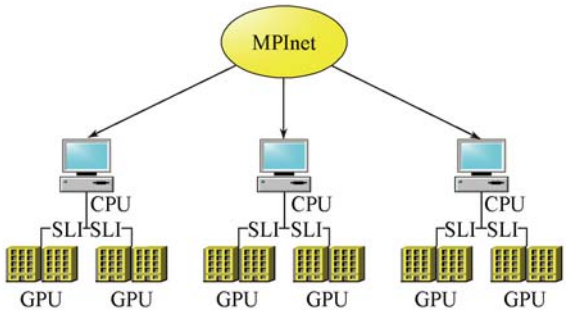

Fig. 1　CPU + GPU and multi-node united program

4　GPU 的计算效率

选取不同大小的一个多糖(45 个原子)和 5 个多肽链作为测试分子(见图 2 和图 3).

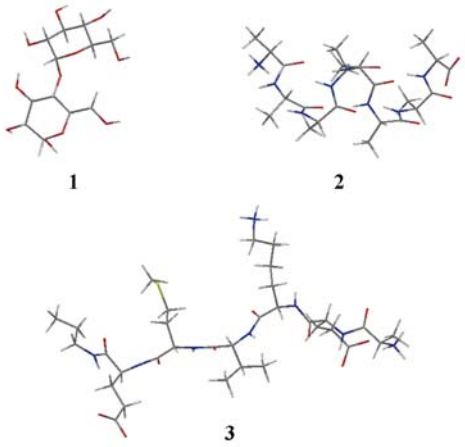

Fig. 2　Testing molecules 1, 2, 3 for molecular dynamics simulation

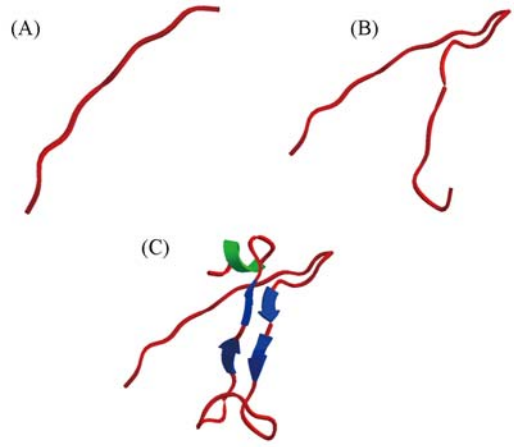

Fig. 3　Testing molecules 4(A), 5(B), 6(C) for molecular dynamics simulation

对于相同的分子，分别使用 GPU 和 CPU 进行了 20 万步(每步 0.1 fs，总共 20 ps)固定步长和总能量的动力学模拟. 其中，显卡型号为 NVIDIA GeForce 8800 GT，CPU 为 AMD 64 位 Athlon 2.0G. 表 1 中列出了不同大小分子在 CPU 上和 GPU 上进行模拟所用时间的比较.

Table 1　Comparison of computation time by CPU and GPU*

Atom number	Simulation step	t_{CPU}/ms	t_{GPU}/ms	t_{GPU}/t_{CPU}	Atom number	Simulation step	t_{CPU}/ms	t_{GPU}/ms	t_{GPU}/t_{CPU}
45	20	38	4	9.5	204	20	148	6.6	22.4
	200	319	37	8.6		200	1302	64	20.3
	2000	3141	330	9.5		2000	13053	624	20.9
	20000	31570	3285	9.6		20000	129602	6208	20.9
	200000	320988	33182	9.7		200000	1290516	62752	20.6
83	20	49	4	12.3	397	20	279	11	25.4
	200	492	37	13.3		200	2501	105	23.8
	2000	4757	375	12.7		2000	24381	1038	23.5
	20000	47775	3716	12.9		20000	246409	10872	22.7
	200000	496147	37417	13.3		200000	2486682	104367	23.8
109	20	72	4	18.0	775	20	524	18.7	28.0

Continued

Atom number	Simulation step	t_{CPU}/ms	t_{GPU}/ms	t_{GPU}/t_{CPU}	Atom number	Simulation step	t_{CPU}/ms	t_{GPU}/ms	t_{GPU}/t_{CPU}
109	200	657	41	16.0	775	200	4883	179	27.3
	2000	6478	418	15.5		2000	48624	1813	26.8
	20000	65287	4108	15.9		20000	486024	18274	26.6
	200000	650815	41645	15.6		200000	4852572	177743	27.3

* Six different molecules are tested. All simulation are done at 0.1 fs time step and the total energy is fixed.

由表 1 可见，相对于 CPU 计算，GPU 计算具有极大的优势，速度的提升基本上在 10 倍以上，甚至达到了将近 30 倍. 图 4 为 GPU 和 CPU 的速度之比随模拟分子中原子数目变化的曲线. 由图 4 可见，分子越大，GPU 计算的速度优势越大，这主要是由于计算较大的体系时计算的分块数目 m 比较大，这样分配到 GPU 的 128 个并行计算单元时剩余空闲单元的数目相对于总计算数目的比例在不断降低. 另外，CPU 计算时，当体系逐渐增大，通过高速一级和二级缓存(cahce)读取数据的命中率会下降，这是导致 CPU 计算的效率不断下降的原因.

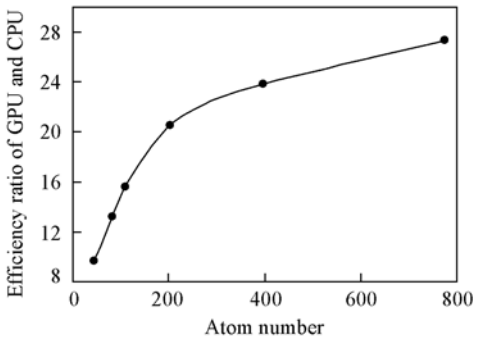

Fig. 4 Efficiency ratio of GPU and CPU relative to the size of molecules

目前，我们在 CPU 上和 GPU 上均使用相同的计算模式，即在 GPU 上运行的代码还没有针对 GPU 运算进行特别的优化. 而且，在计算 CHARMM 力场的时，计算了所有的长程库仑相互作用项. 而这一相互作用的计算速度还可以使用特殊的技术进一步提升，从而使动力学模拟的速度进一步提高. 综合考虑这些因素，可以说明，相比 CPU，在 GPU 上进行分子动力学模拟的确具有非常大的优势.

5 含 397 个原子的多肽链的动力学模拟结果

作为初步测试，还对原子数为 397 的多肽进行了模拟. 图 5 为氧氢键的分布图. 由于氢键相互作用是使一个生物分子具有特定活性和特殊结构的重要原因，所以从氧氢键的分布图中可以了解生物分子中的氢键分布. 由图 5 可以看到，在动力学模拟的初期，分子的氧氢键长还具有比较有规律的周期性的结构(曲线呈现出峰谷的交替)，表明这段时间内多肽主要在分子的某个势能局域的最低点附近做振动. 随着模拟时间的增长，0.25 nm 处的肩峰明显升高，0.32 nm 左右的高峰没有明显变化，肽键长程处有规律的精细结构消失. 表明多肽链跳出了一个势能局域的最低点，由于氢键的相互作用，在肽链的局部形成了某种稳定结构，但整体的肽链却比较松驰，因而短程更加有序，而长程的有序结构消失. 图 6 显示了该肽链在模拟过程中从延展状态到蜷曲状态的逐步过渡过程.

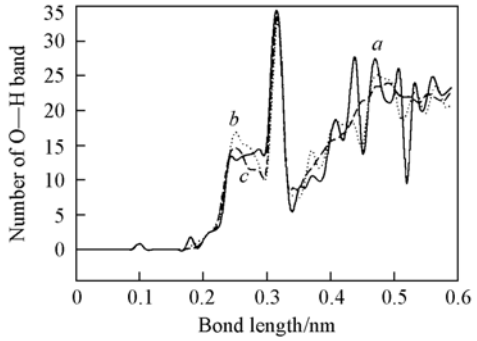

Fig. 5 Population of O—H bond in peptide chain made up of 397 atoms

a. 0.2 ps; *b*. 2 ps; *c*. 20 ps.

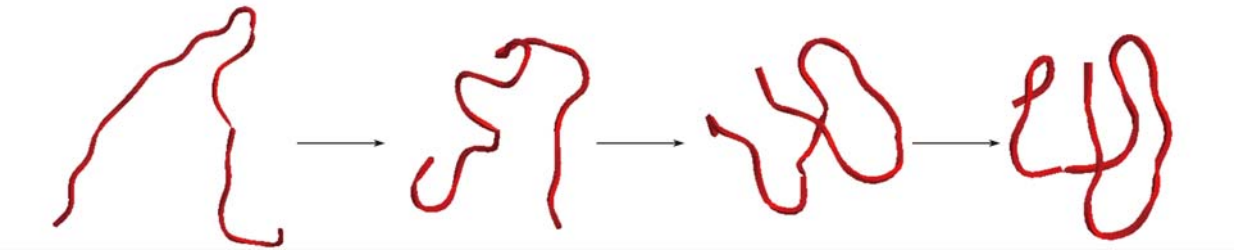

Fig. 6 Evolution of the peptide chain containing 397 atoms in molecular dynamics simulation

6 结 论

本文成功地实现了 WCCS 平台上生物分子动力学模拟的 GPU 运算. 初步的测试结果表明，相对于 CPU 计算，GPU 在计算速度上有非常大的提升. CHARMM 力场的成功使用，将对生物分子的动力学模拟起到极大的促进作用. 本文的工作只是成功开发 GPU 动力学模拟程序的良好开端，将来还会将更多的功能加入到现有的 GPU 动力学模拟程序，并且在计算速度上还会有更大的提升.

参 考 文 献

[1] Vogt L., Olivares-Amaya R., Kermes S., *et al*.. J. Phys. Chem. A[J], 2008, **112**: 2049—2057

[2] Ufimtsev I. S., Martinez T. J.. J. Chem. Theo. Comput. [J] 2008, **4**: 222—231

[3] Yasuda K.. J. Comp. Chem. [J], 2008, **29**: 334—342

[4] Qiao W., Ebert D. S., Entezari A., *et al*.. IEEE Visualization 2005, Proceedings[C], 2005: 319—326

[5] Gossage K.. J. Mol. Diagn. [J], 2007, **9**: 561

[6] Anderson A. G., Goddard W. A. Ⅲ, Schröder P.. Comput. Phys. Commun. [J], 2007, **177**: 298—306 (Online at http://iic.harvard edu/about/index. html)

[7] Frenkel D., Smit B.. Understanding Molecular Simulation: From Algorithms to Applications[M], San Diego: Academic Press, 2002: 1—6

[8] Allen M. P., Tildesley D.. J. Computer Simulation of Liquids[M], Oxford: Oxford University Press, 2005: 1—4

[9] van Meel J. A., Arnold A., Frenkel D., *et al*.. Mol. Simal. [J], 2008, **34**: 259—266

[10] Stone J. E., Phillips J. C., Freddolino P. L., *et al*.. J. Comp. Chem. [J], 2007, **28**: 2618—2640

[11] Brooks B. R., Bruccoleri R. E., Olafson B. D., *et al*.. J. Comput. Chem. [J], 1983, **4**: 187—217

[12] Momany F. A., Rone R.. J. Comput. Chem. [J], 1992, **13**: 888—900

[13] Mackerell A. D., Wiorkiewiczkuczera J., Karplus M.. J. Am. Chem. Soc. [J], 1995, **117**: 11946—11975

[14] Mackerell A. D., Bashford D., Bellott M., *et al*.. J. Phys. Chem. B[J], 1998, **102**: 3586—3616

Molecular Dynamics Simulation Using Graphics Processing Units

SONG Guo-Liang, WENG Jing-Wei, LI Zhen-Hua, WANG Wen-Ning, FAN Kang-Nian*

(*Shanghai Key Laboratory of Molecular Catalysis and Innovative Materials*, *Department of Chemistry*, *Center for Theoretical Chemical Physics*, *Fudan University*, *Shanghai* 200433, *China*)

Abstract In this paper, the molecular dynamics program with CHARMM force field is developed on Graphics processing unit (GPU) at Windows computer cluster server (WCCS) system. From the testing results of peptide chain, the efficiency of GPU is outstanding compared with that of CPU. The efficiency on NVIDIA GeForce 8800 GT GPU is at lease 10 times faster than that on a single Athlon 2.0G CPU. When the total molecule size is enlarged, the number of vacant parallel units in GPU decreases, so the parallel efficiency increases. At the same time, while the fragment size is enlarged, the buffer size of a fragment decreases relatively, so the total efficiency also increases. The maximum efficiency ratio of GPU/CPU reaches to 28 times according to our test. At last, a peptide chain with 397 atoms is tested for simulation and the population of hydrogen bond is described at different time steps.

Keywords Molecular dynamics; Graphics processing unit; CHARMM; Windows computer cluster server

(Ed.: D, I)