Использование технологий параллельного программирования в программах моделирования молекулярной динамики на примере PUMA-CUDA

И.В. Лихачев

Институт математических проблем биологии РАН – филиал ИПМ им. М.В. Келдыша РАН

Аннотация. В работе обсуждается применение технологий параллельного программирования в программах моделирования молекулярной динамики. Даётся обзор аппаратных платформ, а также программ, в которых они применены. На примере программы моделирования собственной разработки PUMA-CUDA объясняется, каким образом технологии помогают ускорить процесс вычислительных экспериментов. Основная вычислительная сложность заключается в вычислении невалентных взаимодействий, для их расчета применяются графические ускорители. При отсутствии таковых – многоядерные процессоры согласно технологии OpenMP. Технологию MPI рекомендуется применяется для задач, которым недостаточно ресурсов одного вычислительного узла.

Ключевые слова: программы моделирования молекулярной динамики, PUMA-CUDA, OpenMP, MPI, CUDA.

Using parallel programming technologies in molecular dynamics simulation: a PUMA-CUDA case study

I.V. Likhachev

Institute of Mathematical Problems of Biology RAS – the Branch of Keldysh Institute of Applied Mathematics of RAS

Abstract. This work describes the use of parallel programming technologies in molecular dynamics modeling programs. A review of hardware platforms is given, as well as the programs in which they are applied. On the example of the PUMA-CUDA modeling program, it is explained how technologies help accelerate the process of computational experiments. The main computational complexity is related to the calculation of non-valent interactions, for the calculation of which graphic accelerators are used. In the absence of such, multi-core processors using OpenMP technology are used. MPI technology is recommended for tasks for which the resources of one computing node are insufficient.

Keywords: molecular dynamics simulation programs, PUMA-CUDA, OpenMP, MPI, CUDA.

1. Введение

Методом моделирования молекулярной динамики пользуются во многих исследовательских лабораториях. Популярность метода обязана в том числе бурному развитию вычислительной техники в 1980-2020 годах. Увеличение вычислительных мощностей зависит от двух главных факторов: частоты процессоров и количества вычислительных ядер. Тактовая частота перестала стремительно расти после выпуска процессора Intel Pentium IV 3,06 ГГц в 2003 году. Спустя 20 лет большинство процессоров имеет те же частоты 3-4 ГГц. С этого же момента начался рост числа вычислительных ядер. Таким образом, один из основных методов ускорения вычислений молекулярной динамики – использование параллельного программирования, чтобы одновременно производить вычисления на всех ядрах центральных процессоров (технологии OpenMP и MPI).

Помимо использования многоядерных процессоров полезно также использовать специализированные вычислители, такие как графические ускорители, работающие на схеме GPGPU. Графические ускорители изначально создавались параллельными. Причем параллелизм был и по количеству вычислительных ядер (которые вначале были узкоспециализированными), и по доступу к памяти.

Выбирая конкретную технологию параллельного программирования, необходимо учитывать количество доступных вычислителей, а также скорость обмена информацией между узлами либо различными типами памяти, что может послужить узким горлышком в процессе функционирования программы.

Для решения одной задачи можно также применять одновременно разные технологии параллельного программирования. Иногда такую связку называют гибридной (hybrid MPI and OpenMP). Технологию не следует воспринимать как нечто новое. Задача разделяется крупными блоками на разные узлы вычислительного кластера при помощи технологии MPI. Подзадача на каждом узле распараллеливается при помощи OpenMP (либо CUDA). Единственно, что нужно – чтобы компиляторы поддерживали соответствующие технологии.

Цель данной работы – дать обзор существующим популярным пакетам моделирования молекулярной динамики, чтобы, учитывая их опыт, создавать собственный программный комплекс, каковым является PUMA-CUDA. Эта программа рассчитана на использование как на персональных компьютерах, оснащенных графическими ускорителями, поддерживающими технологию NVIDIA CUDA, так и вычислительные кластеры (так называемые суперкомпьютеры).

Разработка собственной программы моделирования позволяет проводить численные эксперименты с биологическими и иными системами, задавая уникальные силовые воздействия.

В разделе 2 дан обзор существующих популярных пакетов моделирования и применяемых ими техник параллельного программирования.

Раздел 3 посвящен обзору возможностей собственного пакета моделирования молекулярной динамики PUMA-CUDA.

2. Внедрение технологий параллельного программирования в пакеты моделирования молекулярной динамики.

MPI (Message Passing Interface). Применялся с конца 1990-х — начала 2000-ых в программных пакетах GROMACS [1,2], NAMD [3,4], LAMMPS [5,6], AMBER [7–9], DL_POLY [10,11].

Даёт возможность распараллеливания между узлами суперкомпьютеров. Также предоставляет возможность моделировать очень большие системы на кластерах.

OpenMP (shared memory, многопоточность). Применялся с середины 2000-ых в пакетах моделирования GROMACS, LAMMPS, CHARMM. Позволяет получить ускорение на уровне одного узла (многоядерного центрального процессора).

CUDA/OpenCL (вычисления на графических процессорах). Применяется для моделирования в пакетах GROMACS (с 2010 годов),

AMBER (с 11-ой версии 2010 года), NAMD (активная поддержка CUDA с версии 2.8 2011 года), HOOMD-blue (с самого начала разрабатывался под GPU). Технология позволяет получить ускорение расчётов в 10–100 раз. Применение вычислений на GPU также экономит электричество, используемое в вычислительных центрах.

Hybrid MPI + OpenMP/CUDA. Применяется после 2012 года в GROMACS, NAMD, LAMMPS. Позволяет эффективное использование все ресурсы: CPU + GPU. Даёт гибкую масштабируемость на суперкомпьютерах.

Совместно с «многоядерными» технологиями параллельного программирования в программном пакете GROMACS применялись **векторные инструкции процессоров** [12]. Технология не столь распространена среди других пакетов ввиду высокой сложности разработки. Требуется мыслить в рамках вектора определённого размера, поддерживаемого той или иной технологией векторизации (SSE4.1, AVX, AVX+FMA). Алгоритмы также включают в себя перегруппировку частиц.

Собственные технологии распределения нагрузки (load balancing) внедрены в крупные пакеты моделирования: NAMD (Charm++ runtime, динамическое распределение нагрузки) и GROMACS (пакет domain decomposition + dynamic load balancing).

FPGA (**ПЛИС**, интегральная схема программируемой логики). На данный момент популярные пакеты не поддерживают FPGA. Хотя FPGA могут обеспечить ускорение в отдельных вычислительных задачах, таких как расчёт короткодействующих сил, общая выгода на уровне всего приложения ограничена из-за программных накладных расходов и сложности интеграции с существующими программами. Авторы [13] приходят к выводу, что FPGA вряд ли заменят GPU в качестве основного ускорителя, но могут быть полезны в гибридных системах для задач, чувствительных к задержкам, например, в сетевых коммуникациях между узлами суперкомпьютеров.

Вывод. Современный пакет моделирования молекулярной динамики должен поддерживать работу как с современными многоядерными процессорами, так и с графическими процессорами. Причем поддержка графических процессоров должна быть как отключаемая опция. Подобная универсальность позволить эффективно использовать программу в любой вычислительной среде.

3. Собственный пакет моделирования молекулярной динамики PUMA-CUDA

При создании собственного пакета моделирования молекулярной динамики PUMA-CUDA [14,15] авторами преследовались следующие цели:

- Поддержка общепринятых силовых полей (например, AMBER-99).

- Возможность включения дополнительных силовых воздействий на программном уровне для решения новых задач моделирования.
- Использование технологий параллельного программирования для обеспечения высокой скорости расчетов.
- Прозрачность расчетов.

Судя из названия, программный пакет моделирования молекулярной динамики PUMA-CUDA базируется на потенциалах межатомных взаимодействий программного пакета PUMA [16–18], но позволяет проводить расчеты быстрее за счет технологий параллельного программирования, в первую очередь NVIDIA CUDA, а также используя произвольное количество ядер центральных процессоров как одного компьютера, так и вычислительного кластера.

Начиная с 2010-ых годов технология NVIDIA CUDA представляется как весьма перспективная технология параллельного программирования ввиду возможности её применения на системах различного класса: как на дорогих суперкомпьютерах, так и на доступных персональных компьютерах. И тот и другой факты могут стать решающими для различных исследовательских групп.

PUMA-CUDA использует графические процессоры для вычисления невалентных взаимодействий – электростатических и Ван-дер-Ваальсовых сил и энергий. Ожидаемый выигрыш – до 25 раз (точное число сложно указать ввиду возможности подборки различных связок CPU-GPU). С учетом того, что при использовании параллельного программирования парные силы и потенциалы рассчитываются дважды.

Именно невалентные взаимодействия предполагают взаимодействия всех атомов со всеми, получая вычислительную сложность $O(N^2)$, где N – количество атомов в системе. Полагая, что электростатическое и Ван-дер-Ваальсово взаимодействия распространяются только на ближайшее окружение атомов, введение радиуса взаимодействия позволяет сократить вычислительную сложность до O(N*M), где M – количество атомов в пределах радиуса взаимодействия.

Для этого в программный комплекс PUMA-CUDA внедрена поддержка метода сканирования по пространству и метод составления списков связанных атомов (список Верле). Задача у этих методов одна: зная атом, структуры данных позволяют быстро найти его соседей, не вызывая полный перебор атомов.

Если сравнивать 2010-е и 2020-е годы, то в последнее время стали доступны центральные процессоры с большим количеством ядер (16 ядер/24 потока и более). Ввиду этого факта в PUMA-CUDA внедрена также технология OpenMP для расчета невалентных взаимодействий.

Выбор режима работы – на центральном или на графическом процессоре – задаётся при помощи директив препроцессора. Таким образом

программа способна работать в любой вычислительной среде, как с поддержкой графический процессоров, так и без них.

Расчет валентных взаимодействий – валентных связей, валентных углов, торсионных углов, неправильных торсионных углов – не вызывает большой вычислительной сложности. Но при выполнении расчетов на каждом шаге может замедлить производительность программы в целом. Заметим, что при выполнении вычислений невалентных взаимодействий работает графический ускоритель, а центральный процессор при этом простаивает. В PUMA-CUDA используется это время для расчета валентных взаимодействий путём создания отдельных вычислительных потоков (экземпляров класса C++ std::thread) для каждого типа валентных взаимодействий (всего 3 дополнительных потока). На практике получается, что расчет валентных взаимодействий заканчивается по времени гораздо раньше, чем невалентных. Но всё же в программе предусмотрено ожидание завершения всех вычислительных потоков на тот случай, если расчет невалентных взаимодействия завершится раньше.

Для расчета больших систем в программный комплекс внедрена технология MPI для работы на кластерах. Все координаты копируются на все вычислительные узлы. Но на каждом узле рассчитываются только своя часть невалентных взаимодействий.

На данном этапе развития аппаратной базы технологии ПЛИС решено не использовать, несмотря на имеющийся положительный опыт внедрения этой технологии у авторов программного комплекса. При разработке одномерного пакета моделирования молекулярной динамики цепочек ДНК согласно модели ПБД для повышения быстродействия удачно был использован ПЛИС (FPGA) [19]. Однако, как обсуждается в работе [19], для написания полноценной трехмерной программы моделирования у доступных ускорителей ПЛИС просто не хватит ресурсов.

4. Выводы

В программе моделирования молекулярной динамики PUMA-CUDA предусмотрены следующие уровни параллелизма:

- Расчет невалентных взаимодействий на графических ускорителях при помощи технологии NVIDIA CUDA.
- Расчет невалентных взаимодействий на нескольких ядрах центрального процессора по технологии OpenMP.
- Расчет невалентных взаимодействий на разных узлах вычислительного кластера по технологии MPI.
- Расчет валентных взаимодействий одновременно с невалентными при помощи создания потоков операционной системы (по одному потоку на каждый тип взаимодействий).

Вычислительно наиболее ресурсоемкая задача моделирования – расчет невалентных взаимодействий. Именно от скорости её выполнения зависит быстродействие всей программы. Поэтому алгоритм расчета невалентных взаимодействий реализован на GPU. При отсутствии графического ускорителя, поддерживающего технологию NVIDIA CUDA, расчет невалентных взаимодействий производится на центральном процессоре с применением технологии OpenMP. Этот режим перспективен особенно при дальнейшем повышении количества ядер центральных процессоров.

Для больших систем применяется технология MPI. Однако при текущем уровне развития графических процессоров вся система целиком помещается в память одного GPU. Кроме того, применение технологии MPI ведёт к временным задержкам, требующимся на сетевой обмен между вычислительными узлами кластера. Поэтому, особенно при моделировании различных реализаций одной системы, выгоднее моделирование одной системы на одном вычислительном узле.

Одновременный расчет валентных взаимодействий вместе с невалентными сделан ради дополнительного повышения быстродействия, чтобы уменьшить время простоя ядер центрального процессора, неиспользуемых при расчете невалентных взаимодействий на GPU.

Использование технологий параллельного программирования – неотъемлемая часть современной программы моделирования. Это даёт возможность производить расчеты с приемлемой скоростью. Но само создание своей собственной программы преследует иные цели:

- Детальный контроль над всеми стадиями расчетов.
- Поддержка новых силовых полей.
- Возможность задания новых типов силовых воздействий, не относящихся к силовым полям.

Благодарности

Автор благодарит Суперкомпьютерный центр коллективного пользования ИПМ им. М.В. Келдыша РАН за предоставленные ресурсы.

Финансовая поддержка

Работа выполнена в рамках госзадания ИПМ им. М.В. Келдыша РАН [FFMN-2025-0024].

Литература.

1. Abraham M.J., Murtola T., Schulz R., Páll S., Smith J.C., Hess B., Lindahl E. GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers. // SoftwareX. 2015. Vol. 1–2. P. 19–25. https://doi.org/10.1016/j.softx.2015.06.001.

2. Páll S., Zhmurov A., Bauer P., Abraham M., Lundborg M., Gray A., Hess B., Lindahl E. Heterogeneous parallelization and acceleration of molecular dynamics simulations in GROMACS. // J Chem Phys. 2020. Vol. 153, № 13. P. 134110. https://doi.org/10.1063/5.0018516.
3. Scalable molecular dynamics with NAMD - Phillips. 2005 // Journal of Computational Chemistry - Wiley Online Library. URL: https://onlinelibrary.wiley.com/doi/full/10.1002/jcc.20289.
4. Phillips J.C., Hardy D.J., Maia J.D.C., Stone J.E., Ribeiro J.V., Bernardi R.C., Buch R., Fiorin G., Hénin J., Jiang W., McGreevy R., Melo M.C.R., Radak B.K., Skeel R.D., Singharoy A., Wang Y., Roux B., Aksimentiev A., Luthey-Schulten Z., Kalé L.V., Schulten K., Chipot C., Tajkhorshid E. Scalable molecular dynamics on CPU and GPU architectures with NAMD. // J Chem Phys. 2020. Vol. 153, № 4. P. 044130. https://doi.org/10.1063/5.0014475.
5. Plimpton S. Fast Parallel Algorithms for Short-Range Molecular Dynamics. // Journal of Computational Physics. 1995. Vol. 117, № 1. P. 1–19. https://doi.org/10.1006/jcph.1995.1039.
6. Thompson A.P., Aktulga H.M., Berger R., Bolintineanu D.S., Brown W.M., Crozier P.S., in 't Veld P.J., Kohlmeyer A., Moore S.G., Nguyen T.D., Shan R., Stevens M.J., Tranchida J., Trott C., Plimpton S.J. LAMMPS - a flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales. // Computer Physics Communications. 2022. Vol. 271. P. 108171. https://doi.org/10.1016/j.cpc.2021.108171.
7. Routine Microsecond Molecular Dynamics Simulations with AMBER on GPUs. 2. Explicit Solvent Particle Mesh Ewald // Journal of Chemical Theory and Computation. URL: https://pubs.acs.org/doi/10.1021/ct400314y .
8. The Amber biomolecular simulation programs – Case – 2005 // Journal of Computational Chemistry. Wiley Online Library. URL: https://onlinelibrary.wiley.com/doi/full/10.1002/jcc.20290.
9. Le Grand S., Götz A.W., Walker R.C. SPFP: Speed without compromise—A mixed precision model for GPU accelerated molecular dynamics simulations // Computer Physics Communications. 2013. Vol. 184, № 2. P. 374–380. https://doi.org/10.1016/j.cpc.2012.09.022.
10. DL_POLY: Application to molecular simulation // Molecular Simulation. 2010. Vol 28, No 5. URL: https://www.tandfonline.com/doi/abs/10.1080/08927020290018769.
11. Smith W., Todorov I.T., Leslie M. The DL_POLY molecular dynamics package. // Zeitschrift für Kristallographie - Crystalline Materials. De Gruyter (O). 2005. Vol. 220, № 5–6. P. 563–566. https://doi.org/10.1524/zkri.220.5.563.65076.
12. Páll S., Hess B. A flexible algorithm for calculating pair interactions on SIMD architectures. // Computer Physics Communications. 2013. Vol. 184, № 12. P. 2641–2650. https://doi.org/10.1016/j.cpc.2013.06.003.

13. Schaffner M., Benini L. On the Feasibility of FPGA Acceleration of Molecular Dynamics Simulations // arXiv:1808.04201. arXiv. 2018. https://doi.org/10.48550/arXiv.1808.04201.
14. Likhachev I.V., Balabaev N.K. Parallelism of different levels in the program of molecular dynamics simulation PUMA-CUDA // Доклады Международной конференции “Математическая биология и биоинформатика”. Под ред. В.Д. Лахно. Том 7. Пущино: ИМПБ РАН. 2018. Статья № e44. https://doi.org/10.17537/icmbb18.53.
15. Likhachev I.V. Molecular dynamics simulation package PUMA-CUDA. Working documentation. // http://lmd.impb.ru/Puma/Manual_PUMA-CUDA.pdf.
16. Lemak A.S., Balabaev N.K. A comparison between collisional dynamics and brownian dynamics // Molecular Simulation. Taylor & Francis. 1995. Vol. 15, No. 4. https://doi.org/10.1080/08927029508022336.
17. Lemak A.S., Balabaev N.K. Molecular dynamics simulation of a polymer chain in solution by collisional dynamics method // Journal of Computational Chemistry. John Wiley & Sons. 1996. Vol. 17, No 15. https://doi.org/10.1002/(SICI)1096-987X(19961130)17:15<1685::AID-JCC1>3.0.CO;2-L.
18. Balabaev N.K., Lemak A.S. Molecular dynamics of a linear polymer in a hydrodynamic flow // Russian Journal of Physical Chemistry A. 1995. Vol. 69, No 1. https://elibrary.ru/item.asp?id=21474281.
19. Андреев С.С., Дбар С.А., Лацис А.О., Лихачев И.В., Плоткина Е.А., Фиалко Н.С. Типовая структура программы одномерного моделирования динамики цепочки ДНК и ее схемная реализация. // Препринты ИПМ Им. М.В. Келдыша. 2018. № 171. P. 1–16. https://doi.org/10.20948/prepr-2018-171.

References

1. Abraham M.J., Murtola T., Schulz R., Páll S., Smith J.C., Hess B., Lindahl E. GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers. // SoftwareX. 2015. Vol. 1–2. P. 19–25. https://doi.org/10.1016/j.softx.2015.06.001.
2. Páll S., Zhmurov A., Bauer P., Abraham M., Lundborg M., Gray A., Hess B., Lindahl E. Heterogeneous parallelization and acceleration of molecular dynamics simulations in GROMACS. // J Chem Phys. 2020. Vol. 153, № 13. P. 134110. https://doi.org/10.1063/5.0018516.
3. Scalable molecular dynamics with NAMD - Phillips. 2005 // Journal of Computational Chemistry - Wiley Online Library. URL: https://onlinelibrary.wiley.com/doi/full/10.1002/jcc.20289.
4. Phillips J.C., Hardy D.J., Maia J.D.C., Stone J.E., Ribeiro J.V., Bernardi R.C., Buch R., Fiorin G., Hénin J., Jiang W., McGreevy R., Melo M.C.R.,

Radak B.K., Skeel R.D., Singharoy A., Wang Y., Roux B., Aksimentiev A., Luthey-Schulten Z., Kalé L.V., Schulten K., Chipot C., Tajkhorshid E. Scalable molecular dynamics on CPU and GPU architectures with NAMD. // J Chem Phys. 2020. Vol. 153, № 4. P. 044130. https://doi.org/10.1063/5.0014475.

5. Plimpton S. Fast Parallel Algorithms for Short-Range Molecular Dynamics. // Journal of Computational Physics. 1995. Vol. 117, № 1. P. 1–19. https://doi.org/10.1006/jcph.1995.1039.
6. Thompson A.P., Aktulga H.M., Berger R., Bolintineanu D.S., Brown W.M., Crozier P.S., in 't Veld P.J., Kohlmeyer A., Moore S.G., Nguyen T.D., Shan R., Stevens M.J., Tranchida J., Trott C., Plimpton S.J. LAMMPS - a flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales. // Computer Physics Communications. 2022. Vol. 271. P. 108171. https://doi.org/10.1016/j.cpc.2021.108171.
7. Routine Microsecond Molecular Dynamics Simulations with AMBER on GPUs. 2. Explicit Solvent Particle Mesh Ewald // Journal of Chemical Theory and Computation. URL: https://pubs.acs.org/doi/10.1021/ct400314y.
8. The Amber biomolecular simulation programs – Case – 2005 // Journal of Computational Chemistry. Wiley Online Library. URL: https://onlinelibrary.wiley.com/doi/full/10.1002/jcc.20290.
9. Le Grand S., Götz A.W., Walker R.C. SPFP: Speed without compromise—A mixed precision model for GPU accelerated molecular dynamics simulations // Computer Physics Communications. 2013. Vol. 184, № 2. P. 374–380. https://doi.org/10.1016/j.cpc.2012.09.022.
10. DL_POLY: Application to molecular simulation // Molecular Simulation. 2010. Vol 28, No 5. URL: https://www.tandfonline.com/doi/abs/10.1080/08927020290018769.
11. Smith W., Todorov I.T., Leslie M. The DL_POLY molecular dynamics package. // Zeitschrift für Kristallographie - Crystalline Materials. De Gruyter (O). 2005. Vol. 220, № 5–6. P. 563–566. https://doi.org/10.1524/zkri.220.5.563.65076.
12. Páll S., Hess B. A flexible algorithm for calculating pair interactions on SIMD architectures. // Computer Physics Communications. 2013. Vol. 184, № 12. P. 2641–2650. https://doi.org/10.1016/j.cpc.2013.06.003.
13. Schaffner M., Benini L. On the Feasibility of FPGA Acceleration of Molecular Dynamics Simulations // arXiv:1808.04201. arXiv. 2018. https://doi.org/10.48550/arXiv.1808.04201.
14. Likhachev I.V., Balabaev N.K. Parallelism of different levels in the program of molecular dynamics simulation PUMA-CUDA // Proceedings of the International Conference "Mathematical Biology and Bioinformatics". Edited by V.D. Lakhno. V. 7. Pushchino: IMPB RAS. 2018. Paper No e44. https://doi.org/10.17537/icmbb18.53.

15. Likhachev I.V. Molecular dynamics simulation package PUMA-CUDA. Working documentation. // http://lmd.impb.ru/Puma/Manual_PUMA-CUDA.pdf.
16. Lemak A.S., Balabaev N.K. A comparison between collisional dynamics and brownian dynamics // Molecular Simulation. Taylor & Francis. 1995. Vol. 15, No. 4. https://doi.org/10.1080/08927029508022336.
17. Lemak A.S., Balabaev N.K. Molecular dynamics simulation of a polymer chain in solution by collisional dynamics method // Journal of Computational Chemistry. John Wiley & Sons. 1996. Vol. 17, No 15. https://doi.org/10.1002/(SICI)1096-987X(19961130)17:15<1685::AID-JCC1>3.0.CO;2-L.
18. Balabaev N.K., Lemak A.S. Molecular dynamics of a linear polymer in a hydrodynamic flow // Russian Journal of Physical Chemistry A. 1995. Vol. 69, No 1. https://elibrary.ru/item.asp?id=21474281.
19. Andreev S. S., Dbar S. A., Lacis A. O., Likhachev I. V., Plotkina E. A., Fialko N. S.. Typical structure of the program for dynamics of one-dimensional DNA molecular chain simulation and its hardware implementation // KIAM Preprint № 171, Moscow. 2018. No 171. P. 1–16. https://doi.org/10.20948/prepr-2018-171.