DOI: 10.7242/1999-6691/2012.5.3.31
УДК 519.6, 539.231, 539.6

# МОЛЕКУЛЯРНО–ДИНАМИЧЕСКОЕ МОДЕЛИРОВАНИЕ ОСАЖДЕНИЯ МЕДНЫХ НАНОКЛАСТЕРОВ С ПРИМЕНЕНИЕМ ГРАФИЧЕСКИХ ПРОЦЕССОРОВ

M.S. Ozhgibesov[1], А.В. Уткин[2], В.М. Фомин[2], T.S. Leu[1], C.H. Cheng[1]

[1]*National Cheng Kung University Tainan, Taiwan R.O.C.*
[2]*Институт теоретической и прикладной механики им. С.А. Христиановича СО РАН, Новосибирск, Россия*

В данной работе рассмотрены различные аспекты использования технологии CUDA применительно к решению задач молекулярной динамики. Созданный на основе CUDA комплекс программ позволил провести детальное исследование процесса столкновения медного кластера с металлической подложкой, имеющей углеродную пленку. Установлено, что осаждение кластера не наблюдается, если скорость его падения ниже критической скорости для заданного угла падения. Получена графическая зависимость между критическими значениями скорости и угла падения кластера на поверхность подложки.

*Ключевые слова:* молекулярная динамика, CUDA, параллельные вычисления, осаждение кластеров

# MOLECULAR DYNAMIC SIMULATION OF COPPER NANOCLUSTER DEPOSITION USING GRAPHICS PROCESSING UNITS (GPU)

M.S. Ozhgibesov[1], A.V. Utkin[2], V.M. Fomin[2], T.S. Leu[1] and C.H. Cheng[1]

[1]*National Cheng Kung University Tainan, Taiwan R.O.C.*
[2]*Khristianovich Institute of Theoretical and Applied Mechanics SB RAS, Novosibirsk, Russia*

Various aspects of CUDA technology implementation are considered in the context of the problems of molecular dynamics. The developed CUDA-based program allows us to study in detail physical processes accompanying the collision of a copper cluster with a copper substrate having a one-atom layer of carbon on its top surface. It has been found that cluster deposition cannot be observed if the velocity of the descending cluster is lower than the critical velocity at the prescribed incidence angle. The critical values of velocity are plotted against the critical values of the angle of incidence of the cluster onto the substrate surface.

*Key words:* molecular dynamics, CUDA, parallel computations, cluster deposition

## 1. Введение

Методы молекулярной динамики (МД) являются мощными инструментами, которые позволяют изучать нано- и микропроцессы, однако их алгоритмы предъявляют большие требования к вычислительным ресурсам. Первые исследования с использованием методов МД проводились на примере небольших систем атомов, но появление относительно недорогих вычислительных систем дало толчок к укрупнению моделируемых систем и увеличению продолжительности моделируемых временных отрезков. Отметим, что до середины 1990-х большинство МД программ было серийным, однако развитие параллельных вычислительных систем (компьютерных кластеров) привело к новому витку развития молекулярно-динамических алгоритмов, сделавшему возможным существенное усложнение исследуемых моделей. По сравнению с настольными компьютерами, вычислительные кластеры имеют бóльшую производительность, но и бóльшие стоимость и затраты на обслуживание.

Следующим шагом в эволюции вычислительных систем можно считать появление графических видеопроцессоров общего назначения (General Purpose Graphics Processing Units — GPGPU) и соответствующих языков программирования, таких как графический шейдерный язык (GL Shader Language — GLSL) [1], Си для графики [2], Универсальная архитектура вычислительных устройств (Compute Unified Device Architecture — CUDA) [3] и так далее. Вышеупомянутые языки являются Си-подобными, в то время как в научной среде наиболее популярным языком программирования является Fortran. Компании Portland Group и NVIDIA совместно разработали язык CUDA Fortran, последняя версия которого позволяет задействовать все вычислительные ресурсы видеокарты [4]. С момента своего появления видеокарты активно задействовались только для решения графических задач. Данные устройства разрабатывались с целью снижения нагрузки на центральное процессорное устройство (Central Processing Unit — CPU) при прорисовке сложных трехмерных изображений, следовательно, графические карты потенциально способны стать мощными вычислительными устройствами.

МД алгоритмы хорошо поддаются распараллеливанию, следовательно применение графических карт может быть очень выгодным по причине их низкой стоимости, компактных размеров и меньшей потребляемой мощности по сравнению с вычислительными кластерами. Быстрая память и реализация многопоточности процесса вычислений на уровне «железа» обеспечивают современным видеокартам

превосходство над традиционными CPU (по скорости решения задач) от 10 до 30 раз в большинстве случаев [5], и до 100 раз в случае решения идеализированных задач [6, 7].

Цель данной работы была двоякой: с использованием CUDA технологии требовалось, во-первых, сравнить эффективность различных МД алгоритмов в случае их реализации на видеокартах; во-вторых, исследовать процессы, сопутствующие столкновению сферического медного кластера с медной подложкой, загрязненной углеродом. В связи с этим рассмотрены различные аспекты работы и реализации алгоритмов, включающих технологию CUDA, на примере решения реальных задач.

В ходе вычислительных экспериментов протестированы две графические карты, принадлежащие к различным поколениям: Tesla C1060 с архитектурой Tesla и GeForce GTX460, на базе архитектуры Fermi. Все CUDA версии МД алгоритмов, приводимые ниже, реализованы с применением языка PGI Fortran. Обнаружено, что игровая видеокарта GTX460 работает быстрее, чем Tesla C1060, с алгоритмами, требующими $N^2$ операций, в то же время Tesla C1060 примерно на 10% производительнее при оперировании алгоритмами, требующими $N$ операций.

На основе полученных результатов создана CUDA программа для МД моделирования осаждения медного кластера, позволившая исследовать соударения медного кластера с загрязненной подложкой в широком диапазоне начальных скоростей и углов падения. Обнаружены критические значения скорости и угла падения, ниже которых осаждение медного кластера на подложку не наблюдается. Осуществлено сравнение скоростей, с которыми производятся вычисления CUDA программами и программами, использующими технологию Message passing interface — MPI, с одномерной динамической балансировкой. Программы апробировались на двух компьютерных кластерах.

## 2. Физическая модель и математическая постановка задачи

Первоначальное тестирование и отладка численных алгоритмов и программ проводились на достаточно простой физической модели кубика меди. Кубик представлял собой систему атомов меди, структурированную в кубическую гранецентрированную решетку (ГЦК) с размером ячейки 3,615 Å, что соответствовало параметрам медного кристалла. Начальная температура системы задавалась путем сообщения атомам скоростей в согласии с распределением Максвелла для температуры 300 К. Получив начальные скорости, атомы совершали тепловые колебания, при этом общая структура ГЦК не нарушалась. Моделируемый кубик свободно располагался в пространстве; таким образом, на атомы действовали только внутренние силы (силы взаимодействия с соседними атомами).

Межатомные взаимодействия описывались парным экспоненциальным потенциалом Морзе [8]: $\Phi_M\left(r\right) = D_e\left[e^{-2b\left(r-R_e\right)} - 2e^{-b\left(r-R_e\right)}\right]$, где $D_e = 0,3429$ эВ, $b = 13,588$ нм$^{-1}$, $R_e = 0,2866$ нм — параметры потенциала, а $r$ — межатомное расстояние. Выбор потенциала Морзе для первичного тестирования алгоритма связан в первую очередь с его простотой; он позволяет эффективно отлаживать компьютерные программы и сравнительно легко обнаруживать ошибки. Полное количество атомов меди $N$ в рассматриваемом образце варьировалось от 14896 до 976563. Интегрирование уравнений движения проводилось с использованием схемы первого порядка точности, а общее время развития процесса составляло 20000 временных шагов, величина шага — $10^{-16}$ с.

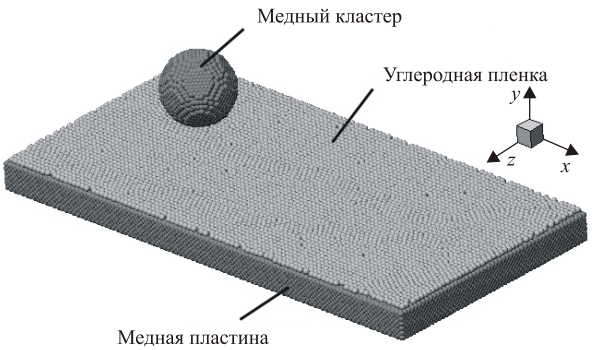


**Рис. 1.** Модель осаждения медного кластера на медную подложку, имеющую углеродную пленку толщиной в один атом

Второй из рассмотренных моделей являлась более сложная физическая система, описывающая столкновение медного кластера с загрязненной углеродом медной подложкой (Рис. 1). Под медным кластером в этой модели понимается сферическое тело, имеющее ГЦК структуру кристаллической меди. В настоящее время существуют работы, описывающие процессы формирования наноструктур [22], однако здесь осаждаемый кластер образуется путем отсечения от кубика меди атомов, лежащих вне сферической области, вписывающейся в кубик. При численном моделировании кластеры имели диаметры 50 Å и 140 Å. В начальный момент времени задавались скорость $V_{inc}$ и угол падения кластера (угол между вектором скорости и нормалью к поверхности) $\alpha_{inc}$, которые являлись контролируемыми внешними параметрами. Следует отметить, что начальные скорости атомов были одинаковыми и равнялись скорости поступательного движения центра масс кластера $V_{inc}$.

На медную подложку, состоящую из 111656 атомов, предварительно был осажден монослой углерода (углеродная пленка), состоящий из 4963 атомов (Рис. 1). Для имитации бесконечного размера подложки, положение которой в пространстве не менялось, использовался хорошо апробированный в численных

экспериментах прием искусственной вязкости, которая действовала на атомы подложки [9]. Это позволило также описать диссипацию энергии, привносимой кластером в бесконечную подложку.

Взаимодействие атомов меди, как внутри кластера, так и между атомами кластера и подложки, описывалось моделью погружённого атома (Embedded Atom Method — EAM), и соответствующий многочастичный потенциал имел вид [10]:

$$\Phi_{EAM} = \sum_i F\left(\rho_i\right) + \sum_i^N \sum_{j>i}^N \Phi\left(r_{ij}\right),$$

где $r_{ij}$ — расстояние между атомами $i$ и $j$; $\rho_i$ — функция электронной плотности; $\Phi\left(r_{ij}\right)$ — парный потенциал.

Взаимодействие атомов углерода между собой и с атомами меди определялось парным потенциалом [11]:

$$\Phi_K = B\exp\left(-C \cdot r_{ij}\right) - A/r_{ij},$$

где $C$, $A$ и $B$ — константы, найденные в работе [12], а $r_{ij}$ — расстояние между атомами.

В качестве основных характеристик, позволяющих исследовать процесс осаждения кластеров на подложку, были выбраны следующие: температура, устанавливаемая исходя из кинетической энергии хаотического движения атомов; температура в области контакта; компоненты скорости центра масс кластера. Оценка величины температуры в области контакта давала возможность выявить наличие плавления вещества после ударного взаимодействия с подложкой и осуществлялась по результатам физического анализа системы атомов кластера, попавших в полусферу радиусом 7 Å с центром в точке контакта нанокластера и подложки.

Изучаемый процесс высокоскоростного взаимодействия кластера с подложкой характеризуется значительными градиентами скоростей, давлений и температур, поэтому интегрирование уравнения движения поводилось с использованием схемы Верле второго порядка точности по временному шагу [13]. При численном моделировании шаг по времени составлял $10^{-16}$с.

В МД алгоритмах важную роль играет механизм сортировки атомов, так как за счет него можно сократить время, требуемое для вычисления сил, действующих на каждый атом. В работе рассмотрены следующие методы сортировки атомов: списки Верле (алгоритм, требующий $N^2$ операций на каждом шаге) [14]; гибридный метод (алгоритм, требующий $N$ операций на каждом шаге) [16].

## 3. Программная реализация

### *3.1. Параллельная программа с использованием технологии MPI, основанная на одномерной параллелизации*

Важным элементом программирования для подобных систем является правильная организация обменов данными между узлами и их синхронизация. В данной работе применен метод разбиения расчетной области на подобласти с последующей динамической корректировкой их размеров. Динамическая корректировка размеров областей (динамическая балансировка) необходима для обеспечения равномерной загрузки всех вычислительных узлов.

В начальный момент времени параллелепипед разбивается на $P$ подобластей, имеющих одинаковую ширину в направлении оси $x$, и каждая такая подобласть «приписывается» к определенному процессору (под процессором понимается вычислительное ядро). Силы, действующие на атом со стороны остальных частиц, внутри такой подобласти находятся методом связанных списков. Это дает дополнительные вычислительные преимущества, поскольку позволяет использовать третий закон Ньютона при расчете сил [13].

Структуру обмена информацией между подобластями можно рассмотреть на примере 1-го процессора. При расчете сил (или потенциальной энергии межатомного взаимодействия) на каждом временном шаге информация из граничных областей 0-го и 2-го процессоров пересылается на 1-й процессор. Соответственно, информация из граничных областей 1-го процессора пересылается на 0-й и 2-й процессоры. Если атом входит в подобласть какого-то другого процессора, то вся информация о частице (набор координат и импульсов) пересылается соответствующему процессору.

В любой момент времени можно вычислить среднее число атомов, которое должно приходиться на каждый процессор при идеальной загрузке: $N_{average} = N/P$. Через определенное число шагов $N_{average}$ сравнивается с числом атомов в каждой подобласти $N_p$. Если хотя бы для одной подобласти выполняется неравенство $\left|\left(N_p - N_{average}\right)/N_{average}\right| \geq \xi$, где $\xi$ задает меру дисбаланса загрузки процессоров, запускается

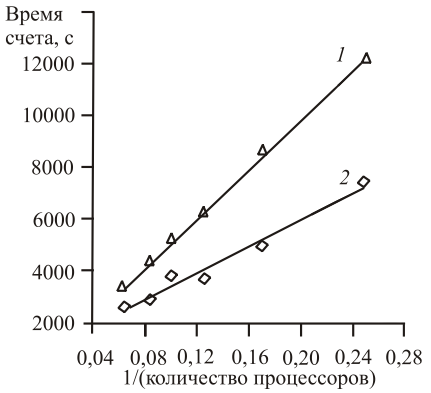


**Рис. 2.** Зависимость времени исполнения программы от параметра, обратного количеству задействованных процессоров вычислительного кластера (Т-Платформы HPC-0012111-001 в ИТПМ СО РАН) при отсутствии (*1*) и наличии (*2*) динамической балансировки; вычислительный процесс содержал 10000 шагов по времени; модель состояла из 709214 атомов.

алгоритм глобальной перестройки подобластей, суть которого состоит в следующем. Проводится анализ подобласти 0-го процессора и выясняется, какое количество атомов необходимо переслать на 1-й процессор или принять от него для выполнения условия $\left|\left(N_p - N_{average}\right)\Big/ N_{average}\right| < \xi$. В зависимости от результата, граница между подобластями, «приписанными» к 0-му и 1-му процессорам, сдвигается либо влево, либо вправо. Затем анализируется подобласть 1-го процессора (с уже измененной границей 0–1) и определяются направления и количество обменов между подобластями 1-го и 2-го процессоров и так далее. Таким образом, последовательно осуществляется глобальная перестройка всей расчетной области.

Эффективность параллельного алгоритма может быть оценена с помощью различных временных зависимостей. На рисунке 2 представлены зависимости полного времени счета программы без динамической балансировки (треугольники) и программы с динамической балансировкой подобластей (ромбики) от количества задействованных процессоров. Из рисунка видно, что обе зависимости полного времени являются линейными с хорошей степенью точности (см. усредняющие сплошные линии), но отсутствие динамической балансировки увеличивает расчетное время.

Следует отметить, что хотя одномерная параллелизация хорошо зарекомендовала себя для моделирования процессов, условно распространяющихся только в одном направлении, для проведения высокоэффективных (с точки зрения вычислительного времени) расчетов физических процессов, характеризующихся сильной пространственной неоднородностью, изменяющейся во времени, необходимо использовать двухмерную и трехмерную пространственную параллелизацию.

### *3.2. Параллельная программа для графического процессора с поддержкой CUDA*

Особенностью реализации программы на графическом процессоре (Graphics Processing Unit — GPU) является выполнение каждым вычислительным процессором одинакового набора инструкций. Современные GPU могут иметь несколько сотен вычислительных ядер и сложную структуру памяти. Особенности архитектуры и работы памяти и мультипроцессоров GPU подробно рассмотрены в [16–18]. Следует отметить, что узкое место в функционировании GPU — это небольшая скорость передачи данных из главной памяти компьютера в память GPU; таким образом, требуется минимизация процесса обмена данными между “host” памятью (оперативной памятью компьютера) и памятью GPU.

В данной работе тестировались два алгоритма решения задачи молекулярной динамики. Первый алгоритм включал сортировку атомов по методу Верле, второй — сортировку с использованием гибридного метода. В первом случае (алгоритм метода списков Верле показан в Листинге 1) все основные расчеты выполнялись на GPU; CPU задействовалось только для определения температуры и вывода данных. Во втором случае (алгоритм гибридного метода представлен в Листинге 2) сортировка атомов по ячейкам осуществлялась на CPU, после чего данные передавались на GPU, где по ним строились списки Верле и производились остальные вычисления (расчет сил и интегрирование уравнений движения). Разделение потоков на блоки по числу атомов было простое одномерное и не применялся третий закон Ньютона, требующий использования атомарных операций (atomic operations) [2, 3]. Несмотря на простоту, эта реализация обеспечивала масштабируемость кода и дала хороший, по сравнению с серийной программой, прирост скорости вычислений.

Однако главной задачей было создание максимального числа параллельных блоков в программном коде с целью минимизации числа обменов между частью программы, работающей на CPU, и GPU.

**Листинг 1.** Алгоритм программы с применением списков Верле

**Параметр:** *N* — количество моделируемых атомов;
**Параметр:** NnMx — максимально возможное количество соседних атомов;
**Параметр:** BlkSz — количество потоков в блоке;
**Параметр:** N/BlkSz+1 — количество блоков, запущенных на GPU;
**Параметр:** jarlst — массив, содержащий списки соседних атомов;
**Параметр:** xyz и Vxyz — массивы координат и скоростей;

**Параметр:** fxyz — массив модулей сил, действующих на атомы.
01. Задание начальных данных
02. call NeighKernel<<<N/BlkSz+1,BlkSz>>>(N,NnMx,xyz,jarlst,…) ! — создание списков Верле
03. call ForceKernel<<<N/BlkSz+1,BlkSz>>>(N,NnMx,xyz,jarlst,fxyz) ! — вычисление сил
04. call VelKernel<<<N/BlkSz+1,BlkSz>>>(N,xyz,fxyz,Vxyz,…) ! — интегрирование уравнений движения
05. do k=1,ntau ! — начало цикла по времени
06. call ForceKernel<<<N/BlkSz+1,BlkSz>>>(N,NnMx,xyz,jarlst,fxyz,…)
07. call VelKernel<<<N/BlkSz+1,BlkSz>>>(N,xyz,fxyz,Vxyz,…)
08. If (максимальное смещение любого из атомов больше, чем 0,5(rc-rn) then
09. call NeighKernel<<<N/BlkSz+1,BlkSz>>>(N,NnMx,xyz,jarlst,…)
10. endif
11. enddo ! — конец цикла по времени

**Листинг 2.** Алгоритм программы с применением «гибридного» метода сортировки

**Параметр:** N — количество моделируемых атомов;
**Параметр:** NnMx и NmxC — максимально возможные количества соседних атомов и атомов в ячейке;
**Параметр:** BlkSz — количество потоков в блоке;
**Параметр:** Ncells — общее количество ячеек;
**Параметр:** NcLst — массив числа атомов в каждой ячейке;
**Параметр:** N/BlkSz+1 — количество блоков, запущенных на GPU;
**Параметр:** jarlst — массив, содержащий списки соседствующих атомов;
**Параметр:** xyz и Vxyz — массивы координат и скоростей;
**Параметр:** fxyz — массив модулей сил, действующих на атомы.
01. Задание начальных данных
02. call CellList(N,NmxC,Ncells,xyz,CellList,NcLst,…)
03. call NeigKernel <<<N/BlkSz+1,BlkSz>>>(n,NnMx,NmxC,Ncells,xyz,jarlst,CellList,NcLst,…) ! — создание списков Верле
04. call ForceKernel<<<N/BlkSz+1,BlkSz>>>(N,NnMx,xyz,jarlst,fxyz,…) ! — вычисление сил
05. call VelKernel<<<N/BlkSz+1,BlkSz>>>(N,xyz,fxyz,Vxyz,…) ! — интегрирование уравнений движения
06. do k=1,ntau ! — начало цикла по времени
07. call ForceKernel<<<N/BlkSz+1,BlkSz>>>(N,NnMx,xyz,jarlst,fxyz,…)
08. call VelKernel<<<N/BlkSz+1,BlkSz>>>(N,xyz,fxyz,Vxyz,…)
09. If (максимальное смещение любого из атомов больше, чем 0,5(rc-rn) then
10. call CellList(N,NmxC,Nxc,Nyc,Nzc,xyz,CellList,NcLst,dxCell,dyCell,dzCell)
11. call NeigKernel<<<N/BlkSz+1,BlkSz>>>(n,NnMx,NmxC,Ncells,xyz,jarlst,CellList,NcLst,…)
12. endif
13. enddo ! — конец цикла по времени

## 4. Описание вычислительных систем

Первыми двумя системами, выбранными для тестирования GPU, были настольные компьютеры с графическими картами TESLA C1060 и GeForce GTX460. Первая видеокарта разрабатывалась для осуществления вычислений и для обработки потокового видео, в то время как вторая видеокарта — игровая. Она имеет меньший объем памяти, но большее количество вычислительных ядер, по сравнению с Tesla C1060. Остальные параметры компьютера следующие: CPU i7-950; материнская плата — P6X58D ASUS с 6Гб ОЗУ.

Для сравнительного тестирования MPI программ также использовались компьютерные кластеры:
– 1-й вычислительный кластер HP BladeSystem c7000, 512 ядер Intel Xeon "Clovertown" 5355 @ 2.6 ГГц, 768 ядер Intel Xeon "Nehalem" 5540 @ 2.5 ГГц, InfiniBand; расположен в Новосибирском государственном университете;
– 2-й вычислительный кластер T-Edge-32 (HPC-0012111-001: 32 узла (2xXeon 5420 2.5 ГГц 16Гб ОЗУ на каждом узле); местоположение — ИТПМ СО РАН.

## 5. Результаты

### *5.1. Тест эффективности вычислений*

В таблице 1 приведена относительная производительность (отношение времени исполнения программы на Tesla C1060 ко времени исполнения на GTX460) в зависимости от размера блока и количества атомов в системе. Результаты свидетельствуют, что МД алгоритм с сортировкой Верле работает быстрее на игровой видеокарте (GTX460), в то время как «гибридный» алгоритм — на Tesla C1060. Данные результаты могут быть объяснены, если принять во внимание в два раза большую ширину интерфейса памяти у C1060

в сравнении с GTX460, то есть C1060 кэширует данные быстрее, чем GTX460. Также с ростом числа необходимых операций возрастает роль количества процессоров. Таким образом, GTX460 «обгоняет» C1060.

Следующим интересным моментом является то, что сортировка по методу Верле требует максимально возможного размера блока, в то время как «гибридный» алгоритм выполняется быстрее в случае, когда блок содержит 64 потока. В таблице 2 показано ускорение программы, которое достигается реализацией алгоритма на графической карте с поддержкой CUDA, по сравнению с серийной программой, запущенной на CPU; очевидно, что чем большее количество атомов моделируется, тем большее ускорение достигается.

Таблица 1. Относительная производительность GPU в зависимости от количества потоков в блоке и числа моделируемых атомов

| Размер блока | 512 | | 256 | | 128 | | 64 | | 32 | | 16 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Сорти-ровка / Число атомов | Верле | Гибрид | Верле | Гибрид | Верле | Гибрид | Верле | Гибрид | Верле | Гибрид | Верле | Гибрид |
| 14896 | 0,936 | 0,863 | 0,929 | 0,861 | 0,941 | 0,873 | 0,992 | 0,894 | 0,744 | 0,741 | 0,596 | 0,573 |
| 34461 | 1,079 | 0,869 | 1,022 | 0,824 | 1,021 | 0,839 | 1,111 | 0,924 | 0,798 | 0,745 | 0,581 | 0,546 |
| 66326 | 1,217 | 0,890 | 1,169 | 0,858 | 1,199 | 0,885 | 1,290 | 0,971 | 0,902 | 0,765 | 0,654 | 0,564 |
| 178956 | 1,395 | 0,929 | 1,368 | 0,896 | 1,344 | 0,917 | 1,226 | 0,991 | 0,841 | 0,792 | 0,702 | 0,593 |
| 265721 | 1,452 | 0,900 | 1,460 | 0,893 | 1,439 | 0,913 | 1,413 | 0,953 | 0,960 | 0,737 | 0,860 | 0,546 |
| 376786 | 1,501 | 0,884 | 1,437 | 0,883 | 1,419 | 0,901 | 1,372 | 0,925 | 0,923 | 0,729 | 0,918 | 0,543 |
| 515151 | 1,681 | 1,272 | 1,569 | 1,185 | 1,544 | 1,212 | 1,325 | 1,331 | 0,954 | 1,104 | 0,761 | 0,811 |
| 976563 | 1,699 | 0,884 | 1,651 | 0,877 | 1,656 | 0,885 | 1,562 | 0,909 | 1,104 | 0,751 | 0,897 | 0,565 |

Таблица 2. Ускорение программы, модифицированной для GPU, в сравнении с серийной версией, выполняемой на одном ядре процессора Intel i7-950; блок содержит 64 потока, метод сортировки гибридный

| Число атомов / Карта GPU | 14896 | 34461 | 66326 | 178956 | 265721 | 376786 | 515151 | 976563 |
|---|---|---|---|---|---|---|---|---|
| Tesla | 26,6 | 28,1 | 29,2 | 28,2 | 33,2 | 33,9 | 22,6 | 34,8 |
| GTX460 | 23,9 | 25,7 | 27,7 | 27,4 | 30,1 | 30,0 | 30,1 | 30,5 |

Во второй части данной работы проведено исследование столкновения кластера меди с загрязненной медной поверхностью. Межатомные взаимодействия описывались многочастичными потенциальными функциями, что потребовало намного бóльших вычислительных ресурсов, чем для алгоритмов, основанных на парных потенциалах. MPI программа выполнялась на 10 процессорах.

Таблица 3 содержит времена решения вышеозначенной задачи на различных системах. К сожалению, GTX460 имеет только 1 Гб памяти, что не позволяет произвести на ней моделирование больших систем с многочастичными потенциалами. Следует отметить, что в случае относительно небольшой системы GTX460 работает быстрее, чем C1060 и 1-й компьютерный кластер. Видеокарта C1060 работает быстрее, чем оба компьютерных кластера, в случае моделирования большой системы. В какой-то мере низкая скорость работы MPI программы обусловлена одномерной параллелизацией, однако использование многомерной параллелизации существенно усложнит программу и увеличит временные затраты на отладку. Главной проблемой реализации многомерной параллелизации является организация динамической балансировки [20, 21].

Таблица 3. Время решения задачи падения медного кластера на подложку (в секундах) с использованием различных вычислительных систем; процесс расчета содержал 20000 шагов по времени

| Количество атомов | 122092 | | | | 238466 | | | |
|---|---|---|---|---|---|---|---|---|
| Вычисли-тельная систе-ма / Размер блока | GTX460 | C1060 | 1-й кластер | 2-й кластер | GTX460 | C1060 | 1-й кластер | 2-й кластер |
| 16 | 20643,17 | 20643,17 | 16540,61 | 8109,10 | Данных нет | 40505,58 | 89832,06 | 43438,04 |
| 32 | 14502,03 | 14502,03 | | | | 34538,99 | | |
| 64 | 11528,20 | 11528,20 | | | | 27726,10 | | |
| 128 | 12567,84 | 12567,84 | | | | 29375,18 | | |
| 256 | 14160,30 | 14160,30 | | | | 32369,74 | | |
| 512 | 16396,22 | 16396,22 | | | | Данных нет | | |

### *5.2. Результаты моделирования*

На рисунке 3 показаны изменения компонент скорости центра масс кластера меди во времени (диаметр кластера меди $d_{clus} = 50$ Å, начальная скорость $V_{inc} = 170$м/с и угол падения $\alpha_{inc} = 90°$ — нормальное падение кластера на подложку). Видно, что образуется связанное состояние между кластером и материалом подложки, наличие которого подтверждается зависимостями скорости центра масс кластера от времени. Закрепившийся кластер совершает колебательные движения на поверхности подложки, которые затухают со временем. В момент удара о подложку наблюдается резкий рост как полной температуры кластера, так и температуры в области контакта за время порядка 2 пс. (см. Рис. 4). Следует отметить, что, несмотря на образование связанного состояния, значения полной температуры кластера и пиковые значения температуры в области контакта не достигают порогового значения температуры плавления меди.

Исследования влияния угла падения в широком диапазоне значений позволили обнаружить критический угол между вектором скорости и поверхностью, меньше которого осаждение кластеров не происходит. На рисунке 5 представлены результаты численного моделирования при начальной скорости кластера 170 м/с и угле падения 30°.

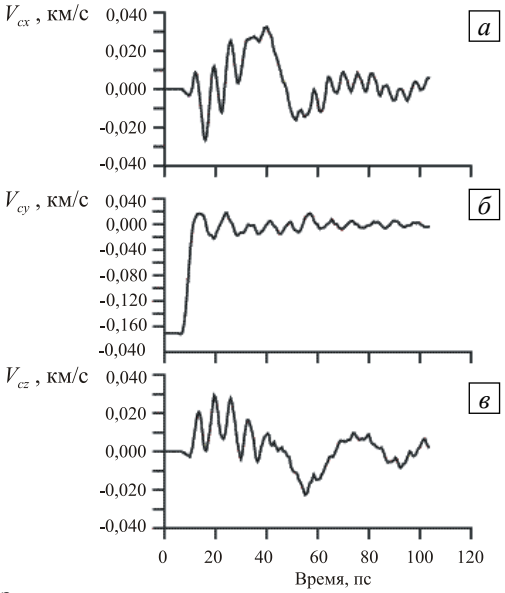


**Рис. 3.** Зависимости компонент скорости центра масс медного кластера $V_{cx}$ (*а*), $V_{cy}$ (*б*), $V_{cz}$ (*в*) от времени при угле падения $\alpha_{inc} = 90°$

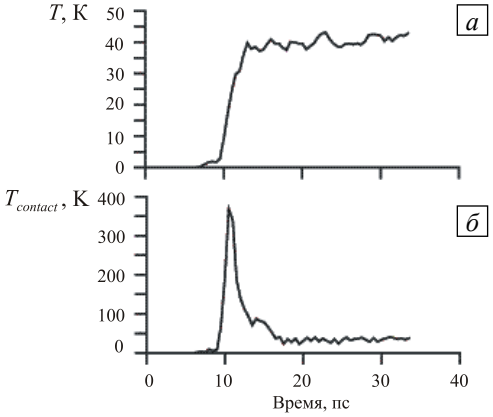


**Рис. 4.** Зависимости от времени полной температуры медного кластера (*а*) и температуры в точке касания (*б*)

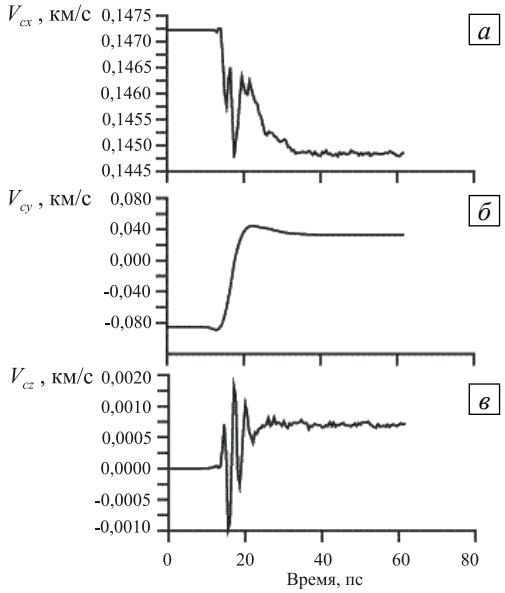


**Рис. 5.** Зависимости компонент скорости центра масс медного кластера $V_{cx}$ (*а*), $V_{cy}$ (*б*), $V_{cz}$ (*в*) от времени при угле падения $\alpha_{inc} = 30°$

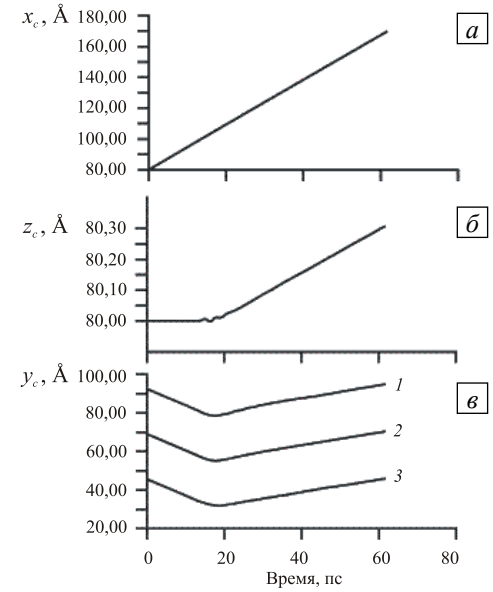


**Рис. 6.** Зависимость координат центра масс медного кластера от времени при угле падения $\alpha_{inc} = 30°$ – *а, б, в* (линия *2*); на фрагменте (*в*) линии *1* и *3* – максимальная и минимальная координаты атомов медного кластера на оси *y* (вертикальная ось)

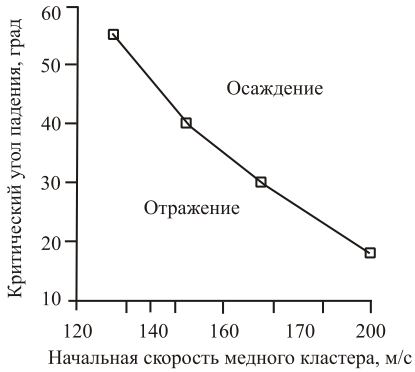


**Рис. 7.** Зависимость критического угла падения от начальной скорости медного кластера (диаметр кластера $d_{clus} = 50$ Å)

Зависимости координат центра масс кластера, приведенные на рисунке 6, показывают, что после удара о подложку кластер отражается от нее. Из анализа зависимостей максимального и минимального положения атомов кластера в пространстве, а также изменения координат его центра масс от времени следует, что при ударе кластера не происходит ни его разрушение, ни закрепление атомов кластера на подложке.

На рисунке 7 изображена связь между критическим углом падения и скоростью медного кластера; очевидно, что увеличение скорости падающего кластера приводит к уменьшению критического угла. Осаждение кластера на подложку не происходит, если начальная скорость ниже 123 м/с вне зависимости от угла падения.

## 6. Заключение

Представленные результаты позволяют заключить, что игровые графические видеокарты обладают потенциальной возможностью стать основой домашнего или настольного суперкомпьютера. Данные видеокарты имеют малую стоимость (по сравнению с профессиональными), однако их главным недостатком является относительно небольшой объем оперативной памяти. Исследования показали также, что оптимальный выбор видеокарты зависит от алгоритма, который реализуется на устройстве.

Проведенная оптимизация программного кода с применением технологии CUDA дала возможность создать эффективную вычислительную модель и рассмотреть осаждение медного кластера на загрязненную медную поверхность в широком диапазоне скоростей и углов столкновения. Была обнаружена зависимость между критическими значениями угла и скоростью падения кластера. Осаждение кластера на подложку не происходит, если начальные значения угла или скорости кластера меньше критических, однако при скорости падения ниже 123 м/с кластер отражается от поверхности вне зависимости от угла падения.

Полученные результаты имеют большое значение для решения задачи высокоскоростного взаимодействия кластеров с подложкой на микроуровне, так как они содержат оптимальные параметры для устойчивого закрепления кластера.

## Литература

1. *Rost R.J.* OpenGL shading language. – Addison Wesley, 2006. – 800 p.
2. *Kilgard M.J.* The Cg tutorial: The definitive guide to programmable real-time graphics. – Addison-Wesley Professional, 2003. – 384 p.
3. NVIDIA CUDA C Programming Guide Version 3.2. – NVIDIA Corporation: Santa Clara, 2010 – 170 p. http://www.serc.iisc.ernet.in/~vss/courses/PPP/CUDA_C_Programming_Guide.pdf (дата обращения 02.07.2012)
4. The Portland Group, T.P. PGI CUDA Fortran Compiler. – http://www.pgroup.com/resources/cudafortran.htm (дата обращения: 04.09.12).
5. *Yang J., Wang Y., Chen Y.* GPU accelerated molecular dynamics simulation of thermal conductivities // J. Comput. Phys. – 2007. – V. 221, N. 2. – P. 799-804. DOI
6. *Ufimtsev I.S., Martinez T.J.* Quantum chemistry on graphical processing units. 1. Strategies for two-electron integral evaluation // J. Chem. Theory Comput. – 2008. – V. 4, N. 2. – P. 222-231. DOI
7. *Friedrichs M.S., Eastman P., Vaidyanathan V., Houston M., Legrand S., Beberg A.L., Ensign D.L., Bruns C.M., Pande V.S.* Accelerating molecular dynamic simulation on graphics processing units // J. Comput. Chem. – 2009. – V. 30, N. 6. – P. 864-872. DOI
8. *Maruyama S.* Molecular Dynamics Method for Microscale Heat Transfer // Advances in Numerical Heat Transfer / Ed. W.J. Minkowycz and E.M. Sparrow. – Taylor & Francis, 2000. – V. 2. – P. 189-226.
9. *Golovnev I.F., Golovneva E.I., Fomin V.M.* Simulation of quasi-static processes in the crystals by molecular dynamics method // Phys. Mesomech. – 2003. – V. 6, N. 5-6. – P. 41-45.
10. *Voter A.F.* Embedded Atom Method Potentials for Seven FCC Metals: Ni, Pd, Pt, Cu, Ag, Au, and Al: Los Alamos Unclassified Technical Report / Los Alamos National Laboratory. – Los Alamos, 1993. – 9 p. – N. LA-UR 93-3901.
11. *Китайгородский А.И.* Молекулярные кристаллы. – М.: Наука, 1971. – 424 с.
12. *Semyannikov P.P., Basova T.V., Trubin S.V., Kol'tsov E.K., Plyashkevich V.A., Igumenov I.K.* Vapor pressure of some metal phthalocyanines // Russ. J. Phys. Chem. A. – 2008. – V. 82, N. 2. – P. 159-163. DOI
13. *Allen M.P., Tildesley D.J.* Computer simulation of liquids. – Oxford Science Publications, 2000. – 385 p.
14. *Verlet L.* Computer "experiments" on classical fluids. I. Thermodynamical properties of Lennard-Jones molecules // Phys. Rev. – 1967. – V. 159, N. 1. – P. 98-103. DOI
15. *Hockney R.W., Eastwood J.W.* Computer simulation using particles. – McGraw-Hill Inc., New-York, 1981. – 540 p.

16. *Auerbach D.J., Paul W., Bakker A.F., Lutz C., Rudge W.E., Abraham F.F.* A special purpose parallel computer for molecular dynamics: motivation, design, implementation, and application // J. Phys. Chem. – 1987. – V. 91, N. 19. – P. 4881-4890. DOI
17. *Stone J.E., Hardy D.J., Ufimtsev I.S., Schulten K.* GPU-accelerated molecular modeling coming of age // J. Mol. Graph. Model. – 2010. – V. 29, N. 2. – P. 116-125. DOI
18. *Anderson J.A., Lorenz C.D., Travesset A.* General purpose molecular dynamics simulations fully implemented on graphics processing units // J. Comput. Phys. – 2008. – V. 227, N. 10, – P. 5342-5359. DOI
19. *Stone J.E., Phillips J.C., Freddolino P.L., Hardy D.J., Trabuco L.G., Schulten K.* Accelerating molecular modeling applications with graphics processors // J. Comput. Chem. – 2007. – V. 28, N. 16. – P. 2618-2640. DOI
20. *Nakano A.* Multiresolution load balancing in curved space: the wavelet representation // Concurrency: Practice and Experience. -1999. – V. 11, N. 7. – P. 343-353. DOI
21. *Deng Y., Peierls R.F., Rivera C.* An adaptive load balancing method for parallel molecular dynamics simulations // J. Comput. Phys. – 2000. – V. 161, N. 1. – P. 250-263. DOI
22. *Вахрушев А.В., Федотов А.Ю.* Исследование вероятностных законов распределения структурных характеристик наночастиц, моделируемых методом молекулярной динамики // Вычисл. мех. сплош. сред. – 2009. – Т. 2, № 2. – С. 14-21. DOI

*Поступила в редакцию 18.01.12; опубликована в электронном виде 22.10.12*

*Сведения об авторах*

*Ozhgibesov Mikhail*, PhD student, National Cheng Kung University (NCKU), Tainan 701, Taiwan R.O.C, No. 1, University Road; E-mail: 48987011@mail.ncku.edu.tw

*Уткин Андрей Вячеславович*, кфмн, Институт теоретической и прикладной механики им. С.А. Христиановича СО РАН (ИТПМ СО РАН), 630090, Новосибирск, ул. Институтская, д. 4/1; E-mail: utkin@itam.nsc.ru

*Фомин Василий Михайлович*, акад., дир., ИТПМ СО РАН; E-mail: admin@itam.nsc.ru

*Tzong-Shyng Leu*, assoc. prof., NCKU; E-mail: tsleu@mail.ncku.edu.tw

*Chin-Hsiang Cheng*, prof., NCKU; E-mail: chcheng@mail.ncku.edu.tw