УДК 519.6
DOI: 10.53403/9785951505460_59

АДАПТАЦИЯ АЛГОРИТМОВ МОЛЕКУЛЯРНОЙ ДИНАМИКИ МЕТОДИЧЕСКОГО ПРИКЛАДНОГО ТЕСТА MD ДЛЯ ГРАФИЧЕСКИХ УСКОРИТЕЛЕЙ

Игнаткова Мария Геннадиевна (*mgignatkova@vniief.ru*),
Ерофеев Алексей Михайлович, *Ветчинников Максим Владимирович*

ФГУП «РФЯЦ-ВНИИЭФ», г. Саров Нижегородской обл.

В работе представлены новые возможности прикладного методического теста MD, адаптированного к счету на графических ускорителях (GPU): перенос всех счетных алгоритмов на GPU для потенциалов Морзе, EAM и MEAM, новый метод расчета сил – метод списка ячеек, оптимизация работы с памятью. Работа содержит результаты численных исследований производительности и масштабируемости теста MD на сервере с двумя GPU. В результате тестирования реализованных алгоритмов получено среднее ускорение для парного потенциала Морзе – в 10 раз, для EAM-потенциала – в 3 раза, для MEAM – в 1,5 раза по сравнению со старой версией на GPU.

Ключевые слова: молекулярная динамика, графические ускорители, алгоритмы, CUDA.

ADAPTATION OF MOLECULAR DYNAMICS ALGORITHMS OF METHODOLOGICAL APPLIED MD TEST FOR GRAPHICAL PROCESSOR UNITS

Ignatkova Mariya Gennadievna (*mgignatkova@vniief.ru*),
Erofeev Aleksey Mikhailovich, *Vetchinnikov Maksim Vladimirovich*

FSUE «RFNC-VNIIEF», Sarov Nizhny Novgorod region

The paper presents new possibilities of applied methodological MD test adapted to the computation on Graphical Processor Units (GPU): transfer of all calculating algorithms to GPU for Morse, EAM, MEAM potentials, a new method of forces computing – cell-list method, optimization of the work with memory. The paper contains the results of numerical research on the efficiency and scalability of MD test on the server with two GPUs. As a result of algorithm testing average acceleration was obtained; it was 10 times for Morse potential, 3 times for EAM potential, and 1,5 times for MEAM potential as compared to the old version on GPU.

Key words: molecular dynamics, graphical processor units, algorithms, CUDA.

Введение

Впервые метод молекулярной динамики был применен в середине 20 века [1]. Сразу стало очевидно, что без помощи мощных вычислительных машин моделировать ансамбли микрочастиц (атомов и молекул) для получения полезной информации с помощью метода невозможно. Поэтому развитие молекулярной динамики неразрывно связано с развитием вычислительной базы.

В 2008 году был создан программный комплекс MoDyS [2], реализующий метод классической молекулярной динамики больших ансамблей частиц. Кроме того, была сделана урезанная тестовая версия программы – тест MD [3], которая в дальнейшем стала полигоном для исследования эволюции кода. Все наработки теста MD со временем вносятся в комплекс программ MoDyS. Тест MD обладает такими особенностями, как хорошая масштабируемость, большое количество независимых арифметических вычислений и относительно небольшое количество обменов и объем передаваемых при этом данных. Это позволяет ему высокоэффективно использовать графические ускорители. Перенос вычислений на GPU осуществляется с помощью программно-аппаратной архитектуры параллельных вычислений CUDA [4].

Описание алгоритма расчета сил в тесте MD

Метод молекулярной динамики основан на численном интегрировании уравнений движения Ньютона для систем частиц с заданным законом межчастичного взаимодействия и используется для задач исследования материалов методом молекулярной динамики:

$$\frac{dr_i}{dt} = v_i \ , \qquad m\frac{dv_i}{dt} = -\sum_{j\neq i} \nabla_r u(r_{ij}), \tag{1}$$

$$F_{ij} = -\nabla_r u(r_{ij}), \tag{2}$$

где r_i – координаты частицы; v_i – скорость частицы; m – масса частицы; $\nabla_r u(r_{ij})$ – градиент межчастичного потенциала взаимодействия; F_{ij} – сила, действующая на частицу с номером i со стороны частицы с номером j; t – временной интервал.

Законы взаимодействия пары частиц или многочастичные взаимодействия определены в виде потенциалов взаимодействия, по которым находятся компоненты сил для каждой конкретной частицы. По полученной силе находим скорости частиц и смещение в пространстве за небольшой временной промежуток – шаг по времени dt.

Подход, реализованный в программе для организации нахождения взаимодействующих частиц, представляет собой сеточный метод, при котором область моделируемого пространства разбивается на параллелепипеды со стороной равной или чуть большей радиусу обрезания – эффективному радиусу взаимодействия, дальше которого потенциал принимает значения 0. Потенциал взаимодействия ищется между частицами, находящимися в своей ячейке и в двадцати шести соседних ячейках.

Частицы в ячейках располагаются в виде связных списков. У ячейки есть адрес первой частицы, их количество в ячейке и адрес последней частицы. В списке каждая частица хранит указатель на следующую за ней, а самая последняя частица ссылается на пустой адрес, на null. При перелете частиц из ячейки в ячейку физически частица не меняет своего местоположения в памяти (если только она не перелетает с процессора на процессор), меняется информация в связных списках.

Вычисления на каждом счетном шаге делятся на следующие этапы: расчет сил, расчет скоростей и координат, отслеживание перелетов частиц между ячейками и обновление связных списков, MPI – обмены.

На ячейки легко ложится OpenMP распараллеливание. Оно уменьшает число MPI процессов, что важно на крупных задачах с большим числом процессоров. В совокупности с применением динамической балансировки в производственных расчетах удается получать эффективность распараллеливания выше 75 %.

Адаптация тестовой программы молекулярной динамики для GPU

Адаптация программы молекулярной динамики MoDyS для графических ускорителей впервые была сделана в 2008–2009 годах [5]. Было определено, что самый трудоемкий участок кода – модуль расчета сил – занимает в среднем от 80 до 95 % от общего времени шага и зависит от типа потенциала взаимодействия, атомной структуры и радиуса обрезания взаимодействия. На тот момент был осуществлен перенос только этой части программы на GPU. Недостаток такого подхода очевиден: необходимость пересылки всей информации о частицах в одну сторону и всех сил обратно на каждом счетном шаге.

Проведено тестирование существующих алгоритмов: был выбран образец меди с ГЦК-решеткой, разогретый до температуры T_0 = 300 К, характерный шаг атомной решетки – 3,615 Å; радиус обрезания – 7,23 Å равен двум характерным шагам. Число частиц – 4 млн. Потенциал взаимодействия – парный потенциал Морзе. Тестирование здесь и далее проводилось на узлах с двумя графическими ускорителями NVIDIA Tesla V100 в узле с двумя CPU Intel Xeon Gold 6154, 3.00GHz. Результаты представлены в табл. 1.

Таблица 1

Профилировка счетного шага при расчете на GPU для потенциала Морзе

	Копирование информации на GPU	Расчет сил (на GPU)	Копирование информации на CPU	Расчет уравнений движения и обновление связных списков (на CPU)	Время счетного шага
Старый код на GPU Версия 1.0	0,12	0,05	0,07	0,13	0,37

Из табл. 1 видно, что обмены занимают 50 % от шага. Одним из путей выхода из данной ситуации становится полный перенос всех вычислений на GPU. В следствие этого исчезает необходимость перекачивания между устройствами большого объема информации о частицах. Обмен между GPU и CPU необходим только для организации пересылок граничной информации с другими процессорами, что по-прежнему идет на CPU с помощью MPI, а это значительно меньше, чем пересылки в первой версии кода. Упрощенная схема расчета шагов показана на рис. 1.

Рис. 1. Схема алгоритма версии 2.0

Расчет скоростей и координат полностью переведен на GPU. Распараллеливание достигается запуском на числе потоков GPU, равном числу частиц на процессоре. Отслеживание перелетов частиц между ячейками и обновление связных списков осуществляется запуском потоков GPU, число которых равно числу ячеек.

Результаты работы по переносу всех этапов вычислений на GPU для потенциала Морзе были взяты за основу для адаптации более сложных потенциалов межчастичного взаимодействия – EAM и MEAM, о чем будет рассказано ниже.

Тестирование нового алгоритма на GPU для потенциала Морзе

Для тестирования эффективности реализованных алгоритмов брались образцы меди различных размеров: от 48 тыс. до 210 млн частиц. Проведено сравнение реализованного алгоритма со старым алгоритмом. Количество счетных шагов в каждом расчете равно 100. Результаты сравнительных расчетов на одном и двух GPU приведены в табл. 2.

Как видно из табл. 2, перенос всех этапов вычислений на GPU позволил получить для потенциала Морзе ускорение в среднем в 5 раз на разных размерах задач. В ходе дальнейших исследований было замечено, что первый шаг в расчете на CPU проходит быстрее следующих. После экспериментов установлено, что причиной этого является запутанность связных списков. После перелета частиц из ячейки в ячейку может оказаться так, что лежащие друг за другом в списке частицы в физической памяти устройства находятся далеко друг от друга. На рис. 2 показано, как запутаны списки.

Была сделана версия программы с переупорядочиванием частиц так, чтобы они в памяти физически лежали последовательно, как показано на рис. 3.

В результате практически везде в коде были убраны циклы с проходами по спискам. Данную модификацию программы обозначим – версия 2.1. Результаты тестирования на одном и двух GPU версии 2.1 в сравнении с версией 2.0 представлены в табл. 3. Как видно из табл. 3, время счета уменьшилось в среднем в 1,5 раза.

Таблица 2

Время работы 100 шагов программы при расчете на одном/двух GPU

Размер, Å (число частиц)	48 (442,368 тыс.)	86 (2,544 млн)	100 (4 млн)	150 (13,5 млн)	172 (20,353 млн)	200 (32 млн)	210 (37,044 млн)
Версия 1.0 GPU	4,281/2,62	28,87/13,09	33,87/29,55	110,69/61,29	162,40/90,37	252,02/138,19	289,9/159,97
Версия 2.0 GPU	0,635/0,51	3,87/1,99	6,76/3,25	20,75/11,4	34,14/18,34	47,84/29,47	61,49/32,78
Ускорение счета T_1.0/T_2.0	6,8/5,13	7,5/6,6	5,0/9,1	5,3/5,4	4,8/4,9	4,8/4,7	4,7/4,9

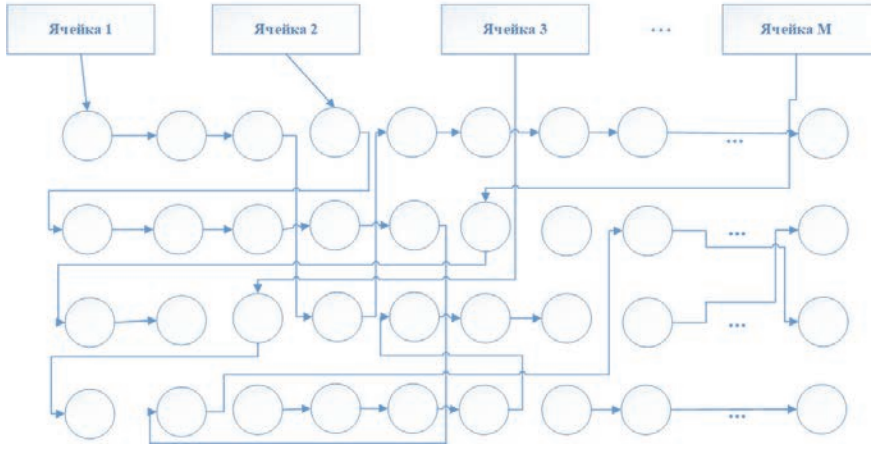

Рис. 2. Запутанность списков частиц в ячейках

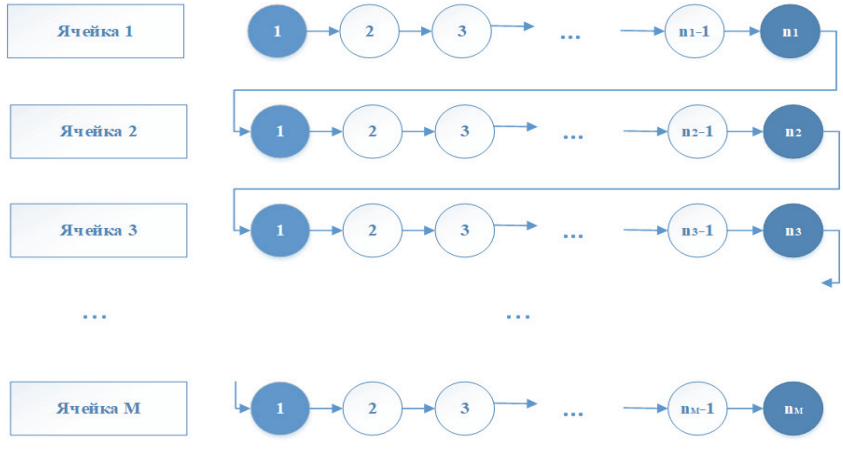

Рис. 3. Непрерывный список ячеек

Таблица 3

Время работы 100 шагов программы при расчете на одном/двух GPU

Размер, Å (число частиц)	48 (442,368 тыс.)	86 (2,544 млн)	100 (4 млн)	150 (13,5 млн)	172 (20,353 млн)	200 (32 млн)	210 (37,044 млн)
Версия 2.0	0,635/0,51	3,87/1,99	6,76/3,25	20,75/11,4	34,14/18,34	47,84/29,47	61,49/32,78
Версия 2.1	0,449/0,398	2,313/1,586	3,71/2,366	12,21/7,67	18,38/11,67	28,21/18,49	33,23/21,31
Ускорение счета T_2.0/T_2.1	1,41/1,28	1,67/1,25	1,82/1,37	1,69/1,48	1,85/1,57	1,70/1,59	1,85/1,53

Описание и тестирование метода списка ячеек

С целью поиска более эффективных методов расчета сил на GPU рассмотрена статья о новом подходе – методе списка ячеек [6]. Авторы статьи утверждают, что в данном методе вычислительная эффективность увеличивается благодаря уменьшению обращений к памяти и объединенному доступу к памяти.

Метод списков ячеек был реализован в тесте MD. Особенностью нового метода является введение двухэтапной схемы расположения атомов и динамическое выделение памяти для ячеек, которые содержат частицы. Динамическая двухэтапная схема расположения частиц подразумевает создание на каждом шаге «списка всех ячеек» и списка ячеек, содержащих атомы («список атомных ячеек»).

На первом этапе на GPU заполняется «список всех ячеек», который содержит следующую информацию: адрес данной ячейки (то есть ее номер в «списке атомных ячеек»), количество частиц в ячейке, количество соседних ячеек и номера соседних ячеек. Длина этого списка равна полному числу ячеек в рассматриваемом образце. Число и номера соседних ячеек инициализируются до начала моделирования. На этом же шаге заполняется список, который приводит в соответствие номера ячеек в «списке атомных ячеек» и в «списке всех ячеек. Длина этого списка равна числу ячеек, содержащих атомы.

На втором этапе на GPU заполняется «список атомных ячеек», содержащий номер частицы и ее координаты. Длина этого списка равна числу ячеек, содержащих атомы. Этот список обновляется на каждом шаге.

Для списков координат и сил используется встроенный тип данных CUDA – double4. При этом 4-й параметр не используется. Использование double4 вместо double3 необходимо, чтобы выровнять обращение к памяти и использовать объединенные запросы при работе с глобальной памятью.

Расчет сил ведется с использованием переменных, расположенных в разделяемой памяти (shared memory) – быстрой памяти, из которой можно считывать и в которую можно записывать данные внутри GPU. В разделяемую память из глобальной памяти записываются основная информация (данные «списка всех ячеек»), как о рассматриваемой ячейке, так и о ее соседях, номера и координаты всех частиц рассматриваемой ячейки и соседних ячеек (данные «списка атомных ячеек»).

Сначала для частиц в рассматриваемой ячейке вычисляются силы взаимодействия с частицами внутри ячейки. Потом вычисляются силы взаимодействия с частицами, расположенными в соседних ячейках.

В результате тестирования на одном GPU была сделана оценка среднего времени, потраченного на расчет сил по новому методу списков ячеек и по старому алгоритму. Данные представлены в табл. 4.

Данные в табл. 4 показывают, что ускорение нового метода расчета сил относительно старого составляет в среднем 1,3 раза.

В конечном итоге, для потенциала Морзе получено среднее ускорение в 10 раз по сравнению с первоначальной версией на GPU.

Таблица 4

Время расчета сил на GPU для метода списка ячеек и метода связных списков, потенциал взаимодействия – Морзе

Размер, Å (число частиц)	48 (442,368 тыс.)	86 (2,544 млн)	100 (4 млн)	150 (13,5 млн)	172 (20,353 млн)	200 (32 млн)	210 (37,044 млн)
Старый код GPU Версия 1.0	0,00491	0,036	0,065	0,195	0,326	0,443	0,582
Новый код GPU	0,00613	0,030	0,045	0,149	0,225	0,353	0,409
Ускорение счета T_gpu_old/ T_gpu_new	0,80	1,22	1,43	1,31	1,45	1,26	1,42

Описание алгоритма расчета сил для EAM-потенциалов

В EAM и MEAM - подходах [7] – [9] энергия частицы *i* в металле E_i состоит из двух компонент:

$$E_i = \Phi_i(\rho_i) + \frac{1}{2}\sum_{j \neq i} \varphi(r_{ij}), \quad (3)$$

где Ф – энергия погружения в металлическую структуру, зависящая от локальной плотности электронов ρ_i в месте нахождения частицы *i*:

$$\rho_i = \sum_{j \neq i} f_j(r_{ij}), \quad (4)$$

где $\varphi(r_{ij})$ – парная отталкивающая ветвь потенциала.

Суммирование происходит по всем частицам системы и обе функции (φ_{ij}, f_{ij}) являются функциями только межчастичных расстояний.

Полная энергия системы из *N* частиц E_{tot} получается суммированием индивидуальных энергий частиц E_i и выглядит так:

$$E_{tot} = \sum_i^N E_i. \quad (5)$$

В первоначальном варианте программы расчета сил на GPU [5] для EAM потенциала схема счетного шага выглядела следующим образом: в начале шага с CPU на GPU копируется информация с координатами и типами частиц. Затем на GPU осуществляется расчет потенциалов и по ним – сил взаимодействия, при этом число нитей на GPU равняется количеству частиц на процессоре. После этого компоненты сил копируются обратно на CPU. Следующие действия на шаге – это расчет на CPU новых скоростей и координат частиц и обновление связных списков.

После проведения оценки времени всех этапов расчета было замечено, что копирование данных на ускоритель в среднем занимает 28 % времени шага, а копирование полученных значений сил с ускорителя на процессор – 17 % времени шага. Точное время указано в табл. 5.

Таблица 5

Профилировка счетного шага при расчете на GPU для EAM-потенциала

	Копирование информации на GPU	Расчет сил и электронных плотностей	Копирование информации на CPU	Расчет уравнений движения и обновление связных списков (на CPU)	Время счетного шага
Старый код на GPU Версия 1.0	0,118	0,1517	0,0728	0,08	0,423

Как и для потенциала Морзе, было решено перенести все этапы вычислений для EAM потенциала на GPU. Для расчета координат и скоростей используются алгоритмы, полученные ранее для потенциала Морзе. Для EAM потенциала дополнительно потребовалось перенести на GPU расчет электронных плотностей для всех частиц, включая фиктивные.

Результаты тестирования программы на GPU для EAM-потенциала

Моделировался образец меди с ГЦК-решеткой при T_0=300 К. Начальный шаг кристаллической решетки – 3,615 Å. Масса частицы – 63,546 а.е.м. Рассматривались образцы меди с ГЦК-решеткой различных размеров – от 55 тыс. частиц до 37 млн частиц. Радиус взаимодействия – 2 a0. В качестве граничных условий использовались периодические граничные условия. Число шагов – 100.

В результате тестирования на одном и двух GPU была сделана оценка среднего времени, потраченного на каждый шаг, при расчете по старому алгоритму и новому варианту. Результаты приведены в табл. 6.

В соответствии с данными в табл. 6 получаем, что коэффициент ускорения нового алгоритма по отношению к старому алгоритму при расчете на одном GPU для EAM потенциала в среднем равен 3,2, при расчете на двух GPU – 2,4.

Таблица 6

Время 100 шагов при расчете на одном/двух GPU, потенциал взаимодействия – EAM

Размер, a0 (число частиц) Время, с	24 (55,296 тыс.)	48 (442,368 тыс.)	86 (2,544 млн)	100 (4 млн)	150 (13,5 млн)	172 (20,353 млн)	200 (32 млн)	210 (37,044 млн)
Старый алгоритм (1)	0,939/0,858	6,072/3,659	34,3/19,7	55,04/30,1	183,3/99,432	283,73/153,44	440,46/239,92	523,61/277,5
Новый алгоритм (2)	0,368/0,514	1,812/1,584	10,459/7,615	17,686/11,43	55,847/39,963	88,509/60,794	130,382/94,78	159,959/107,52
Алгоритм (1)/ Алгоритм (2)	2,55/1,67	3,35/2,3	3,27/2,6	3,11/2,63	3,28/2,49	3,21/2,52	3,38/2,53	3,27/2,58

Описание алгоритма расчета сил для MEAM-потенциалов

Для расчета сил при использовании MEAM потенциала используется «сеточно-списочный» алгоритм. Списки соседей строятся на основе ограниченности зоны влияния частиц друг на друга. Для каждой частицы строится массив соседей, окружающих ее и попавших в ее область влияния. Для этого перед началом работы алгоритма расчета сил или энергии для каждой частицы в процессе, как для действительной, так и фиктивной, производится перебор ее соседей, попавших в 27 ячеек, где 27-й является центральная ячейка, в которую попала выбранная частица. Ячейки сетки кубические, размер ячеек берется увеличенным в 2 раза для обеспечения охвата зоны влияния двух частиц, находящихся на расстоянии радиуса обрезания. Номера тех частиц, что попали в область влияния, заносятся в массив соседей, имеющий размерность (M, n), где M – число частиц, приходящихся на процесс, а n – возможное число соседей, например, для MEAM потенциала Pu (плутония) $n \approx 12$.

Далее для каждой частицы численно рассчитывается градиент полной энергии. Через численный градиент полной энергии ведется расчет сил: для i-й частицы 6 раз (по 2 раза для каждой компоненты силы) вычисляется E_{tot} – полная энергия частиц, попавших в зону влияния i-й частицы, а компоненты силы рассчитываются по формулам (6)–(8):

$$F_x = -\frac{E_{tot}\left(...,\vec{r}_{i-1},\vec{r}_i + h\vec{e}_x,\vec{r}_{i+1},...\right) - E_{tot}\left(...,\vec{r}_{i-1},\vec{r}_i - h\vec{e}_x,\vec{r}_{i+1},...\right)}{2h}, \quad (6)$$

$$F_y = -\frac{E_{tot}\left(...,\vec{r}_{i-1},\vec{r}_i + h\vec{e}_y,\vec{r}_{i+1},...\right) - E_{tot}\left(...,\vec{r}_{i-1},\vec{r}_i - h\vec{e}_y,\vec{r}_{i+1},...\right)}{2h}, \quad (7)$$

$$F_z = -\frac{E_{tot}\left(...,\vec{r}_{i-1},\vec{r}_i + h\vec{e}_z,\vec{r}_{i+1},...\right) - E_{tot}\left(...,\vec{r}_{i-1},\vec{r}_i - h\vec{e}_z,\vec{r}_{i+1},...\right)}{2h}. \quad (8)$$

В первоначальном варианте программы расчета сил на GPU [5] для MEAM потенциала схема счетного шага выглядела следующим образом: в начале шага с CPU на GPU копируется информация с координатами и типами частиц, а также заполненные на CPU списки соседей всех частиц. Затем на GPU осуществляется расчет полной энергии частиц, попавших в зону влияния i-й частицы. После этого значения полной энергии копируются обратно на CPU и рассчитываются компоненты сил взаимодействия. Следующие действия на шаге – это расчет на CPU скоростей и новых координат частиц.

В результате проделанной работы следующие этапы вычислений производятся на GPU:

–заполнение списков соседей (число нитей на GPU равняется количеству частиц на процессоре);

–расчеты полной энергии (распараллеливание на потоки производится по парам взаимодействующих частиц, то есть числу частиц в процессе, помноженному на число соседей);

–расчет сил взаимодействия (число нитей равняется количеству частиц на процессоре),

–вычисление скоростей и координат.

Таким образом, обмен между GPU и CPU необходим только для организации пересылок граничной информации между отдельными GPU с помощью MPI на CPU.

Результаты тестирования и анализ эффективности программы для MEAM-потенциала

Моделировался образец плутония с ГЦК-решеткой при T_0=600 К. Начальный шаг кристаллической решетки – 4.6386 Å. Масса частицы – 239 а.е.м. Радиус обрезания – 4,5 Å. Рассматривались образцы различных размеров – от 55 тыс. частиц до 500 тыс. частиц.

Проведена профилировка реализованного алгоритма на GPU в сравнении со старым алгоритмом для образца плутония размером 500 тыс. частиц. Результаты представлены в табл. 7.

Из табл. 7 видно, что заполнение списков соседей на CPU в старом алгоритме занимало 17 % от времени шага. В результате переноса на GPU время этого этапа уменьшилось в 42 раза.

В соответствии с данными табл. 7 в старом алгоритме копирование с CPU на GPU и обратно в сумме занимало 5,55 % от времени шага, что не столь существенно по сравнению с потенциалами Морзе и EAM, где это значение было около 50 %. Поэтому перенос всех счетных программ на GPU и минимизация обменов между процессором и ускорителем в данном случае дает меньшее ускорение, чем для потенциалов Морзе и EAM, о чем свидетельствуют результаты в табл. 8.

В соответствии с данными в табл. 8 получаем, что коэффициент ускорения нового алгоритма по отношению к старому алгоритму при расчете на одном GPU для образца Pu в среднем равен 1,4, при расчете на двух GPU – 1,52.

Таблица 7

Профилировка счетного шага при расчете на одном GPU для MEAM-потенциала

	Заполнение списков соседей	Копирование информации на GPU	Расчет сил	Копирование информации на CPU	Расчет уравнений движения и обновление связных списков	Время счетного шага
Старый алгоритм (1)	0,208	0,0439	0,9194	0,0237	0,0179	1,2168
Новый алгоритм (2)	0,00487	Только на первом шаге	0,783	–	0,00057	0,7913
% от времени шага, (1)	17,09	3,6	75,56	1,95	1,47	–
% от времени шага, (2)	0,62	–	98,95	–	0,07	–

Таблица 8

Время 100 шагов при расчете на одном/двух GPU, потенциал взаимодействия – MEAM

Размер, a0 (число частиц) Время, с	24 (55,296 тыс.)	36 (186,624 тыс.)	48 (442,368 тыс.)	50 (500 тыс.)
Старый алгоритм (1)	21,015/11,927	55,848/40,259	123,838/63,568	138,456/72,942
Новый алгоритм (2)	13,692/7,964	36,818/22,275	89,793/45,576	116,945/51,968
Алгоритм (1)/ Алгоритм (2)	1,53/1,5	1,52/1,8	1,38/1,39	1,18/1,4

Заключение

В результате проделанной работы осуществлен перенос всех модулей расчета на GPU для потенциалов Морзе, EAM и MEAM, получена новая версия тестовой программы молекулярной динамики. Реализован и протестирован новый метод расчета сил взаимодействия на GPU – метод списка ячеек.

В ходе тестирования реализованных алгоритмов получено среднее ускорение для парного потенциала Морзе – в 10 раз, для EAM-потенциала – в 3 раза, для MEAM – в 1,5 раза по сравнению со старой версией на GPU.

Список литературы

1. Alder B. J., Wainwright T. E. Studies in Molecular Dynamics. I. General Method // J. Chem. Phys. 1957. Vol. 27. P. 1208.

2. Воронин Б. Л., Ерофеев А. М., Ветчинников М. В., Копкин С. В. и др. Комплекс программ молекулярно-динамического моделирования (MoDyS). Свидетельство о государственной регистрации программы для ЭВМ № 2010614974 // Электронный бюллетень. 2010.

3. Алексеев А. В., Беляев С. П., Бочков А. И., Быков А. Н., Ветчинников М. В. и др. Методические прикладные тесты РФЯЦ-ВНИИЭФ для численного исследования параметров высокопроизводительных систем // ВАНТ. Сер. Методики и программы численного решения задач математической физики. 2020. Вып. 2. С. 86–100.

4. Рыбакин Б. П.; под ред. акад. Бетелина В. Б. Параллельное программирование для графических ускорителей. М.: НИИСИ РАН, 2011.

5. Воронин Б. Л., Ерофеев А. М., Крючков И. А., Копкин С. В. и др. Применение графических арифметических ускорителей для расчета задач молекулярной динамики по программному комплексу МД // ВАНТ. Сер. Методики и программы численного решения задач математической физики. 2009. Вып. 2. С. 62–68.

6. Gaobo Xiao, Mingjun Ren, Haibo Hong. 50 million atoms scale molecular dynamics modelling on a single consumer graphics card // Advances in Engineering Software. 2018. N 124. P. 66–72.

7. Mishin Y., Mehl M. J., Papaconstantopoulos D. A., Voter A. F., Kress J. D. // Phys. Rev. 2001. B 63. P. 224106.

8. Baskes M. I. // Phys. Rev. 2000. B 62(23), P. 15532.

9. Baskes M. I., Lawson A. C., Valone S. M. // Phys. Rev. 2005. B. 72(1). P. 14129.