

Исследование эффективности CPU и GPU в моделировании систем электролит/графен/металл методом JDFT

Д. В. Гордиенко^{1,2}, С. В. Павлов¹, В. А. Кисленко¹ и С. А. Кисленко¹

¹ Объединенный институт высоких температур РАН, Ижорская ул., д. 13, стр. 2, Москва, Россия

² Московский физико-технический институт (государственный университет),
Институтский пер., 9, Долгопрудный 141701, Россия

E-mail: gordienko.dv@phystech.edu

Статья поступила в редакцию 1 августа 2025 г.

Аннотация. В данной работе проведено комплексное исследование производительности центральных и графических процессоров при выполнении расчетов, основанных на объединенной теории функционала плотности (JDFT — Joint Density Functional Theory). Были выполнены измерения скоростных характеристик при различных конфигурациях параллельных вычислений с использованием OpenMP и MPI. Осуществлено тестирование моделей электронейтральной поверхности в каноническом ансамбле при заданном числе электронов, а также заряженной поверхности в большом каноническом ансамбле с фиксированным химическим потенциалом электронов в системе электролит/графен/металл с использованием континуальной модели электролита CANDLER, имитирующей 0.1 М водный раствор KCl. Такие расчеты являются ключевыми для точного моделирования электрохимических систем. Результаты показали значительный прирост производительности при применении графических процессоров на данном классе задач. Также приведены оценки объема потребляемой оперативной памяти как для отдельного MPI-процесса, так и для всей программы. На основе полученных данных разработаны рекомендации по практическому применению программного пакета JDFTx.

1. ВВЕДЕНИЕ

Основной мировой тренд развития вычислительных технологий заключается в создании все более мощных вычислительных кластеров. Производительность современных суперкомпьютеров, возглавляющих список TOP-500, уже превышает эксафлопсный барьер [1]. Значительная часть этих ресурсов направлена на решение научных задач [2], хотя спрос на суперкомпьютерные вычисления со стороны промышленности также продолжает расти.

Область электрохимии не является исключением. Методы компьютерного моделирования доказали свою предсказательную способность и стали неотъемлемой частью разработки новых материалов и технологий [3]. Они активно применяются для создания электродных материалов [4, 5], электрокатализаторов [6], электролитов [7], а также для исследования процессов электронного и ионного переноса в электрохимических системах.

Методы моделирования в материаловедении можно разделить по масштабу рассмотрения: от крупномасштабных моделей до низкоуровневых квантово-механических расчетов (см. рисунок 1). С увеличением детализации модели растет и вычислительная сложность, что ограничивает размер исследуемых систем и продолжительность моделируемых процессов. На практике в вычислительной электрохимии наиболее востребованными являются метод молекулярной динамики (MD — Molecular Dynamics) и теория функционала плотности (DFT — Density Functional Theory), а также их комбинация [8]. Эти методы обеспечивают баланс между вычислительной сложностью и точностью. MD подходит для моделирования динамических процессов в электролитах, тогда как DFT незаменим для исследования электронных свойств электродных материалов и интерфейсов между электродом и электролитом.

Существует множество программных реализаций DFT, которые можно разделить на две группы в зависи-

мости от используемого базиса для волновых функций: на основе атомных орбиталей или плоских волн. Базис атомных орбиталей чаще применяется для неперiodических систем, а базис плоских волн — для периодических. Причем, вычислительно базис плоских волн является более сложной задачей из-за размера базиса. Кроме того, моделирование периодических систем требует сэмплирования зоны Бриллюэна (дискретной выборки точек в обратном пространстве) с помощью k-сетки, что также существенно увеличивает вычислительную сложность. Среди популярных программных пакетов, реализующих эти методы, можно выделить Gaussian [10], ORCA [11], VASP [12] и Quantum Espresso [13].

Одной из ключевых задач в электрохимии является моделирование гетерогенных интерфейсов, таких как граница электрод/электролит [14]. Это требует учета как электронных свойств электрода, так и динамики ионов в электролите. В частности, важно рассматривать два сценария: система, находящаяся при потенциале нулевого заряда, когда поверхность электрода электронейтральна, и режим с заданным внешним потенциалом, когда на электроде возникает конечный заряд [15]. Решение таких задач на уровне DFT требуют моделирования тысяч динамических конфигураций атомов электролита в приэлектродной области. При этом расчеты в большом каноническом ансамбле (для учета электрохимического потенциала) становятся крайне ресурсоемкими.

Кроме того, растущий интерес к гетероструктурам на основе графена и других материалов [16], например металлов, требует точного учета перераспределения заряда между графеном и подложкой в зависимости от потенциала электрода. С точки зрения вычислений это приводит к необходимости использования плотных k-сеток из-за низкой плотности электронных состояний графена близи уровня Ферми [17], что опять-таки повышает вычислительную сложность.

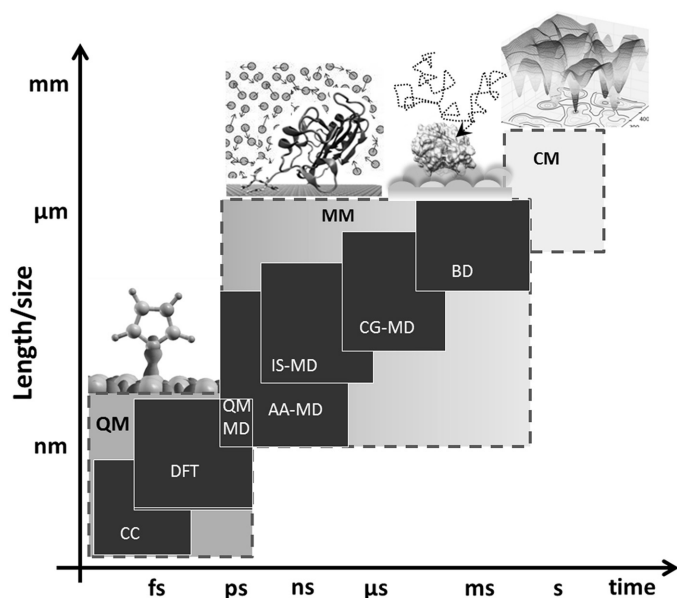


Рисунок 1. Типичные масштабы времени и длины различных методов моделирования: квантовая механика (QM), включая метод связанных кластеров (CC) и теорию функционала плотности DFT; молекулярная механика (MM), включая полноатомную молекулярную динамику (AA-MD), молекулярную динамику с неявным растворителем (IS-MD), крупнозернистую молекулярную динамику (CG-MD) и метод броуновской динамики (BD); и механика сплошных сред (CM). Диапазоны масштабов времени и длины являются приблизительными. [9] (Воспроизведено с разрешения издательства Cambridge University Press).

Ввиду этого, необходимо использование высокопроизводительных вычислительных систем и оптимизации расчетов как на уровне моделей, так и на уровне вычислительных алгоритмов и реализации параллельных вычислений.

Для решения этих проблем хорошо подходит объединенная теория функционала плотности (JDFT — Joint Density Functional Theory) [18]. JDFT сочетает в себе точное квантово-механическое описание электронной структуры электрода с континуальными моделями электролита, что позволяет значительно снизить вычислительную сложность по сравнению с полностью атомистическими подходами. Такой подход особенно полезен для моделирования систем с большим количеством ионов, где полное атомистическое описание становится непрактичным.

В программном пакете JDFTx [19] реализован метод JDFT, при этом код предоставляет широкие возможности для высокопроизводительных вычислений благодаря поддержке различных методов параллелизма, включая OpenMP (OMP, Open Multi-Processing), MPI (Message Passing Interface) и CUDA (Compute Unified Device Architecture). Однако для эффективного использования ресурсов современных вычислительных систем необходимо определить оптимальные параметры параллелизации, такие как число MPI-процессов, OMP-потоков и GPU (Graphics Processing Unit) для оптимизации использования имеющихся вычислительных ресурсов.

В данной работе проводится тестирование программы JDFTx на различных конфигурациях вычислитель-

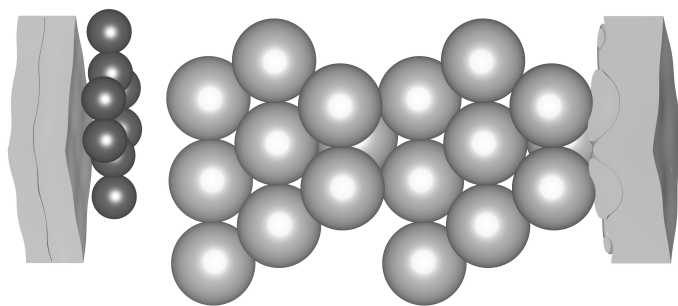


Рисунок 2. Структура тестируемой системы.

ных ресурсов, включая разное число MPI-процессов, OMP-потоков и GPU, для определения оптимальных параметров выполнения задач. Результаты тестирования позволяют предложить рекомендации для достижения максимальной производительности на современных гетерогенных вычислительных системах.

2. МОДЕЛЬ И МЕТОД РАСЧЕТА

Тестирование производилось в рамках объединенной теории функционала плотности (JDFT) [20] в большом каноническом ансамбле, реализованной в программном пакете JDFTx [19] версии 1.7.0, позволяющем использовать вариационную оптимизацию числа электронов совместно с неявной моделью электролита для обеспечения постоянства электрохимического потенциала [21]. Данная версия была скомпилирована с помощью CUDA 11.8, MKL 2021.4.0, GCC 11.2.1 и Intel MPI 2021.4.0.

В качестве тестовой системы была выбрана двумерная гетероструктура графен/золото (рисунок 2). Планарная золотая поверхность (111), состоящая из 6 атомных слоев, по 3 атома в каждом, с одной стороны контактировала с однослойным графеном, построенным из 8 атомов углерода. Из-за несовпадения периодов элементарных решеток графена и золота был выполнен их поворот друг относительно друга на 30° и слабое растяжение длин связей графена на 2%, как предложено в работе [22] (структура $(\sqrt{3} \times \sqrt{3})R30^\circ - GR30^\circ$) с последующим выполнением оптимизации как положения атомов, так и параметров расчетной ячейки. Порог сходимости для разности энергий и среднеквадратичных сил во время ионной оптимизации составил 10^{-6} Хартри и 10^{-6} Хартри/Бор, соответственно. Данная оптимизация была проведена до выполнения всех тестовых замеров скорости работы JDFTx. Тестирование производительности выполнялось для задачи полной электронной оптимизации структуры, находящейся в контакте с электролитом при фиксированной и предварительно оптимизированной геометрии. Электронная оптимизация проводилась в трех различных сценариях: с заданным числом электронов при потенциале нулевого заряда (ПНЗ), при фиксированных потенциалах 0 В и 1 В относительно стандартного водородного электрода (СВЭ). Критерий сходимости электронных шагов составлял 10^{-8} Хартри.

Для описания электролита была использована неявная континуальная модель CANDLE [27] с концентрациями 0.1 М K^+ и 0.1 М Cl^- . Во всех расчетах, в том числе и для предварительной оптимизации геометрии,

Таблица 1. Аппаратная платформа тестирования [23–26].

Характеристика	Значение	
Раздел	Broadwell	A100
Тестируемая платформа	CPU	GPU
Тип процессора	CPU	GPU / CPU
Название	Intel Xeon E5-2697Av4	NVIDIA A100 / Intel Xeon Platinum 8368Q
Частота	2.6 ГГц	0.765 ГГц / 2.6 ГГц
Число ядер	16	6912 / 38
Пиковая производительность	665.6 GFlops	9746 GFlops / 1165.2 GFlops
Объем памяти в узле	128 Гб	2 × 40 Гб / 512 Гб
Число процессоров в узле	2	2 / 2

применялась k-сетка Монкхорста–Пака [28] размерностью $13 \times 13 \times 1$. Плосковолновой базис и электронная плотность обрезались на уровнях 25 Хартри и 175 Хартри соответственно. Применялась Ван-дер-Ваальсова поправка DFT-D3 [29] и обменно-корреляционный функционал GGA-PBE [30]. В качестве псевдопотенциалов были выбраны ультрамягкие псевдопотенциалы GBRV [31]. Размер сетки реального пространства, используемой для скалярных полей, был зафиксирован с размерностью $100 \times 100 \times 700$. Выполнялось холодное размытие (cold smearing function) заполнений электронных уровней [32] с шириной размытия 10^{-3} Хартри. В плоскости гетероструктуры применялись периодические граничные условия, а в ортогональном направлении — непериодические с зазорами по 20 Бор до непериодических границ расчетной ячейки с двух сторон от системы, заполненные электролитом. Кулоновское взаимодействие в непериодической оси рассчитывалось с применением усеченного кулоновского ядра (truncate coulomb interaction) [33]. Потенциал электрода задавался с помощью фиксирования электрохимического потенциала электронов в явной части расчетной системы. Для этого применялся модифицированный метод прямой вариационной минимизации с использованием вспомогательного гамильтониана [34].

Тестирование выполнялось на ресурсах МСЦ РАН [23]. В качестве CPU-платформы (Central Processing Unit) был выбран раздел Broadwell, в качестве GPU-платформы применялось оборудование раздела A100. Характеристики вычислительных узлов представлены в таблице 1.

3. РЕЗУЛЬТАТЫ

3.1. Производительность и масштабируемость

На рисунке 3 представлены итоги замеров скорости вычислений на CPU- и GPU-платформах. Стоит отметить, что расчет в JDFTx делится на несколько стадий. Сначала строятся начальные приближения волновых функций методом построения линейных комбинаций атомных орбиталей, затем выполняется этап DFT-счета системы в вакууме, после чего проводится оптимизация в электролите. В данной работе представлена средняя производительность одной итерации последнего этапа алгоритма оптимизации (рисунки 3(а, б)) совместно с обратным временем работы всего алгоритма (рисунки 3(в, г)). На рисунках 3(а, в) изображена зависимость среднего обратного времени выполнения одной

итерации от произведения количества MPI-процессов на количество OMP-потоков. При этом каждый поток запускался на одном ядре, чтобы не создавать ситуацию конкуренции за ресурсы между потоками.

Результаты тестирования CPU-системы демонстрируют нелинейный рост, который совпадает с экстраполяцией результатов в рамках модели идеального масштабирования с последующим замедлением ускорения времени выполнения одной итерации минимизации из-за необходимости обмена информацией и синхронизации данных между MPI-процессами. Данные, полученные в пределах одного вычислительного узла, свидетельствуют о необходимости запуска одного MPI-процесса не более чем на одном сокете, так как происходит существенное ослабление роста скорости счета при запуске одного MPI-потока в пределах всего двухпроцессорного вычислительного узла.

Однако нужно учитывать, что MPI-параллелизм в коде организован только для распределения по k-точкам, а значит число MPI-процессов не должно превышать количество рассчитываемых k-точек. В целом данный график демонстрирует, что производительность алгоритма минимизации слабо зависит от соотношения числа OpenMP-потоков и MPI-процессов при равном общем числе вычислительных потоков, но в среднем наблюдается слабый (не более $\sim 10\%$) прирост производительности для конфигураций с большим числом MPI-процессов. Здесь стоит отметить, что остальные этапы алгоритма оказались более чувствительными к типу применяемого механизма параллелизации и показывают более существенное расхождение результатов (рисунок 3(в)).

Результаты тестирования GPU-системы, представленные на рисунке 3(б), позволяют сделать вывод, что наиболее оптимальным режимом счета с точки зрения производительности является задание ровно одного MPI-процесса на один GPU, так как именно такие данные укладываются на пропорциональность, предсказываемую моделью идеального масштабирования. Увеличение числа MPI-потоков, запущенных на одном графическом процессоре, не приводит к существенному приросту, а, наоборот, несколько снижает скорость счета. Это свидетельствует о том, что в процессе работы все MPI-процессы, запущенные в пределах одной GPU используют все доступные CUDA-ядра, а запуск двух и более MPI-потоков на одной графической карте приводит к необходимости синхронизации потоков и дополнительным затратам времени на эту процедуру.

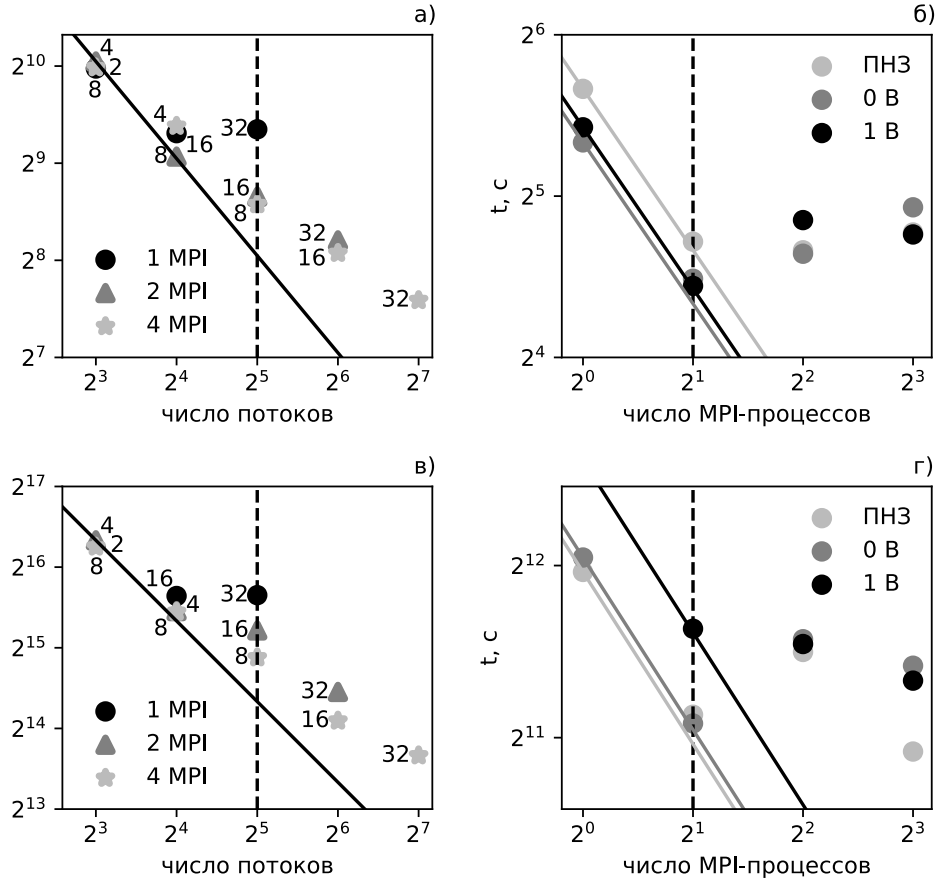


Рисунок 3. Результаты тестирования производительности и масштабирования. Представлены времена для одной итерации минимизации энергии системы в электролите (а), (б) и общего времени работы алгоритма (в), (г) в зависимости от общего числа задействованных потоков для CPU-платформы (а), (в) и от числа запущенных MPI-процессов для GPU-платформы (б), (г). Сплошной линией везде указан тренд идеального масштабирования, слева от пунктира ограничена область точек, когда задача выполняется на одном вычислительном узле (а), (в) и когда на каждую GPU приходится ровно один MPI-процесс (б), (г). На рисунках (а) и (в) цифрами рядом с точками указано число запущенных OMP-потоков внутри одного MPI-процесса.

Результаты исследования общего времени выполнения алгоритма, представленные на рисунках 3(в) и 3(г), в целом демонстрируют те же асимптотики, что и соответствующие графики на рисунках 3(а) и 3(б). Однако стоит отметить, что некоторую нестабильность в итоги вносит различие общего числа итераций при проведении оптимизации системы, что наиболее заметно для результатов, полученных на GPU. Это связано с накоплением ошибок округления в процессе счета и, возможно, применением алгоритмов стохастического интегрирования.

Сопоставление результатов счета в разных режимах: при ПНЗ и заданных фиксированных значениях потенциала 0 В и 1 В относительно СВЭ показывает, что их производительности не существенно отличаются, причем наблюдается, что общее время работы алгоритма при ПНЗ меньше, чем в других случаях, а время выполнения одной итерации больше. Такое несоответствие связано как с необходимостью проведения меньшего числа итераций при ПНЗ, так и с применением иного алгоритма минимизации по сравнению с расчетами при ПНЗ (при фиксированном значении потенциала, а не числа электронов) [34].

Сравнение производительности счета на разных платформах (рисунок 4) по методике, предложенной ра-

нее в работе [35], показывает несомненное преимущество применяемого графического процессора над центральным: времена выполнения одной итерации и всего алгоритма в целом отличаются на порядок. Этот результат соответствует соотношению их пиковых производительностей. Поскольку на вычислительном кластере МСЦ РАН был только один вычислительный узел с двумя видеокартами NVIDIA A100, не было возможным проверить дальнейшее масштабирование задачи на большее количество видеокарт, находящихся, в том числе, на разных узлах.

3.2. Использование памяти

Отмеченное выше преимущество GPU-систем над CPU на практике ограничивается объемом доступной памяти, которую можно существенно увеличить в платформах на основе центральных процессоров без относительно высоких финансовых затрат, но невозможно произвести для графических процессоров, так как для них эта операция сопряжена с необходимостью закупки дополнительных видеокарт или их замены на другие модели. Поэтому становится важным учет размера потребляемой памяти для выбора аппаратной платформы при проведении реальных расчетов [36]. Анализ

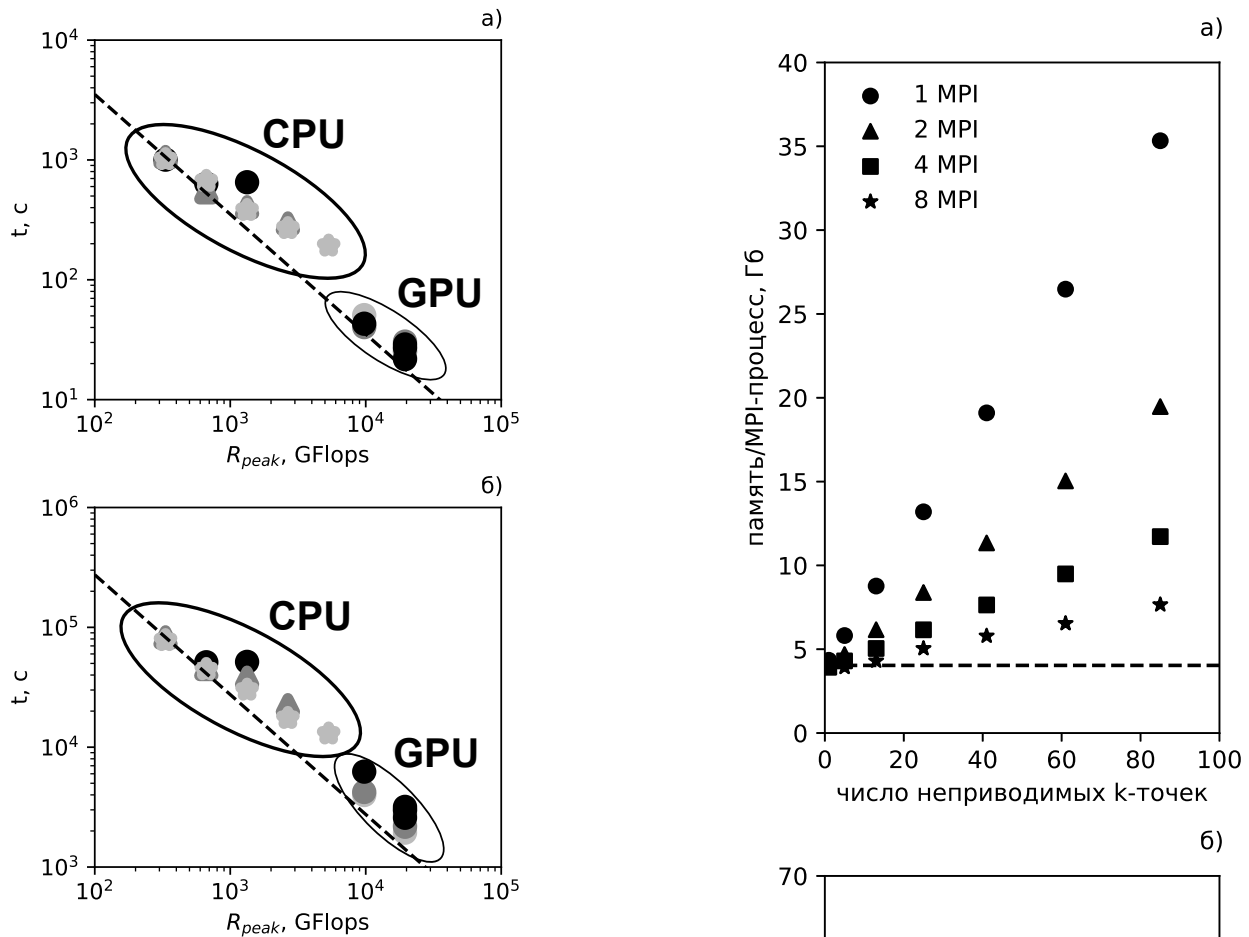


Рисунок 4. Сравнение производительности вычислений на платформе CPU и GPU (подписаны на графиках). Представлены времена для одной итерации минимизации энергии системы в электролите (а) и общего времени работы алгоритма (б) в зависимости от доступной пиковой производительности применяемого оборудования. Пунктирной линией везде указан тренд идеального шкалирования.

необходимого объема памяти на проведение расчетов в различных условиях представлен на рисунке 5.

Одним из наиболее существенных параметров с точки зрения объема потребляемой памяти является размер k -сетки в пространстве Фурье. Для эффективного проведения расчетов на ней автоматически выполняется процедура уменьшения числа расчетных точек с учетом симметрий системы. Эта процедура создает меньший набор k -точек, которые называются неприводимыми k -точками. На рисунке 5(а) представлена зависимость требуемого на один MPI-процесс объема памяти от числа таких неприводимых k -точек для различного количества одновременно запущенных MPI-потоков. Наблюдается линейный рост всех зависимостей с коэффициентом наклона, понижающимся по мере повышения числа MPI-процессов, причем он обратно пропорционален количеству MPI-потоков. Все зависимости стремятся к одинаковому объему памяти при экстраполяции к ситуации с нулевым числом k -точек. Это соответствует ситуации, когда выделение памяти происходит только для данных, рассчитываемых на сетках в реальном пространстве.

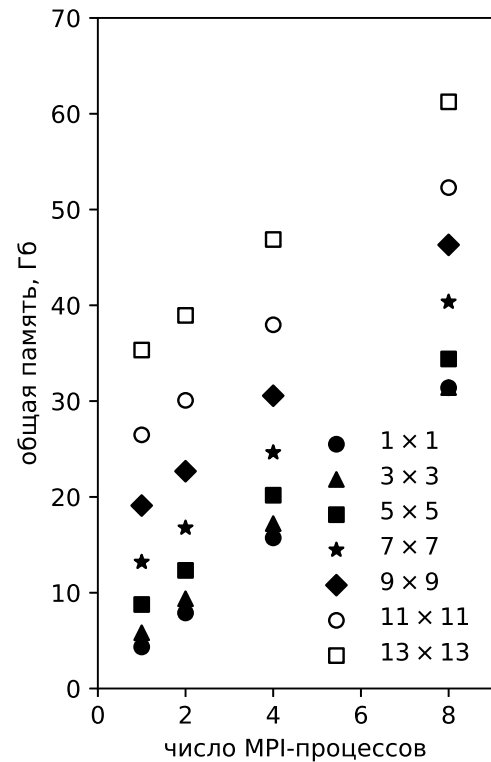


Рисунок 5. Результаты тестирования количества потребляемой памяти. Представлены зависимости размера памяти, приходящейся на один MPI-процесс (а) и общей потребляемой алгоритмом памяти (б) от числа неприводимых k -точек (а) и MPI-процессов (б) при различном количестве задействованных MPI-процедур (а) и размеров k -сеток (б). Пунктиром указана минимальная величина потребляемой памяти (а).

Рисунок 5(б) визуализирует зависимость общей памяти, необходимой для всего расчета, от числа запущенных MPI-процессов на разных k-сетках. Эти графики тоже представляют собой линейные зависимости, причем с одинаковыми коэффициентами наклона, что связано с тем, что создание одного MPI-потока сопровождается дополнительным выделением памяти на дублирование данных для проведения вычислений на сетках реального пространства, данные же по k-точкам распределяются между потоками, так как внутри кода JDFTx MPI-параллелизация осуществляется только по k-точкам. Этот результат также подтверждается тем, что данные на малой k-сетке 1×1 , когда вклад данных от k-точек пренебрежимо мал, образует зависимость, проходящую через начало координат.

Вышеприведенные результаты позволяют оценить функциональную зависимость нужного для проведения вычислений объема памяти. Данные с графика на рисунке 5(а) при одном MPI-процессе методом наименьших квадратов аппроксимируются линейной зависимостью:

$$s_{1MPI} \approx 0.368 \cdot N_k + 3.972 \text{ [Гб]}, \quad (1)$$

где s_{1MPI} — объем потребляемой памяти при расчете в одном MPI-процессе, N_k — число неприводимых k-точек. Все остальные результаты отличаются с точностью до деления коэффициента наклона на число MPI-потоков. Тогда итоговые зависимости принимают следующий вид:

$$s_{MPI} \approx 0.368 \cdot N_k / N_{MPI} + 3.972 \text{ [Гб]}, \quad (2)$$

$$s_{\text{общее}} \approx 0.368 \cdot N_k + 3.972 \cdot N_{MPI} \text{ [Гб]}, \quad (3)$$

где s_{MPI} — объем потребляемой памяти в одном MPI-процессе, $s_{\text{общее}}$ — общий объем потребляемой памяти, N_{MPI} — число запущенных MPI-потоков.

Для обобщения полученных результатов на более широкий класс задач нужно внести правки в выражения 2 и 3, связанные с числом рассчитываемых электронных состояний в пределах каждой k-точки, размером плоскостного базиса и сетки реального пространства. Расчет ведется для каждого состояния в каждой k-точке на заданном базисе, поэтому можно предположить, что объем памяти зависит от этих параметров так же, как и от размера k-сетки. Выше было предположено, что свободный член в формуле (1) связан с созданием копий сеток реального пространства в разных MPI-процессах, а значит он должен быть пропорционален размеру этих сеток. Тогда, с учетом того, что в тестовой системе было 194 электронных уровня, средний размер плоскостного базиса на k-точку 31253.994 и сетка реального пространства $100 \times 100 \times 700$, можно предположить, что результаты выражений (2) и (3) оценочно обобщаются следующим образом:

$$s_{MPI} \approx 0.368 \cdot N_k / N_{MPI} \cdot N_{\text{сост}} / 194 \cdot N_{\text{базис}} / 31254 + 3.972 \cdot N_{\text{сетка}} / (7 \cdot 10^6) \text{ [Гб]}, \quad (4)$$

$$s_{\text{общее}} \approx 0.368 \cdot N_k \cdot N_{\text{сост}} / 194 \cdot N_{\text{базис}} / 31254 + 3.972 \cdot N_{MPI} \cdot N_{\text{сетка}} / (7 \cdot 10^6) \text{ [Гб]}, \quad (5)$$

Таблица 2. Сопоставление теоретических предсказаний объема нужного для расчета размера памяти с величинами, полученными на практике.

Величина	Значение			
$N_{\text{слоев}}$	1	2	3	4
k-сетка	13×13	13×13	13×13	13×13
N_k	85	85	85	85
$N_{\text{сетка}}$	7×10^6	7.7×10^6	8.4×10^6	9.1×10^6
$N_{\text{сост}}$	194	226	258	290
$N_{\text{базис}}$	31254	34436	37592	40786
N_{MPI}	2	2	2	2
S_{MPI} (эксперимент)	19.5 Гб	24.3 Гб	29.6 Гб	35.5 Гб
S_{MPI} (теория)	19.6 Гб	24.4 Гб	29.8 Гб	35.7 Гб
$\omega_{\text{отн}}$	0.7 %	0.7 %	0.6 %	0.6 %

где S_{MPI} — примерный обобщенный объем потребляемой памяти в одном MPI-процессе, $S_{\text{общее}}$ — примерный обобщенный общий объем потребляемой памяти, $N_{\text{сост}}$ — число электронных состояний, $N_{\text{базис}}$ — средний размер плоскостного базиса на одну k-точку, $N_{\text{сетка}}$ — число точек сетки реального пространства.

Предсказания выражения (4) сопоставлены с расчетными значениями, полученными при варьировании числа слоев графена ($N_{\text{слоев}}$) в системе, ранее визуализированной на рисунке 2. Это приводит к изменению всех трех вышеприведенных величин: $N_{\text{полос}}$, $N_{\text{базис}}$ и $N_{\text{сетка}}$. Результаты представлены в таблице 2. Наблюдается хорошее согласие расчетных теоретических величин между собой с относительной погрешностью $\omega_{\text{отн}} \sim 1\%$.

4. ЗАКЛЮЧЕНИЕ

В ходе работы было проведено тестирование программного пакета JDFTx в задачах электронной оптимизации гетероструктуры графен/золото с неявной моделью электролита, в различных условиях: при потенциале нулевого заряда и при фиксированном электрохимическом потенциале — на CPU- и GPU-платформах. Результаты демонстрируют существенный выигрыш от применения графических процессоров, однако на практике возможность их применения ограничена объемом доступной видеопамяти. Для расчета необходимого объема памяти, инициализируемого в пределах одного MPI-процесса и всей программы, получены аналитические оценки в виде выражений (4) и (5) соответственно.

Результаты исследования позволяют привести следующие рекомендации по эффективному применению JDFTx на практике:

- при работе на GPU-системах запускать не более одного MPI-процесса на одной GPU, что эффективней как с точки зрения скорости вычислений, так и необходимого объема памяти;
- при работе на CPU-системах задавать как можно большее число MPI-потоков, не менее одного на сокет, но не больше числа неприводимых k-точек, внутри каждого MPI-процесса запускать наибольшее возможное число OpenMP-потоков;

- Выбор между имеющимися в наличии GPU- и CPU-системой можно осуществлять на основе доступного объема памяти, и оценочных формул (4) и (5): если для решения задачи достаточно объема памяти графического процессора, то считать на нем, иначе проводить вычисления на центральном процессоре. Эти же оценки можно применять при планировании закупок вычислительного оборудования.

При планировании строительства новых вычислительных кластеров стоит обращать внимание на гибридные (гетерогенные) системы с графическими ускорителями с объемом видеопамати более 100 Гб. Это подтверждается и мировым опытом из наиболее мощных вычислительных кластеров из списка top500 [1], использующих именно такие решения.

БЛАГОДАРНОСТИ

Работа выполнена при поддержке Министерства науки и высшего образования Российской Федерации (Государственное задание № 075-00269-25-00).

СПИСОК ЛИТЕРАТУРЫ

- List of 500 supercomputers, november 2024 URL <https://top500.org/lists/top500/2024/11>
- Wei J, Chen M, Wang L, Ren P, Lei Y, Qu Y, Jiang Q, Dong X, Wu W, Wang Q *et al* 2022 *CCF Transactions on High Performance Computing* **4** 211–230
- Yang X H, Zhuang Y B, Zhu J X, Le J B and Cheng J 2022 *Wiley Interdisciplinary Reviews: Computational Molecular Science* **12** e1559
- Zhu X, Yang Y, Shu X, Xu T and Jing Y 2023 *Wiley Interdisciplinary Reviews: Computational Molecular Science* **13** e1660
- Bi S, Knijff L, Lian X, van Hees A, Zhang C and Salanne M 2024 *ACS nano* **18** 19931–19949
- Levell Z *et al* 2024 *Chemical Reviews* **124** 8620–8656
- Yao N, Chen X, Fu Z H and Zhang Q 2022 *Chemical Reviews* **122** 10970–11021
- He Y, Liu S, Wang M, Ji H, Zhang L, Cheng Q, Qian T and Yan C 2022 *Advanced Functional Materials* **32** 2208474
- Ozboyaci M, Kokh D B, Corni S and Wade R C 2016 *Quarterly reviews of biophysics* **49** e4
- Frisch M *et al* 2009 *Inc., Wallingford CT* **121** 150–166
- Neese F 2012 *Wiley Interdisciplinary Reviews: Computational Molecular Science* **2** 73–78
- Kresse G and Furthmüller J 1996 *Physical review B* **54** 11169
- Giannozzi P *et al* 2009 *Journal of physics: Condensed matter* **21** 395502
- Groß A and Sakong S 2019 *Current Opinion in Electrochemistry* **14** 1–6
- Melander M M, Wu T, Weckman T and Honkala K 2024 *npj Computational Materials* **10** 5
- Inozemtseva A, Savin V, Itkis D and Yashina L 2024 *Current Opinion in Electrochemistry* **49** 101632
- Pavlov S V, Kozhevnikova Y O, Kislenko V A and Kislenko S A 2022 *Physical Chemistry Chemical Physics* **24** 25203–25213
- Sundararaman R, Vigil-Fowler D and Schwarz K 2022 *Chemical reviews* **122** 10651–10674
- Sundararaman R, Letchworth-Weaver K, Schwarz K A, Gunceler D, Ozhabes Y and Arias T A 2017 *SoftwareX* **6** 278–284
- Letchworth-Weaver K and Arias T 2012 *Physical Review B—Condensed Matter and Materials Physics* **86** 075140
- Xiao H, Cheng T, Goddard III W A and Sundararaman R 2016 *Journal of the American Chemical Society* **138** 483–486
- Yortanlı M and Mete E 2019 *The Journal of Chemical Physics* **151** 214701
- Savin G, Shabanov B, Telegin P and Baranov A 2019 *Lobachevskii Journal of Mathematics* **40** 1853–1862
- МСП Руководство пользователя раздела MBC-10П МПЗ а100 ВЕРСИЯ ОТ 07.02.2024 URL <https://www.jscc.ru/documents/2024/02/instruktsiya-po-ispolzovaniyu-razdela-a100.pdf>
- NVIDIA Nvidia a100 pcie 40 gb URL <https://www.techpowerup.com/gpu-specs/a100-pcie-40-gb.c3623>
- Intel Export compliance metrics for intel microprocessors URL <https://www.intel.com/content/www/us/en/support/articles/000005755/processors.html>
- Sundararaman R and Goddard W A 2015 *The Journal of chemical physics* **142** 064107
- Monkhorst H J and Pack J D 1976 *Physical review B* **13** 5188
- Grimme S, Antony J, Ehrlich S and Krieg H 2010 *The Journal of chemical physics* **132** 154104
- Perdew J P, Burke K and Ernzerhof M 1996 *Physical review letters* **77** 3865
- Garrity K F, Bennett J W, Rabe K M and Vanderbilt D 2014 *Computational Materials Science* **81** 446–452
- Marzari N, Vanderbilt D, De Vita A and Payne M 1999 *Physical review letters* **82** 3296
- Sundararaman R and Arias T 2013 *Physical Review B—Condensed Matter and Materials Physics* **87** 165122
- Sundararaman R, Goddard W A and Arias T A 2017 *The Journal of chemical physics* **146** 114104
- Stegailov V V, Orekhov N D and Smirnov G S 2015 HPC hardware efficiency for quantum and classical molecular dynamics *International conference on parallel computing technologies* pp 469–473
- Stegailov V, Smirnov G and Vecher V 2019 *Concurrency and Computation: Practice and Experience* **31** e5136