Как быстрее всего решать задачи квантового и классического атомистического моделирования, используя современное суперкомпьютерное программное и аппаратное обеспечение?*

Г.С. Смирнов, В.В. Стегайлов

Объединенный институт высоких температур РАН

Развитие аппаратного обеспечения для высокопроизводительных расчетов опережает адаптацию алгоритмов для таких фундаментальных математических алгоритмов как классическая и квантовая молекулярная динамика. Большое разнообразие выбора обуславливает необходимость ясных критериев, основанных на вычислительной эффективности определенного алгоритма на определенном аппаратном обеспечении. Тест LINPACK не может более служить этой цели. В этой работе мы рассматриваем основанную на практических соображениях метрику «время решения — пиковая производительность». В этой метрике мы сравниваем различное аппаратное обеспечение (как современное, так и вышедшее из употребления) на примере тестов программного обеспечения LAMMPS, GROMACS, NAMD и CP2K, широко используемых для атомистического моделирования. Показано, что рассмотренная метрика может служить для однозначного сравнения различных комбинаций центральных процессоров, ускорителей и интерконнекта.

1. Введение

Гибридные вычислительные системы, использующие несколько различных вычислительных устройств, получили широкое распространение в последние годы. Анализ списка Топ-500 [1] лучших суперкомпьютеров мира показывает, что наибольшей популярностью на сегодняшний день пользуются машины, использующие в качестве укорителей видеокарты NVIDIA архитектуры Kepler или сопроцессоры Intel Xeon Phi архитектуры MIC (Many Integrated Core).

Резкий рост количества машин с гибридной архитектурой связан в первую очередь с перспективой существенного увеличения производительности суперкомпьютера без значительного увеличения затрат. Пиковая производительность современных ускорителей достигает нескольких ТФлопс (10^{12} операций с плавающей точкой в секунду). Однако использование значительной доли пиковой вычислительной мощности R_{peak} в реальных приложениях не всегда возможно. Это связано с необходимостью полного переписывания исходного кода для обычных CPU, решения проблем с доступом к памяти ускорителя, шириной канала передачи данных между центральным процессором и ускорителем, а также особенностями работы АЛУ ускорителей.

Предпринимаются попытки создания специализированных суперкомпьютеров для конкретных приложений, однако они не входят в список Топ-500. Для задач молекулярной динамики можно выделить суперкомпьютеры на программируемых пользователем вентильных матрицах (разновидность ПЛИСов) [2], MDGRAPE [3] и ANTON [4] на интегральных схемах специального назначения. Использование специальной архитектуры позволяет добиться наиболее эффективного использования вычислительной мощности для конкретной задачи.

При этом «классические» многопроцессорные суперкомпьютеры, использующие для вычислений только центральные процессоры, продолжают доминировать в списке Топ-500, и на такие системы продолжают ориентироваться при разработке ПО для классической

*Работа частично поддержана грантами РФФИ 13-01-12070-офи_м и 14-08-31550 мол_а.

молекулярной динамики [5,6]. На них проводятся рекордные по числу частиц молекулярно-динамические расчеты [7].

В сложившейся ситуации при наличии большого разнообразия аппаратного обеспечения для высокопроизводительных суперкомпьютерных расчетов особенную актуальность приобретает вопрос сравнения между собой различных альтернативных решений. Целью анализа должна быть эффективность связи 1) аппаратного обеспечения, 2) конкретной математической модели или класса моделей и 3) численных алгоритмов, в том числе с учетом уже существующего программного обеспечения (ПО) и сложности их адаптации на новые типы аппаратного обеспечения.

Популярный тест LINPACK, использующийся для ранжирования суперкомпьютеров, не отражает особенностей многих современных алгоритмов, в том числе и алгоритмов атомистического моделирования, являющихся предметом этой статьи. Для данного класса задач это приводит к существенному отличию между реальной и теоретически максимальной производительностью [8, 9]. Отсюда следует, что необходим анализ эффективности работы конкретных программ и алгоритмов на различных архитектурах. В качестве примеров аппаратного обеспечения рассматриваются лучшие суперкомпьютерные системы России. Анализируемые математические модели соответствуют стандартным примерам: леннард-джонсовская жидкость (типичная система для задач статистической физики, физики конденсированных сред и физической химии), белковая молекула в водном растворе и квантовый расчет молекул воды. В качестве примера ПО мы используем популярные пакеты для классических и квантовых задач атомистического моделирования: LAMMPS, GROMACS, NAMD, CP2K.

Развитие современных многомасштабных моделей в физике, химии, биологии, материаловедении и других областях существенным образом основано на моделировании процессов на атомистическом уровне. При этом даже достигнутый на сегодня рекордный размер моделей в триллионы частиц [7] соответствует, например, для металла при нормальной плотности объему всего в несколько мкм^3. Задача увеличения максимальных доступных времен молекулярно-динамических расчетов еще сложнее. Разработка подобных вычислительных методов неразрывно связана с прогрессом в суперкомпьютерных технологиях.

2. Сравнение современных ускорителей

2.1. NVIDIA GPU

Распространение графических процессоров для научных вычислений произошло за счет появления специальных технологий программирования CUDA и OpenCL. В настоящее время наиболее распространены две архитектуры компании NVIDIA: Fermi и Kepler. Пик популярности архитектуры Fermi пришелся на 2012 год, теперь она вытесняется более новой архитектурой Kepler. Архитектура Kepler обеспечивает большее быстродействие, имеет больший объем памяти, а также поддерживает последние версии спецификации CUDA. В работе исследовалась работа видеокарт NVIDIA X2070 архитектуры GF100 (Fermi) и NVIDIA K40 архитектуры GK110 (Kepler). Задачи на видеокартах выполняются по принципу SIMD (одиночный поток команд и множественный поток данных single instruction multiple data). Базовой единицей при запуске задачи является поток (thread), выполняемый на одном ядре. Потоки объединяются в блоки (block), запускаемые на одном мультипроцессоре. При этом они имеют доступ к одним и тем же данным в регистрах и разделяемой памяти. Потоки запускаются группами из 32 единиц, называемых варпами (warp). Принцип SIMD подразумевает, что одна команда применяется ко всем потокам в варпе.

Видеокарты архитектуры GF100 имеют 14 потоковых мультипроцессоров, каждый состоит из 32 вычислительных ядер, 4 устройств для расчета трансцендентных функций с одинарной точностью. Вычислительные ядра работают на частоте 1150 МГц, только половина из них может выполнять операции с двойной точностью. Развитие производитель-

ности видеокарт идет по пути увеличения вычислительной мощности мультипроцессоров. Мультипроцессор архитектуры GK110 имеет 192 вычислительных ядра, 64 из них могут выполнять операции с двойной точностью. Ядра работают на частоте 745 МГц. На видеокарте установлено 15 таких мультипроцессоров. Число ядер увеличилось более чем в 6 раз, для работы с ними число одновременно запускаемых потоков на мультипроцессоре также увеличено. Вдвое увеличено число регистров на поток для повторного использования переменных. Также появилась поддержка динамического параллелизма — потоки GPU могут генерировать новые потоки без обращения к CPU, что может давать преимущества на некоторых задачах.

Тестирование видеокарт поколения Fermi и более старого Tesla на алгоритмах молекулярной динамики проводилось в работе [10]. Показано, что относительная производительность видеокарт от числа ядер хорошо описывается законом Амдала, то есть имеет место ограничение роста производительности из-за невозможности обеспечить стопроцентную параллелизацию алгоритма.

2.2. Intel Xeon Phi

Сопроцессоры Intel Xeon Phi впервые были представлены в 2012 году, на сегодняшний день представлено три серии сопроцессоров. В зависимости от модели сопроцессор имеет 57, 60 или 61 ядро. В данной работе рассматривалась модель Intel Xeon Phi SE10X с 61 ядром, установленная на суперкомпьютере МВС-10П МВС РАН. Каждое ядро имеет кэш инструкций и кэш данных по 32 КБ (первый уровень); инклюзивный кэш 512 КБ (второй уровень) и 512-битное векторное АЛУ. Все ядра соединены между собой двунаправленной кольцевой шиной. В отличие от GPU, на сопроцессоре запускается отдельная операционная система, что уменьшает на единицу число реально выделенных для расчетов ядер. Пиковая производительность сопроцессора при работе с двойной точностью достигает 1.0736 ТФлопс.

Существует три способа использования сопроцессора Intel Xeon Phi: 1) режим разгрузки (offload mode), 2) «родной» режим (native mode) сопроцессора и 3) симметричный режим (symmetrical mode). В режим разгрузки основная программа запускается на обычном процессоре, на сопроцессоре выполняются определенные участки программного кода. Это можно делать автоматически при использовании библиотеки Intel Math Kernel Library (MKL), либо в явном виде при использовании директив компилятора путем модификации исходного кода. В «родном» режиме программа запускается исключительно на сопроцессорах, и, наконец, в симметричном режиме одна программа одновременно запускается на CPU и сопроцессоре. Однако из-за различной частоты процессора (∼2–4 ГГц для Intel Xeon) и сопроцессора (∼1 ГГц для Intel Xeon Phi) в симметричном режиме могут возникнуть серьезные трудности с балансировкой нагрузки. В «родном» режиме можно добиться ускорения за счет оптимизации кода, однако при использовании суперкомпьютеров в данном случае будут простаивать CPU на занятых узлах.

При начале производства Intel Xeon Phi основной упор в рекламе новой технологии делался на возможность запуска программ без переписывания исходного кода. Однако многочисленная практика показывает, что это не приводит к ускорению работы конкретных приложений [11–14]. Для получения хотя бы минимального ускорения затраты на переписывание исходного кода программ становятся сравнимыми с аналогичными затратами на видеокартах.

Хотя с точки зрения производительности ускорители GPU и сопроцессоры Intel Xeon Phi довольно близки, использование их вычислительных возможностей на 100%, не всегда возможно. В литературе обсуждаются сложности, возникающие при портировании молекулярно-динамических кодов на архитектуру Intel Xeon Phi [15].

Таблица 1. Сравнение ускорителей, используемых для расчетов в данной работе

Ускоритель	NVIDIA X2070	NVIDIA K40	Intel Xeon Phi SE10X
Число ядер	448	2880	61
Объем памяти, Гб	6	12	8
Тактовая частота, ГГц	1.15	0.745	1.1
R_{peak} с двойной точностью, ТФлопс	0.515	1.43	1.0736
R_{peak} с одинарной точностью, ТФлопс	1.03	4.29	2.1472
Потребляемая мощность, Вт	247	235	300

2.3. Особенности реализации операций с плавающей точкой на ускорителях

Как на ускорителях NVIDIA GPU, так и на сопроцессорах Intel Xeon Phi, операции двойной точности с плавающей точкой поддерживают набор инструкций умножения-сложения с однократным округлением (fused multiply-add — FMA). При этом за один такт может выполняться одна операция сложения и одна операция умножения вида y=a*x+b без потери точности. Это позволяет добиться ускорения в алгоритмах, требующих суммирования произведений, например, перемножение матриц или вычисление значения многочлена по схеме Горнера.

Некоторые особенности возникают при расчете пиковой производительности в режиме одинарной точности. Производительность Intel Xeon Phi увеличивается в 2 раза за счет того, что в каждый регистр помещается в 2 раза больше данных. Видеокарты NVIDIA имеют ядра, которые работают только в режиме одинарной точности (single precision units — SPU), и ядра, поддерживающие режим двойной точности (double precision units DPU). Рост производительности в этом случае зависит от отношения SPU к DPU. Каждый SPU за 1 такт может выполнять одну операцию FMA и одну операцию умножения, то есть три операции с плавающей точкой на один такт. Также есть ядра, поддерживающие на аппаратном уровне вычисление трансцендентных функций с одинарной точностью. Их обычно не учитывают при расчете пиковой производительности.

В алгоритмах молекулярной динамики большую часть времени расчета занимает построение списка соседей и вычисление сил взаимодействия между атомами. Однако доля операций, в которых можно использовать инструкции FMA, ничтожна. В результате выполнения любой одиночной операции сложения/умножения фактически приводит к тому, что половина вычислительной мощности ускорителя или сопроцессора остается незадействованной. В отличие от ускорителей процессоры Intel поколения Sandy Bridge и Ivy Bridge, установленные сегодня на большинстве современных суперкомпьютеров, не поддерживают набор инструкций FMA на аппаратном уровне.

Используемые в работе ускорители сравниваются в таблице 1.

3. Реализация атомистических алгоритмов на ускорителях

В данной работе для тестов использовались популярные пакеты программ для атомистического моделирования LAMMPS [16], GROMACS [17], NAMD (классические задачи) и CP2K (квантовые задачи).

Таблица 2. Поддерживаемые архитектуры различных модулей пакета программ LAMMPS

Гибридная архитектура	USER-CUDA	GPU	KOKKOS	USER-INTEL
GPU	+	+	+	-
Intel Xeon Phi, «родной» режим	-	-	+	-
Intel Xeon Phi, режим разгрузки	-	-	-	+

Модули предназначены для расширения базового функционала, часть из них позволяет проводить расчеты на гибридной архитектуре. Для расчетов с использованием GPU-ускорителей существует 3 разных модуля, которые имеют названия «GPU» [19], «USER-CUDA» и «KOKKOS» [20]. Они отличаются особенностями реализации и могут давать различные результаты по производительности в зависимости от задачи и от используемого оборудования. Стоит отметить, что модуль «KOKKOS» разрабатывался для карт NVIDIA Kepler последнего поколения и может работать медленно на картах предыдущего поколения NVIDIA Fermi. Поддержка ускорителей Intel Xeon Phi в стадии активной разработки, она частично реализована в модулях «KOKKOS» («родной» режим) и «USER-INTEL» (режим разгрузки) для небольшого числа межатомных потенциалов. Данные по указанным выше модулям приведены в таблице 2.

В пакетах GROMACS, NAMD и CP2K также существует возможность запуска задач на видеокартах. Реализована поддержка Intel Phi в «родном» режиме (GROMACS, CP2K) и режиме разгрузки (NAMD).

Создание эффективных алгоритмов для сложных многочастичных потенциалов не всегда тривиально, как показывает опыт разработки на GPU [21–23]. Одной из основных проблем является неоднородный доступ к памяти. В результате либо приходится использовать более ресурсоемкие атомарные операции, либо усовершенствовать имеющиеся алгоритмы. Стоит отметить, что для эффективного использования многопроцессорных вычислительных систем также приходится оптимизировать алгоритмы, в частности, для обеспечения динамической балансировки нагрузки на ядра [24].

4. Результаты тестов

4.1. Леннард-джонсовская жидкость

В данном разделе приводится анализ производительности пакета программ LAMMPS на различных гибридных архитектурах. Задачи запускались на суперкомпьютере МГУ им. М.В. Ломоносова «Ломоносов» [25] (видеокарты NVIDIA X2070) и суперкомпьютере МСЦ РАН МВС-10П (сопроцессоры Intel Xeon Phi). Видеокарта K40 исследовалась на тестовом компьютере, предоставленном компанией NVIDIA. Для оценки эффективности алгоритмов рассматривалась леннард-джонсовская жидкость из 10^6 атомов, длина траектории составляла 100 шагов. В качестве критерия сравнения использовалось время расчета одного шага на один атом, так как оно слабо зависит от полного числа атомов и служит объективным критерием быстродействия. Производительность гибридных архитектур соотносилась с производительностью обычных процессоров в метрике их пиковой производительности R_{peak}.

Рассмотрим подробнее данные на рисунке 1. В части а) приведено сравнение обычной версии LAMMPS (без векторизации кода) и с ручной векторизацией (модуль «USER-INTEL») на суперкомпьютерах МВС-10П и «Ломоносов». Для сравнения приведены данные однопроцессорных тестов на некоторых вышедших из эксплуатации машинах начала 2000-

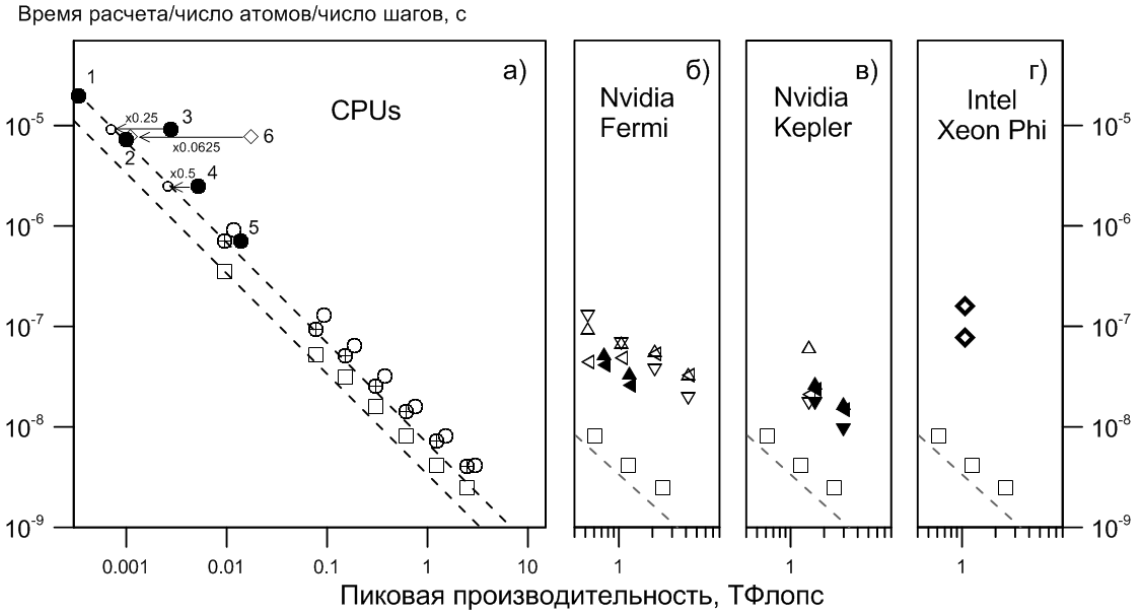

Рис. 1. Сравнение быстродействия МД-пакета LAMMPS на различных платформах: а) Расчеты на суперкомпьютере МВС-10П без (⊕) и с векторизацией (□) кода с использованием модуля USER-INTEL; на суперкомпьютере «Ломоносов» без векторизации кода (○). Для сравнения приведены данные однопроцессорных тестов с сайта LAMMPS [26] (●, 1 — Pentium II 333 МГц, 2 — DEC Alpha 500 МГц, 3 — PowerPC 440, 4 — Power4 1.3 ГГц, 5 — Intel Xeon 3.47 ГГЦ). Точка 6 — время расчета на одном ядре Intel Xeon Phi; б) сравнение модулей USER-CUDA (◁), GPU (△) и KOKKOS (▽) на видеокартах NVIDIA X5670 GPU (суперкомпьютер Ломоносов). Также приведены данные тестов на видеокартах C2075 с сайта LAMMPS (заполненные символы); в)сравнение модулей USER-CUDA (◁), GPU (△) и KOKKOS (▽) на NVIDIA K40 GPU. Заполненные символы — данные тестов с сайта LAMMPS для видеокарт Tesla K20x; г) быстродействие ускорителя Intel Xeon Phi на МВС-10П (◇). Верхняя точка — запуск в native-mode без оптимизации, нижняя — использование оптимизации модуля KOKKOS.

х гг. (таблица 3). Пиковая производительность рассчитывалась как произведение тактовой частоты на число ядер и на число операций с плавающей точкой на один такт (в соответствии со значениями, приводящимися в списке Топ-500). Крайние левые точки для МВС-10П и «Ломоносова» соответствуют одному вычислительному ядру, крайние правые — 256 ядрам. Пунктиром показана идеальная масштабируемость, когда увеличение вычислительного поля в два раза приводит к уменьшению времени расчета в 2 раза для той же задачи. Видно, что оптимизация «USER-INTEL» позволят добиться двукратного роста скорости расчета.

Рост производительности современных суперкомпьютеров идет как по пути наращивания вычислительных ядер, так и наращивания сложности самого ядра, внедрения операций FMA и поддержки векторных инструкций. Это приводит к увеличению теоретической пиковой производительности, которая зачастую не может использоваться на 100% для многих алгоритмов.

Эту ситуацию можно проиллюстрировать на примере процессоров Power4 и PowerPC 440 (точки 3 и 4 на рис. 1а), которые выбиваются из обратной пропорциональности времени расчета от пиковой производительности ядра. Однако эти точки ложатся на общую линию, если уменьшить реальное число операций на такт с 4 до 2 для Power4 и с 4 до 1 для PowerPC 440 (и, соответственно, пиковую производительность ядра). В первом случае 4 операции на такт обеспечиваются двумя операциями FMA, во втором случае — двойной операцией FMA (одна операция FMA применяется к двум сегментам данных). Таким образом, в данном случае реальная производительность составляет 50% и 25% от R_{peak} даже

Таблица 3. Сравнительная характеристика некоторых процессоров

Компьютер	Модель процессора	Тактовая частота, ГГц	Число операций с двойной точностью на такт	Пиковая производительность одного ядра, ГФлопс
МВС-10П	Xeon E5-2690	1.2	8	9.6
Ломоносов	Xeon X5570	2.93	4	11.73
Ross	DEC Alpha	0.5	2	1
BlueGene/L	PowerPC 440	0.7	4	2.8
Cheetah	Power4	1.3	4	5.2
Настольный	Xeon X5690	3.47	4	13.88

на одном ядре. Аналогичный результат справедлив и для ускорителя Intel Xeon Phi (точка 6 на рис. 1а), теоретически можно выполнять операции FMA над векторами из 8 элементов (16 операций на такт), при этом время расчета ложится на общую зависимость при уменьшении числа операций на такт с 16 до 1.

В части б) рисунка 1 показано сравнение времен расчета на кластере «Ломоносов» с использованием различных модулей LAMMPS. Аналогичное исследование для видеокарты NVIDIA Tesla K40 показано в части в) рисунка 1. Программная реализация алгоритмов на GPU сильно влияет на скорость расчета, для одних и тех же данных она может отличаться в несколько раз. Например, модуль «USER-CUDA» хорошо оптимизирован для запусков задач на одной видеокарте, но увеличение их числа не дает соответствующего увеличения скорости расчета. Несмотря на указанный разброс, во всех случаях их производительность существенно уступает производительности CPU в метрике R_{peak}. При этом использование видеокарт требует больших временных затрат на оптимизацию кода. Аналогичный вывод справедлив и при запуске задач на всех ядрах ускорителя Intel Xeon Phi SE10X в «родном» режиме, данные приведены в части г) рисунка 1. Аналогично одноядерному случаю сопроцессоры проигрывают обычным CPU по эффективности использования R_{peak} (к тому же выводу приходят авторы работы [6]). Таким образом, из одинаковых значений R_{peak} не следует одинаковое время решения задач. На это влияют два фактора, упоминавшихся ранее. Во-первых, в алгоритмах, не требующих операций FMA, фактически невозможно использовать более половины заявленной величины R_{peak}. Во-вторых, рост вычислительной мощности современных ускорителей не сопровождается соответствующим ростом каналов передачи данных, что может приводить к простою из-за ожидания новых данных для дальнейших вычислений.

4.2. Белок в водной системе

Аналогичное сравнение суперкомпьютеров по эффективности использования пиковой вычислительной мощности R_{peak} сделано для задачи расчета свойств белка в водной системе. В сравнение по данной метрике включены и специализированные архитектуры типа ANTON.

Суперкомпьютер ANTON-2 основан на интегральной схеме специального назначения (ASIC), разработанной для численной реализации алгоритма классической молекулярной динамики в моделях с межчастичными потенциалами, ориентированными на биомолекулярные задачи. Суперкомпьютер содержит 512 узлов, объединенных специализированным

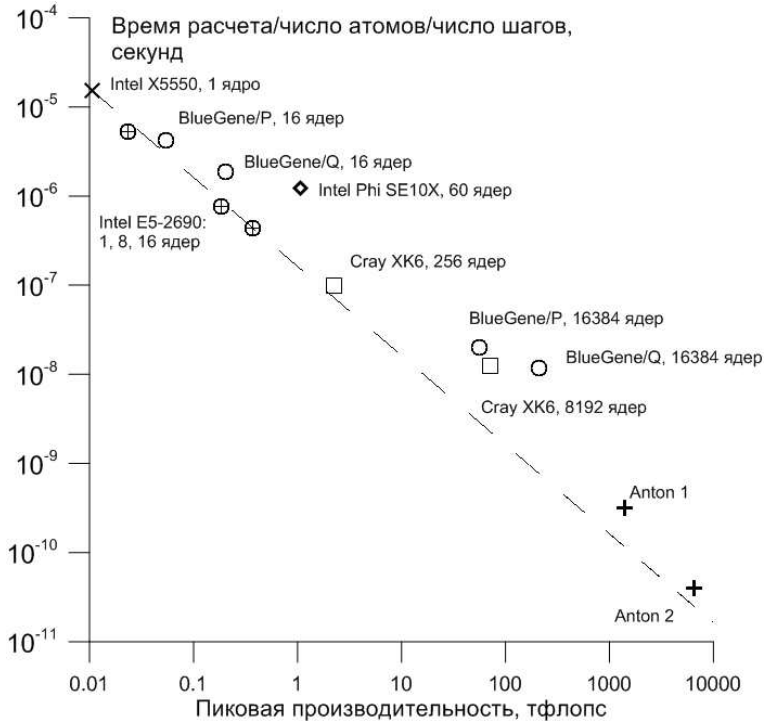

Рис. 2. Сравнение быстродействия теста ApoA1 на классической и специализированной архитектуре. Символы $\oplus$ и $\diamond$ показывают производительность ПО Gromacs на суперкомпьютере МВС-10П, остальные — ПО NAMD на различных суперкомпьютерах.

интерконнектом с топологией 3-х мерного тора. При создании соответствующих специализированных микропроцессоров ASIC и интерконнекта разработчики преследовали цель реализации максимально возможного уровня параллелизации молекулярно-динамического алгоритма [4].

На рисунке 2 сравниваются расчетные времена для стандартного теста ApoA1 (белок аполипротеин A-1, 92244 атома, 500 и более шагов). Приведены данные тестов ПО NAMD на суперкомпьютерах Cray XK6 [27], IBM BlueGene/P и BlueGene/Q [28], специализированной платформы ANTON первого и второго поколения (512 узлов) [4] и ПО Gromacs на суперкомпьютере МВС-10П. Несмотря на различную архитектуру, число ядер и программную реализацию молекулярно-динамического кода, данные хорошо согласуются между собой. Однако на суперкомпьютерах IBM BlueGene хорошо заметна вышеуказанная проблема с невозможностью полноценного использования операций FMA. Также на суперкомпьютерах IBM BlueGene и Cray при использовании большого числа ядер заметно увеличение времени расчета из-за обмена данными между узлами.

Полученный график показывает универсальность числа R_{peak} для сравнения различных архитектур, а также показывают перспективность узкоспециализированных суперкомпьютеров (типа ANTON-2) для задач классической молекулярной динамики.

4.3. Квантовый МД-расчет молекул воды

Метрика «время решения — пиковая производительность» пригодна и при рассмотрении квантовых МД-задач. Для сравнения различных суперкомпьютеров использовался стандартный тест воды из пакета программ CP2K. На рисунке 3 приведены времена расчета для различного числа молекул воды на суперкомпьютерах Cray XT3 и XT5, IBM BlueGene/P и К-100 ИПМ РАН.

Для одного узла производительность в секундах на 1 шаг довольна близка для различных систем. Роль интерконнекта становится очевидной при использовании нескольких

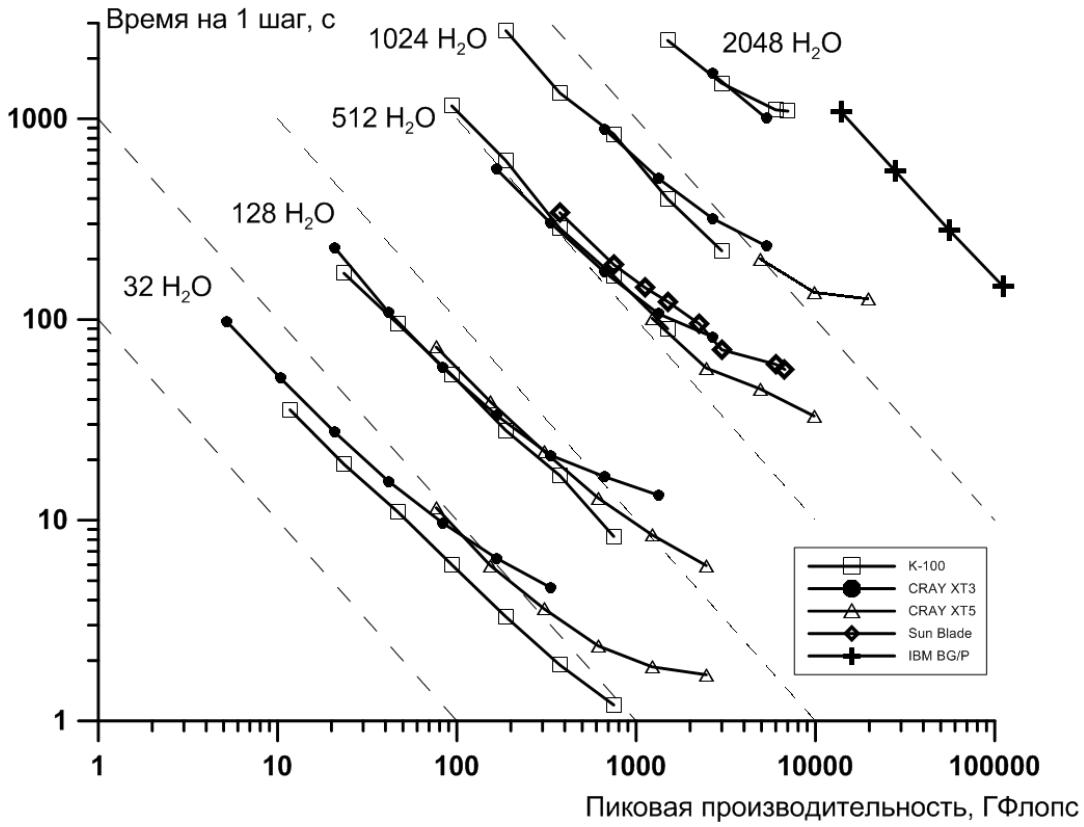

Рис. 3. Квантовый МД-расчет молекул воды (от 32 до 2048) с использованием пакета CP2K на различных суперкомпьютерах

узлов, когда происходит падение производительности из-за интенсивного обмена данными. Для суперкомпьютера IBM BlueGene/P также заметна потеря значительной доли пиковой производительности из-за операций FMA.

5. Выводы

Введена очевидная, но ранее не используемая метрика для ранжирования высокопроизводительных вычислительных систем «время решения — пиковая производительность (в единицах флопс)». Данная метрика использована для сравнения гибридных, специализированных и других систем на примерах моделей классической молекулярной динамики. Наилучшую эффективность в данной метрике на модели леннард-джонсовской жидкости показывают процессоры Intel Xeon с использованием алгоритмов с ручной векторизацией из пакета «USER-INTEL» в LAMMPS. Эффективность аппаратного обеспечения NVIDIA GPU повышается в моделях последнего поколения с использованием новейших программных решений (пакета «KOKKOS» в LAMMPS), однако на сегодняшний день значительно проигрывает эффективности Intel Xeon. Таким образом, при одинаковой вычислительной мощности наименьшее время расчета достигается на классических процессорах, хотя они проигрывают NVIDIA GPU по энергопотреблению и стоимости. Ускорители Intel Xeon Phi на текущий момент не эффективны для ускорения молекулярно-динамических расчетов и не оправдывают своей стоимости.

Результаты тестов показывают, что предложенная метрика выявляет аппаратное обеспечение, оптимизированное под тесты типа LINPACK и являющееся существенно менее эффективным для приложений иного типа. В частности, подобный результат был ожидаем для архитектур NVIDIA GPU и Intel Xeon Phi, однако и для процессоров IBM Power оказалось, что максимальная эффективность операций с плавающей точкой для задач молекулярной

динамики соответствует лишь половине декларируемой пиковой производительности.

Показано, что архитектура таких специализированных решений, как ANTON и ANTON-2, обеспечивает близкое к 100% использование полной пиковой производительность этих суперкомпьютеров даже на молекулярно-динамических задачах малого размера.

Литература

1. Список 500 лучших суперкомпьютеров. URL: http://top500.org (дата обращения: 16.04.2015).

2. High-Performance Computing Using FPGAs / Ed. Vanderbauwhede W., Benkrid K. —New York: Springer Verlag, 2013, 803 p.

3. Ohmura I., Morimoto G., Ohno Y., Hasegawa A.,Taiji M. MDGRAPE-4: a special-purpose computer system for molecular dynamics simulations // Phil Trans R Soc A, 2014, vol. 372, p. 20130387.

4. Shaw D.E. et al. Anton 2: Raising the Bar for Performance and Programmability in a Special-Purpose Molecular Dynamics Supercomputer // SC14 Int. Conf. High Perform. Comput. Networking, Storage Anal. —Piscataway:IEEE Press, 2014, p. 41–53.

5. Подрыга В.О., Поляков С.В. Молекулярно-динамическое моделирование установления термодинамического равновесия в никеле // Математическое моделирование, 2015, т. 27, № 3, с. 3–19.

6. Подрыга В.О., Поляков С.В., Пузырьков Д.В. Суперкомпьютерное молекулярное моделирование термодинамического равновесия в микросистемах газ-металл // Вычислительные методы и программирование, 2015, т. 16 , №. 1, с. 123–138.

7. Eckhardt W. et al. 591 TFLOPS multi-trillion particles simulation on SuperMUC // Supercomputing, vol. 7905, 2013, 473p.

8. Стегайлов В.В., Норман Г.Э. Проблемы развития суперкомпьютерной отрасли в России?: взгляд пользователя высокопроизводительных систем // Программные системы: теория и приложения, 2014, т. 1, № 19, с. 111–152.

9. Куксин А.Ю., Ланкин А.В., Морозов И.В., Норман Г.Э., Орехов Н.Д., Писарев В.В., Смирнов Г.С., Стариков С.В., Стегайлов В.В., Тимофеев А.В. ЗАЧЕМ и КАКИЕ нужны суперкомпьютеры эксафлопсного класса?? Предсказательное моделирование свойств и многомасштабных процессов в материаловедении // Программные системы: теория и приложения, 2014, т. 1, № 19, с. 191–244.

10. Пестряев Е.М. Тестирование многоядерных графических процессоров на алгоритме молекулярной динамики // Математическое моделирование, 2014, т. 26, №1, с. 1–12.

11. Reid F., Bethune I. Optimising CP2K for the Intel Xeon Phi. URL: http://www.prace-ri.eu/IMG/pdf/wp140.pdf (дата обращения: 16.04.2015).

12. Jeong H. et al. Performance of Kepler GTX Titan GPUs and Xeon Phi System. URL: http://arxiv.org/abs/1311.0590 (дата обращения: 16.04.2015).

13. Chan E.Y.K. Benchmarks for Intel MIC Architecture. URL: http://www.clustertech.com/wp-content/uploads/2014/01/MICBenchmark.pdf (дата обращения: 16.04.2015).

14. URL: http://www.nvidia.ru/object/gpu-computing-facts-ru.html (дата обращения: 16.04.2015).

15. Pennycook S.J. et al. Exploring SIMD for molecular dynamics, using Intel Xeon processors and Intel Xeon Phi coprocessors // IPDPS-13 Proceedings, 2013, 1338 c.

16. Plimpton S. Fast Parallel Algorithms for Short-Range Molecular Dynamics // J. Comput. Phys., 1995, vol. 117, № 1, p. 1–19.

17. Pronk, S. et al. GROMACS 4.5: a highthroughput and highly parallel open source molecular simulation toolkit. // Bioinformatics, 2013, vol. 29, № 7, 845–854

18. Phillips J.C. et al. Scalable molecular dynamics with NAMD // J. Comput. Chem. 2005. Vol. 26, № 16. P. 1781–1802.

19. Brown W.M. et al. Implementing molecular dynamics on hybrid high performance computers–short range forces // Comput. Phys. Commun, 2011, vol. 182, № 4, p. 898–911.

20. Carter Edwards H., Trott C.R., Sunderland D. Kokkos: Enabling manycore performance portability through polymorphic memory access patterns // J. Parallel Distrib. Comput., 2013, vol. 74, № 12, p. 3202–3216.

21. Morozov I.V., Kazennov A.M., Bystryi R.G., Norman G.E., Pisarev V.V., Stegailov V.V. Molecular dynamics simulations of the relaxation processes in the condensed matter on GPUs // Comput. Phys. Commun., 2011, vol. 182, № 9, p. 1974–1978.

22. Brown W.M., Kohlmeyer A., Plimpton S.J., Tharrington A.N. Implementing molecular dynamics on hybrid high performance computers – Particle–particle particle-mesh // Comput. Phys. Commun., 2012, vol. 183, № 3, p. 449–459.

23. Brown W.M., Yamada M. Implementing molecular dynamics on hybrid high performance computers - Three-body potentials // Comput. Phys. Commun., 2013, vol. 184, № 12, p. 2785–2793.

24. Begau C., Sutmann G. Adaptive dynamic load-balancing with irregular domain decomposition for particle simulations // Comput. Phys. Commun, 2015, vol. 190, p. 51–61.

25. Воеводин Вл.В., Жуматий С.А., Соболев С.И., Антонов А.С., Брызгалов П.А., Никитенко Д.А., Стефанов К.С., Воеводин Вад.В. Практика суперкомпьютера “Ломоносов” // Открытые системы. 2012. т. 7, с. 36–39.

26. URL: http://lammps.sandia.gov/bench/lj_one.html (дата обращения: 16.04.2015).

27. Sun Y., Zheng G., Mei C., Bohm E.J., Phillips J.C., Kale L. V., Jones T.R. // In proc. Int. Conf. High Perform. Comput. Networking, Storage Anal, 2012.

28. Kumar S., Sun Y., Kale L. V. Acceleration of an asynchronous message driven programming paradigm on IBM Blue Gene/Q // IPDPS-13 Proceedings, 2013, 1338 p.