ФИЗИКА ЭЛЕМЕНТАРНЫХ ЧАСТИЦ И АТОМНОГО ЯДРА
2024. Т. 55, вып. 3. С. 615–618

ВЫСОКОПРОИЗВОДИТЕЛЬНЫЕ ВЫЧИСЛЕНИЯ НА МАТЕМАТИЧЕСКИХ СОПРОЦЕССОРАХ И ГРАФИЧЕСКИХ УСКОРИТЕЛЯХ С ИСПОЛЬЗОВАНИЕМ PYTHON

*С. В. Борзунов *, А. В. Романов **, С. Д. Кургалин ***, К. О. Петрищев *****

Воронежский государственный университет, Воронеж, Россия

Рассмотрено использование модулей языка программирования Python для решения ресурсоемких задач на высокопроизводительном компьютерном кластере. Показано применение кластера для решения наиболее сложных задач в области ядерной физики. На примере перемножения вещественных матриц продемонстрированы методы высокопроизводительных вычислений на базе математических сопроцессоров и графических ускорителей. Использование алгоритмического языка Python в высокопроизводительных вычислениях во многих случаях создает значительные удобства для разработки, отладки и тестирования программного кода.

This paper considers the use of Python modules for solving resource-intensive tasks on a high-performance computer cluster. It shows the application of the cluster for solving the most complex problems in the field of nuclear physics. It also demonstrates methods of high-performance computing based on mathematical coprocessors and graphics accelerators on the example of multiplication of real matrices. The use of the Python algorithmic language in high-performance computing in many cases significantly expands the convenience of creating, debugging and testing program code.

PACS: 07.05.Tp; 02.60.–x

Многие современные вычислительные комплексы, входящие в список топ-500 ведущих суперкомпьютеров мира, собраны по гибридной схеме. Они имеют в своем составе не только универсальные процессоры, но и энергоэффективные математические сопроцессоры, такие как Intel Xeon Phi или Nvidia Tesla. Подобные машины являются примером устройств, созданных специально для выполнения массивных параллельных вычис-

* E-mail: sborzunov@gmail.com
** E-mail: alphard.rm@gmail.com
*** E-mail: kurgalin@bk.ru
**** E-mail: vrn.kostyan.p@mail.ru

лений. Например, сопроцессор Intel Xeon Phi предоставляет возможность производить вычисления с использованием до 240 логических ядер, а математический сопроцессор Nvidia Tesla A100 включает в себя 6912 потоковых ядер CUDA. У таких суперкомпьютеров достаточно сложная архитектура взаимодействия между процессором и сопроцессором, которая сильно усложняет процесс подготовки и поддержки программного кода. Программирование подобных систем с использованием полного функционала средств распараллеливания вычислений имеет свои особенности, поскольку современные GPU в отличие от центральных процессоров представляют собой массивно-параллельные вычислительные устройства с относительно большим количеством вычислительных ядер и иерархически организованной собственной памятью.

Суперкомпьютерный центр Воронежского государственного университета [1] имеет в своем составе высокопроизводительный вычислительный кластер, состоящий из 10 узлов. Семь узлов кластера содержат по два ускорителя Intel Xeon Phi, а три остальных узла — по два ускорителя Nvidia Tesla. Узлы кластера объединены сетью InfiniBand.

Сразу же после создания в 2002 г. вычислительный кластер начал применяться для решения наиболее сложных задач в области ядерной физики. Так, расчет эффективных чисел W [2] нуклонных кластеров ^{8}Be и ^{16}O в ядрах, связанный с 20-кратным суммированием, существенно ускоряется при использовании параллельных алгоритмов. Получение величин W нередко требует решения уравнения Шредингера для разных, часто весьма сложных, видов ядерных взаимодействий. При этом широко применяется метод Нумерова [3]. Расчет потенциала взаимодействия V, входящего в уравнение и определяющего поведение ядерной системы, при использовании «фолдинг-потенциалов» [2] приводит к многократному интегрированию по нескольким переменным и, следовательно, к большим затратам машинного времени. Поэтому разработанный алгоритм метода Нумерова с распараллеливанием, как и алгоритмы эффективного параллельного суммирования, применяется для получения вероятностей кластерных распадов [2], спектров внутреннего тормозного излучения при α-, кластерном и протонном распадах [4], при оценке эффекта кваркового усиления «жестких» процессов с вылетом дейтрона [5], при анализе свойств α-частичных решений многонуклонной задачи [6] и т. д.

В связи с тем, что в узлах вычислительного кластера находятся ускорители вышеуказанных типов, актуальной является задача обеспечения их эффективного использования.

Согласно современным тенденциям разработки программных инструментов более высокого уровня для облегчения программирования сложных вычислительных систем созданы специальные модули языка Python, которые позволяют упростить методы работы с сопроцессором. В качестве примеров таких расширений укажем модули PyMIC и PyCUDA [7]. Принцип действия у них совпадает, он заключается в предоставлении интерфейса к основным операциям процессор/сопроцессор. Разумеется,

реализация перемножения вещественных матриц на языке Python не может похвалиться высокой скоростью вычислений, поэтому математические операции подобного рода выполняются, как правило, средствами сторонних библиотек, чаще всего написанных на языках C или Fortran [8]. Примером этого могут служить вычисления, производимые с помощью пакета NumPy. В этом случае в качестве сторонних библиотек используются подпрограммы, реализованные на языке Fortran. Соответственно, первым шагом к переносу вычислений на сопроцессор будет реализация пользовательской библиотеки на компилируемом языке и ее интеграция в код на языке Python.

Наиболее доступными с точки зрения простоты программирования являются вычисления с использованием математического сопроцессора Intel Xeon Phi. Это объясняется сходством архитектуры сопроцессора с универсальными процессорами фирм Intel и AMD, что позволяет использовать уже известные приемы программирования. Работу с таким сопроцессором легко продемонстрировать на примере задачи перемножения двух вещественных матриц. Авторами был разработан вариант исходного кода пользовательской библиотеки для интеграции с модулем PyMIC (библиотека libmult.so). Отметим, что в этом случае код практически не отличается от стандартной реализации перемножения матриц на языке C.

Эта библиотека выполняет расчеты на сопроцессоре Intel Xeon Phi, получая данные из программы на языке Python, и возвращает результат вычисления в среду Python. Код программы численных расчетов с вызовом библиотеки libmult.so для генерации матриц использует средства пакета NumPy.

С учетом использования для вычислений до 240 потоков (одно физическое ядро зарезервировано под нужды операционной системы) сопроцессор Intel Xeon Phi предоставляет широкие возможности при исследовании вопросов параллельных вычислений без необходимости глубокого понимания его архитектуры и специальных диалектов языка C.

Подготовка программного кода, ориентированного на работу с GPU фирмы Nvidia, значительно сложнее. Основная трудность написания кода для сопроцессора Nvidia Tesla объясняется особенностями работы потоковых ядер CUDA.

Программа перемножения двух матриц, написанная для сопроцессора Nvidia Tesla с применением модуля PyCUDA, требует большего времени для подготовки и отладки. Как и в случае кода для Intel Xeon Phi, программа состоит из инициализации данных, их пересылки на сопроцессор и обратно, а также из ядра, которое осуществляет вычисления. Однако математическое ядро здесь сильно отличается от аналога для сопроцессора Intel из-за разницы архитектур.

Использование алгоритмического языка Python в высокопроизводительных вычислениях во многих случаях создает значительные удобства для разработки, отладки и тестирования программного кода. Более того, для обучения специалистов работе на суперкомпьютерах целесообразно

начинать обучение с более простых в программировании сопроцессоров фирмы Intel с переходом на сопроцессоры Nvidia, которые стали сегодня признанным стандартом в отрасли. В итоге применение операций верхнего уровня модулей PyMIC и PyCUDA позволяет существенно упростить работу с суперкомпьютером, предоставить бесшовную интеграцию с инструментами популярного языка программирования Python.

СПИСОК ЛИТЕРАТУРЫ

1. *Kurgalin S. D., Borzunov S. V.* Using the Resources of the Supercomputer Center of Voronezh State University in Learning Processes and Scientific Researches // Тр. междунар. конф. «Суперкомпьютерные дни в России», Москва, 24–25 сент. 2018 г. М.: Изд-во Моск. ун-та, 2018. С. 972–977.
2. *Кадменский С. Г., Кургалин С. Д., Чувильский Ю. М.* Кластерные состояния атомных ядер и процессы кластерного распада // ЭЧАЯ. 2007. Т. 38, вып. 6. С. 1333–1412.
3. *Hamming R. W.* Numerical Methods for Scientists and Engineers. 2nd ed. New York: Dover Publ., 1987. 752 p.
4. *Кургалин С. Д., Чувильский Ю. М., Чуракова Т. А.* Внутреннее тормозное излучение сильновзаимодействующих заряженных частиц // ЯФ. 2016. Т. 79, № 6. С. 635–642.
5. *Кургалин С. Д., Чувильский Ю. М.* Распределения 6q-флуктонов в ядрах и кварковое усиление жестких процессов с вылетом дейтрона // ЯФ. 1989. Т. 49, вып. 1. С. 126–134.
6. *Гнилозуб И. А., Кургалин С. Д., Чувильский Ю. М.* Свойства альфа-частичных решений многонуклонной задачи // ЯФ. 2006. Т. 69, № 6. С. 1043–1059.
7. Pycuda 2022.2.2 Documentation. https://documen.tician.de/pycuda/ (accessed 12.05.2023).
8. *Kurgalin S., Borzunov S.* A Practical Approach to High-Performance Computing. Springer Intern. Publ., 2019. 206 p.