АНАЛИЗ ЭФФЕКТИВНОСТИ ГИБРИДНОГО ПАРАЛЛЕЛЬНОГО ПРОГРАММИРОВАНИЯ НА ПРИМЕРЕ СИСТЕМЫ BLUE GENE/P

Е.А. Глазкова, Н.Н. Попова

Современные компьютеры для высокопроизводительных вычислений, как правило, многоядерные: каждый узел в них представляет собой систему с разделяемой памятью. Скорее всего, именно многоядерность останется основной тенденцией в развитии процессоров в ближайшее время. Использование многоядерных архитектур и SMP-кластеров в высокопроизводительных вычислениях требует разработки для них эффективных методов программирования [1]. Одним из подходов к созданию параллельных программ для таких архитектур является гибридный (MPI + OpenMP) подход.

Гибридный подход предполагает, что алгоритм разбивается на параллельные процессы, при этом каждый процесс включает в себя несколько легковесных процессов – нитей. Таким образом, гибридный подход включает в себя два уровня параллелизма: параллелизм между подзадачами и параллелизм внутри подзадачи. В этой модели программирования MPI используется для организации взаимодействий между узлами (процессами), а OpenMP для многонитевого программирования внутри узла.

Как отмечалось в работах [1], [2], [3], за счет укрупнения MPI-процессов и уменьшения их числа гибридная модель может частично устранить некоторые недостатки использования MPI, такие как большие накладные расходы на передачу сообщений и слабая масштабируемость приложения с возрастанием числа MPI-процессов. Но, несмотря на свои потенциальные достоинства, гибридная модель имеет целый ряд трудностей. Одним из проблемных мест является обращение OpenMP-нитей к общей памяти, которое часто требует критических секций и атомарных операций [4]. Достаточно часто использование гибридной модели программирования приводит, наоборот, к уменьшению эффективности [1].

Для того чтобы гибридные программы выполнялись эффективно, необходимо понимать, какие параметры влияют на время выполнения и возможность масштабируемости. В системе Blue Gene/P имеется ряд параметров настройки (Рис.1), которые могут влиять на производительность системы и значениями которых пользователь может управлять. Такими параметрами являются режим выполнения, определяющий соотношение числа MPI-процессов и OpenMP-нитей на вычислительном узле, и mapping – способ назначения MPI-процессов на процессоры системы. Настраиваемые параметры присутствуют и в самой тестовой задаче (способ распределения данных и виртуальная топология).

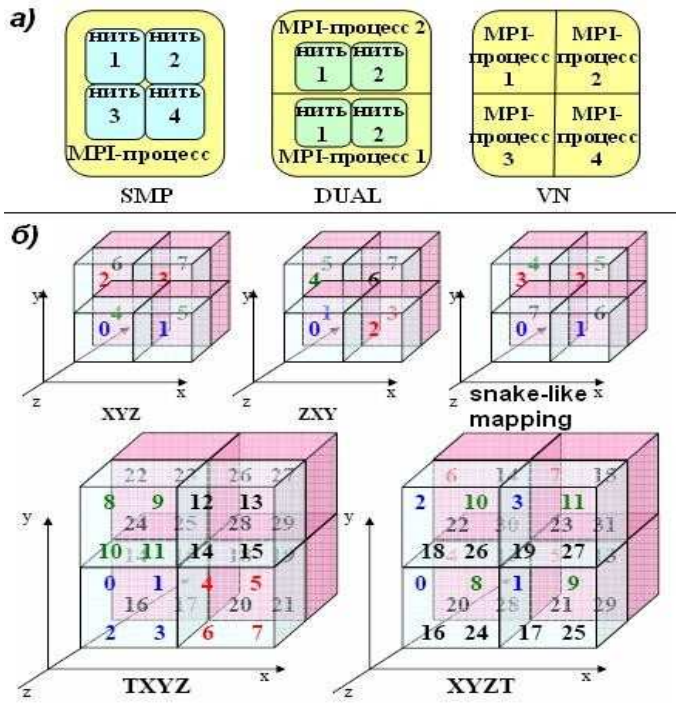

Рис.1. Параметры системы Blue Gene/P: а)режимы выполнения программы(SMP, DUAL, VN) б)mapping. Вверху – примеры для режима SMP, внизу – для режимов VN и DUAL

Целью работы является разработка методов настройки эффективности гибридной параллельной программы на примере модельной задачи (решения разностной задачи Дирихле методом Якоби в трехмерной области (параллелепипед)). Предполагается, что исследование даст возможность определить набор параметров, наиболее сильно влияющих на производительность параллельной программы и возможность ее масштабируемости, и выработать рекомендации для пользователей по выбору значений этих параметров.

В качестве целевой архитектуры, на которой проводится исследование, используется система Blue Gene/P. Базовыми элементами системы Blue Gene/P являются вычислительные узлы, состоящие из четырех ядер PowerPC 450, работающих над общей оперативной памятью размером 2Гб. Вычислительные узлы соединены между собой несколькими сетями, в том числе 3D-тором (сетью для взаимодействий точка-точка), сетью для коллективных операций и сетью для барьерной синхронизации [5]. Система Blue Gene/P имеет иерархическую структуру, для которой гибридное программирование потенциально может дать выигрыш в производительности. Модельная задача имеет виртуальную MPI-топологию решетки, которая должна хорошо отображаться на физическую топологию процессоров системы Blue Gene/P.

В ходе работы был проведен ряд экспериментов на системе Blue Gene/P, чтобы исследовать время работы тестовой программы при различных значениях перечисленных параметров.

Первая серия экспериментов проводилась на минимальном доступном в системе размере партиции - 128 вычислительных узлов (512 ядер), и при размере задачи 2304*1280*2560 элементов типа double. Такой размер задачи был выбран, как компромисс между необходимостью максимально использовать память вычислительного узла и желанием иметь в качестве размера задачи числа, делящиеся нацело на степени двойки, что позволило бы разделить данные между узлами блоками одинакового размера. Первая серия экспериментов включала в себя сначала исследование и анализ времени выполнения программы при распределении данных полосами, а затем аналогичный эксперимент для распределения данных блоками. Для параметров системы и задачи, соответствующих наиболее характерным результатам экспериментов, был проведен более глубокий анализ с использованием библиотеки для профилирования MPI-приложений mpiP [6].

Вторая серия экспериментов состояла в исследовании влияния некоторых избранных параметров на масштабируемость приложения. Под масштабируемостью понималась способность приложения при увеличении размеров задачи, пропорциональном увеличению числа процессоров, сохранять прежнее время выполнения.

Результаты первой серии экспериментов показали, что исследуемые параметры (способ распределения процессов по процессорам, виртуальная топология MPI и режим выполнения программы в системе Blue Gene/P) достаточно сильно влияют на время выполнения программы. Разница между худшим временем (67.6854 сек), полученном при режиме выполнения VN, случайном назначении процессов на процессоры и виртуальной топологии 1*1*512, и лучшим временем выполнения программы (34.4916 сек), полученном при VN режиме, порядке TYXZ назначения процессов на процессоры и виртуальной топологии 8*4*16, составляет примерно 1.96 раза.

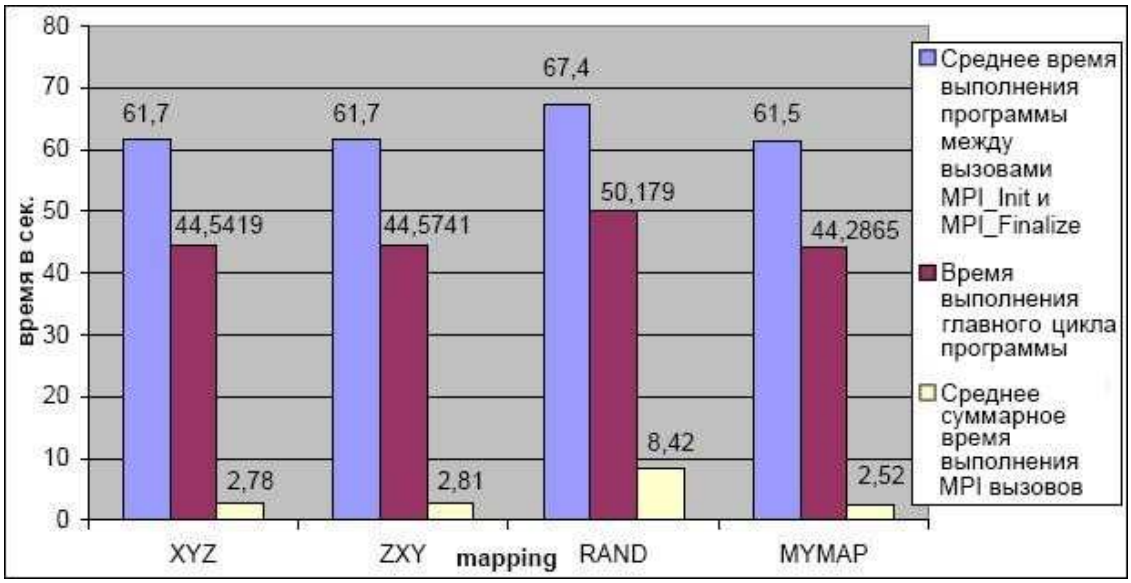

Рис.2. Среднее время выполнения программы при различных способах назначения процессов на процессоры в режиме SMP на 128 вычислительных узлах при виртуальной топологии 1*1*128.

Использование гибридных (DUAL и SMP) режимов выполнения при счете модельной задачи было эффективнее по времени, чем использование режима VN, только при распределении данных полосами, которое требовало передачи очень больших по объему сообщений между MPI-процессами. При блочном распределении

данных объем коммуникаций между MPI-процессами был существенно меньше и гибридные (DUAL, SMP) режимы выполнения проигрывали режиму VN, использующему только MPI для коммуникаций.

Для режима SMP и относительно небольшого размера партиции (128 выч. узлов) способ назначения процессов на процессоры еще не оказывает значительного влияния на время выполнения программы, только случайный mapping сильно отличается от других в худшую сторону. Суммарное время выполнения MPI-вызовов при случайном назначении процессов на процессоры примерно в 3.34 раза хуже, чем для «змеевидного» способа назначения (Рис.2).

В режиме DUAL(Рис.3) и VN(Рис.4) mapping оказывает уже гораздо большее влияние на время выполнения программы. При использовании стандартных способов назначения процессов на процессоры порядок координат X,Y,Z в перестановке при фиксированном положении T практически не влияет на время выполнения программы. Разница составляет примерно 0.3%, что сравнимо с различием времен при нескольких выполнениях программы с одним значением параметров. Само же положение координаты T в перестановке (первым или последним) достаточно существенно влияет на время выполнения программы: разница составляет примерно 3,5% в DUAL режиме и 13% в VN режиме.

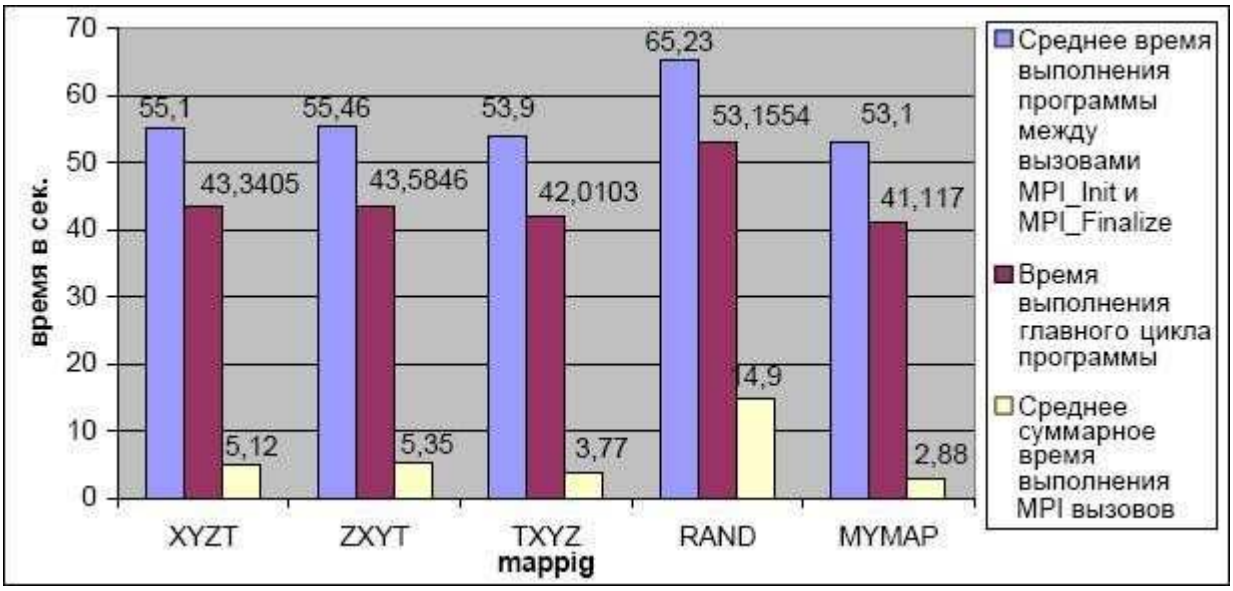

Рис.3. Среднее время выполнения программы при различных способах назначения процессов на процессоры в режиме DUAL на 128 вычислительных узлах при виртуальной топологии 1*1*256.

Можно сделать вывод, что влияние способа назначения процессов на процессоры возрастает не только с числом вычислительных узлов, но и с числом MPI процессов внутри вычислительного узла.

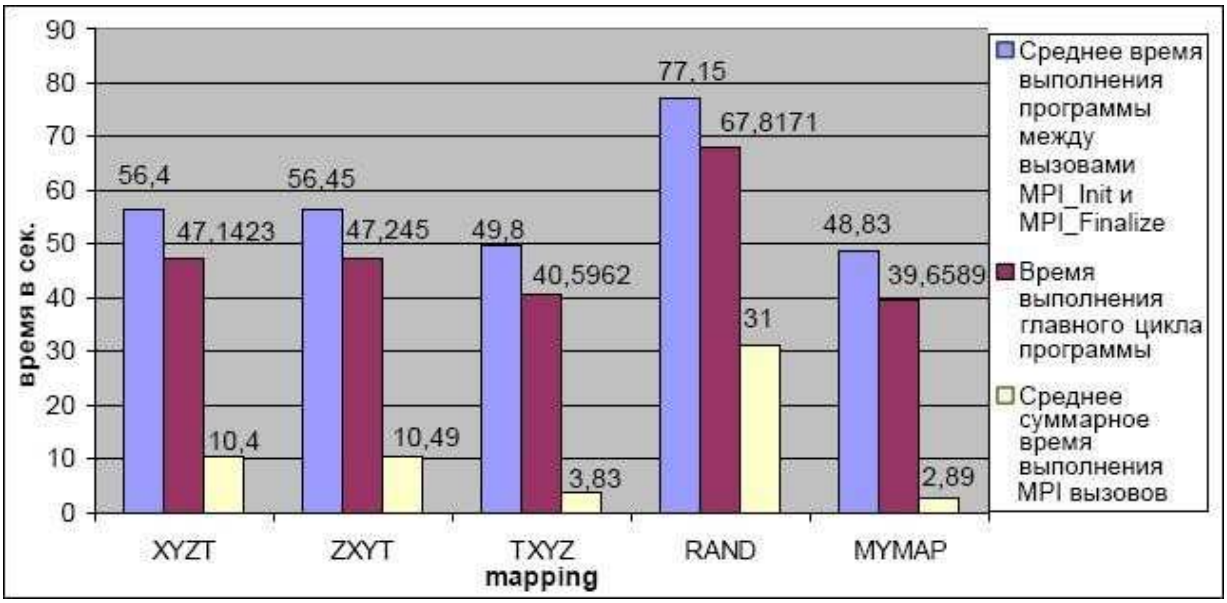

Рис.4. Среднее время выполнения программы при различных способах назначения процессов на процессоры в режиме VN на 128 вычислительных узлах при виртуальной топологии 1*1*512.

Эксперимент по исследованию масштабируемости проводился таким образом, чтобы не изменять размер передаваемых 1 узлом сообщений, то есть исследовалась масштабируемость по вычислениям, а не по коммуникациям (Рис.5).

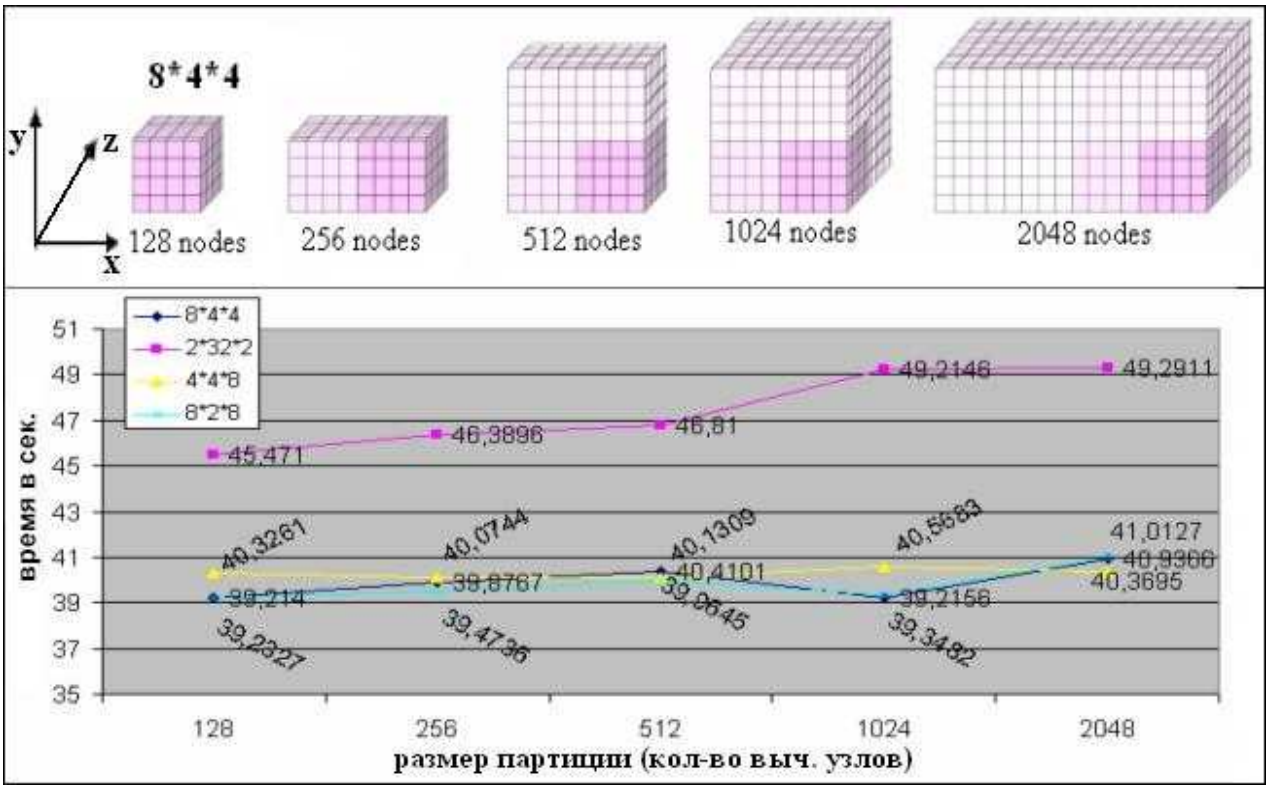

Рис.5. Вверху - организация эксперимента по исследованию масштабируемости (направления увеличения размера задачи и партиции). Внизу - время выполнения главного цикла программы для различных виртуальных топологий и размеров партиции.

При неудачно выбранной виртуальной топологии (2*32*2) сильно возрастает длина пути передачи сообщений при увеличении размера партиции, потому приложение плохо масштабируется. Для виртуальных топологий (8*4*4, 4*4*8, 8*2*8) близких к физической топологии системы наблюдается хорошая масштабируемость. Иногда с увеличением размера партиции происходило даже уменьшение времени выполнения задачи. Скорее всего, это связано с уменьшением нагрузки на сеть, создаваемой другими пользователями, и появлением новых путей передачи сообщений.

ЛИТЕРАТУРА:

1. Piotrowski M. Mixed Mode Programming on Clustered SMP Systems. 2006 [PDF] (http://www2.epcc.ed.ac.uk/msc/dissertations/dissertations-0506/2391976-9i-dissertation1.1.pdf)
2. Бахтин В.А., Коновалов Н.А., Крюков В.А., Поддерюгина Н.В., Сазанов Ю.Л. Разработка параллельных программ для решения больших вычислительных задач на SMP-кластерах. [PDF] (http:// lvk.cs.msu.su/files/mco2003/bahtin.pdf)
3. Makris I. Mixed Mode Programming on Clustered SMP Systems. 2005 [PDF] (http://www2.epcc.ed.ac.uk/msc/dissertations/dissertations-0405/6333284-9f-dissertation1.2.pdf)
4. He Y., Ding C. Hybrid OpenMP and MPI Programming and Tuning. 2004 [PPT] (http://www.nersc.gov/nusers/services/training/classes/NUG/Jun04/NUG2004_yhe_hybrid.ppt)
5. IBM System Blue Gene Solution: Blue Gene/P Application Development. 2008 [PDF] (http://www.redbooks.ibm.com/redbooks/pdfs/sg247179.pdf)
6. mpiP: Lightweight, Scalable MPI Profiling [HTML] (http://mpip.sourceforge.net)