ТЕСТИРОВАНИЕ ВЫСОКОПРОИЗВОДИТЕЛЬНОГО ПАРАЛЛЕЛЬНОГО КОМПЬЮТЕРНОГО КЛАСТЕРА

С.Д. Кургалин, А.Н. Кривцов, Е.Н. Паршина
Воронежский государственный университет

В Воронежском государственном университете с 2002 г. работает 20-процессорный параллельный компьютерный кластер с пиковой производительностью 28 Гфлопс [1]. Узлы кластера первого уровня (12 процессоров в составе одного четырехпроцессорного и четырех двухпроцессорных узлов) связаны по протоколу Gigabit Ethernet, для связи остальных восьми однопроцессорных узлов используется протокол Fast Ethernet. Связь обеспечивается коммутаторами Rapier G6f и Rapier 24 фирмы Allied Telesyn.

Одним из важных этапов включения компьютерного кластера в научные исследования вуза и учебный процесс является проведение его тестирования. Тестирование позволяет оценить возможности его использования как при решении научных проблем, так и в образовательных целях. При создании тестирующих программ и последующем анализе получаемых результатов происходит детальное освоение системы и проводится работа по повышению ее эффективности, постоянно продолжающаяся во время эксплуатации. Тестирование также дает возможность обеспечить использование всех преимуществ применения кластера в задачах математического и компьютерного моделирования.

Тестирование кластера проводилось в следующих направлениях: а) тестирование *уровня программного и аппаратного обеспечения* - оценка эффективности работы кластера в течение определенного периода времени *(нагрузочное тестирование)*; б) тестирование *базового коммуникационного уровня* - определение параметров среды взаимодействия параллельных процессов, эффективности выполнения основных коммуникационных процедур и примитивов синхронизации. Оно включает *определение латентности* и *скорости передачи данных* по коммуникационной сети кластера в различных режимах, а также эффективности работы конструкций MPI (Message Passing Interface) [2]; в) тестирование *коммуникационного уровня приложений* - оценка эффективности отображения различных логических топологий процессов на коммуникационную среду кластера.

Нагрузочное тестирование. В любой компьютерной системе периодически возникают сбои, связанные с работой программного обеспечения или оборудования. Для оценки надежности компьютерного кластера применяется следующий тест: запускается прием от всех процессов целочисленного массива заданной длины; всем процессам рассылается такой же массив, заполненный счетчиком, сдвинутым на собственный номер; завершается прием от всех процессов и проверяется счетчик. Счетчик, присланный от каждого процесса, должен быть сдвинут на номер отправителя. Выдаются сообщения о числе несовпадений значений счетчиков. Если в результате тестирования получаются нулевые значения, то считается, что система функционирует корректно.

Определение латентности и пропускной способности. Пропускная способность R сети - это количество информации, передаваемой между узлами сети в единицу времени. Реальная пропускная способность снижается за счет передачи служебной информации. Латентностью (задержкой) S называется время, затрачиваемое программным обеспечением и устройствами сети на подготовку к передаче информации по данному каналу. Под латентностью можно понимать время, необходимое для передачи сигнала или сообщения нулевой длины.

Для измерения пропускной способности систем типа "точка-точка" используется следующая методика. Процесс с номером 0 посылает процессу с номером 1 сообщение длиной L байт. Процесс 1, приняв сообщение от процесса 0, посылает ему ответное сообщение той же длины. Используются блокирующие *(blocking)* вызовы MPI (MPI_Send и MPI_Recv) [2]. Эти действия повторяются N раз с целью минимизации погрешности за счет усреднения. Процесс 0 измеряет время T, затраченное на все эти обмены. Пропускная способность R определяется по формуле $R = 2NL / T$.

Пропускная способность двунаправленных обменов определяется аналогичным образом. Используются неблокирующие (non-blocking) вызовы *MPI* (MPI_Isend и MPI_Irecv) [2]. Измеряется время, затрачиваемое процессом 0 на передачу сообщения процессу 1 и прием ответа от него при условии, что процессы начинают передачу информации одновременно после синхронизации.

Измерение эффективности основных операций MPI. Время работы параллельной программы, как и любой другой, определяется не только объемом передаваемых данных, но и эффективностью используемых операций. Для определения эффективности основных операций MPI применяется следующая методика. Определяется время DT на один замер времени: t_1 = MPI_Wtime ( ); t_2 = MPI_Wtime( ); $DT = t_2 - t_1$; и время DB на операцию синхронизации: MPI_Barrier (comm); t_1 = MPI_Wtime ( ); MPI_Barrier (comm); t_2 = MPI_Wtime ( ); $DB = t_2 - t_1 - DT$. Все ос-

тальные замеры времени осуществляются по схеме: производится синхронизация с помощью операции MPI_Barrier; замеряется текущее время t_1; исполняются вызовы MPI, относящиеся к тестируемой конструкции; вновь производится барьерная синхронизация; измеряется текущее время t_2; время, затраченное на исполнение тестируемой конструкции, вычисляется как $t = t_2 - t_1 - DT - DB$. С целью минимизации погрешности эта процедура повторяется несколько раз. Величина t усредняется по всем итерациям.

Определение коэффициента полноты. Полнота сети есть мера того, насколько сильно одновременно происходящие обмены "мешают" друг другу. Достаточно типична ситуация, когда коммутаторы разных моделей, реализующих попарно независимые обмены без потерь быстродействия, сильно различаются по времени выполнения одной и той же тестовой серии частично пересекающихся обменов. Для оценки качества коммутаторов все обмены на одном узле запускаются одновременно, чтобы исключить их зависимость друг от друга по порядку выполнения. Если среднее время выполнения теста равно T, число процессоров N, а X - объем данных, передаваемых каждым процессором каждому процессору, то суммарный объем данных, переданных одним узлом D, будет равен: $D = X(N - 1)$.

Если сеть идеальна, то есть обмен не мешает другому, то этот объем данных может быть передан из узла за время $T_i = D/S = (X(N - 1))/S$. В действительности же этот тест будет выполняться за реальное время T_e. Отношение T_i и T_e дает экспериментальный коэффициент полноты $F_e = T_i / T_e$. Чем ближе F_e к единице, тем более независимо осуществляются одновременные обмены в данной сети.

Исследование эффективности отображения логических топологий процессов на коммуникационную среду. Под логическим каналом подразумевается пара узлов (A, B), которые могут обмениваться сообщениями. В ходе теста по каждому логическому каналу между узлами A и B передается в обе стороны L байт информации. Пусть для данной топологии задействованы $N(P)$ логических каналов, где P - число узлов. Тогда суммарный объем передаваемой по сети информации есть $I = 2LN$. Пусть узел i имеет $N_i(P)$ логических связей с другими узлами. Тогда этот узел передает и принимает всего $2LN_i(P)$ байт информации. Для определения пропускной способности сети производится синхронизация с помощью операции MPI_Barrier; измеряется "начальное" время t_o; инициализируется прием/передача информации "нужному" (в зависимости от топологии) узлу с помощью операций MPI_Send или MPI_Recv; производится барьерная синхронизация; измеряется "конечное" время t_1. Общая пропускная способность сети - отношение количества всей переданной по сети информации к затраченному времени. Средняя пропускная способность одного логического канала - отношение общей пропускной способности к количеству задействованных каналов.

Результаты нагрузочного тестирования кластера показали, что несовпадения значений счетчиков нет, то есть система функционирует без ошибок.

Результаты определения пропускной способности кластера ВГУ представлены на рис. 1. Для процессоров, связанных по протоколу Fast Ethernet, достигается латентность 90 мкс, а скорость пересылки данных составляет 80 Мбит/с для сообщений размером 256 Кбайт. Для процессоров, связанных по протоколу Gigabit Ethernet, достигается латентность 100 мкс, а скорость пересылки составляет 320 Мбит/с для сообщений размером 256 Кбайт.

Для одного узла достигается латентность 95 мкс, а скорость пересылки составляет 360 Мбит/с для сообщений размером 256 Кбайт.

При изменении размера сообщения до 32 Кбайт пропускная способность возрастает до 450 Мбит/с, а затем снижается до 350 Мбит/с. Это связано с тем, что при увеличении размера сообщения увеличивается количество кадров для передачи информации и время на разбиение сообщения, а размер кадра Ethernet при этом не меняется.

Рис. 1. Зависимость пропускной способности кластера ВГУ от размера сообщения

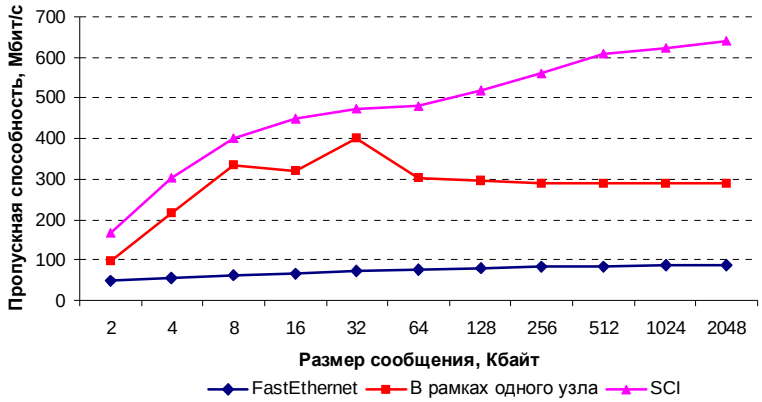

Рис. 2. Пропускная способность SCI-кластера НИВЦ МГУ в зависимости от размера сообщения

На рис. 2 для сравнения приведены аналогичные зависимости для SCI-кластера НИВЦ МГУ. Использование специального коммуникационного оборудования и программного обеспечения позволяет в этом случае обеспечить непрерывный рост пропускной способности кластера в зависимости от размера сообщения.

Измерение времени выполнения основных операций MPI (см. табл. 1) позволяет оценить время работы всей параллельной программы и выбрать те процедуры, которые обеспечат наибольшую производительность. Например, при наличии достаточного объема памяти системы эффективнее создавать переменную для каждого процесса, чем рассылать ее значение с помощью операции MPI_BCAST.

Таблица 1

Время выполнения основных операций MPI

Операции MPI	Время выполнения, мкс
MPI_Wtime	2,2
MPI_Barrier	310
MPI_ALLREDUCE (NONVEC)	340
MPI_ALLREDUCE (VEC)	350
MPI_REDUCE (NONVEC)	770
MPI_REDUCE (VEC)	1200
MPI_BCAST	260
MPI_GATHER	2100
MPI_SEND (NONBLOCKING WITH WAIT)	48
MPI_SEND (BLOCKING)	47
MPI_SENDRECV	100
MPI_SEND & MPI_RECV	1600

Определение коэффициента полноты дает возможность оценить степень независимости одновременных обменов в сети кластера.

Для сети Gigabit Ethernet (коммутатор Rapier G6f) был получен коэффициент полноты F_e = 0,9; для сети Fast Ethernet (коммутатор Rapier 24) F_e =0,5. Это означает, что при использовании протокола Gigabit Ethernet одновременные обмены происходят практически независимо, при использовании протокола Fast Ethernet их взаимозависимость достаточно велика.

Исследование эффективности отображения логических топологий связи процессов на коммуникационную среду позволяет выявить наиболее эффективную топологию для решения конкретной задачи. Результаты определения пропускной способности для различных топологий соединения процессоров кластера представлены в табл. 2, 3. Тестирование проводилось для четырех процессоров. Как видно из результатов, наиболее удачной топологией соединения процессоров является "кольцо".

При тестировании кластера также применяются алгоритмы, разрабатываемые в рамках создаваемой в ВГУ библиотеки параллельных программ.

Для обеспечения использования компьютерного кластера возникает необходимость создания параллельных алгоритмов или "перефразирования" имеющихся последовательных алгоритмов.

Так, при реализации параллельного алгоритма умножения квадратных матриц процессоры средствами MPI объединяются в топологию квадратной решетки размером $k \times k$, где k^2 - общее число используемых процессоров. Для вычисления произведения квадратных матриц $A \times B = C$ размером $n \times n$ каждая из них представляется в блочном виде. Обозначим A_{ij}, B_{ij} и C_{ij} - блоки соответствующих матриц размером $m \times m (m = n/k)$, которые распределяются по k^2 процессорам. На каждом шаге L $(1 \le L \le k)$ выполняются следующие действия: а) для каждой строки i процессорной решетки процессор k_{ij} пере-

Таблица 2

Пропускная способность в сети Fast Ethernet

Топология сети	Средняя пропускная способность, Мбит/с	Общая пропускная способность, Мбит/с
звезда	3,26	9,8
кольцо	3,11	12,45
полный граф	0,42	2,52

Таблица 3

Пропускная способность в сети Gigabit Ethernet

Топология сети	Средняя пропускная способность, Мбит/с	Общая пропускная способность, Мбит/с
звезда	19,49	58,47
кольцо	39,99	159,96
полный граф	9,03	54,21

дает свой блок A_{ij} на все процессоры этой же строки (т.е. строки i), где столбец j вычисляется по формуле [redacted]; б) полученный блок A_{ij} умножается на блок B_{ij} процессора k_{ij} и прибавляется к блоку C_{ij} на процессоре k_{ij}; в) блоки B_{ij} каждого процессора k_{ij} пересылаются процессорам, являющимся соседями сверху в процессорной решетке, причем процессоры первой строки пересылают блоки процессорам последней строки решетки.

Представленный метод умножения матриц является примером распределения данных между процессорами кластера с учетом близости их расположения и топологии соединения.

В параллельном алгоритме приближенного расчета многомерных интегралов и многократных сумм функций, зависящих от нескольких переменных, задается число k, которое показывает, какая часть из общего числа n слагаемых исходной суммы будет учитываться при суммировании ($n = k \cdot m$, где m - число слагаемых, которые учитываются при суммировании). Затем с помощью датчика случайных чисел генерируются номера слагаемых исходной суммы общим числом m. После вычисления выбранных таким образом слагаемых на параллельных процессорах кластера результат суммирования умножается на k.

Параллельный алгоритм решения "задачи коммивояжера" определяет маршрут, при котором коммивояжер посещает все назначенные ему пункты один раз и возвращается в исходный. При этом "стоимость" его пути (сумма "весов" дуг графа, соединяющих пункты маршрута) должна быть минимальной. Алгоритм представляется в виде ненаправленного взвешенного графа, узлами которого являются населенные пункты, а дугами, соединяющими узлы, - дороги между ними. Задача решается приближенным методом. Алгоритм разбивается на два шага: на первом - случайным образом создается один из возможных путей (сложность алгоритма определяется как $O(N)$, где N - число пунктов); на втором - подсчитывается "стоимость" выбранного пути. Для распараллеливания алгоритма весь набор возможных маршрутов распределяется между процессорами. Граф данной задачи представляется в виде дерева, вершиной которого является узел, обозначающий пункт, из которого отправляется коммивояжер, а ветками дерева являются возможные пути обхода всех узлов. Для выполнения задачи на многих процессорах следует выделить каждому процессору конкретный набор ветвей полученного дерева так, чтобы они независимо друг от друга создавали уникальные пути движения коммивояжера.

Алгоритм параллельной сортировки методом "пузырька" (мы рассматриваем сортировку по возрастанию) состоит из ряда этапов: а) массив сортируемых данных делится на k блоков (k - число процессоров кластера), которые распределяются по всем процессорам; б) выполняется циклический процесс, на каждом шаге которого происходит взаимообмен блоками сортируемых данных между двумя соседними процессорами (их номера отличаются на единицу), а номер большего из пары процессоров совпадает по четности с номером итерации; в) полученный от соседнего процессора блок данных соединяется с уже имеющимся на данном процессоре. В результате на каждом процессоре такой пары теперь будет находиться одинаковый набор данных. С ним выполняется обычная сортировка методом "пузырька"; г) процессор с меньшим номером "забирает" половину отсортированных данных с меньшими значениями, а процессор с большим - остальную часть массива (с большими значениями). После k итераций на каждом процессоре будет находиться блок массива, все элементы которого по величине меньше элементов, находящихся на процессорах с большими номерами; д) заключительным этапом сортировки является сбор на одном процессоре блоков от всех процессоров в порядке возрастания их номеров и объединение в единый массив.

В алгоритме нахождения числа e с заданной точностью объявляется массив, элементы которого содержат соответствующие десятичные разряды этого числа. Число e раскладывается в ряд, представляющий собой сумму обратных величин факториалов. Работа алгоритма заключается в следующем: а) все множество слагаемых распределяется между процессорами. Каждый из них вычисляет фак-

ториал, начиная с номера своего первого слагаемого, и складывает обратные величины факториалов; б) полученная сумма умножается на произведение последних элементов всех процессоров с номером меньшим, чем у текущего процессора; в) на завершающем этапе происходит сбор и суммирование на одном процессоре полученных результатов.

Рассмотренные выше и реализованные на компьютерном кластере ВГУ алгоритмы представляют собой часть создающейся библиотеки параллельных алгоритмов и программ, которая должна обеспечить широкое использование кластера для расчетов в различных областях исследований.

На рис. 3 приведен график зависимости времени выполнения программы от количества разрядов в числе, которое раскладывается на простые множители, для одного и двух процессоров. Как видно из рисунка, при использовании программой двух процессоров время выполнения алгоритма уменьшается в два раза по сравнению с непараллельной программой. Это происходит вследствие того, что процесс вычисления разветвляется на два почти полностью независимых потока.

На рис. 4 показана зависимость времени выполнения алгоритма для "задачи коммивояжера" от числа городов в маршруте на одном, четырех и восьми

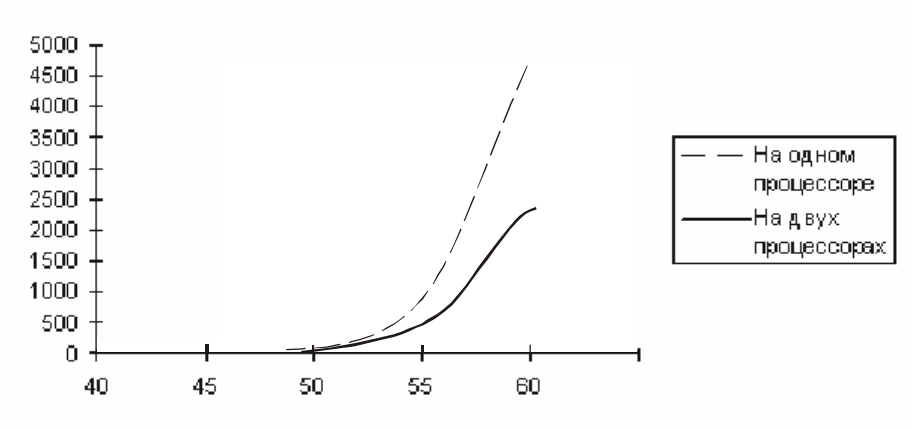

Рис. 3. Зависимость времени выполнения программы (в с) от количества разрядов в числе, которое раскладывается на простые множители, для одного и двух процессоров

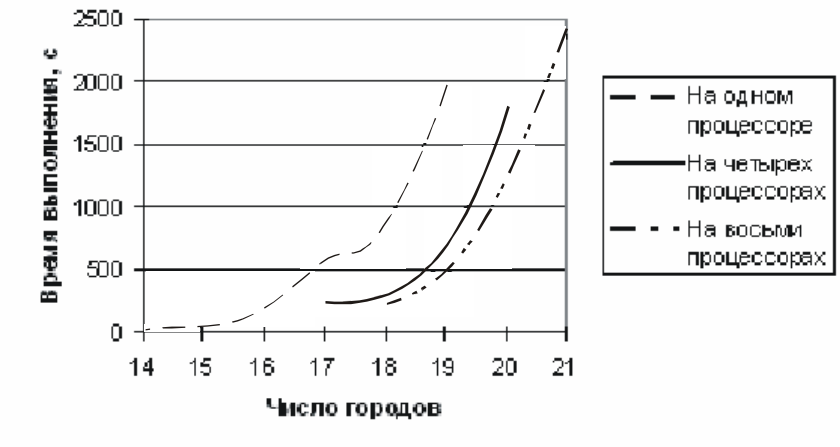

Рис. 4. Зависимость времени выполнения алгоритма (в с) для "задачи коммивояжера" от числа городов в маршруте на одном, четырех и восьми процессорах

процессорах. В этом случае проявляется нелинейная зависимость времени выполнения программы от числа процессоров, что связано с особенностями решаемой задачи.

В целом тестирование кластера показало, что использование коммутатора Rapier G6f и сети Gigabit Ethernet позволяет до 20 раз в некоторых случаях увеличить общую пропускную способность по сравнению с применением протокола Fast Ethernet. Это особенно важно при использовании сложных топологий соединения процессоров с большим числом связей.

Литература

1. *Запрягаев, С.А.* Высокопроизводительный вычислительный кластер в учебном процессе Воронежского университета / С.А. Запрягае, С.Д. Кургалин // Математика, компьютер, образование : 10 междунар. конф. - Пущино, 2003.- С.26.

2. *Корнеев, В.Д.* Параллельное программирование в MPI / В.Д. Корнеев. - М.-Ижевск : Институт компьютерных исследований, 2003. - 304 с.