Имитационное моделирование подсети коллективных операций сети «Ангара»

А. В. Мукосей, А. С. Семенов, И. А. Пожилов
ОАО «НИЦЭВТ»

В ОАО «НИЦЭВТ» разрабатывается высокоскоростная коммуникационная сеть «Ангара» с топологией «многомерный тор». Для исследования и оценки производительности разрабатываемой сети при большом количестве используемых узлов создана параллельная потактовая имитационная модель сети. Сеть «Ангара» имеет аппаратную поддержку двух коллективных операций — broadcast и reduce. В статье описана реализация коллективных операций в имитационной модели и представлены результаты исследования их производительности при помощи модели. Исследование проводилось на базовых тестах broadcast и reduce, а также на прикладных задачах — умножение разреженной матрицы на вектор и численное решение нелинейного уравнения теплопроводности.

1. Введение

В настоящее время суперкомпьютеры содержат сотни тысяч вычислительных ядер. Эффективность одновременной работы ядер на задачах с интенсивным обменом данными между ними (задачи моделирования, задачи на графах и нерегулярных сетках, вычисления с использованием разреженных матриц) в основном определяется производительностью коммуникационной сети, соединяющей вычислительные узлы высокоскоростными каналами связи (линками).

В ОАО «НИЦЭВТ» разрабатывается высокоскоростная коммуникационная сеть «Ангара» с топологией «многомерный тор» [1–6]. В 2013 году выпущен кристалл маршрутизатора этой сети [7], на его основе в 2015 году ожидается построение суперкомпьютера.

Для некоторых прикладных задач требуется эффективное выполнение коллективных коммуникационных операций, в которых задействовано сразу много вычислительных узлов. В сети «Ангара» реализована аппаратная поддержка двух коллективных операций — broadcast и reduce [8]. Для этого добавлена виртуальная подсеть, состоящая из двух виртуальных каналов с особыми правилами маршрутизации. Виртуальная подсеть имеет топологию дерева, наложенную на «многомерный тор».

Для оценки производительности на тестовых программах и для исследования новых архитектур на языке Charm++ создана параллельная потактовая имитационная модель разрабатываемой коммуникационной сети [4]. Однако поддержка коллективных операций в модели отсутствовала.

Имитационное моделирование позволяет исследовать характеристики сетей и суперкомпьютеров, состоящих из большого числа узлов, что особенно важно для коллективных операций, эффективная реализация которых начинает проявляться при большом числе используемых узлов. Имитационное моделирование большого числа узлов важно из экономических соображений, так как большой суперкомпьютер-макет построить дорого по экономическим соображениям.

Целью данной работы является реализация и исследование производительности подсети коллективных операций при помощи имитационной модели. Статья организована следующим образом. Во втором разделе описывается архитектура поддержки коллективных операций в коммуникационной сети. Третий раздел посвящен реализации поддержки коллективных операций в потактовой имитационной модели сети. В четвертом разделе описывается проведенное исследование коллективных операций при помощи имитационной модели сети. В пятом разделе перечисляются основные результаты работы и планы дальнейших

исследований.

2. Коллективные операции в маршрутизаторе коммуникационной сети «Ангара»

Коллективные операции используются для обмена данными между несколькими узлами системы. Их относят к основным примитивам взаимодействия вычислительных процессов в большинстве стандартов параллельного программирования, ориентированных на выполнение на системах с распределенной памятью (MPI [9], Shmem [10], PGAS-языки — UPC [11], X10 [12]); они могут составлять значительную часть коммуникационных обменов в процессе работы [13]. Реализация коллективных операций с использованием операций типа «точка-точка» имеет ряд недостатков, таких как большая доля дублирующего трафика, плохая масштабируемость [14], поэтому их аппаратная поддержка способствует повышению масштабируемости параллельной программы (см., например, [15]).

Высокоскоростная коммуникационная сеть «Ангара» с топологией «многомерный тор» поддерживает детерминированную и адаптивную передачу пакетов, неблокирующие записи, чтения, атомарные операции, отказоустойчивость на канальном уровне и обход отказавших каналов и узлов. В рамках данной сети реализована аппаратная поддержка двух коллективных операций — broadcast и reduce [8]. Для этого добавлена виртуальная подсеть, состоящая из двух виртуальных каналов с отдельными буферами и специальными правилами маршрутизации. Виртуальная подсеть имеет топологию дерева (см. рис. 1), построенного в торе. Выбирается корневой узел, от которого строится дерево с учетом порядка измерений: X, Y, Z, W (это позволяет предотвратить возможные дедлоки). В построенном дереве существует два направления движения: от корня и к корню. Каждому направлению соответствует свой виртуальный канал. В системе могут быть транзитные узлы, в них процессоры не посылают и не получают данных.

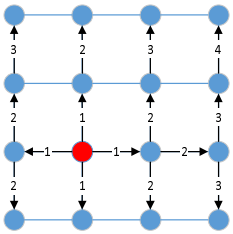


Рис. 1. Виртуальная подсеть коллективных операций на примере 2D-решетки. Стрелками обозначено направление движения от корня, цифрами обозначены этапы обхода дерева.

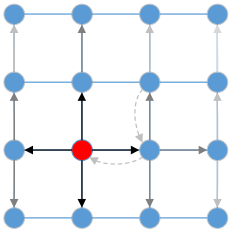


Рис. 2. Схема выполнения операции broadcast не из корневого узла на примере 2D-решетки. Пунктирной линией выделен путь до корня. Сплошной — распространение от корня.

При выполнении операции broadcast каждый узел при получении пакета от узла, находящегося выше по дереву, рассылает его всем узлам, находящимся ниже по дереву (см. рис. 1). При инжекции пакета в сеть не в корневом узле сначала генерируется запрос на broadcast, который посылается корню (см. рис. 2).

При выполнении операции reduce (или варианта all reduce) узел ожидает пакеты от хоста, если узел не транзитный, и всех узлов, находящихся ниже по дереву; выполняет над ними указанную в пакете коммутативную ассоциативную бинарную операцию и отправляет готовый результат вверх к корню. В текущей реализации поддерживаются операции максимума, минимума и суммы целых чисел. Операция reduce в корне завершается аппаратной отправкой (без эжекции) результата заданному узлу посредством операции «точка-точка» (broadcast для all reduce).

Для определения направления к корню и от корня на каждом узле задается таблица маршрутизации коллективной подсети, при этом указываются следующие поля: направления на узлы ниже и выше по дереву; является ли узел транзитным или корневым. Для задания корректного дерева должны выполняться следующие критерии:

- корень ровно один;
- если в каком-то узле выставлено направление вниз по дереву, то в этом направлении должен находиться принадлежащий дереву узел, в котором направление вверх по дереву выставлено противоположным данному;
- направления на узлы ниже по дереву могут быть только: 1) по измерениям, следующим за направлением вверх по дереву в рамках порядка направлений, 2) по направлению, противоположному направлению вверх по дереву.

В рамках сети можно задавать различные пересекающиеся деревья. Поддерживается 16 деревьев, каждому соответствует свой идентификатор TreeId, по которому производится выборка из таблицы маршрутизации при принятии решения по маршрутизации пакета. Одновременно маршрутизатор поддерживает до 16 различных пакетов reduce по каждому TreeId. Каждый reduce, выполняющийся по данному дереву, имеет свой идентификатор ReduceId (от 0 до 15).

С точки зрения прикладного программиста, базовые версии коллективных операций — односторонние асинхронные операции. Процессор не блокируется после отправки сообщения, а результат записывается в память без активного участия принимающей стороны, это позволяет совмещать ожидание окончания операции со счетом. Для того, чтобы узнать, что коллективная операция завершилась и результат доступен вычислительным узлам, существуют механизмы синхронизации, основанные на коллективных операциях.

3. Реализация коллективных операций в параллельной имитационной модели

Для оценки производительности и исследования новых архитектур высокоскоростной коммуникационной сети разработана и используется параллельная потактовая имитационная модель [4]. Модель разработана на языке Charm++ и позволяет моделировать на вычислительном кластере конфигурации с большим количеством моделируемых узлов сети. Модель маршрутизатора устроена достаточно гибко и имеет большое количество конфигурационных параметров. Это позволяет подстраивать модель для различных типов сетей. В модели реализованы следующие топологии: тор произвольной размерности, сеть Кэли, сеть Клоса.

На рис. 3 представлена общая схема маршрутизатора рассматриваемой сети, реализованная в имитационной модели.

Маршрутизатор имеет два типа входов: межузловые и инжекционные. Аналогично, имеется два типа выходов: межузловые и эжекционные. Межузловые входы (выходы) соединяются с выходами (входами) других узлов соответственно, посредством физических каналов, так называемых межузловых линков. Считается, что помехи отсутствуют. Инжекционные (эжекционные) каналы, так называемые процессорные, служат для связи процессора с маршрутизатором. Каждый процессор соединяется с маршрутизатором посредством одного или нескольких инжекционных и такого же количества эжекционных каналов. Каналы представляют собой FIFO-буфера, доступные только для чтения или только для записи. Минимальная длина передаваемых данных в сети — 128 бит (флит данных).

Для поддержки виртуальных подсетей в маршрутизаторе предусмотрены виртуальные каналы (VC). Виртуальные каналы представляют собой FIFO-буфера с блоком маршрутизации. Каждый пакет из линка через блок анализа данных BAD попадает в заданный типом пакета FIFO-буфер.

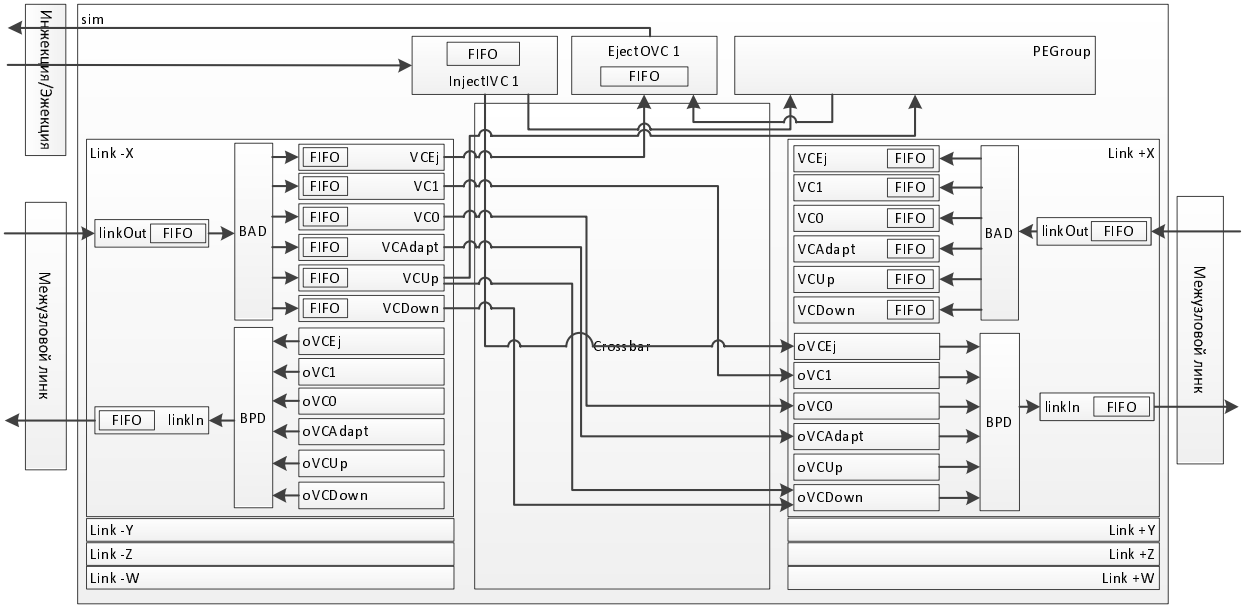


Рис. 3. Общая схема маршрутизатора коммуникационной сети с топологией «многомерный тор», реализованная в имитационной модели сети.

Для реализации подсети коллективных операций добавлены два виртуальных канала: для движения к корню (VCUp) и для движения к листьям (VCDown), а также блок PEGroup для эжекции, инжекции и обработки коллективных пакетов.

3.1. Виртуальные каналы VCUp и VCDown

За основу виртуальных каналов VCUp и VCDown взят детерминированный виртуальный канал (VCDet). При этом изменена маршрутизация, а также добавлен механизм выдачи копий пакета. Маршрутизация осуществляется по данным из таблицы маршрутизации и по заголовку пакета. Таблица маршрутизации заполняется на этапе инициализации модели. В таблице 16 строк, по строке на каждое дерево. У строк есть следующие поля: $TreeId$ — номер дерева подсети коллективных операций, $isRoot$ — определяет является ли узел корнем, $toRoot$ — направление вверх к корню, Pe — участвует ли узел в коллективной операции или он транзитный, $Dirs$ — направления вниз по дереву, Dir_sum — количество направлений (нужно для контроля выданных пакетов). Процесс прохождения пакета по виртуальным каналам коллективных операций состоит из следующих этапов (см. рис. 4):

- ожидание приема всего пакета в буфер виртуального канала;
- чтение таблицы маршрутизации, анализ головного флита пакета;
- в зависимости от виртуального канала и полученной выше информации составляется список направлений на передачу: детям, в вычислительный узел или к корню.

3.2. Блок PEGroup

Блок PEGroup (см. рис. 5) предназначен для приёма/передачи пакетов типа broadcast, reduce-пакетов и кредитных пакетов. Он разделен на три основные части: Broadcaster — блок рассылки broadcast пакетов, Reducer — блок для пакетов reduce и блок анализа кредитных пакетов. Инжектированные пакеты коллективных операций от процессора сначала попадают в очередь PEGroup Fifo блока PEGroup. После чего блок маршрутизации, ана-

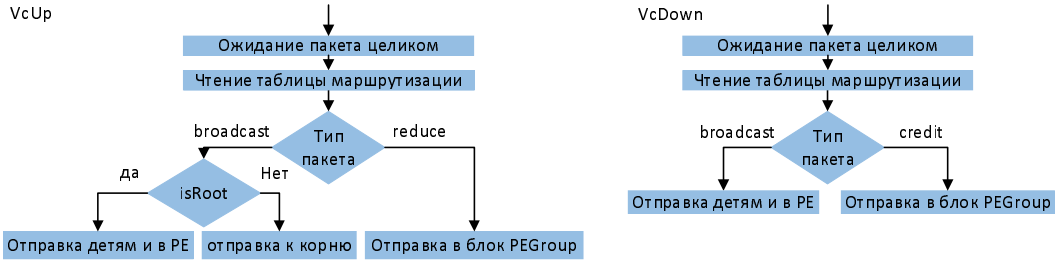


Рис. 4. Логика работы виртуальных каналов VCUp и VCDown.

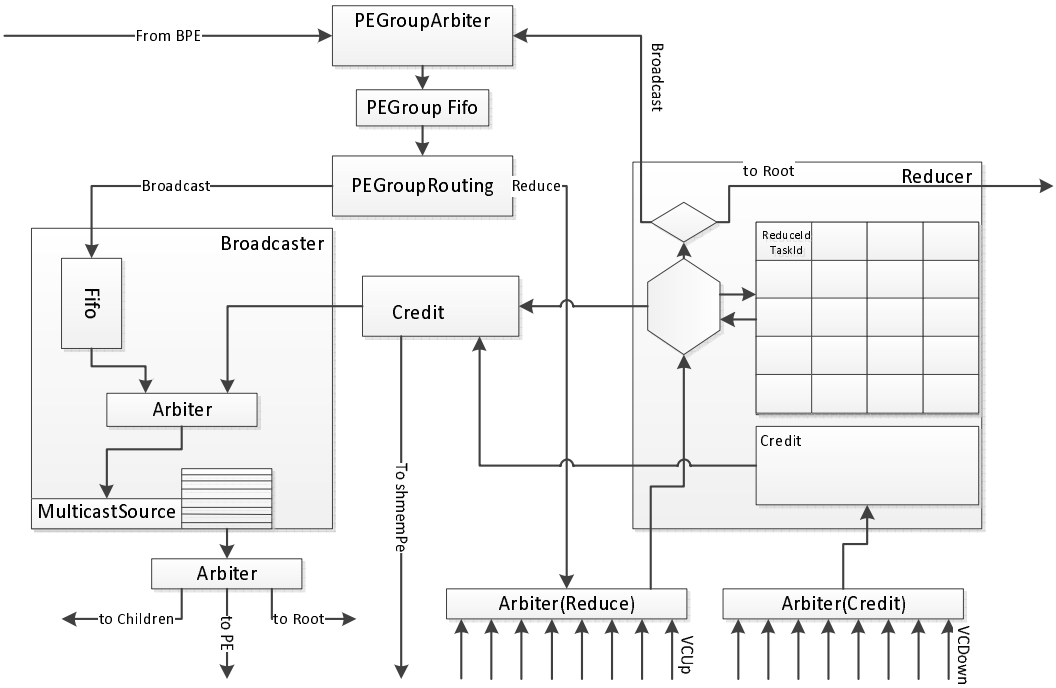


Рис. 5. Общая схема блока PEGroup.

лизируя пакеты, отправляет их в соответствующий блок. Reduce- и кредитные пакеты, пришедшие из виртуальных каналов, попадают в блок Reducer.

3.3. Блок Reducer

Блок Reducer принимает пакеты типа reduce. Над данными из пакета типа reduce блок Reducer производит заданную операцию (определяется по данным из головного флита пакета), сохраняет значения у себя в памяти для дальнейших операций или отправляет дальше по сети. Для каждого TreeId и ReduceId в блоке Reducer имеется изначально равный нулю счетчик принятых пакетов. Когда в Reducer из виртуального канала приходит пакет reduce, то его процесс прохождения по блоку Reducer будет следующим:

- Из заголовочного флита пакета считываются TreeId и ReduceId.
- Увеличивается на 1 значение счетчика принятых пакетов по TreeId и ReduceId.
- Считываются значения из памяти по TreeId и ReduceId (если это первый пакет, то ничего не происходит).
- Производится заданная арифметическая операция над каждым флитом данных из вновь пришедшего пакета и соответствующим значением из памяти (если это первый

пакет, то ничего не происходит).

- Полученные значения вновь записываются в память по TreeId и ReduceId (если это первый пакет, то в качестве значений берутся флиты данных пакета).
- Если пришедший пакет был последним, то полученный результат выдается в блок маршрутизации. В кредитном блоке увеличивается на 1 значение счетчика выданных результирующих пакетов reduce по TreeId.

Если узел не был корневым, то маршрутизатор выставит запрос к корню вверх по дереву. Если узел корневой, то будет выставлен запрос на передачу в очередь PEGroup Fifo, для рассылки результата операции редукции (reduce убрать).

3.4. Блок анализа кредитных пакетов

Блок анализа кредитных пакетов создан для контроля количества одновременно обрабатываемых reduce пакетов в блоке Reducer; допустимо 16 операций по одному дереву. Блок отправляет кредитный пакет в PE и всем своим детям. Отправка детям происходит каждый раз, когда из узла было отправлено вверх по дереву или в процессор 8 пакетов с результатом редукции для каждого фиксированного дерева. Отправка в PE осуществляется каждые 128 отправленных пакетов с результатом редукции по всем TreeId или каждые 4096 тактов. В процессор отправляются значения всех счетчиков выполненных операций редукции по всем TreeId. Также блок анализирует приходящие кредитные пакеты. Каждый такой пакет означает, что узел-родитель может принять еще 8 пакетов по данному TreeId. Кредитный блок осуществляет передачу пакетов через блок Broadcaster.

3.5. Блок Broadcaster

Блок Broadcaster берет поочередно пакеты из очереди PEGroup Fifo и кредитного блока, анализирует заголовки пакетов и таблицу маршрутизации и принимает решение о маршрутизации: вниз по дереву, вверх к корню или на инжекцию в PE.

В модели реализован механизм построения временных диаграмм прохождения пакета по маршрутизатору. Для каждого реализованного блока с помощью этого механизма построены временные диаграммы. Все диаграммы соответствую временным диаграммам аппаратной реализации. Пример диаграммы прохождения пакета по виртуальному каналу VCUp приведен на рис. 6.

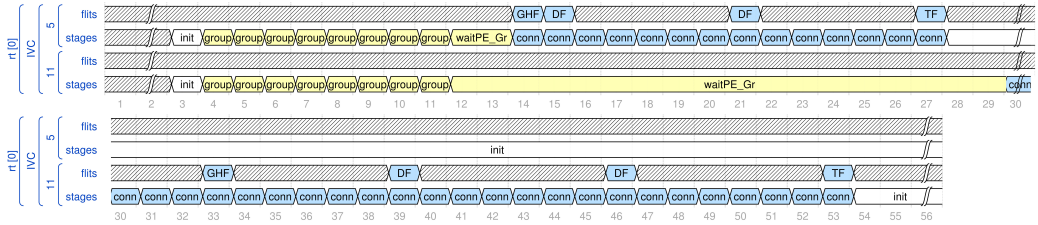


Рис. 6. Временная диаграмма прохождения пакета по каналу VCUp.

4. Исследование производительности коллективных операций на базовых тестах и прикладных задачах

Исследование производительности коллективных операций проводилось в два этапа. На первом этапе исследовались базовые операции broadcast и all reduce, имеющие непосредственную аппаратную поддержку в подсети коллективных операций. На втором этапе

исследовалась производительность небольших прикладных тестов-задач. Выбраны следующие задачи: умножение разреженной матрицы на вектор, численное решение задачи с нелинейным уравнением теплопроводности.

Исследования проводились на имитационной модели сети, которая описана в третьем разделе. Параметры модели сети соответствуют значениям параметрам СБИС маршрутизатора сети «Ангара», выпущенного в 2013 году и приведены в табл. 1. Необходимые вычисления проводились на процессоре, параметры которого также приведены в табл. 1.

Таблица 1. Параметры моделируемого суперкомпьютера.

Параметры	**Значение**
Процессор	Intel Xeon E5-2660
Тактовая частота процессора, ГГц	2.2
Количество ядер в узле	8
Размер кэша L3, МБ	20
Пиковая производительность узла, Гфлопс	140.8
Тактовая частота маршрутизатора, МГц	500
ПС канала сети, Гбит/с	75
Интерфейс процессора с сетью	PCIe gen2 x16
Задержка на инжекцию (эжекцию) пакета через PCIe, нс	300
Задержка передачи по линку, нс	80
Топология сети	3D/4D-тор

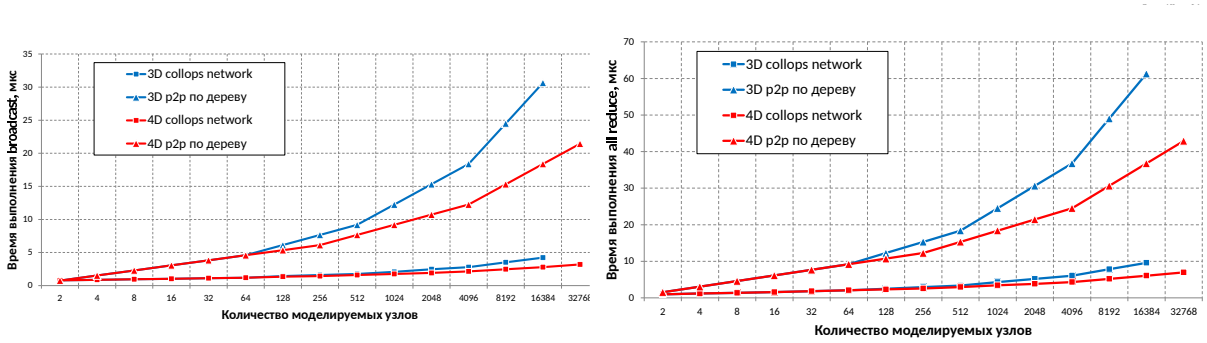
4.1. Базовые тесты broadcast и all reduce

На этапе тестирования базовых операций исследовалась производительность базовых операций broadcast и all reduce. Тесты заключались в посылке одного пакета максимального размера в 16 флитов (256 байт) при помощи операций broadcast и all reduce для разного количества узлов моделируемой сети.

На рис. 7 представлены результаты выполнения операций broadcast и all reduce. Синим и красным показано время выполнения (в мкс) коллективной операции для сети с топологией 3D- и 4D-тор соответственно. Квадратными маркерами отмечено время выполнения коллективной операции, реализованной с помощью сети коллективных операций, треугольными маркерами — при помощи коммуникационных операций «точка-точка» по тому же дереву, что и в подсети коллективных операций.

Для операции broadcast в сети с топологией 3D-тор выигрыш для 8 узлов (тор 2x2x2) составляет 2.39 раз, а для 8096 узлов (тор 16x16x32) — 6.97 раз. Для топологии 4D-тор выигрыш для 8096 узлов (тор 8x8x8x16) составляет 6.18 раз. Превосходство аппаратной поддержки коллективных операций обусловлено отсутствием накладных расходов на эжекцию/инжекцию пакетов. Сети с топологией 4D-тор обладают меньшим диаметром по сравнению с сетями 3D-тор при одинаковом количестве узлов, поэтому для сетей с топологией

4D-тор выигрыш меньше. Для операции all reduce разница в производительности примерно такая же. Таким образом, реализация операций с помощью коллективной подсети дает значительный выигрыш по сравнению с использованием операций «точка-точка».



(a) broadcast

(b) all reduce

Рис. 7. Время выполнения операции broadcast (a) и all reduce (b) в микросекундах в зависимости от количества вычислительных узлов для 3D- и 4D-торов для реализаций через сеть коллективных операций (collops network) и при помощи операций «точка-точка» (p2p).

4.2. Умножение разреженной матрицы на вектор

Операция умножения разреженной матрицы на вектор лежит в основе метода сопряженных градиентов CG. CG — метод нахождения локального минимума функции на основе информации о её значениях и её градиенте. Один из способов распараллеливания этой операции заключается в следующем [16]: пусть дана разреженная матрица A и вектор x, требуется найти $y = Ax$. Строки матрицы блоками равномерно распределяются по узлам:

$$A = (A_1, A_2, \ldots, A_{np})^T \Rightarrow Ax = (A_1x, A_2x, \ldots, A_{np}x)^T, \quad (1)$$

где A_i — строки матрицы A, хранящиеся на i-ом узле. При умножении строк матрицы на вектор получаются результирующие части искомого вектора y: $y_j = A_jx$, где $j = 1, 2, \ldots, np$, np — количество вычислительных узлов. Для сборки результирующего вектора y на каждом из вычислительных узлов для последующих вычислений понадобится коллективная операция all gather.

Для реализации all gather через операцию «точка-точка» используется алгоритмом «рекурсивное удвоение» [17]. Алгоритм заключается в следующем: сначала все узлы попарно обмениваются друг с другом, затем пары обмениваются попарно таким образом, что каждый узел пары обменивается данными с другим узлом пары и т.д. Схема работы алгоритма изображена на рис. 8.

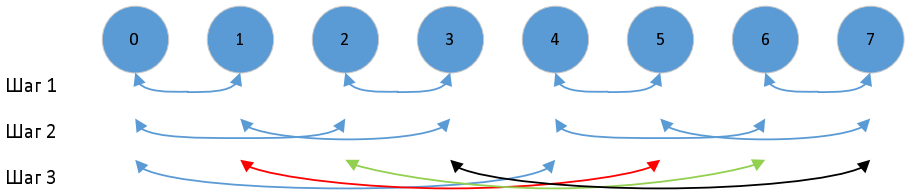


Рис. 8. Схема работы алгоритма «рекурсивное удвоение» для системы из 8 узлов.

Для реализации all gather через коллективные операции используется broadcast — все узлы рассылают свою часть вектора, инициируя операцию broadcast. Используются все 16 деревьев, корни которых равномерно распределены по системе.

Для расчетов выбрана матрица с d = 10 ненулевыми элементами в строке и размером N = 20480000. Измерение времени умножения строк матрицы на плотный вектор проводилось на реальном процессоре Intel Xeon E5-2660 с использованием тестовой программы на C/OpenMP отдельно для каждого количества строк матрицы, попадающих на вычислительный узел при увеличении количества узлов сети и дроблении задачи. Производительность на одном узле составляет 591 Мфлопс. Время выполнения операции all gather получено с помощью имитационной модели сети.

На рис. 9 видно, что использование аппаратных коллективных операций дало максимальный прирост производительности 28% на 2048 узлах для сети 4D-тор по сравнению с использованием коллективных операций, реализованных с помощью операций «точка-точка».

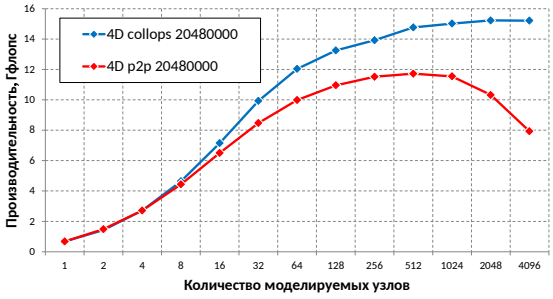


Рис. 9. Производительность задачи умножения разреженной матрицы на вектор (N=20480000, d=10) в зависимости от количества вычислительных узлов для сети с топологией 4D-тор.

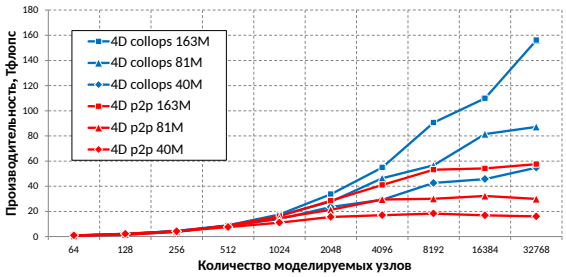


Рис. 10. Производительность двухмерной задачи теплопроводности на различных сетках в зависимости от количества вычислительных узлов.

4.3. Задача с нелинейным уравнением теплопроводности

Рассматривается двухмерная нелинейная задача теплопроводности:

$$\frac{\partial u}{\partial t} = \sigma(x_1, x_2, t)\left(\frac{\partial^2 u}{\partial x_1^2} + \frac{\partial^2 u}{\partial x_2^2}\right) + f(x_1, x_2, t) \tag{2}$$

$$t\epsilon[0, t_k], x_1\epsilon[a, b], x_2\epsilon[c, d], \forall x_1, x_2, t : \sigma \geqslant 0 \tag{3}$$

Явная разностная схема:

$$\frac{U_{j,k}^{n+1} - U_{j,k}^n}{\Delta t} = \sigma_{j,k}^n \left(\frac{U_{j+1,k}^n - 2U_{j,k}^n + U_{j-1,k}^n}{h_j^2} + \frac{U_{j,k+1}^n - 2U_{j,k}^n + U_{j,k-1}^n}{h_k^2}\right) + f_{j,k}^h \tag{4}$$

Условие устойчивости:

$$\Delta t \leqslant \frac{1}{2\sigma(x_1, x_2, t)\left(\frac{1}{h_1^2} + \frac{1}{h_2^2}\right)} \leqslant \frac{1}{2max[\sigma(x_1, x_2, t)]\left(\frac{1}{h_1^2} + \frac{1}{h_2^2}\right)} \tag{5}$$

Двухмерная область распределяется поровну по вычислительным узлам. В явной схеме для вычисления значения в ячейке на следующем слое по времени необходимы 9 значений ячеек с предыдущего слоя (см. рис. 11). Для этого на каждый узел требуется область больше на единицу в каждую сторону, чтобы была возможность рассчитать границы области. После каждого расчета области необходим обмен гранями между узлами и вычисление

оптимального шага по времени. Для вычисления временного шага, требуется вычислить максимум коэффициента температуропроводности (усл. устойчивости 5). Вычисление максимума происходит при помощи коллективной операции all reduce. Общая схема параллельной реализации задачи представлена на рис. 12.

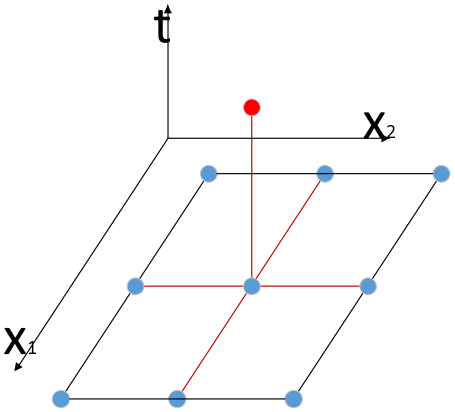


Рис. 11. Шаблон явной схемы двухмерной нелинейной задачи теплопроводности.

Рис. 12. Схема параллельной реализации явной схемы для нелинейного уравнения теплопроводности.

Производительность вычисляется по формуле: $\frac{N_{op}}{T_S+T_{max}} * 10^{-12}$ [Тфлопс], где N_{op} — количество операций с плавающей точкой. Рассматривались задачи с общим количеством ячеек, равным 40960000, 81920000 и 163840000 (на рис. 10 обозначены 40M, 81M и 163M соответственно). Время расчета области (T_S) измерялось на реальном процессоре Intel Xeon E5-2660 с использованием тестовой программы на C/OpenMP отдельно для каждого размера области, приходящейся на вычислительный узел при увеличении количества узлов сети и дроблении задачи. Производительность на одном узле составляет приблизительно 11 Гфлопс. Время вычисления максимума коэффициента температуропроводности (T_{max}) получено при помощи имитационного моделирования операции all reduce.

На данной задаче (см. рис. 10) аппаратная поддержка коллективных операций в сети 4D-тор сказывается, когда узлов в сети более 64. На большом числе вычислительных узлов (32768) производительность варианта с использованием подсети коллективных операций превышает производительность варианта с реализацией на основе операций «точка-точка» в 3 раза для задачи с 40M ячеек в сетке, для остальных размеров задач выигрыш составляет 2.7 раз.

5. Заключение

В ходе выполнения данной работы получены следующие основные результаты:

1. Исследование задержки выполнения базовых коллективных операций reduce и broadcast при помощи разработанной имитационной модели показало значительный выигрыш (более 2-х раз для 8 узлов, более 6-ти раз для 8192 узлов) при использовании аппаратной подсети коллективных операций относительно программной реализации этих операций при помощи операции «точка-точка». С увеличением количества вычислительных узлов разница в производительности увеличивается.

2. Исследование производительности прикладных задач проводилось на имитационной модели на примере задачи умножения разреженной матрицы на вектор и задачи численного решения явной схемы нелинейного уравнения теплопроводности. Для задачи умноже-

ния матрицы на вектор выбраны следующие параметры: 20480000 — размер матрицы с 10 ненулевыми элементами в строке, задача теплопроводности рассматривалась на сетках с количеством ячеек 40960000, 81920000 и 163840000.

На задаче умножения разреженной матрицы на вектор, реализованной с помощью операции all gather, выигрыш от использования аппаратной поддержки коллективных операций по сравнению с реализацией на основе операции «точка-точка» небольшой, растет с увеличением количества вычислительных узлов и максимально составляет 28% на 2048 узлах для сети с топологией 4D-тор.

На задаче численного решения явной схемы нелинейного уравнения теплопроводности на тысячах и десятках тысяч вычислительных узлов производительность варианта с использованием подсети коллективных операций превышает производительность варианта с реализацией на основе операций «точка-точка» от 2.7 до 3 раз. При увеличении количества вычислительных узлов разница в производительности растет.

Выполненная работа позволила получить оценки производительности тестов с использованием аппаратной поддержки коллективных операций в коммуникационной сети с топологией «многомерный-тор» и показала возможность получения высокой производительности с использованием таких коллективных операций для выбранных задач. При этом выбраны конкретные размеры демонстрационных задач.

С использованием представленных в данной статье результатов планируется разработка новой архитектуры подсети коллективных операций для следующего поколения маршрутизатора сети «Ангара».

Литература

1. Макагон Д.В., Сыромятников Е.Л. Сети для суперкомпьютеров // Открытые системы. — 2011. — №7.

2. Корж А.А. , Макагон Д.В., Жабин И.А., Сыромятников Е.Л. Отечественная коммуникационная сеть 3D-тор с поддержкой глобально адресуемой памяти для суперкомпьютеров транспетафлопсного уровня производительности. // Паралельные вычислительные технологии (ПаВТ'2010): Труды международной научной конференции (Уфа, 29 марта-2 апреля 2010 г.): С. 227-237, Челябинск: Издательский центр ЮУрГУ, ISBN 978-5-696-03987-9, 2010.
`http://omega.sp.susu.ac.ru/books/conference/PaVT2010/full/134.pdf`

3. Симонов А.С., Жабин И.А., Макагон Д.В. Разработка межузловой коммуникационной сети с топологией «многомерный тор» и поддержкой глобально адресуемой памяти для перспективных отечественных суперкомпьютеров. // Научно-техническая конференция «Перспективные направления развития вычислительной техники», ОАО «НИЦЭВТ», 2011.

4. Эйсымонт Л.К., Семенов А.С., Соколов А.А., Фролов А.С., Шворин А.Б. Моделирование российского суперкомпьютера «Ангара» на суперкомпьютере // В сборнике «Суперкомпьютерные технологии в науке, образовании и промышленности» под редакцией академика В.А. Садовничего, академика Г.И. Савина, чл.-корр. РАН Вл.В. Воеводина. — М.: Издательство Московского университета, 2009. — С. 145-150.

5. Симонов А.С., Макагон Д.В., Жабин И.А., Щербак А.Н., Сыромятников Е.Л., Поляков Д.А. Первое поколение высокоскоростной коммуникационной сети "Ангара"// Наукоемкие технологии. — 2014. — Т. 15, №1. — С. 21-28.

6. Слуцкин А.И., Симонов А.С., Жабин И.А., Макагон Д.В., Сыромятников Е.Л. Разработка межузловой коммуникационной сети ЕС8430 «Ангара» для перспективных суперкомпьютеров // Успехи современной радиоэлектроники. — 2012. — №1.

7. Жабин И.А., Макагон Д.В., Симонов А.С. Кристалл для Ангары // Суперкомпьютеры. — Зима-2013. — С. 46-49.

8. Макагон Д.В., Сыромятников Е.Л., Парута С.И., Румянцев А.А. Реализация аппаратной поддержки коллективных операций в маршрутизаторе высокоскоростной коммуникационной сети с топологией «многомерный тор» // Успехи современной радиоэлектроники. — 2012.

9. Message Passing Interface Forum, MPI: A Message-Passing Interface Standard, 1995. http://www.mpi-forum.org/docs/mpi-11-html/node64.html

10. Karl Feind, Cray Research. Shared Memory Access (SHMEM) Routines, 1995. https://cug.org/5-publications/proceedings_attendee_lists/1997CD/S95PROC/303_308.PDF

11. Elizabeth Wiebel, David Greenberg, Steve Seidel. UPC Collective Operations Specifications, 2003. http://upc.gwu.edu/docs/UPC_Coll_Spec_V1.0.pdf

12. Vijay Saraswat, Bard Bloom, Igor Peshansky, Olivier Tardieu, David Grove. X10 Language Specification, 2011. http://dist.codehaus.org/x10/documentation/languagespec/x10-latest.pdf

13. Geoffrey Fox, Mark Johnson, Gregory Lyzenga, Steve Otto, John Salmon and David Walker. Solving Problems on Concurrent Processors. Vol. 1: General techniques and regular problems, 1988, Prentice-Hall, Inc., ISBN 0-13-823022-6.

14. Vasanth Bala, Jehoshua Bruck, Robert Cypher, Pablo Elustondo, Alex Ho, Ching-Tien Ho, Shlomo Kipnis, Marc Snir. CCL: A Portable and Tunable Collective Communication Library for Scalable Parallel Computers, Parallel Processing Symposium: c. 835-844, Proceedings, ISBN 0-8186-5620-6, 1994. http://citeseer.ist.psu.edu/viewdoc/download;jsessionid=3D465188A4C42E5F58002758BE8B57C3?doi=10.1.1.155.3612&rep=rep1&type=pdf

15. George Almasi, Gabor Dozsa, C. Erway, Burkhardt Steinmacher-Burow. Efficient Implementation of Allreduce on BlueGene/L Collective Network, Recent Advances in Parallel Virtual Machine and Message Passing Interface: c.57-66, Springer Berlin // Heidelberg, 2005. http://dx.doi.org/10.1007/11557265_12

16. Пожилов И. А., Семенов А. С., Макагон Д. В. Прогнозирование масштабируемости задачи умножения разреженной матрицы на вектор при помощи модели коммуникационной сети // Вестник УГАТУ. — 2012. — Т. 16, № 6 (51). — С. 158-163.

17. Rajeev Thakur, Rolf Rabenseifner, William Gropp. Optimization of Collective Communication Operations in MPICH. http://www.lrr.in.tum.de/~gerndt/home/Teaching/HPCSeminar/mpich_multi_coll.pdf