3. Оценка коммуникационной трудоемкости параллельных алгоритмов

3. Оценка коммуникационной трудоемкости параллельных алгоритмов 1
3.1. Общая характеристика механизмов передачи данных 1
3.1.1. Алгоритмы маршрутизации 1
3.1.2. Методы передачи данных 2
3.2. Анализ трудоемкости основных операций передачи данных 2
3.2.1. Передача данных между двумя процессорами сети 3
3.2.2. Передача данных от одного процессора всем остальным процессорам сети 3
3.2.3. Передача данных от всех процессоров всем процессорам сети 4
3.2.4. Обобщенная передача данных от одного процессора всем остальным процессорам сети 5
3.2.5. Обобщенная передача данных от всех процессоров всем процессорам сети 6
3.2.6. Циклический сдвиг 7
3.3. Методы логического представления топологии коммуникационной среды 8
3.3.1. Представление кольцевой топологии в виде гиперкуба 8
3.3.2. Отображение топологии решетки на гиперкуб 9
3.4. Оценка трудоемкости операций передачи данных для кластерных систем 9
3.5. Краткий обзор раздела 11
3.6. Обзор литературы 11
3.7. Контрольные вопросы 11
3.8. Задачи и упражнения 12

Как уже отмечалось ранее, временные задержки при передаче данных по каналам связи для организации взаимодействия раздельно функционирующих процессов могут в значительной степени определять эффективность параллельных вычислений. Данный раздел посвящен вопросам анализа информационных потоков, возникающих при выполнении параллельных алгоритмов. В разделе дается общая характеристика механизмов передачи данных, проводится анализ трудоемкости основных операций обмена информацией, рассматриваются методы логического представления структуры МВС. Более подробно изучаемый в данном разделе пособия учебный материал излагается в работах Kumar (1994), Quinn (2004).

3.1. Общая характеристика механизмов передачи данных

3.1.1. Алгоритмы маршрутизации

Алгоритмы маршрутизации определяют путь передачи данных от процессора-источника сообщения до процессора, к которому сообщение должно быть доставлено. Среди возможных способов решения данной задачи различают:

– *оптимальные*, определяющие всегда наикратчайшие пути передачи данных, и *неоптимальные* алгоритмы маршрутизации;

– *детерминированные* и *адаптивные* методы выбора маршрутов (адаптивные алгоритмы определяют пути передачи данных в зависимости от существующей загрузки коммуникационных каналов).

К числу наиболее распространенных оптимальных алгоритмов относится класс *методов покоординатной маршрутизации* (*dimension-ordered routing*), в которых поиск путей передачи данных осуществляется поочередно для каждой размерности топологии сети коммуникации. Так, для двумерной решетки такой подход приводит к маршрутизации, при которой передача данных сначала выполняется по одному направлению (например, по горизонтали до достижения вертикали процессоров, в которой располагается процессор назначения), а затем данные передаются вдоль другого направления (данная схема известна под названием *алгоритма XY-маршрутизации*).

Для гиперкуба покоординатная схема маршрутизации может состоять, например, в циклической передаче данных процессору, определяемому первой различающейся битовой позицией в номерах процессоров, на котором сообщение располагается в данный момент времени и на который сообщение должно быть передано.

3.1.2. Методы передачи данных

Время передачи данных между процессорами определяет коммуникационную составляющую (*communication overhead*) длительности выполнения параллельного алгоритма в многопроцессорной вычислительной системе. Основной набор параметров, описывающих время передачи данных, состоит из следующего ряда величин:

– **время начальной подготовки** ($t_н$) характеризует длительность подготовки сообщения для передачи, поиска маршрута в сети и т.п.;

– **время передачи служебных данных** ($t_с$) между двумя соседними процессорами (т.е. для процессоров, между которыми имеется физический канал передачи данных); к служебным данным может относиться заголовок сообщения, блок данных для обнаружения ошибок передачи и т.п.;

– **время передачи одного слова данных** по одному каналу передачи данных ($t_к$); длительность подобной передачи определяется полосой пропускания коммуникационных каналов в сети.

К числу наиболее распространенных методов передачи данных относятся следующие два основные способа коммуникации (см., например, Kumar (1994)). Первый из них ориентирован на *передачу сообщений* (*МПС*) как неделимых (атомарных) блоков информации (*store-and-forward routing* or *SFR*). При таком подходе процессор, содержащий сообщение для передачи, готовит весь объем данных для передачи, определяет процессор, которому следует направить данные, и запускает операцию пересылки данных. Процессор, которому направлено сообщение, в первую очередь осуществляет прием полностью всех пересылаемых данных и только затем приступает к пересылке принятого сообщения далее по маршруту. Время пересылки данных $t_{пд}$ для метода передачи сообщения размером m байт по маршруту длиной l определяется выражением

$$t_{пд} = t_н + (mt_к + t_с)l \ .$$

При достаточно длинных сообщениях временем передачи служебных данных можно пренебречь и выражение для времени передачи данных может быть записано в более простом виде

$$t_{пд} = t_н + mt_к l \ .$$

Второй способ коммуникации основывается на представлении пересылаемых сообщений в виде блоков информации меньшего размера (*пакетов*), в результате чего передача данных может быть сведена к *передаче пакетов* (*МПП*). При таком методе коммуникации (*cut-through routing* or *CTR*) принимающий процессор может осуществлять пересылку данных по дальнейшему маршруту непосредственно сразу после приема очередного пакета, не дожидаясь завершения приема данных всего сообщения. Время пересылки данных при использовании метода передачи пакетов будет определяться выражением

$$t_{пд} = t_н + mt_к + t_с l \ .$$

Сравнивая полученные выражения, можно заметить, что в большинстве случаев метод передачи пакетов приводит к более быстрой пересылке данных; кроме того, данный подход снижает потребность в памяти для хранения пересылаемых данных для организации приема-передачи сообщений, а для передачи пакетов могут использоваться одновременно разные коммуникационные каналы. С другой стороны, реализация пакетного метода требует разработки более сложного аппаратного и программного обеспечения сети, может увечить накладные расходы (время подготовки и время передачи служебных данных); при передаче пакетов возможно возникновения конфликтных ситуаций (дедлоков).

3.2. Анализ трудоемкости основных операций передачи данных

При всем разнообразии выполняемых операций передачи данных при параллельных способах решения сложных научно-технических задач определенные процедуры взаимодействия процессоров сети могут быть отнесены к числу основных коммуникационных действий, которые или наиболее широко распространены в практике параллельных вычислений, или к которым могут быть сведены многие другие процессы приема-передачи сообщений. Важно отметить также, что в рамках подобного базового набора для большинства операций коммуникации существуют процедуры, обратные по действию исходным операциям (так, например, операции передачи данных от одного процессора всем имеющимся процессорам сети соответствует операция приема в одном процессоре сообщений от всех остальных процессоров). Как результат, рассмотрение коммуникационных процедур целесообразно выполнять попарно, поскольку во многих случаях алгоритмы выполнения прямой и обратной операций могут быть получены исходя из общих положений.

Рассмотрение основных операций передачи данных в данном разделе будет осуществляться на примере таких топологий сети, как кольцо, двумерная решетка и гиперкуб. Для двумерной решетки будет предполагаться также, что между граничными процессорами в строках и столбцах решетки имеются каналы передачи данных (т.е. топология сети представляет из себя тор). Как и ранее, величина m будет означать

размер сообщения в байтах, значение p определяет количество процессоров в сети, а переменная N задает размерность топологии гиперкуба.

3.2.1. Передача данных между двумя процессорами сети

Трудоемкость данной коммуникационной операции может быть получена путем подстановки длины максимального пути (диаметра сети – см. табл. 1.1) в выражения для времени передачи данных при разных методах коммуникации (см. п. 3.1.2).

Таблица 3.2. Время передачи данных между двумя процессорами

Топология	**Передача сообщений**	**Передача пакетов**
Кольцо	$t_н + mt_к \lfloor p/2 \rfloor$	$t_н + mt_к + t_c \lfloor p/2 \rfloor$
Решетка-тор	$t_н + 2mt_к \lfloor \sqrt{p}/2 \rfloor$	$t_н + mt_к + 2t_c \lfloor \sqrt{p}/2 \rfloor$
Гиперкуб	$t_н + mt_к \log_2 p$	$t_н + mt_к + t_c \log_2 p$

3.2.2. Передача данных от одного процессора всем остальным процессорам сети

Операция передачи данных (одного и того же сообщения) от одного процессора всем остальным процессорам сети (*one-to-all broadcast* or *single-node broadcast*) является одним из наиболее часто выполняемых коммуникационных действий; двойственная операция передачи – прием на одном процессоре сообщений от всех остальных процессоров сети (*single-node accumulation*). Подобные операции используются, в частности, при реализации матрично-векторного произведения, решении систем линейных уравнений при помощи метода Гаусса, поиска кратчайших путей и др.

Простейший способ реализации операции рассылки состоит в ее выполнении как последовательности попарных взаимодействий процессоров сети. Однако при таком подходе большая часть пересылок является избыточной и возможно применение более эффективных алгоритмов коммуникации. Изложение материала будет проводиться сначала для метода передачи сообщений, затем – для пакетного способа передачи данных (см. п. 3.1.2).

Передача сообщений. Для **кольцевой топологии** процессор-источник рассылки может инициировать передачу данных сразу двум своим соседям, которые, в свою очередь, приняв сообщение, организуют пересылку далее по кольцу. Трудоемкость выполнения операции рассылки в этом случае будет определяться соотношением

$$t_{пд} = (t_н + mt_к)\lceil p/2 \rceil.$$

Для топологии типа **решетки-тора** алгоритм рассылки может быть получен из способа передачи данных, примененного для кольцевой структуры сети. Так, рассылка может быть выполнена в виде двухэтапной процедуры. На первом этапе организуется передача сообщения всем процессорам сети, располагающимся на той же горизонтали решетки, что и процессор-инициатор передачи; на втором этапе процессоры, получившие копию данных на первом этапе, рассылают сообщения по своим соответствующим вертикалям. Оценка длительности операции рассылки в соответствии с описанным алгоритмом определяется соотношением

$$t_{пд} = 2(t_н + mt_к)\lceil \sqrt{p}/2 \rceil$$

Для **гиперкуба** рассылка может быть выполнена в ходе N- этапной процедуры передачи данных. На первом этапе процессор-источник сообщения передает данные одному из своих соседей (например, по первой размерности) – в результате после первого этапа имеется два процессора, имеющих копию пересылаемых данных (данный результат можно интерпретировать также как разбиение исходного гиперкуба на два таких одинаковых по размеру гиперкуба размерности $N-1$, что каждый из них имеет копию исходного сообщения). На втором этапе два процессора, задействованные на первом этапе, пересылают сообщение своим соседям по второй размерности и т.д. В результате такой рассылки время операции оценивается при помощи выражения

$$t_{пд} = (t_н + mt_к)\log_2 p$$

Сравнивая полученные выражения для длительности выполнения операции рассылки, можно отметить, что наилучшие показатели имеет топология типа гиперкуба; более того, можно показать, что данный результат является наилучшим для выбранного способа коммуникации с помощью передачи сообщений.

Передача пакетов. Для топологии типа **кольца** алгоритм рассылки может быть получен путем логического представления кольцевой структуры сети в виде гиперкуба. В результате на этапе рассылки процессор-источник сообщения передает данные процессору, находящемуся на расстоянии *p/2* от исходного процессора. Далее, на втором этапе оба процессора, уже имеющие рассылаемые данные после первого этапа, передают сообщения процессорам, находящиеся на расстоянии *p/4* и т.д. Трудоемкость выполнения операции рассылки при таком методе передачи данных определяется соотношением

$$t_{пд} = \sum_{i=1}^{\log_2 p}(t_н + mt_к + t_c p/2^i) = (t_н + mt_к)\log_2 p + t_c(p-1)$$

(как и ранее, при достаточно больших сообщениях, временем передачи служебных данных можно пренебречь).

Для топологии типа **решетки-тора** алгоритм рассылки может быть получен из способа передачи данных, примененного для кольцевой структуры сети, в соответствии с тем же способом обобщения, что и в случае использования метода передачи сообщений. Получаемый в результате такого обобщения алгоритм рассылки характеризуется следующим соотношением для оценки времени выполнения:

$$t_{пд} = (t_н + mt_к)\log_2 p + 2t_c(\sqrt{p} - 1) .$$

Для **гиперкуба** алгоритм рассылки (и, соответственно, временные оценки длительности выполнения) при передаче пакетов не отличается от варианта для метода передачи сообщений.

3.2.3. Передача данных от всех процессоров всем процессорам сети

Операция передачи данных от всех процессоров всем процессорам сети (*all-to-all broadcast* or *multinode broadcast*) является естественным обобщением одиночной операции рассылки; двойственная операция передачи – прием сообщений на каждом процессоре от всех процессоров сети (*multinode accumulation*). Подобные операции широко используются, например, при реализации матричных вычислений.

Возможный способ реализации операции множественной рассылки состоит в выполнении соответствующего набора операций одиночной рассылки. Однако такой подход не является оптимальным для многих топологий сети, поскольку часть необходимых операций одиночной рассылки потенциально может быть выполнена параллельно. Как и ранее, материал будет рассматриваться раздельно для разных методов передачи данных (см. п. 3.1.2).

Передача сообщений. Для **кольцевой топологии** каждый процессор может инициировать рассылку своего сообщения одновременно (в каком-либо выбранном направлении по кольцу). В любой момент времени каждый процессор выполняет прием и передачу данных; завершение операции множественной рассылки произойдет через (*p-1*) цикл передачи данных. Длительность выполнения операции рассылки оценивается соотношением:

$$t_{пд} = (t_н + mt_к)(p-1) .$$

Для топологии типа **решетки-тора** множественная рассылка сообщений может быть выполнена при помощи алгоритма, получаемого обобщением способа передачи данных для кольцевой структуры сети. Схема обобщения состоит в следующем. На первом этапе организуется передача сообщений раздельно по всем процессорам сети, располагающимся на одних и тех же горизонталях решетки (в результате на каждом процессоре одной и той же горизонтали формируются укрупненные сообщения размера $m\sqrt{p}$, объединяющие все сообщения горизонтали). Время выполнения этапа

$$t'_{пд} = (t_н + mt_к)(\sqrt{p} - 1) .$$

На втором этапе рассылка данных выполняется по процессорам сети, образующим вертикали решетки. Длительность этого этапа

$$t''_{пд} = (t_н + m\sqrt{p}t_к)(\sqrt{p} - 1) .$$

Как результат, общая длительность операции рассылки определяется соотношением:

$$t_{пд} = 2t_н(\sqrt{p} - 1) + mt_к(p-1).$$

Для **гиперкуба** алгоритм множественной рассылки сообщений может быть получен путем обобщения ранее описанного способа передачи данных для топологии типа решетки на размерность гиперкуба *N*. В результате такого обобщения схема коммуникации состоит в следующем. На каждом этапе i, $1 \le i \le N$, выполнения алгоритма функционируют все процессоры сети, которые обмениваются своими данными со своими соседями по i размерности и формируют объединенные сообщения. Время операции рассылки может быть получено при помощи выражения

$$t_{по} = \sum_{i=1}^{\log_2 p} (t_н + 2^{i-1} m t_к) = t_н \log_2 p + m t_к (p-1).$$

Передача пакетов. Применение более эффективного для кольцевой структуры и топологии типа решетки-тора метода передачи данных не приводит к какому-либо улучшению времени выполнения операции множественной рассылки, поскольку обобщение алгоритмов выполнения операции одиночной рассылки на случай множественной рассылки приводит к перегрузке каналов передачи данных (т.е. к существованию ситуаций, когда в один и тот же момент времени для передачи по одной и той линии передачи имеется несколько ожидающих пересылки пакетов данных). Перегрузка каналов приводит к задержкам при пересылках данных, что и не позволяет проявиться всем преимуществам метода передачи пакетов.

Возможным широко распространенным примером операции множественной рассылки является задача **редукции** (*reduction*), определяемая в общем виде как процедура выполнения той или иной обработки данных, получаемых на каждом процессоре в ходе множественной рассылки (в качестве примера такой задачи может быть рассмотрена проблема вычисления суммы значений, находящихся на разных процессорах, и рассылки полученной суммы по всем процессорам сети). Способы решения задачи редукции могут состоять в следующем:

– непосредственный подход заключается в выполнении операции множественной рассылки и последующей затем обработке данных на каждом процессоре в отдельности;

– более эффективный алгоритм может быть получен в результате применения операции одиночного приема данных на отдельном процессоре, выполнения на этом процессоре действий по обработке данных, и рассылке полученного результата обработки всем процессорам сети;

– наилучший же способ решения задачи редукции состоит в совмещении процедуры множественной рассылки и действий по обработке данных, когда каждый процессор сразу же после приема очередного сообщения реализует требуемую обработку полученных данных (например, выполняет сложение полученного значения с имеющейся на процессоре частичной суммой). Время решения задачи редукции при таком алгоритме реализации в случае, например, когда размер пересылаемых данных имеет единичную длину ($m = 1$) и топология сети имеет структуру гиперкуба, определяется выражением

$$t_{по} = (t_н + t_к) \log_2 p .$$

Другим типовым примером использования операции множественной рассылки является **задача нахождения частных сумм** последовательности значений S_i (в литературе эта задача известна под названием *prefix sum problem*)

$$S_k = \sum_{i=1}^{k} x_i, \ 1 \le k \le p$$

(будем предполагать, что количество значений совпадает с количеством процессоров, значение x_i располагается на i процессоре и результат S_k должен получаться на процессоре с номером k).

Алгоритм решения данной задачи также может быть получен при помощи конкретизации общего способа выполнения множественной операции рассылки, когда процессор выполняет суммирование полученного значения (но только в том случае, если процессор-отправитель значения имеет меньший номер, чем процессор-получатель).

3.2.4. Обобщенная передача данных от одного процессора всем остальным процессорам сети

Общий случай передачи данных от одного процессора всем остальным процессорам сети состоит в том, что все рассылаемые сообщения являются различными (*one-to-all personalized communication* or *single-node scatter*). Двойственная операция передачи для данного типа взаимодействия процессоров – обобщенный прием сообщений (*single-node gather*) на одном процессоре от всех остальных процессоров сети (отличие данной операции от ранее рассмотренной процедуры сборки данных на одном процессоре (single-node accumulation) состоит в том, что обобщенная операция сборки не предполагает какого-либо взаимодействия сообщений (типа редукции) в процессе передачи данных).

Трудоемкость операции обобщенной рассылки сопоставима со сложностью выполнения процедуры множественной передачи данных. Процессор-инициатор рассылки посылает каждому процессору сети сообщение размера m и, тем самым, нижняя оценка длительности выполнения операции характеризуется величиной $m t_к (p-1)$.

Проведем более подробный анализ трудоемкости обобщенной рассылки для случая топологии типа **гиперкуб**. Возможный способ выполнения операции состоит в следующем. Процессор-инициатор рассылки

передает половину своих сообщений одному из своих соседей (например, по первой размерности) – в результате, исходный гиперкуб становится разделенным на два гиперкуба половинного размера, в каждом из которых содержится ровно половина исходных данных. Далее действия по рассылке сообщений могут быть повторены и общее количество повторений определяется исходной размерностью гиперкуба. Длительность операции обобщенной рассылки может быть охарактеризована соотношением:

$$t_{пд} = t_н \log_2 p + mt_к (p-1)$$

(как и отмечалась выше, трудоемкость операции совпадает с длительностью выполнения процедуры множественной рассылки).

3.2.5. Обобщенная передача данных от всех процессоров всем процессорам сети

Обобщенная передача данных от всех процессоров всем процессорам сети (*total exchange*) представляет собой наиболее общий случай коммуникационных действий. Необходимость в выполнении подобных операций возникает в параллельных алгоритмах быстрого преобразования Фурье, транспонирования матриц и др.

Выполним краткую характеристику возможных способов выполнения обобщенной множественной рассылки для разных методов передачи данных (см. п. 3.1.2).

Передача сообщений. Общая схема алгоритма для **кольцевой топологии** состоит в следующем. Каждый процессор производит передачу всех своих исходных сообщений своему соседу (в каком-либо выбранном направлении по кольцу). Далее процессоры осуществляют прием направленных к ним данных, затем среди принятой информации выбирают свои сообщения, после чего выполняет дальнейшую рассылку оставшейся части данных. Длительность выполнения подобного набора передач данных оценивается при помощи выражения:

$$t_{пд} = (t_н + \frac{1}{2} mpt_к)(p-1).$$

Способ получения алгоритма рассылки данных для топологии типа **решетки-тора** является тем же самым, что и в случае рассмотрения других коммуникационных операций. На первом этапе организуется передача сообщений раздельно по всем процессорам сети, располагающимся на одних и тех же горизонталях решетки (каждому процессору по горизонтали передаются только те исходные сообщения, что должны быть направлены процессорам соответствующей вертикали решетки); после завершения этапа на каждом процессоре собираются p сообщений, предназначенных для рассылки по одной из вертикалей решетки. На втором этапе рассылка данных выполняется по процессорам сети, образующим вертикали решетки. Общая длительность всех операций рассылок определяется соотношением

$$t_{пд} = (2t_н + mpt_к)(\sqrt{p}-1).$$

Для **гиперкуба** алгоритм обобщенной множественной рассылки сообщений может быть получен путем обобщения способа выполнения операции для топологии типа решетки на размерность гиперкуба N. В результате такого обобщения схема коммуникации состоит в следующем. На каждом этапе i, $1 \le i \le N$, выполнения алгоритма функционируют все процессоры сети, которые обмениваются своими данными со своими соседями по i размерности и формируют объединенные сообщения. При организации взаимодействия двух соседей канал связи между ними рассматривается как связующий элемент двух равных по размеру подгиперкубов исходного гиперкуба, и каждый процессор пары посылает другому процессору только те сообщения, что предназначены для процессоров соседнего подгиперкуба. Время операции рассылки может быть получено при помощи выражения:

$$t_{пд} = (t_н + \frac{1}{2} mpt_к)\log_2 p$$

(кроме затрат на пересылку, каждый процессор выполняет $mp\log_2 p$ операций по сортировке своих сообщений перед обменом информацией со своими соседями).

Передача пакетов. Как и в случае множественной рассылки, применение метода передачи пакетов не приводит к улучшению временных характеристик для операции обобщенной множественной рассылки. Рассмотрим для примера более подробно выполнение данной коммуникационной операции для сети с топологией типа **гиперкуб**. В этом случае рассылка может быть выполнена за $p-1$ последовательных итераций. На каждой итерации все процессоры разбиваются на взаимодействующие пары процессоров, причем это разбиение на пары может быть выполнено таким образом, чтобы передаваемые между разными парами сообщения не использовали одни и те же пути передачи данных. Как результат, общая длительность операции обобщенной рассылки может быть определена в соответствии с выражением:

$$t_{пд} = (t_н + mt_к)(p-1) + \frac{1}{2} t_c p \log_2 p.$$

3.2.6. Циклический сдвиг

Частный случай обобщенной множественной рассылки есть процедура перестановки (*permutation*), представляющая собой операцию перераспределения информации между процессорами сети, в которой каждый процессор передает сообщение определенному неким способом другому процессору сети. Конкретный вариант перестановки – *циклический q-сдвиг* (*cirlular q-shift*), при котором каждый процессор i, $1 \le i \le N$, передает данные процессору с номером $(i+q) \bmod p$. Подобная операция сдвига используется, например, при организации матричных вычислений.

Поскольку выполнение циклического сдвига для кольцевой топологии может быть обеспечено при помощи простых алгоритмов передачи данных, рассмотрим возможные способы выполнения данной коммуникационной операции только для топологий решетки-тора и гиперкуба при разных методах передачи данных (см. п. 3.1.2).

Передача сообщений. Общая схема алгоритма циклического сдвига для топологии типа **решетки-тора** состоит в следующем. Пусть процессоры перенумерованы по строкам решетки от *0* до $p-1$. На первом этапе организуется циклический сдвиг с шагом $q \bmod \sqrt{p}$ по каждой строке в отдельности (если при реализации такого сдвига сообщения передаются через правые границы строк, то после выполнения каждой такой передачи необходимо осуществить компенсационный сдвиг вверх на 1 для процессоров первого столбца решетки). На втором этапе реализуется циклический сдвиг вверх с шагом $\lfloor q/\sqrt{p} \rfloor$ для каждого столбца решетки. Общая длительность всех операций рассылок определяется соотношением

$$t_{пд} = (t_н + mt_к)(2\lfloor \sqrt{p}/2 \rfloor + 1) .$$

Для **гиперкуба** алгоритм циклического сдвига может быть получен путем логического представления топологии гиперкуба в виде кольцевой структуры. Для получения такого представления установим взаимно-однозначное соответствие между вершинами кольца и гиперкуба. Необходимое соответствие может быть получено, например, при помощи известного кода Грея, который можно использовать для определения процессоров гиперкуба, соответствующих конкретным вершинам кольца. Более подробное изложение механизма установки такого соответствия осуществляется в подразделе 3.3; для наглядности на рис. 3.1 приводится вид гиперкуба для размерности $N=3$ с указанием для каждого процессора гиперкуба соответствующей вершины кольца. Положительным свойством выбора такого соответствия является тот факт, что для любых двух вершин в кольце, расстояние между которыми является равным $l=2^i$ для некоторого значения i, путь между соответствующим вершинами в гиперкубе содержит только две линии связи (за исключением случая $i=0$, когда путь в гиперкубе имеет единичную длину).

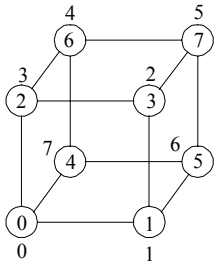

Рис. 3.1. Схема отображения гиперкуба на кольцо (в кружках приведены номера процессоров гиперкуба)

Представим величину сдвига q в виде двоичного кода. Количество ненулевых позиций кода определяет количество этапов в схеме реализации операции циклического сдвига. На каждом этапе выполняется операция сдвига с величиной шага, определяемой наиболее старшей ненулевой позицией значения q (например, при исходной величине сдвига $q=5=101_2$, на первом этапе выполняется сдвиг с шагом 4, на втором этапе шаг сдвига равен 1). Выполнение каждого этапа (кроме сдвига с шагом 1) состоит в передаче данных по пути, включающему две линии связи. Как результат, верхняя оценка для длительности выполнения операции циклического сдвига определяется соотношением:

$$t_{пд} = (t_н + mt_к)(2\log_2 p - 1) .$$

Передача пакетов. Использование пересылки пакетов может повысить эффективность выполнения операции циклического сдвига для топологии **гиперкуб**. Реализация всех необходимых коммуникационных действий в этом случае может быть обеспечена путем отправления каждым процессором всех пересылаемых данных непосредственно процессорам назначения. Использование метода покоординатной маршрутизации (см. п. 3.1.1) позволит избежать коллизий при использовании линий передачи данных (в каждый момент времени для каждого канала будет существовать не более одного готового для отправки сообщения). Длина наибольшего пути при такой рассылке данных определяется как $\log_2 p - \gamma(q)$, где $\gamma(q)$ есть наибольшее

целое значение j такое, что 2^j есть делитель величины сдвига q. Тогда длительность операции циклического сдвига может быть определена при помощи выражения

$$t_{по} = t_н + mt_к + t_c(\log_2 p - \gamma(q))$$

(при достаточно больших размерах сообщений временем передачи служебных данных можно пренебречь и время выполнения операции может быть определено как $t_{по} = t_н + mt_к$).

3.3. Методы логического представления топологии коммуникационной среды

Как показало рассмотрение основных коммуникационных операций в п. 3.1, ряд алгоритмов передачи данных допускает более простое изложение при использовании вполне определенных топологий сети межпроцессорных соединений. Кроме того, многие методы коммуникации могут быть получены при помощи того или иного логического представления исследуемой топологии. Как результат, важным моментом является при организации параллельных вычислений возможность *логического представления разнообразных топологий* на основе конкретных (физических) межпроцессорных структур.

Способы логического представления (отображения) топологий характеризуются следующими тремя основными характеристиками:

- **уплотнение дуг** (*congestion*), выражаемое как максимальное количество дуг логической топологии, отображаемых в одну линию передачи физической топологии;
- **удлинение дуг** (*dilation*), определяемое как путь максимальной длины физической топологии, на который отображаемая дуга логической топологии;
- **увеличение вершин** (*expansion*), вычисляемое как отношение количества вершин в логической и физической топологиях.

Для рассматриваемых в рамках пособия топологий ограничимся изложением вопросов отображения топологий кольца и решетки на гиперкуб; предлагаемые ниже подходы для логического представления топологий характеризуются единичными показателями уплотнения и удлинения дуг.

3.3.1. Представление кольцевой топологии в виде гиперкуба

Установление соответствия между кольцевой топологией и гиперкубом может быть выполнено при помощи *двоичного рефлексивного кода Грея* G(i,N) (*binary reflected Gray code*), определяемого в соответствии с выражениями:

$$G(0,1) = 0, \quad G(1,1) = 1,$$

$$G(i, s+1) = \begin{cases} G(i,s), & i < 2^s, \\ 2^s + G(2^{s+1} - 1 - i, s), & i \geq 2^s, \end{cases}$$

где i задает номер значения в коде Грея, а N есть длина этого кода. Для иллюстрации подхода на рис. 3.2 показывается отображение кольцевой топологии на гиперкуб для сети из $p = 8$ процессоров.

Код Грея для N=1	Код Грея для N=2	Код Грея для N=3	Номера процессоров	
			гиперкуба	кольца
0	0 0	0 0 0	0	0
1	0 1	0 0 1	1	1
	1 1	0 1 1	3	2
	1 0	0 1 0	2	3
		1 1 0	6	4
		1 1 1	7	5
		1 0 1	5	6
		1 0 0	4	7

Рис. 3.2. Отображение кольцевой топологии на гиперкуб при помощи кода Грея

Важным свойством кода Грея является тот факт, что соседние значения $G(i,N)$ и $G(i+1,N)$ имеют только одну различающуюся битовую позицию. Как результат, соседние вершины в кольцевой топологии отображаются на соседние процессоры в гиперкубе.

3.3.2. Отображение топологии решетки на гиперкуб

Отображение топологии решетки на гиперкуб может быть выполнено в рамках подхода, использованного для кольцевой структуры сети. Тогда для отображения решетки $2^r x 2^s$ на гиперкуб размерности $N=r+s$ можно принять правило, что элементу решетки с координатами (i,j), будет соответствовать процессор гиперкуба с номером

$$G(i,r)\,||\,G(j,s),$$

где операция $||$ означает конкатенацию кодов Грея.

3.4. Оценка трудоемкости операций передачи данных для кластерных систем

Для кластерных вычислительных систем (см. п. 1.2.2) одним из широко применяемых способов построения коммуникационной среды является использование концентраторов (*hub*) или переключателей (*switch*) для объединения процессорных узлов кластера в единую вычислительную сеть. В этих случаях топология сети кластера представляет собой *полный граф*, в котором, однако, имеются определенные ограничения на одновременность выполнения коммуникационных операций. Так, при использовании концентраторов передача данных в каждый текущий момент времени может выполняться только между двумя процессорными узлами; переключатели могут обеспечивать взаимодействие нескольких непересекающихся пар процессоров.

Другое часто применяемое решение при создании кластеров состоит в использовании *метода передачи пакетов* (реализуемого, как правило, на основе протокола TCP/IP) в качестве основного способа выполнения коммуникационных операций.

1. Если выбрать для дальнейшего анализа кластеры данного распространенного типа (топология в виде полного графа, пакетный способ передачи сообщений), то трудоемкость операции коммуникации между двумя процессорными узлами может быть оценена в соответствии с выражением (*модель А*)

$$t_{пд}(m) = t_н + m * t_k + t_c,$$

оценка подобного вида следует из соотношений для метода передачи пакетов при единичной длине пути передачи данных, т.е. при $l = 1$. Отмечая возможность подобного подхода, вместе с этим можно заметить, что в рамках рассматриваемой модели время подготовки данных $t_н$ предполагается постоянным (не зависящим от объема передаваемых данных), время передачи служебных данных t_c не зависит от количества передаваемых пакетов и т.п. Эти предположения не в полной мере соответствуют действительности и временные оценки, получаемые в результате использования модели, могут не обладать необходимой точностью.

2. Учитывая все приведенные замечания, схема построения временных оценок может быть уточнена; в рамках новой расширенной модели трудоемкость передачи данных между двумя процессорами определяется в соответствии со следующими выражениями (*модель В*):

$$t_{пд} = \begin{cases} t_{нач_0} + m \cdot t_{нач_1} + (m + V_c) \cdot t_к, & n = 1 \\ t_{нач_0} + (V_{\max} - V_c) \cdot t_{нач_1} + (m + V_c \cdot n) \cdot t_к, & n > 1 \end{cases},$$

где $n = \lceil m/(V_{\max} - V_c) \rceil$ есть количество пакетов, на которое разбивается передаваемое сообщение, величина $V_{\max}$ определяет максимальный размер пакета, который может быть доставлен в сети (по умолчанию для операционной системы MS Windows в сети Fast Ethernet $V_{\max}$=1500 байт), а V_c есть объем служебных данных в каждом из пересылаемых пакетов (для протокола TCP/IP, ОС Windows 2000 и сети Fast Ethernet V_c=78 байт). Поясним также, что в приведенных соотношениях константа $t_{нач_0}$ характеризует аппаратную составляющую латентности и зависит от параметров используемого сетевого оборудования, значение $t_{нач_1}$ задает время подготовки одного байта данных для передачи по сети. Как результат, величина латентности

$$t_н = t_{нач_0} + v \cdot t_{нач_1}$$

увеличивается линейно в зависимости от объема передаваемых данных. При этом предполагается, что подготовка данных для передачи второго и всех последующих пакетов может быть совмещена с пересылкой по сети предшествующих пакетов и латентность, тем самым, не может превышать величины

$$t_н = t_{нач_0} + (V_{\max} - V_c) \cdot t_{нач_1} .$$

Помимо латентности, в предлагаемых выражениях для оценки трудоемкости коммуникационной операции можно уточнить также правило вычисления времени передачи данных

$$(m + V_c \cdot n) \cdot t_к ,$$

что позволяет теперь учитывать эффект увеличения объема передаваемых данных при росте числа пересылаемых пакетов за счет добавления служебной информации (заголовков пакетов).

3. Завершая анализ проблемы построения теоретических оценок трудоемкости коммуникационных операций, следует отметить, что для практического применения перечисленных моделей необходимо выполнить оценку значений параметров используемых соотношений. В этом отношении полезным может оказаться использование и более простых способов вычисления временных затрат на передачу данных – одной из известных схем подобного вида является подход (*модель Хокни* (*Hockney*) – см., например, Hockney (1994)), в котором трудоемкость операции коммуникации между двумя процессорными узлами кластера оценивается в соответствии с выражением (*модель C*)

$$t_{пд}(m) = t_н + m t_к .$$

4. Для проверки адекватности рассмотренных моделей реальным процессам передачи данных приведем результаты выполненных экспериментов в сети многопроцессорного кластера Нижегородского университета (компьютеры IBM PC Pentium 4 1300 Мгц и сеть Fast Etherrnet). При проведении экспериментов для реализации коммуникационных операций использовалась библиотека MPI.

Часть экспериментов была выполнена для оценки параметров моделей:

- значение латентности $t_н$ для моделей *A* и *C* определялось как время передачи сообщения нулевой длины;
- величина пропускной способности R устанавливалась максимально наблюденной в ходе экспериментов скорости передачи данных, т.е.

$$R = \max_m \quad (t_{пд}(m)/m) ,$$

и полагалось $t_к = 1/R$;

- значения величин $t_{нач_0}$ и $t_{нач_1}$ оценивались при помощи линейной аппроксимации времен передачи сообщений размера от 0 до $V_{\max}$.

В ходе экспериментов осуществлялась передача данных между двумя процессорами кластера, размер передаваемых сообщений варьировался от 0 до 8 Мб. Для получения более точных оценок выполнение каждой операции осуществлялось многократно (более 100000 раз), после чего результаты временных замеров усреднялись. Для иллюстрации ниже приведен результат одного эксперимента, при проведении которого размер передаваемых сообщений изменялся от 2000 до 60000 байт.

В табл. 3.3 приводится ряд числовых данных по погрешности рассмотренных моделей трудоемкости коммуникационных операций (величина погрешности дается в виде относительного отклонения от реального времени выполнения операции передачи данных).

Таблица 3.3. Погрешность моделей трудоемкости операций передачи данных (по результатам вычислительных экспериментов)

Объем сообщения (байт)	Время передачи (мкс)	Погрешность теоретической оценки времени передачи данных, в %		
		Модель A	Модель B	Модель C
2000	495	33.45%	7.93%	34.80%
10000	1184	13.91%	1.70%	14.48%
20000	2055	8.44%	0.44%	8.77%
30000	2874	4.53%	-1.87%	4.76%
40000	3758	4.04%	-1.38%	4.22%
50000	4749	5.91%	1.21%	6.05%
60000	5730	6.97%	2.73%	7.09%

Как можно заметить по результатам проведенных экспериментов, оценки трудоемкости операций передачи данных по модели *B* имеют меньшую погрешность.

Вместе с этим важно отметить, что для предварительного анализа временных затрат на выполнение коммуникационных операций точности модели *C* может оказаться достаточно. Кроме того, данная модель

носит наиболее простой вид среди всех рассмотренных моделей. С учетом последнего обстоятельства, далее во всех последующих разделах для оценки трудоемкости операций передачи данных будет применяться именно модель *C* (модель Хокни); при этом для модели будет использоваться форма записи, более широко используемая в литературе:

$$t_{пд}(m) = \alpha + m/\beta ,$$

где α есть латентность сети передачи данных (т.е., $\alpha = t_н$), а β обозначает пропускную способность сети (т.е., $\beta = R = 1/t_к$).

3.5. Краткий обзор раздела

Данный раздел посвящен оценке коммуникационной сложности параллельных алгоритмов.

В подразделе 3.1 представлена общая характеристика алгоритмов маршрутизации и методов передачи данных. Для подробного рассмотрения выделены *метод передачи сообщений* (*store-and-forward routing*) и метод передачи пакетов (*cut-through routing*), для которых определены оценки времени выполнения коммуникационных операций.

В подразделе 3.2 определены основные типы операций передачи данных, выполняемых в ходе параллельных вычислений. К основным коммуникационным операциям относятся:

- Передача данных между процессорами сети,
- Передача данных от одного процессора всем остальным процессорам сети (*one-to-all broadcast* or *single-node broadcast*) и обратная операция приема на одном процессоре сообщений от всех остальных процессоров сети (*single-node accumulation*),
- Передача данных от всех процессоров всем процессорам сети (*all-to-all broadcast* or *multinode broadcast*) и обратная операция приема сообщений на каждом процессоре от всех процессоров сети (*multinode accumulation*),
- Обобщенная[1)] передачи данных от одного процессора всем остальным процессорам сети (*one-to-all personalized communication* or *single-node scatter*) и обратная операция обобщенного приема сообщений на одном процессоре от всех остальных процессоров сети (*single-node gather*),
- Обобщенная передача данных от всех процессоров всем процессорам сети (*total exchange*).

Для всех перечисленных операций передачи данных рассмотрены алгоритмы их выполнения на примере топологий кольца, решетки и гиперкуба. Для каждого из представленных алгоритмов приведены оценки их временной трудоемкости как для метода передачи сообщений, так и для метода передачи пакетов.

В подразделе 3.3 рассмотрены *методы логического представления топологий* на основе конкретных (физических) межпроцессорных структур. Использование логических топологий позволяет получить более простое изложение для ряда алгоритмов передачи данных, снизить затраты на реализацию коммуникационных операций и т.п.

В подразделе 3.4 обсуждаются более подробно модели, при помощи которых могут быть получены оценки времени выполнения операций передачи данных для кластерных вычислительных систем. Точность формирования временных оценок сравнивается при помощи проведения вычислительных экспериментов. По результатам экспериментов определена наиболее точная модель (модель *B*). Кроме того, отмечается, что для предварительного анализа временной трудоемкости коммуникационных операций целесообразно использовать более простую модель - модель *C* (*модель Хокни*).

3.6. Обзор литературы

В качестве дополнительного учебного материала для данного раздела могут быть рекомендованы работы Kumar (1994) и Quinn (2003).

Вопросы построения моделей для оценки времени выполнения коммуникационных операций широко обсуждаются в литературе. При изучении раздела могут быть полезны работы Culler, et al. (1996), Skillicorn and Talia (1998), Andrews (2000). *Модель Хокни* впервые была опубликована в Hockney (1994). Модель *B* из подраздела 3.4 представлена в работе Гергель, Стронгин (2001).

3.7. Контрольные вопросы

1. Какие основные характеристики используются для оценки топологии сети передачи данных? Приведите значения характеристик для конкретных типов коммуникационных структур (полный граф, линейка, решетка и др.).

[1)] Обобщение операции состоит в том, что разным процессорам рассылаются различные сообщения; для обратной операции приема все собираемые сообщения также являются разными.

2. Какие основные методы применяются при маршрутизации передаваемых данных по сети?

3. В чем состоят основные методы передачи данных? Приведите для этих методов аналитические оценки времени выполнения?

4. Какие операции передачи данных могут быть выделены в качестве основных?

5. В чем состоят алгоритмы выполнения передачи данных от одного процессора всем процессорам сети для топологий кольца, решетки и гиперкуба? Приведите оценки временной трудоемкости для этих алгоритмов.

6. В чем состоят алгоритмы выполнения передачи данных от всех процессоров всем процессорам сети для топологий кольца, решетки и гиперкуба? Приведите оценки временной трудоемкости для этих алгоритмов.

7. В чем состоят возможные алгоритмы выполнения операции редукции? Какой из алгоритмов является наилучшим по времени выполнения?

8. В чем состоят алгоритм выполнения операции циклического сдвига?

9. В чем состоит полезность использования логических топологий? Приведите примеры алгоритмов логического представления структуры коммуникационной сети?

10. В чем различие моделей для оценки времени выполнения операций передачи данных в кластерных вычислительных системах? Какая модель является более точной? Какая модель может быть использована для предварительного анализа временной трудоемкости коммуникационных операций?

3.8. Задачи и упражнения

1. Разработайте алгоритмы выполнения основных операций передачи данных для топологии сети в виде 3-мерной решетки.

2. Разработайте алгоритмы выполнения основных операций передачи данных для топологии сети в виде двоичного дерева.

3. Примените модель B из подраздела 3.4 для оценки временной сложности операций передачи данных. Сравните получаемые показатели.

4. Примените модель C из подраздела 3.4 для оценки временной сложности операций передачи данных. Сравните получаемые показатели.

5. Разработайте алгоритмы логического представления двоичного дерева для различных физических топологий сети.

Литература

Гергель, В.П., Стронгин, Р.Г. (2001). Основы параллельных вычислений для многопроцессорных вычислительных систем. - Н.Новгород, ННГУ (2 изд., 2003).

Culler, D.E., et al. (1996). LogP: A practical model for parallel computation. – Comm. Of the ACM, 39, 11, pp. 75-85.

Hockney, R. (1994). The communication challenge for MPP: Intel Paragon and Meiko CS-2. – Parallel Computing, 20 (3), pp. 389-398.

Kumar V., Grama, A., Gupta, A., Karypis, G. (1994). Introduction to Parallel Computing. - The Benjamin/Cummings Publishing Company, Inc. (2nd edn., 2003)

Quinn, M. J. (2004). Parallel Programming in C with MPI and OpenMP. – New York, NY: McGraw-Hill.

Skillicorn, D.B., Talia, D. (1998). Models and languages for parallel computation. – ACM Computing surveys, 30, 2.