Нижегородский государственный университет им. Н.И.Лобачевского

Факультет Вычислительной математики и кибернетики

Образовательный комплекс

Введение в методы параллельного программирования

Раздел 6.

Принципы разработки параллельных методов

Гергель В.П., профессор, д.т.н.
Кафедра математического обеспечения ЭВМ

Содержание

- Моделирование параллельных программ
- Методика разработки параллельных алгоритмов
  - Разделение вычислений на независимые части
  - Выделение информационных зависимостей
  - Масштабирование имеющегося набора подзадач
  - Распределение подзадач между процессорами
- Пример применение методики - параллельное решение гравитационной задачи N тел
- Заключение

Введение…

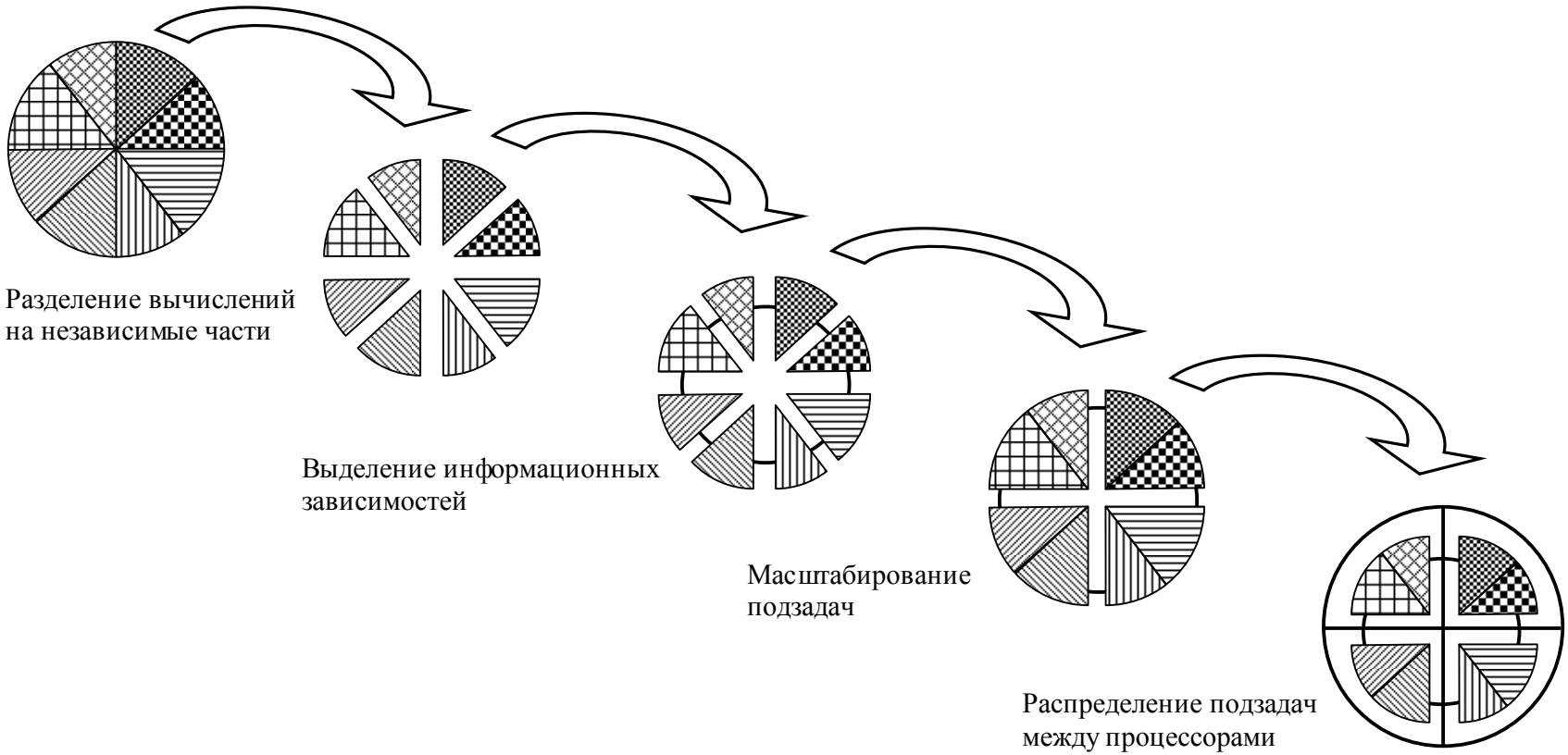
- Для определения эффективных способов организации параллельных вычислений необходимо:
  - Выполнить анализ имеющихся вычислительных схем и осуществить их разделение (*декомпозицию*) на части (*подзадачи*), которые могут быть реализованы независимо друг от друга,
  - Выделить для набора подзадач *информационные взаимодействия*,
  - Определить необходимую (или доступную) для решения задачи *вычислительную систему* и выполнить *распределение* имеющего набора подзадач между процессорами системы.

Введение…

- Объем вычислений для каждого процессора должен быть примерно одинаков – это позволит обеспечить равномерную вычислительную загрузку (*балансировку*) процессоров
- Распределение подзадач между процессорами должно быть выполнено таким образом, чтобы наличие информационных связей (*коммуникационных взаимодействий*) между подзадачами было минимальным.

Введение…

- Схема разработки параллельных алгоритмов



Введение…

- После выполнения всех этапов проектирования необходимо оценить эффективность разрабатываемых параллельных методов
- По результатам проведенного анализа может оказаться необходимым повторение некоторых (или всех) этапов разработки:
  - корректировка состава сформированного множества задач - подзадачи могут быть укрупнены (*агрегированы*) или *детализированы*. Данные действия могут быть определены как *масштабирование* разрабатываемого алгоритма.

Введение

- На следующем шаге необходимо выполнить разработку программ для решения сформированного набора подзадач
- Для проведения вычислений программы запускаются на выполнение, для реализации информационных взаимодействий программы должны иметь в своем распоряжении *каналы передачи сообщений*
- Обычно каждый процессор выделяется для решения одной подзадачи, но при наличии большого количества подзадач на процессорах может выполняться одновременно несколько программ (*процессов*)

Моделирование параллельных программ…

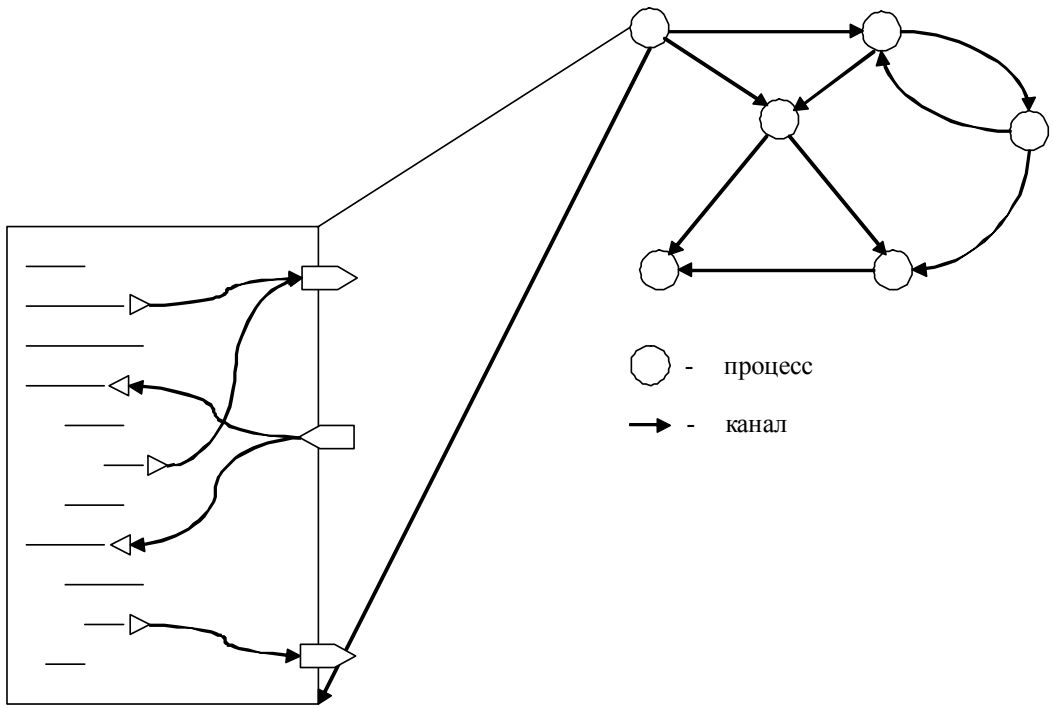
- На стадии проектирования параллельный метод может быть представлен в виде ***графа "подзадачи – сообщения"*** (*агрегированное* представление графа *"операции-операнды"*)
- Модель "подзадачи - сообщения" позволяет:
  - Определить подзадачи одинаковой вычислительной сложности,
  - Обеспечить низкий уровень информационной зависимости между подзадачами.

Моделирование параллельных программ…

- На стадии выполнения для описания параллельной программы может быть использована модель в виде ***графа "процессы – каналы"***, в которой вместо подзадач используется понятие процессов, вместо информационных зависимостей - каналы передачи сообщений)
- Модель *"процессы – каналы"* позволяет:
  - Осуществить оптимальное распределение подзадач по процессорам,
  - Выполнить анализ эффективности разработанного параллельного метода,
  - Обеспечить возможность контроля и управления процессом выполнения параллельных вычислений.

Моделирование параллельных программ…

- Модель параллельной программы в виде графа "процессы-каналы"…



- операции приема (передачи)

- входные (выходные) каналы для взаимодействия процессов

Моделирование параллельных программ

- **Модель "процессы-каналы"**:
  - *Процесс* - выполняемая на процессоре *программа*, использует для свой работы часть локальной *памяти* процессора и содержит ряд *операций приема/передачи* данных для организации информационного взаимодействия между выполняемыми процессами параллельной программы,
  - *Канал передачи данных* - *очередь сообщений*, в которую один или несколько процессов могут отправлять пересылаемые данные и из которой процесс-адресат может извлекать сообщения:
    - каналы возникают динамически в момент выполнения первой операции приема/передачи с каналом,
    - канал может соответствовать одной или нескольким командам приема/передачи данных различных процессов,
    - емкость канала неограничена,
    - операции приема сообщений могут приводить к *блокировкам* (запрашиваемые данные еще не были отправлены процессами-источниками).

Методика разработки параллельных алгоритмов…

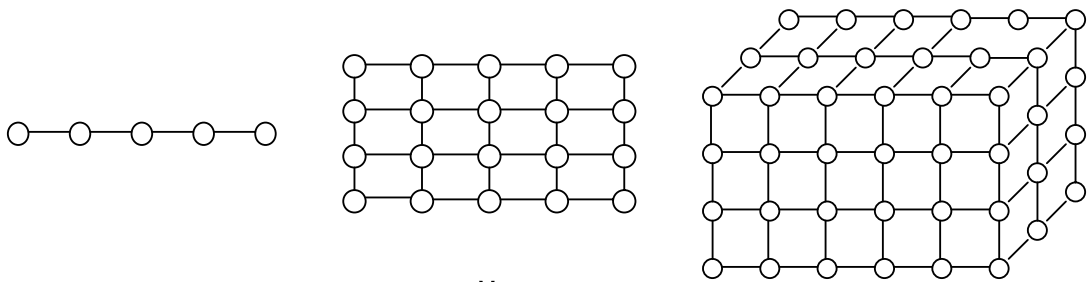
- Для разработки параллельных алгоритмов необходимо выполнить:
  - выделение подзадач,
  - определение информационных зависимостей,
  - масштабирование,
  - распределения подзадач по процессорам вычислительной системы
- Для демонстрации рекомендаций будем использовать задачу поиска максимального значения среди элементов матрицы ***A***:

$$y = \max_{1 \le i, j \le N} a_{ij}$$

Методика разработки параллельных алгоритмов…

Разделение вычислений на независимые части…

- Типовые вычислительные схемы:
  - выполнение однотипной обработки большого набора данных - ***параллелизм по данным***. В этом случае выделение подзадач сводится к разделению имеющихся данных.
    - Для большого количества решаемых задач разделение вычислений по данным приводит к порождению одно-, двух- и трех- мерных наборов подзадач, для которых информационные связи существуют только между ближайшими соседями (*сетки* или *решетки*):

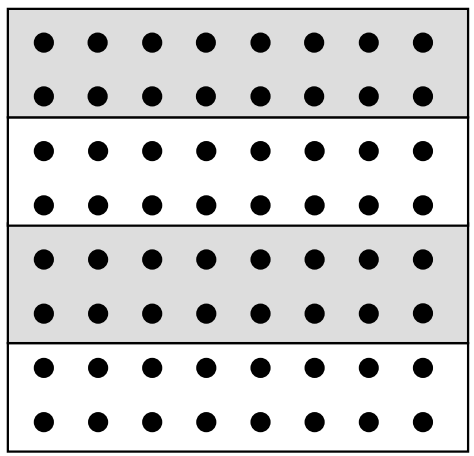


  - выполнение разных операций над одним и тем же набором данных - ***функциональный параллелизм.***

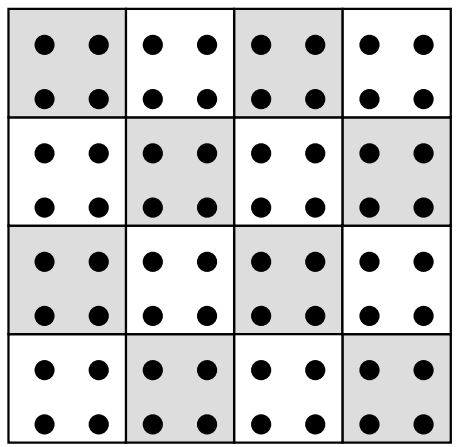
Методика разработки параллельных алгоритмов…

Разделение вычислений на независимые части…

- **Пример**: нахождение максимального элемента матрицы
  - исходная матрица ***A*** может быть разделена на отдельные строки (или последовательные группы строк) – *ленточная схема* разделения данных
  - прямоугольные наборы элементов – *блочная схема* разделения данных



а)



б)

Методика разработки параллельных алгоритмов…

Разделение вычислений на независимые части…

- Уровень декомпозиции вычислений
  - Формирование максимально возможного количества подзадач*:*
    - Обеспечивает использование предельно допустимого параллелизма,
    - Затрудняет анализ параллельных вычислений.
  - Использование достаточно крупных подзадач:
    - Приводит к ясной схеме параллельных вычислений,
    - Затрудняет эффективное использование большого числа процессоров.
  - Промежуточный уровень - использование в качестве элементов декомпозиции только тех подзадач, для которых методы параллельных вычислений известны. Выбираемые подзадачи при таком подходе будем именовать далее *базовыми*, которые могут быть *элементарными* (не допускают дальнейшего разделения) или *составными.*

Методика разработки параллельных алгоритмов…

Разделение вычислений на независимые части

- Для оценки корректности этапа разделения вычислений на независимые части можно воспользоваться контрольным списком вопросов:
  - Выполненная декомпозиция не увеличивает объем вычислений и необходимый объем памяти?
  - Возможна ли при выбранном способе декомпозиции равномерная загрузка всех имеющихся процессоров?
  - Достаточно ли выделенных частей процесса вычислений для эффективной загрузки имеющихся процессоров (с учетом возможности увеличения их количества)?

Методика разработки параллельных алгоритмов…

Выделение информационных зависимостей…

- Базовые понятия…
  - *Локальные и глобальные схемы* передачи данных
    - для локальных схем в каждый момент передачи данных выполняются только между небольшим числом подзадач (располагаемых, как правило, на соседних процессорах),
    - для глобальных операций передачи данных в процессе коммуникации принимают участие все подзадачи
  - *Структурные и произвольные способы взаимодействия* –
    - для структурных способов организация взаимодействий приводит к формированию некоторых стандартных схем коммуникации (например, в виде кольца, прямоугольной решетки и т.д.),
    - для произвольных структур взаимодействия схема выполняемых операций передач данных не носит характер однородности

Методика разработки параллельных алгоритмов…

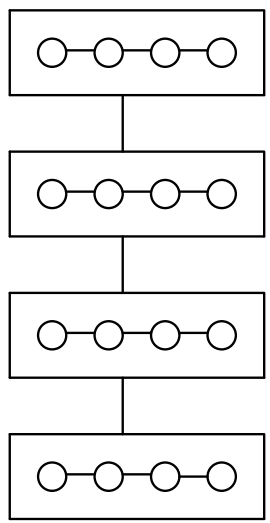
Выделение информационных зависимостей…

- Базовые понятия
  - *Статические или динамические схемы* передачи данных –
    - для статических схем моменты и участники информационного взаимодействия фиксируются на этапах проектирования и разработки параллельных программ,
    - для динамического варианта взаимодействия структура операции передачи данных определяется в ходе выполняемых вычислений
  - *Синхронные и асинхронные способы взаимодействия* –
    - для синхронных способов операции передачи данных выполняются только при готовности всех участников взаимодействия и завершаются только после полного окончания всех коммуникационных действий,
    - при асинхронном выполнении операций участники взаимодействия могут не дожидаться полного завершения действий по передаче данных

Методика разработки параллельных алгоритмов…

Выделение информационных зависимостей…

- **Пример**: нахождение максимального элемента матрицы
  - Достаточный уровень декомпозиции может состоять в разделении матрицы ***A*** на множество отдельных строк и получении на этой основе набора подзадач поиска максимальных значений в отдельных строках,
  - Структура информационных связей имеет вид:



Методика разработки параллельных алгоритмов…

Выделение информационных зависимостей

- Для оценки правильности этапа выделения информационных зависимостей можно воспользоваться контрольным списком вопросов:
  - Соответствует ли вычислительная сложность подзадач интенсивности их информационных взаимодействий?
  - Является ли одинаковой интенсивность информационных взаимодействий для разных подзадач?
  - Является ли схема информационного взаимодействия локальной?
  - Не препятствует ли выявленная информационная зависимость параллельному решению подзадач?

Методика разработки параллельных алгоритмов…

Масштабирование набора подзадач…

- Масштабирование проводится в случае, если количество имеющихся подзадач не совпадает с числом доступных процессоров
- Для сокращения количества подзадач необходимо выполнить укрупнение (*агрегацию*) вычислений:
  - подзадачи должны иметь одинаковую вычислительную сложность, а объем и интенсивность информационных взаимодействий между подзадачами должны быть минимально возможными,
  - первыми претендентами на объединение являются подзадачи с высокой степенью информационной взаимозависимости
- При недостаточном количестве подзадач для загрузки всех доступных к использованию процессоров необходимо выполнить *декомпозицию* вычислений

Методика разработки параллельных алгоритмов…

Масштабирование набора подзадач

- Список контрольных вопросов для оценки правильности этапа масштабирования:
  - Не ухудшится ли локальность вычислений после масштабирования имеющегося набора подзадач?
  - Имеют ли подзадачи после масштабирования одинаковую вычислительную и коммуникационную сложность?
  - Соответствует ли количество задач числу имеющихся процессоров?
  - Зависят ли параметрически правила масштабирования от количества процессоров?

Методика разработки параллельных алгоритмов…

Распределение подзадач между процессорами…

- Управление распределением нагрузки для процессоров возможно только для вычислительных систем с распределенной памятью. Для мультипроцессоров распределение вычислительной нагрузки между процессорами обычно выполняется операционной системой автоматически
- Этап распределения подзадач между процессорами является избыточным, если количество подзадач совпадает с числом имеющихся процессоров, а топология сети передачи данных вычислительной системы представляет собой полный граф

Методика разработки параллельных алгоритмов…

Распределение подзадач между процессорами…

- *Эффективность использования процессоров* - относительная доля времени, в течение которого процессоры использовались для вычислений, связанных с решением исходной задачи
- Пути достижения хороших показателей эффективности:
  - равномерное распределение вычислительной нагрузки между процессорами,
  - минимальное количество сообщений, передаваемых между процессорами.
- Оптимальное решение проблемы распределения подзадач между процессорами основывается на анализе информационной связности графа "подзадачи - сообщения"

Методика разработки параллельных алгоритмов…

Распределение подзадач между процессорами…

- *Динамическое распределение* вычислительной нагрузки необходимо в ситуациях, когда количество подзадач может изменяться в ходе вычислений.
- Одна из часто используемых схем - *схема "менеджер - исполнитель"* (*manager-worker scheme*)
  - для управления распределением нагрузки в системе выделяется отдельный процессор-менеджер, которому доступна информация обо всех имеющихся подзадачах,
  - Остальные процессоры системы являются исполнителями, которые для получения вычислительной нагрузки обращаются к процессору-менеджеру.
  - Порождаемые в ходе вычислений новые подзадачи передаются обратно процессору-менеджеру и могут быть получены для решения при последующих обращениях процессоров-исполнителей,
  - Завершение вычислений происходит в момент, когда процессоры-исполнители завершили решение всех переданных им подзадач, а процессор-менеджер не имеет каких-либо вычислительных работ для выполнения.

Методика разработки параллельных алгоритмов

Распределение подзадач между процессорами

- Перечень контрольных вопросов для проверки этапа распределения подзадач состоит в следующем:
  - Не приводит ли распределение нескольких задач на один и тот же процессор к росту дополнительных вычислительных затрат?
  - Существует ли необходимость динамической балансировки вычислений?
  - Не является ли процессор-менеджер "узким" местом при использовании схемы "менеджер-исполнитель"?

Пример применения методики: *Гравитационная задача N тел…*

- ***Гравитационная задача N тел*** (или просто *задача N тел*), как и многие задачи в области физики, сводится к операциям обработки данных для каждой пары объектов имеющейся физической системы:
  - Дано большое количество тел (планет, звезд и т.д.), для каждого из которых известна масса, начальное положение и скорость,
  - Под действием гравитации положение тел меняется,
  - Требуемое решение задачи состоит в моделировании динамики изменения системы *N* тел на протяжении некоторого интервала времени.

Пример применения методики: *Гравитационная задача N тел…*

- Для проведения моделирования интервал времени разбивается на временные отрезки небольшой длительности, на каждом шаге моделирования вычисляются силы, действующие на каждое тело, и обновляются скорости и положения тел
- Очевидный алгоритм решения задачи *N* тел состоит в рассмотрении на каждом шаге моделирования всех пар объектов физической системы и выполнении для каждой получаемой пары всех необходимых расчетов
- При таком подходе время выполнения одной итерации моделирования (*τ* есть время перевычисления параметров одной пары тел):

$$T_1 = \tau N(N-1)/2$$

Пример применения методики: *Гравитационная задача N тел…*

- **Разделение вычислений на независимые части**
  - *Базовая подзадача* - весь набор вычислений, связанных с обработкой данных одного какого-либо тела физической системы
- **Выделение информационных зависимостей**
  - Выполнение вычислений в подзадаче возможно только в случае, когда имеются данные обо всех телах физической системы,
  - Перед началом каждой итерации моделирования каждая подзадача должна получить все необходимые сведения от всех других подзадач системы,
  - Такая процедура передачи данных именуется *операцией обобщенного сбора данных* (*multi-node gather or all gather*).

Пример применения методики: *Гравитационная задача N тел…*

- **Выделение информационных зависимостей. Операция обобщенного сбора данных.**
  - **Метод 1**: Обмен данными осуществляется в ходе последовательности шагов, на каждом из которых все имеющиеся подзадачи разбиваются попарно и обмен данными осуществляется между подзадачами образовавшихся пар (*N-1* итерация).
  - **Метод 2**: Первый шаг метода выполняется точно также, как в методе 1 - после выполнения этого шага подзадачи будут содержать свои данные и данные подзадач, с которыми они образовывали пары. Как результат, на втором шаге пары подзадач могут быть образованы для обмена данными сразу о двух телах физической системы. После завершения второго шага каждая подзадача будет содержать сведения о четырех телах системы и т.д. Тем самым, общее количество шагов для выполнения всех требуемых обменов является равным $log_2 N$.

Пример применения методики: *Гравитационная задача N тел…*

- **Масштабирование и распределение подзадач по процессорам**
  - Если число тел физической системы *N* значительно превышает количество процессоров *p*, рассмотренные ранее подзадачи следует укрупнить, объединив в рамках одной подзадачи вычисления для группы (*N/p*) тел,
  - После проведения подобной агрегации число подзадач и количество процессоров будет совпадать,
  - При распределении подзадач между процессорами необходимо обеспечить наличие прямых коммуникационных линий между процессорами с подзадачами, между которыми имеются информационные обмены при выполнении операции сбора данных

Пример применения методики: *Гравитационная задача N тел…*

- **Анализ эффективности параллельных вычислений…**
  - Предложенные варианты отличаются только методами выполнения информационных обменов и для сравнения подходов достаточно определить длительность операции обобщенного сбора данных. Используем для оценки времени передачи сообщений модель, предложенную Хокни,
  - Длительность выполнения операции сбора данных для первого метода реализации может быть выражена как:

$$T_p^1(comm) = (p-1)(\alpha + m(N/p)/\beta)$$

  - При использовании второго метода информационного обмена для итерации с номером *i* объем сообщений оценивается как $2^{i-1}(Nm/p)$. Тем самым, длительность выполнения операции сбора данных в этом случае является равной

$$T_p^2(comm) = \sum_{i=1}^{\log p}(\alpha + 2^{i-1} m(N/p)/\beta) = \alpha \log p + m\,(N/p)(p-1)/\beta$$

Пример применения методики: *Гравитационная задача N тел*

- **Анализ эффективности параллельных вычислений**

  Сравнение полученных выражений показывает, что второй разработанный способ параллельных вычислений имеет существенно более высокую эффективность, несет меньшие коммуникационные затраты и допускает лучшую масштабируемость при увеличении количества используемых процессоров

Заключение…

- Определены основные требования к разрабатываемым алгоритмов параллельных вычислений:
  - Достижение равномерной загрузки процессоров,
  - Обеспечение низкого уровня информационного взаимодействия отдельных подзадач
- Представлены две модели для описания параллельных алгоритмов и программ:
  - Модель "подзадачи-сообщения" для использования на стадии проектирования параллельных алгоритмов,
  - Модель "процессы-каналы" для применения на стадии разработки параллельных программ

Заключение

- Рассмотрена методика разработки параллельных алгоритмов, которая включает этапы:
  - Разделение вычислений на независимые части,
  - Выделение информационных зависимостей,
  - Масштабирование имеющегося набора подзадач,
  - Распределение подзадач между процессорами
- Приведен пример использования методики разработки параллельных алгоритмов для параллельного решения гравитационной задачи *N* тел

Вопросы для обсуждения

- В чем состоят исходные предположения для возможности применения рассмотренной в разделе методики разработки параллельных алгоритмов?
- Каковы основные этапы проектирования и разработки методов параллельных вычислений?
- Какие основные требования должны быть обеспечены при разработке параллельных алгоритмов?
- Какой метод параллельных вычислений был разработан для решения гравитационной задачи *N* тел?
- Какой способ выполнения операции обобщенного сбора данных в задаче *N* тел является более эффективным?

Темы заданий для самостоятельной работы...

- Разработайте схему параллельных вычислений, используя рассмотренную в разделе методику:
  - для задачи поиска максимального значения среди минимальных элементов строк матрицы (такая задача имеет место для решения матричных игр)

$$y = \max_{1 \le i \le N} \min_{1 \le j \le N} a_{ij}$$

(обратите особое внимание на ситуацию, когда число процессоров превышает порядок матрицы, т.е. $p>N$),

  - для задачи вычисления определенного интеграла с использованием метода прямоугольников

$$y = \int_a^b f(x)dx \approx h\sum_{i=0}^{N-1} f_i,\ f_i = f(x_i),\ x_i = i\,h,\ h = (b-a)/N$$

Темы заданий для самостоятельной работы

- Выполните реализацию рассмотренных способов выполнения обобщенной операции сбора данных в задаче *N* тел и сравните время их выполнения. Сопоставьте получаемые временные характеристики с имеющими теоретическими оценками. Выполните сравнение со временем выполнения функции MPI_Allgather библиотеки MPI

Литература

- **Andrews**, G. R. (2000). Foundations of Multithreaded, Parallel, and Distributed Programming.. – Reading, MA: Addison-Wesley (русский перевод Эндрюс Г.Р. Основы многопоточного, параллельного и распределенного программирования. – М.: Издательский дом "Вильямс", 2003)
- **Bertsekas**, D.P., Tsitsiklis, J.N. (1989) Parallel and distributed Computation. Numerical Methods. - Prentice Hall, Englewood Cliffs, New Jersey.
- **Buyya,** R. (Ed.) (1999). High Performance Cluster Computing. Volume1: Architectures and Systems. Volume 2: Programming and Applications. - Prentice Hall PTR, Prentice-Hall Inc.

Литература

- **Kahaner, D.**, Moler, C., Nash, S. (1988). Numerical Methods and Software. – Prentice Hall (русский перевод Каханер Д., Моулер Л., Нэш С. Численные методы и программное обеспечение. М.: Мир, 2001)
- **Foster**, I. (1995). Designing and Building Parallel Programs: Concepts and Tools for Software Engineering. Reading, MA: Addison-Wesley.
- **Quinn**, M. J. (2004). Parallel Programming in C with MPI and OpenMP. – New York, NY: McGraw-Hill.
- **Wilkinson**, B., Allen, M. (1999). Parallel programming. – Prenrice Hall.

Следующая тема

- **Параллельные методы умножения матрицы на вектор**

Авторский коллектив

Гергель В.П., профессор, д.т.н., руководитель

Гришагин В.А., доцент, к.ф.м.н.

Абросимова О.Н., ассистент (раздел 10)

Лабутин Д.Ю., ассистент (система ПараЛаб)

Курылев А.Л., ассистент (лабораторные работы 4, 5)

Сысоев А.В., ассистент (раздел 1)

Гергель А.В., аспирант (раздел 12, лабораторная работа 6)

Лабутина А.А., аспирант (разделы 7,8,9, лабораторные работы 1, 2, 3, система ПараЛаб)

Сенин А.В., аспирант (раздел 11, лабораторные работы по Microsoft Compute Cluster)

Ливерко С.В. (система ПараЛаб)

О проекте

Целью проекта является создание образовательного комплекса "Многопроцессорные вычислительные системы и параллельное программирование", обеспечивающий рассмотрение вопросов параллельных вычислений, предусматриваемых рекомендациями Computing Curricula 2001 Международных организаций IEEE-CS и ACM. Данный образовательный комплекс может быть использован для обучения на начальном этапе подготовки специалистов в области информатики, вычислительной техники и информационных технологий.

Образовательный комплекс включает **учебный курс "Введение в методы параллельного программирования" и лабораторный практикум "Методы и технологии разработки параллельных программ"**, что позволяет органично сочетать фундаментальное образование в области программирования и практическое обучение методам разработки масштабного программного обеспечения для решения сложных вычислительно-трудоемких задач на высокопроизводительных вычислительных системах.

Проект выполнялся в Нижегородском государственном университете им. Н.И. Лобачевского на кафедре математического обеспечения ЭВМ факультета вычислительной математики и кибернетики (http://www.software.unn.ac.ru). Выполнение проекта осуществлялось при поддержке компании Microsoft.