СУПЕРКОМПЬЮТЕРНЫЙ КОНСОРЦИУМ УНИВЕРСИТЕТОВ РОССИИ

Проект

Создание системы подготовки высококвалифицированных кадров в области суперкомпьютерных технологий и специализированного программного обеспечения

Московский государственный университет им. М.В. Ломоносова

Нижегородский государственный университет им. Н.И. Лобачевского

- Национальный исследовательский университет -

Учебный курс

ВЫСОКОПРОИЗВОДИТЕЛЬНЫЕ ВЫЧИСЛЕНИЯ ДЛЯ МНОГОПРОЦЕССОРНЫХ МНОГОЯДЕРНЫХ СИСТЕМ

Лекция 3.

Принципы разработки параллельных методов

Гергель В.П., профессор, д.т.н.

Содержание

- Моделирование параллельных программ
- Этапы разработки параллельных алгоритмов
  - Разделение вычислений на независимые части
  - Выделение информационных зависимостей
  - Масштабирование набора подзадач
  - Распределение подзадач между вычислительными элементами
- Параллельное решение гравитационной задачи N тел
  - Разделение вычислений на независимые части
  - Выделение информационных зависимостей
  - Масштабирование и распределение подзадач по процессорам

Принципы разработки параллельных методов …

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

Принципы разработки параллельных методов …

- После выполнения всех перечисленных этапов проектирования можно оценить эффективность разрабатываемых параллельных методов:
  - Ускорение,
  - Эффективность,
  - Масштабируемость.
- По результатам проведенного анализа может оказаться необходимым повторение отдельных этапов разработки.
  - Возврат к предшествующим шагам разработки может происходить на любой стадии проектирования параллельных вычислительных схем.

Принципы разработки параллельных методов …

- Часто выполняют корректировку состава сформированного множества задач после определения имеющегося количества процессоров:
  - подзадачи могу быть укрупнены (*агрегированы*) при наличии малого числа процессоров или, наоборот, *детализированы* в противном случае.
- Данные действия могут быть определены как *масштабирование* разрабатываемого алгоритма.

Принципы разработки параллельных методов …

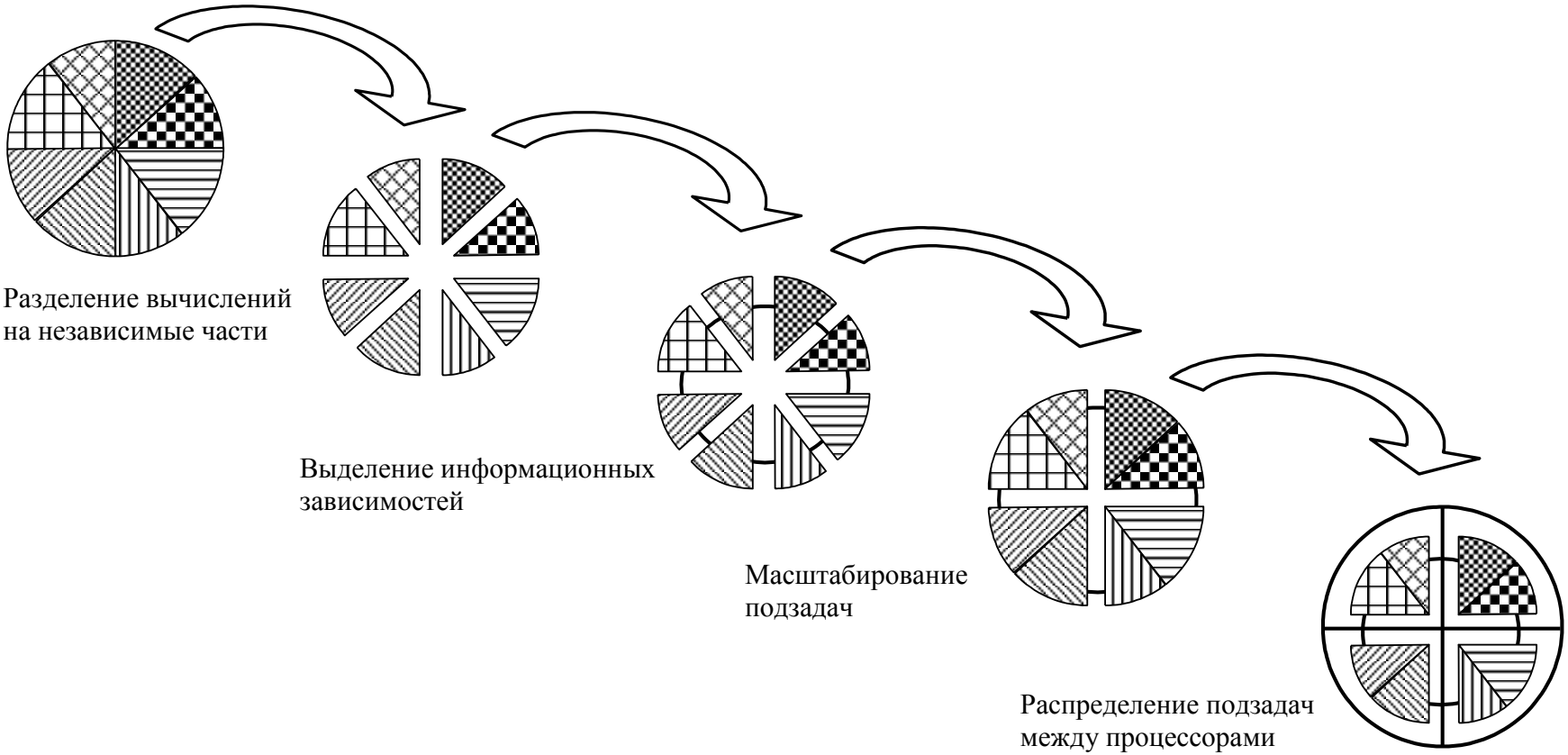
- Для применения получаемого параллельного метода необходимо выполнить его программную реализацию и разделить разработанный программный код по вычислительным элементам.
- Можно разработать для решения каждой подзадачи отдельную программу, однако чаще всего создается программа, которая объединяет в себе все действия, необходимые для решения всех имеющихся подзадач - *метапрограмма*:
  - Для проведения вычислений метапрограмма может копироваться для выполнения на все вычислительные элементы (MPI программы),
  - Метапрограмма может использоваться для порождения множества отдельных командных *потоков* (OpenMP программы).

Принципы разработки параллельных методов …

- Для реализации информационных взаимодействий параллельно выполняемые части программы (процессы или потоки) должны иметь в своем распоряжении средства обмена данными:
  - *Каналы передачи сообщений* для систем с распределенной памятью,
  - *Общие переменные* для систем с общей разделяемой памятью.

Принципы разработки параллельных методов …

- Общая схема разработки параллельных алгоритмов



Принципы разработки параллельных методов …

- Каждый вычислительный элемент (процессор или ядро процессора) компьютерной системы обычно выделяется для решения одной единственной подзадачи:
  - При наличии большого количества подзадач или использовании ограниченного числа вычислительных элементов это правило может не соблюдаться и, в результате, на вычислительных элементах может выполняться одновременно несколько параллельных частей программы (процессов или потоков).
- При расположении на одном вычислительном элементе параллельные части программы выполняются в режиме разделения времени.

Принципы разработки параллельных методов …

- Разработанная схема проектирования и реализации параллельных вычислений первоначально была предложена для вычислительных систем с распределенной памятью.
  - В данном подходе информационные взаимодействия реализуются при помощи передачи сообщений.
- Данная схема может быть использована без потери какой-либо эффективности параллельных вычислений и для разработки параллельных методов для систем с общей памятью.
  - Механизмы передачи сообщений для обеспечения информационных взаимодействий заменяются операциями доступа к общим (разделяемым) переменным.
- Для снижения сложности излагаемого далее учебного материала *схема проектирования и реализации параллельных вычислений будет конкретизирована применительно к вычислительным системам с общей памятью*.

Принципы разработки параллельных методов

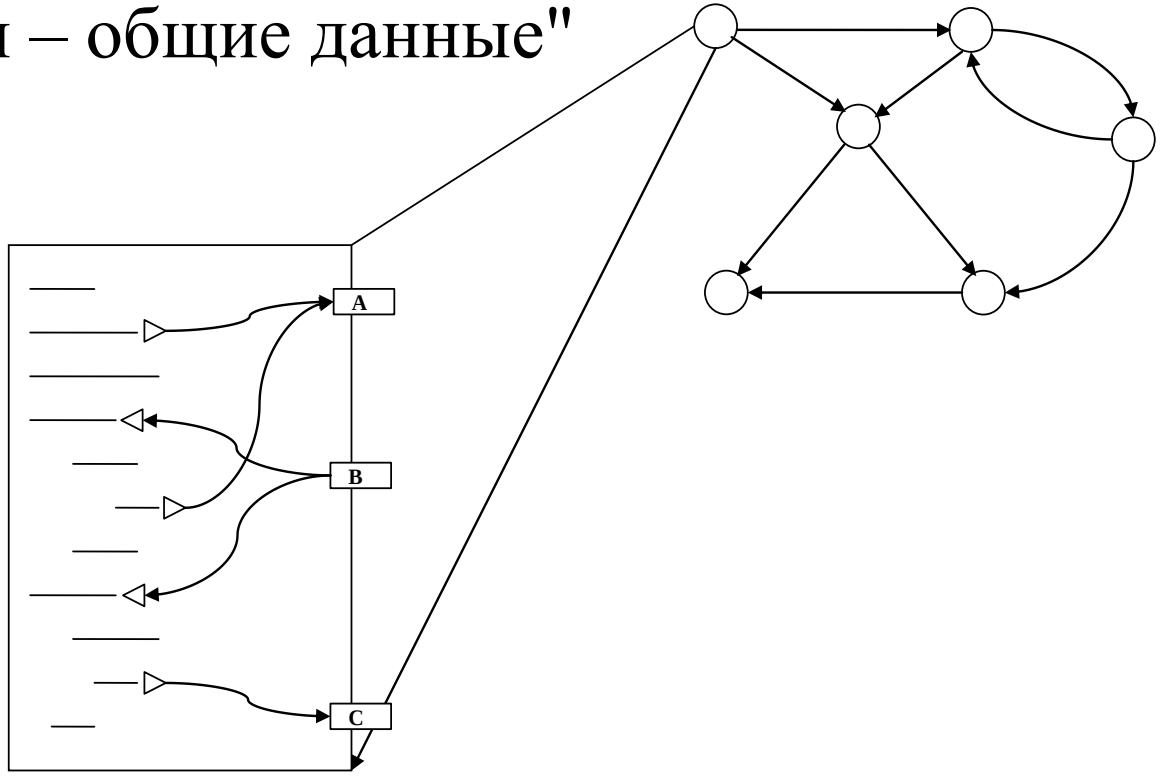
- Для вычислительных систем с общей памятью активно применяется и другой – "обратный" – способ разработки параллельных программ:
  - За основу берется тот или иной последовательный прототип, который постепенно преобразуется к параллельному варианту.
  - В частности, именно так предлагается использовать технологию OpenMP на начальном этапе освоения.
- Такой подход позволяет достаточно быстро получить начальные варианты параллельных программ без значительных дополнительных усилий.
  - Однако достаточно часто такая методика приводит к получению параллельных программ со сравнительно низкой эффективностью.
- **Достижение максимально возможных показателей ускорения вычислений можно обеспечить только при проектировании параллельных вычислений на начальных этапах разработки методов решения поставленных задач.**

Моделирование параллельных программ …

- Рассмотренная схема проектирования и реализации параллельных вычислений дает способ понимания параллельных алгоритмов и программ.
- На стадии проектирования параллельный метод может быть представлен в виде *графа "подзадачи – информационные зависимости"*, который представляет собой не что иное, как укрупненное представление графа информационных зависимостей.
- На стадии выполнения для описания параллельной программы может быть использована модель в виде *графа "потоки – общие данные"*.
  - В модели вместо подзадач используется понятие потоков, а информационные зависимости реализуются за счет использования общих данных.
  - На этой модели может быть показано распределение потоков по вычислительным элементам компьютерной системы, если количество подзадач превышает число вычислительных элементов.

Моделирование параллельных программ …

- Модель параллельной программы в виде графа "потоки – общие данные"



◁ [▷] - операции чтения (записи) общих данных

X - отдельные модули (объекты) общих данных

Моделирование параллельных программ …

- Первая модель:
  - Граф "подзадачи – информационные зависимости" – позволяет сосредоточиться на вопросах выделения подзадач одинаковой вычислительной сложности, обеспечивая при этом низкий уровень информационной зависимости между подзадачами.
- Вторая модель:
  - Граф "потоки – общие данные" – концентрирует внимание на вопросах распределения подзадач по вычислительным элементам и позволяет лучше анализировать эффективность разработанного параллельного метода и обеспечивает возможность более адекватного описания процесса выполнения параллельных вычислений.

Моделирование параллельных программ …

- В модели "потоки – общие данные":
  - Под *потоком* в рамках данного учебного материала будем понимать логически выделенную с точки зрения операционной системы *последовательность команд*, которая может исполняться на одном из вычислительном элементе компьютерной системы и которая содержит ряд *операций доступа (чтения/записи) к общим данным* для организации информационного взаимодействия между выполняемыми потоками параллельной программы;
  - Все потоки параллельной программы имеют общее адресное пространство;
  - В отличие от процессов операции создания и завершения потоков являются менее трудоемкими;
  - *Общие данные* с логической точки зрения могут рассматриваться как *общий (разделяемый) между потоками ресурс*; общие данные могут использоваться параллельно выполняемыми потоками для чтения и записи значений.

Моделирование параллельных программ

- Важно отметить, что для обеспечения корректности использование общих данных должно осуществляться в соответствии с правилами *взаимного исключения*:
  - В каждый момент времени общие данные могут использоваться только одним потоком.
  - При несоблюдении этих правил при использовании общих данных может иметь место ситуация *гонки потоков*, когда результат вычислений будет зависеть от взаимного соотношения темпа выполнения команд в потоках.
- Следует отметить важное достоинство рассмотренной модели "потоки – общие данные".
  - В модели проводится четкое разделение локальных вычислений и действий по организации информационного взаимодействия одновременно выполняемых потоков.
  - Такой подход значительно снижает сложность анализа эффективности параллельных методов и существенно упрощает проблемы разработки параллельных программ.

Этапы разработки параллельных алгоритмов

- Рассмотрим более подробно изложенную выше методику разработки параллельных алгоритмов.
- Методика включает этапы:
  - Разделение вычислений на независимые части,
  - Выделение информационных зависимостей,
  - Масштабирование набора подзадач,
  - Распределение подзадач между вычислительными элементами.
- Для демонстрации приводимых рекомендаций далее будет использоваться учебная задача поиска максимального значения среди элементов матрицы $\boldsymbol{A}$:

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

Разделение вычислений на независимые части …

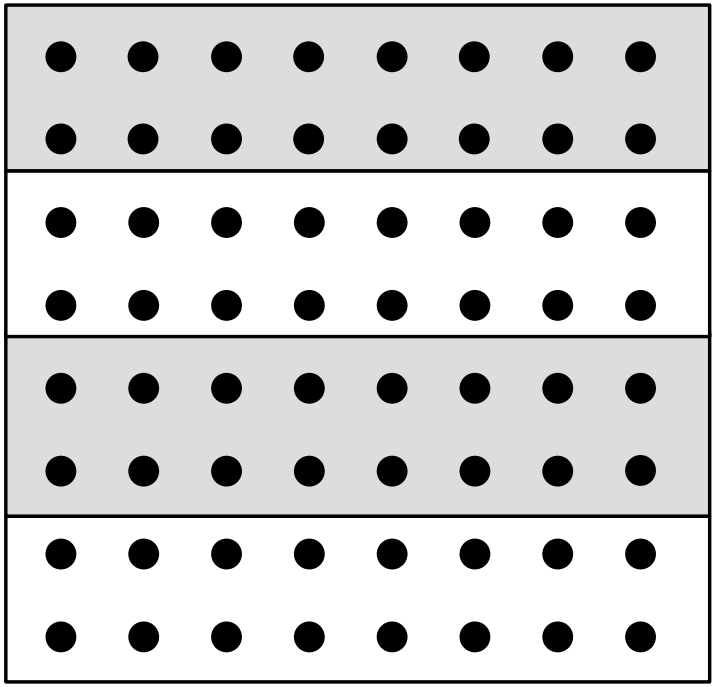
- Выбор способа разделения вычислений на независимые части основывается на анализе вычислительной схемы решения исходной задачи.
- Требования, которым должен удовлетворять выбираемый подход, обычно состоят в обеспечении *равного объема вычислений* в выделяемых подзадачах и *минимума информационных зависимостей* между этими подзадачами.

Разделение вычислений на независимые части …

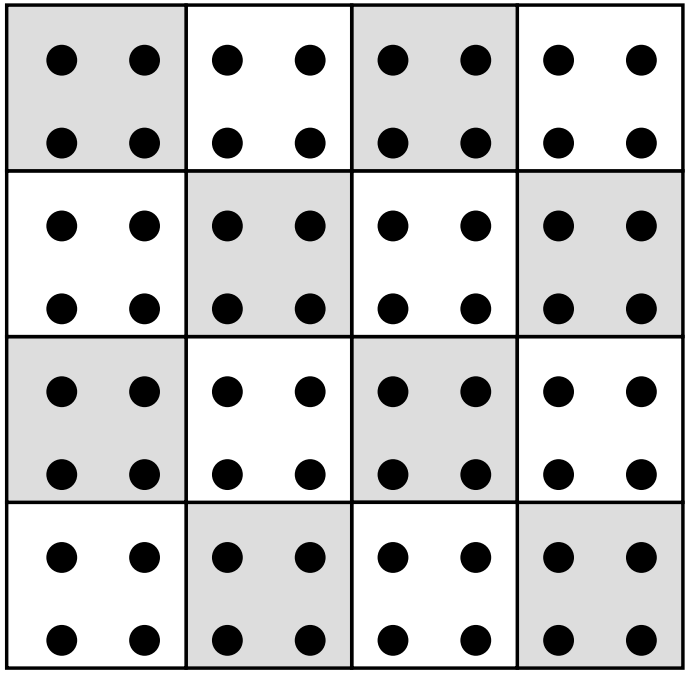
- Существует два часто встречающихся типов вычислительных схем:
  - Для большого класса задач вычисления сводятся к выполнению однотипной обработки большого набора данных:
    - В этом случае говорят, что существует *параллелизм по данным*, и выделение подзадач сводится к разделению имеющихся данных.
    - Для учебной задачи поиска максимального значения исходная матрица ***A*** может быть разделена на отдельные строки – *ленточная схема* разделения данных или на прямоугольные наборы элементов – *блочная схема* разделения данных.
    - Разделение вычислений по данным приводит к порождению одно-, двух- и трех- мерных наборов подзадач, для которых информационные связи существуют только между ближайшими соседями - такие схемы обычно именуются *сетками* или *решетками*.

Разделение вычислений на независимые части …

- Разделение данных для матрицы *A*: а) ленточная схема, б) блочная схема



а)



б)

Разделение вычислений на независимые части …

– Для другой части задач вычисления могут состоять в выполнении разных операций над одним и тем же набором данных:
  - В этом случае говорят о существовании *функционального параллелизма*.
  - Очень часто функциональная декомпозиция может быть использована для организации конвейерной обработки данных:
    – Так, например, при выполнении каких-либо преобразований данных вычисления могут быть сведены к функциональной последовательности ввода, обработки и сохранения данных.

Разделение вычислений на независимые части …

- Важный вопрос при выделении подзадач состоит в выборе нужного *уровня декомпозиции* вычислений:
  - Формирование максимально возможного количества подзадач обеспечивает использование предельно достижимого уровня параллелизма решаемой задачи, однако затрудняет анализ параллельных вычислений.
  - Использование при декомпозиции вычислений только достаточно "крупных" подзадач приводит к ясной схеме параллельных вычислений, однако может затруднить эффективное использование достаточно большого количества процессоров.
  - Возможное разумное сочетание этих двух подходов может состоять в использовании в качестве конструктивных элементов декомпозиции только тех подзадач, для которых методы параллельных вычислений являются известными.

Разделение вычислений на независимые части …

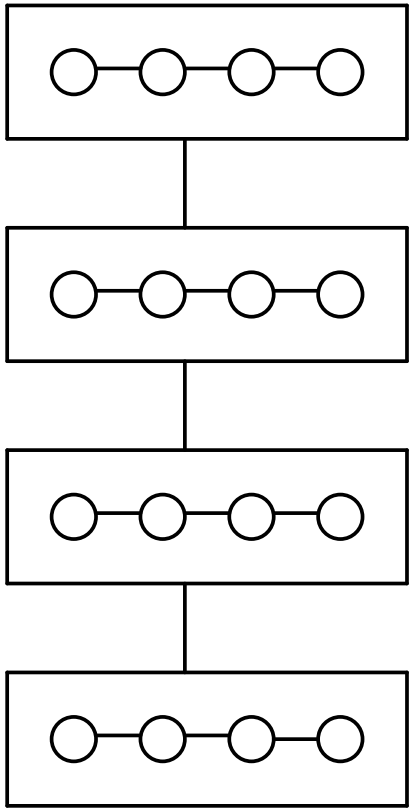
- При анализе задачи матричного умножения в качестве подзадач можно использовать методы скалярного произведения векторов или алгоритмы матрично-векторного произведения.
  - Подобный промежуточный способ декомпозиции вычислений *позволит обеспечить* и *простоту* представления вычислительных схем, и *эффективность* параллельных расчетов.
- Выбираемые подзадачи при таком подходе будем именовать далее *базовыми*, которые могут быть *элементарными*, если не допускают дальнейшего разделения, или *составными* в противном случае.

Разделение вычислений на независимые части …

- Для рассматриваемой учебной задачи достаточный уровень декомпозиции может состоять, например, в разделении матрицы ***A*** на множество отдельных строк и получении на этой основе набора подзадач поиска максимальных значений в отдельных строках;
- Порождаемая при этом структура информационных связей соответствует линейному графу.

Разделение вычислений на независимые части …

- Структура информационных связей учебной задачи



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 вычислительных элементов компьютерной системы?
  - Достаточно ли выделенных частей процесса вычислений для эффективной загрузки имеющихся вычислительных элементов (с учетом возможности увеличения их количества)?

Выделение информационных зависимостей …

- При наличии вычислительной схемы решения задачи после выделения базовых подзадач определение информационных зависимостей между подзадачами обычно не вызывает больших затруднений.
- Выделения подзадач и информационных зависимостей достаточно сложно поддаются разделению:
  - Выделение подзадач должно происходить с учетом возникающих информационных связей;
  - После анализа объема и частоты необходимых информационных обменов между подзадачами может потребоваться повторение этапа разделения вычислений.

Выделение информационных зависимостей …

- При проведении анализа информационных зависимостей между подзадачами следует различать:
  - *Локальные и глобальные схемы* информационного взаимодействия – для локальных схем в каждый момент времени информационная зависимость существует только между небольшим числом подзадач, для глобальных зависимостей информационное взаимодействие имеет место для всех подзадач,
  - *Структурные и произвольные способы взаимодействия* – для структурных способов организация взаимодействий приводит к формированию некоторых стандартных схем коммуникации, для произвольных структур взаимодействия схема информационных зависимостей не носит характер какой-либо однородности,
  - *Статические или динамические схемы* информационной зависимости – для статических схем моменты и участники информационного взаимодействия фиксируются на этапах проектирования и разработки параллельных программ, для динамического варианта взаимодействия структура зависимостей определяется в ходе выполняемых вычислений.

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?

Масштабирование набора подзадач …

- Масштабирование разработанной вычислительной схемы параллельных вычислений проводится в случае, если количество имеющихся подзадач отличается от числа планируемых к использованию вычислительных элементов.
  - Для сокращения количества подзадач необходимо выполнить укрупнение (*агрегацию*) вычислений.
- Применяемые здесь правила совпадают с рекомендациями начального этапа выделения подзадач
  - Определяемые подзадачи, как и ранее, должны иметь одинаковую вычислительную сложность, а объем и интенсивность информационных взаимодействий между подзадачами должны оставаться на минимально-возможном уровне.
  - Первыми претендентами на объединение являются подзадачи с высокой степенью информационной взаимозависимости.

Масштабирование набора подзадач …

- При недостаточном количестве имеющегося набора подзадач для загрузки всех доступных к использованию вычислительных элементов необходимо выполнить детализацию (*декомпозицию*) вычислений.
  - Как правило, проведение подобной декомпозиции не вызывает каких-либо затруднений, если для базовых задач методы параллельных вычислений являются известными.

Масштабирование набора подзадач …

- Выполнение этапа масштабирования вычислений должно свестись, в конечном итоге, к **разработке правил агрегации и декомпозиции подзадач, которые должны параметрически зависеть от числа вычислительных элементов,** применяемых для вычислений.
- Для рассматриваемой учебной задачи поиска максимального значения:
  - Агрегация вычислений может состоять в объединении отдельных строк в группы,
  - При декомпозиции подзадач строки исходной матрицы ***A*** могут разбиваться на несколько частей (блоков).

Масштабирование набора подзадач

- Список контрольных вопросов, для оценки правильности этапа масштабирования, выглядит следующим образом:
  - 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 вычислительных элементов?
  - Зависят ли параметрически правила масштабирования от количества вычислительных элементов?

Распределение подзадач между вычислительными элементами …

- Распределение подзадач между вычислительными элементами является завершающим этапом разработки параллельного метода.
- Управление распределением нагрузки для процессоров возможно только для вычислительных систем с распределенной памятью.
  - Для мультипроцессоров (систем с общей памятью) распределение нагрузки обычно выполняется операционной системой автоматически.
- Данный этап распределения подзадач между процессорами является избыточным, если количество подзадач совпадает с числом имеющихся вычислительных элементов.

Распределение подзадач между вычислительными элементами …

- Основной показатель успешности выполнения данного этапа – *эффективность использования вычислительных элементов*, определяемая как относительная доля времени, в течение которого вычислительные элементы использовались для вычислений, связанных с решением исходной задачи.
- Пути достижения хороших результатов в этом направлении остаются прежними необходимо:
  - Обеспечить равномерное распределение вычислительной нагрузки между вычислительными элементами,
  - Минимизировать количество информационных взаимодействий, существующих между ними.
- Оптимальное решение проблемы распределения подзадач между вычислительными элементами основывается на анализе информационной связности графа "подзадачи – информационные зависимости".

Распределение подзадач между вычислительными элементами …

- Требование минимизации информационных обменов между вычислительными элементами может противоречить условию равномерной загрузки процессоров вычислительной системы.
  - Можно разместить все подзадачи на одном процессоре и полностью устранить межпроцессорную передачу данных, однако, понятно, загрузка большинства процессоров в этом случае будет минимальной.
- Для учебной задачи поиска максимального значения распределение подзадач между вычислительными элементами не вызывает каких-либо затруднений – размещение подзадач может быть выполнено непосредственно операционной системой.

Распределение подзадач между вычислительными элементами …

- Решение вопросов балансировки вычислительной нагрузки значительно усложняется, если схема вычислений может изменяться в ходе решения задачи. Причиной этого могут быть, например:
  - Неоднородные сетки при решении уравнений в частных производных, разреженность матриц.
  - Используемые на этапах проектирования оценки вычислительной сложности решения подзадач могут иметь приближенный характер.
  - Количество подзадач может изменяться в ходе вычислений.
- В таких ситуациях может потребоваться перераспределение базовых подзадач между процессорами уже непосредственно в процессе выполнения параллельной программы.

Распределение подзадач между вычислительными элементами …

- Для нашей простой учебной задачи так же может наблюдаться различная вычислительная сложность сформированных базовых задач.
  - Количество операций при поиске максимального значения для строки, в которой максимальное значение имеет первый элемент, и строки, в которой значения являются упорядоченными по возрастанию, различается в два раза.

Распределение подзадач между вычислительными элементами …

- Рассмотрим широко используемый способ динамического управления вычислительной нагрузки – *"менеджер - исполнитель"* (*manager-worker scheme*):
  - При использовании данного подхода подзадачи могут возникать и завершаться в ходе вычислений, при этом информационные взаимодействия между подзадачами либо полностью отсутствует, либо минимальны.
  - Для управления распределением нагрузки в системе выделяется вычислительный элемент-менеджер, которому доступна информация обо всех имеющихся подзадачах.
  - Остальные вычислительные элементы системы являются исполнителями.
  - Порождаемые в ходе вычислений новые подзадачи передаются обратно менеджеру и могут быть получены для решения при последующих обращениях исполнителей.
  - Завершение вычислений происходит в момент, когда исполнители завершили решение всех переданных им подзадач, а менеджер не имеет каких-либо вычислительных работ для выполнения.

Распределение подзадач между вычислительными элементами

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

Параллельное решение гравитационной задачи N тел …

- Рассмотрим более сложный пример задачи для демонстрации приведенной выше схемы проектирования и реализации параллельных вычислений.
- Многие задачи в области физики сводятся к операциям обработки данных для каждой пары объектов имеющейся физической системы.
- Такой задачей является, в частности, проблема, широко известная в литературе как *гравитационная задача N тел* (или просто *задача N тел*).

Параллельное решение гравитационной задачи N тел …

- В самом общем виде, задача может быть описана следующим образом:
  - Пусть 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, и требуемое решение задачи состоит в моделировании динамики изменения системы N тел на протяжении некоторого задаваемого интервала времени.
- Для проведения такого моделирования заданный интервал времени обычно разбивается на временные отрезки небольшой длительности и далее на каждом шаге моделирования вычисляются силы, действующие на каждое тело, а затем 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.

Параллельное решение гравитационной задачи N тел …

Разделение вычислений на независимые части:

- Выбор способа разделения вычислений не вызывает каких-либо затруднений – очевидный подход состоит в выборе в качестве базовой подзадачи всего набора вычислений, связанных с обработкой данных одного какого-либо тела физической системы.

Параллельное решение гравитационной задачи N тел …

Выделение информационных зависимостей:

- Выполнение вычислений, связанных с каждой подзадачей, становится возможным только в случае, когда в подзадачах имеются данные (положение и скорости передвижения) обо всех телах имеющейся физической системы.
  - Как результат, перед началом каждой итерации моделирования каждая подзадача должна получить все необходимые сведения от всех других подзадач системы.
- Такая процедура информационного взаимодействия обычно именуется *операцией сбора данных* (*single-node gather*).
- В рассматриваемом алгоритме данная операция должна быть выполнена для каждой подзадачи – такой вариант информационного взаимодействия часто именуется как *операция обобщенного сбора данных* (*multi-node gather* or *all gather*).

Параллельное решение гравитационной задачи N тел

Масштабирование и распределение подзадач по процессорам:

- Как правило, число тел физической системы N значительно превышает количество вычислительных элементов p.
  - Как результат, рассмотренные ранее подзадачи следует укрупнить, объединив в рамках одной подзадачи вычисления для группы (N/p) тел – после проведения подобной агрегации число подзадач и количество вычислительных элементов будет совпадать.
- Учитывая характер имеющихся информационных зависимостей распределение подзадач между вычислительными элементами может быть непосредственно операционной системой.

Заключение …

- В лекции была рассмотрена методика разработки параллельных алгоритмов.
- Данная методика включает этапы:
  - выделения подзадач,
  - определения информационных зависимостей,
  - масштабирования и распределения подзадач по вычислительным элементам компьютерной системы.
- Основные требования, которые должны быть обеспечены при разработке параллельных алгоритмов – обеспечение равномерной загрузки вычислительных элементов при низком информационном взаимодействии сформированного множества подзадач.

Заключение

- Рассмотрены две модели:
  - "подзадачи – информационные зависимости",
  - "потоки – общие данные".
- В завершение демонстрируется применение рассмотренной методики разработки параллельных алгоритмов на примере решения гравитационной задачи N тел.

Темы заданий для самостоятельной работы

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

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

  - для задачи вычисления определенного интеграла с использованием метода прямоугольников

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

- Разработайте схему параллельных вычислений для задачи умножения матрицы на вектор, используя рассмотренную в разделе методику проектирования и разработки параллельных методов.

Обзор литературы …

- Рассмотренная в разделе методика разработки параллельных алгоритмов впервые была предложена в
  - Foster I. Designing and Building Parallel Programs: Concepts and Tools for Software Engineering. Reading, MA: Addison-Wesley, 1995.
- В этой работе изложение методики проводится более детально; кроме того, в работе содержится несколько примеров использования методики для разработки параллельных методов для решения ряда вычислительных задач.

Обзор литературы

- Полезной при рассмотрении вопросов проектирования и разработки параллельных алгоритмов может оказаться также работа
  - Quinn M.J. Parallel Programming in C with MPI and OpenMP. – New York, NY: McGraw-Hill, 2004
- Гравитационная задача *N* тел более подробно рассматривается в
  - Andrews, G. R. Foundations of Multithreaded, Parallel, and Distributed Programming.. – Reading, MA: Addison-Wesley, 2000 (русский перевод Эндрюс Г.Р. Основы многопоточного, параллельного и распределенного программирования. – М.: Издательский дом "Вильямс", 2003)

Следующая тема

- Основы параллельного программирования

О проекте

Целью проекта является создание национальной системы подготовки высококвалифицированных кадров в области суперкомпьютерных технологий и специализированного программного обеспечения.

Задачами по проекту являются:

Задача 1. Создание сети научно-образовательных центров суперкомпьютерных технологий (НОЦ СКТ).

Задача 2. Разработка учебно-методического обеспечения системы подготовки, переподготовки и повышения квалификации кадров в области суперкомпьютерных технологий.

Задача 3. Реализация образовательных программ подготовки, переподготовки и повышения квалификации кадров в области суперкомпьютерных технологий.

Задача 4. Развитие интеграции фундаментальных и прикладных исследований и образования в области суперкомпьютерных технологий. Обеспечение взаимодействия с РАН, промышленностью, бизнесом.

Задача 5. Расширение международного сотрудничества в создании системы суперкомпьютерного образования.

Задача 6. Разработка и реализации системы информационного обеспечения общества о достижениях в области суперкомпьютерных технологий.

См. http://www.hpc-education.ru