Московский государственный университет им. М.В. Ломоносова

Введение в методы параллельных вычислений

Разработчик:
А.В. Старченко, д.ф.-м.н., профессор
E-mail: starch@math.tsu.ru
Томский государственный университет

Направление 010300.68
«Фундаментальная информатика и информационные технологии»

Проект комиссии Президента по модернизации и техническому развитию экономики России «Создание системы подготовки высококвалифицированных кадров в области суперкомпьютерных технологий и специализированного программного обеспечения»

Разработка курса выполнена в рамках Проекта комиссии Президента РФ по модернизации и техническому развитию экономики России «Создание системы подготовки высококвалифицированных кадров в области суперкомпьютерных технологий и специализированного программного обеспечения»

Применение потенциала суперкомпьютерных технологий (СКТ) как значимой составляющей инновационного развития страны является задачей государственной важности, относится к приоритетному направлению и находится под постоянным контролем Президента и Правительства России. Одним из сдерживающих факторов развития страны в этом направлении является острая нехватка высококвалифицированных кадров в области СКТ, поскольку подготовка таких специалистов сейчас отсутствует как элемент системы высшего профессионального образования.

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

http://hpc-education.ru.

Содержание курса

- Введение
- Рекуррентные формулы
- Параллельные вычисления определенных и кратных интегралов
- Умножение матрицы на вектор. Умножение матриц
- Прямые методы решения систем линейных уравнений на многопроцессорных системах Организация межпроцессорных обменов
- Трехдиагональные системы. Параллельная реализация прямых методов решения систем линейных уравнений
- Параллельная реализация итерационных методов решения СЛАУ
- Параллельная реализация быстрого преобразования Фурье

Содержание лекции

- Введение
- Дискретное преобразование Фурье
- Лемма Даниэльсона-Ланцоша
- Быстрое преобразование Фурье
- Алгоритм БПФ Кули-Тьюки
- Параллельная реализация БПФ
- Оценка ускорения и эффективности
- Заключение

Введение

- Математические преобразования играют важную роль в численном анализе.
- Среди них наиболее известным является дискретное преобразование Фурье, так как существует быстрый алгоритм его вычисления – так называемый алгоритм быстрого преобразования Фурье (БПФ или Fast Fourier Transform, FFT).
- В последние годы в связи с интенсивным развитием цифровой вычислительной техники внимание исследователей стали привлекать полные системы прямоугольных ортогональных функций Уолша, Хаара и др., для которых также существуют быстрые процедуры построения.

Введение

- Ортогональные дискретные преобразования находят свое применение:
  - в методах вычислений – при решении краевых задач для дифференциальных уравнений с частными производными, при интерполировании функций;
  - при анализе временных рядов для определения спектра частот, вычисления автокорреляций;
  - в области обработки изображений и речевых сигналов;
  - при кодировании и декодировании информации и др.

Дискретное преобразование Фурье

- Дискретное преобразование Фурье (ДПФ) вектора x длиной $n \in \mathbb{N}$ определяется следующим образом:

$$y_m = \sum_{k=0}^{n-1} x_k \exp\left(i\frac{2\pi km}{n}\right); \quad m = 0,1,\ldots,n-1; \quad i^2 = -1;$$

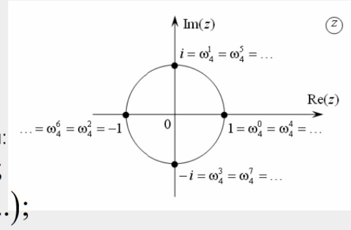
- Обратное дискретное преобразование Фурье определяется:

$$x_k = \frac{1}{n}\sum_{m=0}^{n-1} y_m \exp\left(-i\frac{2\pi mk}{n}\right); \quad k = 0,1,\ldots,n-1.$$

Дискретное преобразование Фурье

- Обозначим

$$\omega_n = \exp\left(i\frac{2\pi}{n}\right) = \sqrt[n]{1},$$



- Можно отметить следующие свойства введенного понятия:

$$\omega_n^{m\cdot n} = 1, \ (m = 0,1,2,\ldots);$$
$$\omega_n^{m\cdot n/2} = -1, (m = 1,3,5,\ldots);$$
$$\omega_n^{k+m\cdot n} = \omega_n^k, \ (k,m = 0,1,2,\ldots).$$

Дискретное преобразование Фурье

- С учетом этих обозначений прямое вычисление ДПФ

$$y_m = \sum_{k=0}^{n-1} x_k \exp\left(i\frac{2\pi km}{n}\right) = \sum_{k=0}^{n-1} x_k \omega_n^{km}; \ m = 0,1,\ldots,n-1.$$

- потребует ***n*** комплексных умножений и ***n*** комплексных сложений на одну компоненту вектора ***y*** или, в целом, ***8n²*** вещественных арифметических операций (флопов) для вычисления вектора ***y***.

Дискретное преобразование Фурье

- На протяжении последних двухсот лет математики неоднократно предлагали эффективные методы для вычисления ДПФ. Возможность рекурсивного представления ДПФ была известна Карлу Фридриху Гауссу, который в 1805 году использовал этот подход для интерполирования траекторий астероидов Паллас и Джуно.
- В 1942 году Даниэльсон и Ланцош представили вывод алгоритма эффективного вычисления ДПФ. Они показали, что ДПФ длиной ***n*** (***n***-четное) может быть записано как сумма двух ДПФ длиной ***n/2***: одно формируется из компонентов вектора ***x*** с четными индексами, другое из компонентов ***x*** с нечетными индексами .

Лемма Даниэльсона-Ланцоша

- Доказательство этого утверждения выглядит следующим образом:

$$y_m = \sum_{k=0}^{n-1} \omega_n^{km} x_k = \sum_{k=0}^{n/2-1} \omega_n^{2km} x_{2k} + \sum_{k=0}^{n/2-1} \omega_n^{(2k+1)m} x_{2k+1} =$$
$$= \sum_{k=0}^{n/2-1} \omega_{\underline{n/2}}^{km} x_{2k} + \omega_n^m \sum_{k=0}^{n/2-1} \omega_{\underline{n/2}}^{km} x_{2k+1}; m = 0,1,\ldots,n-1.$$

- Замечательным в лемме Даниэльсона-Ланцоша является то, что она может быть применена рекурсивно, если $n = 2^q$, до тех пор, пока не останутся двухточечные преобразования Фурье.

Быстрое преобразование Фурье

- При ***n=2^q*** уменьшение количества комплексных сложений и умножений получается, если ДПФ с помощью следствия леммы, представить в виде кратных сумм с двумя слагаемыми:

$$y_m = \sum_{s_0=0}^{1} \omega_n^{ms_0} \left(\sum_{k=0}^{n_1-1} \omega_{n_1}^{km} x_{2k+s_0} \right) = \sum_{s_0=0}^{1} \omega_n^{ms_0} \left(\sum_{k=0}^{n_1-1} \omega_{n_1}^{km} z_k^{(s_0)} \right) =$$
$$= \sum_{s_0=0}^{1} \omega_n^{ms_0} \left(\sum_{s_1=0}^{1} \omega_{n_1}^{ms_1} \left(\ldots \left(\sum_{s_{q-1}=0}^{1} \omega_{n_{q-1}}^{ms_{q-1}} \left(\sum_{k=0}^{n_q} \omega_{n_q}^{km} z_k^{(s_0,s_1,\ldots,s_{q-1})} \right) \right) \right) \right)$$
$$n_l = n/2^l; z_k^{(s_0,\ldots,s_{l-1})} = z_{2k+s_{l-1}}^{(s_0,\ldots,s_{l-2})}; k = 0,\ldots,n_l - 1; l = 1,\ldots,q;$$

Быстрое преобразование Фурье

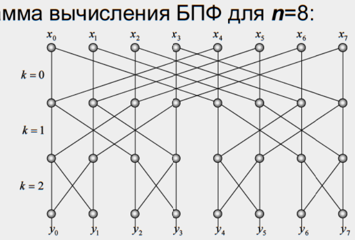
- Для нахождения одной компоненты вектора ***y*** требуется $q=\log_2 n$ комплексных умножений и $q=\log_2 n$ комплексных сложений или $8n\cdot\log_2 n$ флопов.
- Различие между $n\cdot\log_2 n$ и n^2 огромно. Так, например, при $n=10^6$ это есть разница между **0,03 секундами** и **1200 секундами** времени центрального процессора на компьютере с тактовой частотой 1 ГГц.
- Существование алгоритма БПФ стало широко известным лишь в середине 60-х годов прошлого века из опубликованной статьи Дж.Кули и Дж.Тьюки.

Алгоритм БПФ Кули-Тьюки

```
do i = 0, n-1        ! инициализация вектора R
   R(i) = x(i)
end do
   q = log2(n)
do k = 0, q-1
  do i=0, n-1
    S(i) = R(i)
  end do
  do i = 0, n-1
     (b0, b1, ..., bq-1) = binary(i) ! определение двоичного представления i
     i1 = (b0, b1, ..., bk-1, 0, bk+1, ..., bq-1)  ! вычисление индексов
     i2 = (b0, b1, ..., bk-1, 1, bk+1, ..., bq-1)
     i3 = (bk, bk-1, ..., b0, 0, 0, ..., 0)
    R(i) = S(i1) + ω**i3 * S(i2)
  end do
end do
do i = 0, n-1        ! передача результата вектору y
  y(i)=R(i)
end do
```


Алгоритм БПФ Кули-Тьюки

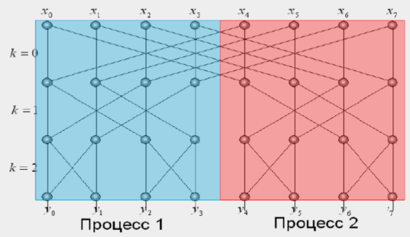
- Диаграмма вычисления БПФ для ***n***=8:



$$y_m=\left\{\left(x_0+(-1)^m x_4\right)+\omega_8^{2m}\left(x_2+(-1)^m x_6\right)\right\}+$$
$$+\omega_8^m\left\{\left(x_1+(-1)^m x_5\right)+\omega_8^{2m}\left(x_3+(-1)^m x_7\right)\right\};m=0,\dots,7.$$

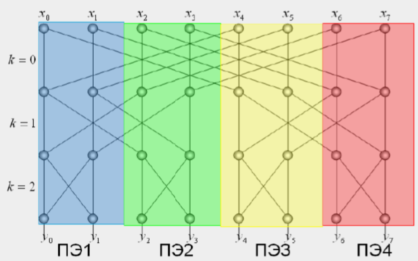
Параллельная реализация БПФ

- Чтобы получить мелкозернистый параллельный алгоритм быстрого преобразования Фурье, распределим исходный вектор равномерно между $p=2^r$ процессами (***n=8, p=2***).



Параллельная реализация БПФ

- Чтобы получить мелкозернистый параллельный алгоритм быстрого преобразования Фурье, распределим исходный вектор равномерно между $p=2^r$ процессами (***n=8, p=4***).



Параллельная реализация БПФ

- Чтобы получить меньшее количество укрупненных подзадач, объединим вместе вычисление ***n/p*** компонентов вектора ***y*** и ***n/p*** компонентов входного вектора ***x***.
- При таком способе декомпозиции пересылки данных требуются лишь на первых $\log_2 p$ стадиях, причем каждым процессом передается одному другому по ***n/p*** чисел.
- Оставшиеся $\log_2(n/p)$ стадий параллельного алгоритма выполняются локально без коммуникаций.
- Для такого параллельного алгоритма наиболее подходящей является топология межпроцессорных соединений - гиперкуб.

Параллельная реализация БПФ

- Проведем оценку временных затрат рассмотренного параллельного алгоритма БПФ.

$$T_p^{comp} \approx t_a \frac{8n}{p} \log_2 n; \quad T_p^{comm} \approx t_{comm} \frac{n}{p} \log_2 p;$$

- Теоретические оценки ускорения и эффективности:

$$S_p = \frac{T_1}{T_p} \approx \frac{p}{1 + \frac{\kappa \log_2 p}{8 \log_2 n}}; \quad \kappa = \frac{t_{comm}}{t_a}.$$

- При увеличении числа процессов ***p*** эффективность алгоритма при фиксированном ***n*** падает за счет увеличения количества обменов.

Заключение

- Рассмотрены формулы дискретного преобразования Фурье, проведены оценки количества арифметических операций для преобразования вектора заданной длины.
- Выведены формулы леммы Даниэльсона-Ланцоша, позволяющие сократить число арифметических операций, особенно в случае когда длина преобразуемого вектора $n=2^q$.
- Представлен псевдокод алгоритма быстрого преобразования Фурье Кули-Тьюки и его параллельная реализация, базирующаяся на декомпозиции данных и вычислений по процессорным элементам.
- Получены оценки ускорения и эффективности параллельного алгоритма БПФ.

Вопросы для обсуждения

- Где применяется дискретное преобразование Фурье?
- Запишите формулу прямого ДПФ вектора и оцените его вычислительную трудоемкость.
- Сформулируйте лемму Даниэльсона-Ланцоша и объясните ее роль в ускорении вычисления ДПФ.
- Оцените вычислительную трудоемкость БПФ для случая $n=2^q$.
- Нарисуйте диаграмму алгоритма БПФ Кули-Тьюки (***n***=8), дайте краткую характеристику этому алгоритму.
- Как следует проводить декомпозицию данных и вычислений для построения параллельного алгоритма БПФ?
- Какое количество процессов можно использовать в параллельном алгоритме БПФ Кули-Тьюки?
- Нарисуйте диаграмму параллельного алгоритма БПФ для случая (***n***=8, ***p***=4) и опишите его особенности.
- При каких условиях параллельный алгоритм БПФ будет иметь ускорение, близкое к идеальному?

Темы заданий для самостоятельной работы

- Подготовьте последовательные программы ДПФ и БПФ (алгоритм Кули-Тьюки) и сравните их время счета для больших размеров преобразуемого вектора.
- Нарисуйте диаграмму параллельного алгоритма БПФ для случая (***n***=16, ***p***=4) и опишите его особенности.
- Подготовьте MPI-программу параллельного алгоритма БПФ Кули-Тьюки и сравните ее ускорение с выполненными теоретическими оценками.

Основная литература

1. *Хокни Р., Джессхоуп К.* Параллельные ЭВМ. Архитектура, программирование и алгоритмы. М: Радио и связь, 1986.
2. *Каханер Д., Моулер К., Нэш С.* Численные методы и программное обеспечениею – М.: Мир, 2001.
3. *Миллер Р., Боксер Л.* Последовательные и параллельные алгоритмы. – М.: БИНОМ. Лаборатория знаний, 2006.
4. *Марчук Г.И.* Методы вычислительной математики. – М.: Наука, 1977.
5. *Quinn, M.* Parallel Computing - Theory and Practice, McGraw-Hill, New-York, 1994.

Курс окончен!

- Введение
- Рекуррентные формулы
- Параллельные вычисления определенных и кратных интегралов
- Умножение матрицы на вектор. Умножение матриц
- Прямые методы решения систем линейных уравнений на многопроцессорных системах Организация межпроцессорных обменов
- Трехдиагональные системы. Параллельная реализация прямых методов решения систем линейных уравнений
- Параллельная реализация итерационных методов решения СЛАУ
- Параллельная реализация быстрого преобразования Фурье