Лабораторная работа

Слащинин Сергей, группа 17 МАГ-ИАД

17 января 2018 г.

В данной работе рассматриваются параллельные алгоритмы, реализующие различные методы для численного нахождения определенных интегралов

Цель

Целью данной работы является анализ и реализация параллельных алгоритмов для численного интегрирования. Используя различные методы, необходимо вычислить:

$$\int_0^1 \frac{4}{x^2+1}\,\mathrm{d}x$$

Следует рассмотреть следующие методы численного интегрирования:

- метод прямоугольников
- метод трапеций
- метод Симпсона
- метод Гаусса 3-го порядка точности

Для каждого из них необходимо написать параллельную реализацию, используя программный интерфейс MPI

Пункт 1

Построить график (или таблицу) зависимости среднего времени работы алгоритма (например, 100 запусков) от числа процессов при некотором фиксированном значении шага интегрирования (для персонального компьютера и кластера);

Как можно заметить из графиков, наименьшее среднее время работы программы достигается при количестве процессов p=8 и p=16 для персонального компьютера и кластера, соответственно. Это совпадает с количеством виртуальных ядер каждого процессора. проанализировать, как будет меняться время работы алгоритма при количестве процессов, превышающем количество ядер/нод (для персонального компьютера и кластера);

Для каждого из методов ожидалось, при постепенном увеличении числа процессов, когда их общее число превышает количество ядер процессора, среднее время работы программы будет расти. Это объясняется тем, что системе необходимо тратить дополнительное время на переключение ядер между одновременно выполняемыми процессами и синхронизацию (а также запуск) процессов. Как видно из результатов в п.1, эксперименты подтвердили эти ожидания. Наименьшее вермя работы на персональном компьютере достигается при количетсве потоков p=8, а на кластере при p=16, а затем начинает постепенно расти.

Пункт 3

Посчитать ускорение и эффективность параллельного алгоритма для произвольного числа процессов;

Пусть p - число параллельных процессов, n - число шагов интегрирования, $T(n,p)$ - общее время работы программы, $T_0(p)$ - время, необходимое на инициализацию, запуск и синхронизацию процессов, $C(n)$ - время, необходимое на вычислений на n отрезках интегрирования. Тогда:

$$T(n,p) = T_0(p) + C(\frac{n}{p})$$

$C(1)$ для каждого метода отличается, т.к. для вычисления приближения на отрезке для каждого метода вычисляется функция в разном количестве точек и выполняется разное количество вспомогательных

операций. Т.к. операции вычисления приближения интеграла на каждом из отрезков выполняются последовательно, а также включают в себя одинаковое число операций, можно считать, что $C(n) = p \cdot C(\frac{n}{p})$ и также $T_0(1) = 0$ при $n \gg p$. Получаем следующую формулу для ускорения:

$$S(p) = \frac{C(n)}{T_0(p) + C(\frac{n}{p})} \approx p$$

Эффективность:

$$E(p) = \frac{C(n)}{T_0(p) + p \cdot C(\frac{n}{p})} \approx 1$$

Для $p = 8$ и $n = 10000000$ на практике были получены следующие результаты:

- Метод прямоугольников: $S(8) = 4.2987$
- метод трапеций: $S(8) = 4.2365$
- метод Симпсона: $S(8) = 6.7279$
- метод Гаусса 3-го порядка точности: $S(8) = 5.9721$
- $E(8) = 99.8$ для всех методов численного интегрирования, программа в среднем загружает каждое ядро на 99,8% на протяжении всего времени работы

Пункт 4

Сравнить теоретические значения ускорения и эффективности с полученными на практике значениями; построить график (или таблицу) зависимости среднего времени работы алгоритма, применяемого для вычисления интеграла Ожидается, что при увеличении верхнего предела интегрирования, среднее время работы программы будет расти линейно (как и число шагов)

$$\int_0^N \frac{4}{x^2 + 1} \, \mathrm{d}x$$

Как показали результаты экспериментов, зависимость времени работы от N линейная.