В этом разделе описаны задачи, необходимые для достижения цели. Приведено математическое и детальное описание трех алгоритмов умножения матриц:

1) классический;

2) винограда;

3) винограда Модифицированный.

0.1 Постановка задачи

Цель: изучение способа распараллеливания алгоритма Винограда. В данной работе рассматривается распараллеленная реализация стандартного алгоритма умножения матриц, алгоритма Винограда.

Задачи:

1) рассмотреть задачу умножения матриц;

2) произвести анализ сложности алгоритмов умножения матриц;

3) запрограммировать алгоритмы и сделать их работу параллельной;

4) сделать замеры времени для алгоритмов;

5) результаты экспериментов сравнить с теоретическими оценками трудоёмкости;

6) сделать выводы.

0.2 Классический подход

Предположим, что необходимо получить матрицу $C_{(a,c)} = A_{(a,b)} * B_{(b,c)}$. Для нахождения значений элементов матрицы $C_{(a,c)}$ используют следующие выражение[**makkonell**]

$$C_{i,j} = \sum_{K}(A_{i,k}B_{i,k}) \tag{1}$$

Классический алгоритм напрямую реализует эту формулу

0.3 Алгоритм Винограда

Можно заметить, что элементы из суммы выражения 1 можно переписать как:

$$A_{i,k-1}B_{k-1,j}+A_{i,k}B_{k,j} = (A_{i,k-1}+B_{k,j})(A_{i,k}+B_{k-1,j})-A_{i,k-1}A_{i,k}-B_{k-1}B_{k,j} \tag{2}$$

т. е. как сумму произведения сумм и двух произведений. Учитывая, что упомянутые два произведения можно рассчитать заранее для обработки двух элементов матрицы теперь нужно не сложение и два умножения, а умножение и два сложения, что проще с точки зрения вычислений. Таким образом, алгоритм Винограда состоит в следующем:

1. совершить расчет заранее двух произведений для каждого ряда и столбца матрицы-результата (одно произведение считается для ряда, другое для столбца). Для хранения результатов используется промежуточный буфер;
2. по вышеприведённой формуле осуществить расчёт каждого элемента матрицы;
3. в случае, если в произведении $A_{(a,b)XB_{b,c}}$ b – нечётное число, пройтись во второй раз по матрице, дополняя элементы $C_{i,j}$ недостающим элементом (который не был описан вышеописанной суммой). Можно заметить, что пункт 3 необходимо выполнять только в некоторых случаях, но если это происходит, то получается существенное увеличение времени работы алгоритма.

0.4 Оптимизированный алгоритм Винограда

1. Внутри тройного цикла накапливать результат в буфер, а вне цикла сбрасывать его в ячейку матрицы.
2. Заменить MulH[i] = MulH[i] + ... на MulH[i] += ... (аналогично для MulV), где MulH и MulB – временные массивы для предварительного рассчета сумм произведений

0.5 Вывод

В данном разделе детально описаны 3 алгоритма умножения матриц: Классический, Винограда, модифицированная версия Винограда.