## Лабораторная работа №1

## «Исследование автоматического распараллеливания в GCC»

### Порядок выполнения работы

1. На компьютере с многоядерным процессором установить ОС Linux и компилятор GCC версии **не ниже 4.7.2**. При невозможности установить Linux или отсутствии компьютера с многоядерным процессором можно выполнять лабораторную работу на виртуальной машине VMWare в а.372, предварительно попросив разрешение у заведующего лабораторией Филиппова Владимира Геннадьевича.
2. На **языке Cи** написать консольную программу lab1.c, решающую задачу, указанную в п.IV (см. ниже). В программе **нельзя** использовать библиотечные функции сортировки, выполнения матричных операций и расчёта статистических величин. Задача должна решаться 100 раз с разными начальными значениями генератора случайных чисел (ГСЧ). Структура программы *примерно* следующая:

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  #include <sys/time.h>  int main(int argc, char\* argv[])  {  int i, N;  struct timeval T1, T2;  long delta\_ms;  N = atoi(argv[1]); /\* инициализировать число N, используя 1-й параметр командной строки \*/  gettimeofday(&T1, NULL); /\* запомнить текущее время T1 \*/  for (i=0; i<100; i++) /\* провести 100 экспериментов с разными начальными значениями ГСЧ \*/  {  srand(i); /\* инициализировать начальное значение(ГСЧ) \*/  /\* Заполнить массив исходных данных размером NxN случайными числами \*/  /\* Решить поставленную задачу и заполнить массив с результатами \*/  /\* Отсортировать массив с результатами указанным методом \*/  }  gettimeofday(&T2, NULL); /\* запомнить текущее время T2 \*/  delta\_ms = 1000\*(T2.tv\_sec - T1.tv\_sec) + (T2.tv\_usec - T1.tv\_usec)/1000;  printf("\nN=%d. Milliseconds passed: %ld\n", N, delta\_ms); /\* вывести затраченное время \*/  return 0;  } |
|  |

1. Скомпилировать написанную программу без использования автоматического распараллеливания с помощью следующей команды:   
   ***/home/user/gcc -O2 -Wall -Werror -o lab1-seq lab1.c***
2. Скомпилировать написанную программу, используя средства автоматического распараллеливания с помощью следующей команды (переменную K следует изменять от 2 до 202 с шагом 10):   
   ***/home/user/gcc -O2 -Wall -Werror -floop-parallelize-all -ftree-parallelize-loops=K lab1.c -o lab1-par-K***
3. В результате получится 1 нераспараллеленная программа и 20 распараллеленных.
4. Закрыть **все** работающие в операционной системе прикладные программы (включая Winamp, uTorrent, браузеры и Skype), чтобы они не влияли на результаты последующих экспериментов.
5. Запускать файл lab1-seq из командной строки, увеличивая значения N до значения N1, при котором время выполнения превысит 0.5 с. Продолжать запускать lab1-seq, увеличивая значения N до значения N2, при котором время выполнения превысит 100 с.
6. Используя найденные значения N1 и N2, выполнить следующие эксперименты (для автоматизации проведения экспериментов рекомендуется написать скрипт):
   1. запускать **lab1-seq** для значений N = {N1, N1+∆, N1+2∆, N1+3∆,…, N2} и записывать получающиеся значения времени *delta\_ms(N)* в функцию *seq(N)*;
   2. запускать **lab1-par-K** для значений N = {N1, N1+∆, N1+2∆, N1+3∆,…, N2} и записывать получающиеся значения времени *delta\_ms(N)* в функцию *par-K(N).*

Значение ∆ выбрать так: ∆ = (N2- N1)/10.

1. Написать отчёт о проделанной работе.
2. Подготовиться к устным вопросам на защите.
3. \* Изменить программу так, чтобы эффект от распараллеливания увеличился (это дополнительное необязательное задание повышенной сложности).

### Состав отчета

1. Титульный лист с названием вуза, ФИО студента и названием работы.
2. Содержание отчета (с указанием номера страниц и т.п.).
3. Описание решаемой задачи (взять из п.I и п.IV).
4. Краткая характеристика использованного для проведения экспериментов процессора, операционной системы и компилятора GCC (официальное название, номер версии/модели, разрядность, количество ядер, ёмкость ОЗУ и т.п.).
5. Полный текст программы lab1.c в виде отдельного файла.
6. Таблицы значений и графики функций seq(N), par-K(N) с указанием величины параллельного ускорения.
7. Подробные выводы с анализом приведённых графиков и полученных результатов.

Отчёт предоставляется на бумажном носителе или на флешке, или высылается на почту [vsosnin@gmail.com](mailto:vsosnin@gmail.com) и [pvbalakshin@gmail.com](mailto:pvbalakshin@gmail.com).

### Подготовка к защите

1. Уметь объяснить каждую строку программы, представленной в отчёте.
2. Уметь объяснить выводы, полученные в результате работы.
3. Знать о назначении и основных особенностях GCC, а также о назначении всех использованных в работе ключей компиляции GCC.
4. Знать, что способствует развитию и что замедляет развитие параллельных вычислений (см. материал лекции 1).

### Варианты заданий

Вариант задания выбирается в соответствии с Таблицей. Порядок вычислений должен быть следующим:

1. Сформировать матрицу М1 размерностью N x N, заполнив её случайными вещественными числами, имеющими закон Гамма-распределения с математическим ожиданием А и дисперсией В.
2. Из получившейся матрицы М1 нужно сформировать матрицу М2 (размерностью N x N) в соответствии с параметром С вашего варианта.
3. Сформировать матрицу М3 (размерностью N x N), выполнив матричное умножение М3 = М1 х М2.
4. Сформировать вектор V размерностью N х 1, используя задание, указанное параметром D.
5. Полученный вектор V необходимо отсортировать методом, указанным в параметре Е (для этого нельзя использовать библиотечные функции).

Таблица с параметрами задания

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ФИО студента** | **Математическое ожидание** | **Дисперсия** | **Преобразование исходной матрицы** | **Способ формирования вектора с результатами** | **Метод сортировки вектора с результатами** |
| **A** | **B** | **C** | **D** | **Е** |
| Абдеев Диас Умирсерикович | -65 | 34 | 4 | 6 | 3 |
| Бевза Роман Александрович | -48 | 92 | 5 | 9 | 1 |
| Бедаш Дмитрий Сергеевич | -45 | 65 | 1 | 2 | 2 |
| Гаврина Мария Андреевна | -10 | 90 | 6 | 8 | 4 |
| Гапонов Андрей Игоревич | -25 | 34 | 1 | 9 | 8 |
| Дмитриев Антон Николаевич | -11 | 85 | 4 | 2 | 7 |
| Жуков Максим Андреевич | -3 | 26 | 6 | 5 | 2 |
| Кобцев Сергей Владимирович | -66 | 26 | 5 | 5 | 5 |
| Коюшев Сергей Алексеевич | -48 | 90 | 6 | 7 | 5 |
| Подсухина Юлия Андреевна | -10 | 23 | 2 | 8 | 4 |
| Прус Виталий Андреевич | -89 | 18 | 3 | 7 | 11 |
| Смирновский Юрий Игоревич | -37 | 90 | 3 | 6 | 3 |
| Ткешелашвили Нино Мерабиевна | -11 | 80 | 5 | 4 | 1 |
| Фиц Владислав Александрович | -3 | 73 | 1 | 2 | 4 |
| Алейников Андрей Александрович | -76 | 73 | 4 | 5 | 8 |
| Бойцов Сергей Владимирович | -5 | 34 | 3 | 7 | 8 |
| Криммель Герман Константинович | -58 | 88 | 6 | 7 | 2 |
| Кудреватых Евгений Васильевич | -3 | 34 | 3 | 3 | 10 |
| Репин Алексей Владимирович | -11 | 18 | 1 | 8 | 9 |
| Шатунова Ксения Юрьевна | -24 | 34 | 6 | 5 | 8 |
| Грудина Анна Михайловна | -65 | 94 | 5 | 7 | 3 |

Метод получения матрицы М2 (параметр «С»)

1. Умножение М1 на скаляр 5.
2. Вычитание из матрица М1 матрицы Е размером N x N.
3. Транспонирование матрицы М1.
4. Инвертирование знака матрица М1.
5. Каждый элемент М2 равен синусу симметричного элемента в М1.
6. Каждый элемент М2 равен десятичному логарифму модуля симметричного элемента в М1.
7. Каждый элемент М2 равен элементу M1, умноженному на число Х, где:

* X равен числу Пи (3,14159265358...), если сумма номера строки и номера столбца нечётная.
* X равен числу e (2,718281828...), если сумма номера строки и номера столбца чётная.

Способ формирования вектора с результатами (параметр «D»)

1. Математическое ожидание каждой строки.
2. Выборочная дисперсия каждой строки.
3. Коэффициент вариации каждой строки.
4. Выборочное среднеквадратичное отклонение каждой строки.
5. Минимальный элемент каждой строки.
6. Максимальный элемент каждой строки.
7. Количество отрицательных элементов в строке.
8. Количество положительных элементов в строке.
9. Математическое ожидание отрицательных элементов.

Методы сортировки (параметр «Е»)

1. [Сортировка выбором](http://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0_%D0%B2%D1%8B%D0%B1%D0%BE%D1%80%D0%BE%D0%BC) (Selection sort).
2. [Сортировка Шелла](http://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0_%D0%A8%D0%B5%D0%BB%D0%BB%D0%B0) (Shell sort) .
3. [Сортировка расчёской](http://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0_%D1%80%D0%B0%D1%81%D1%87%D1%91%D1%81%D0%BA%D0%BE%D0%B9) (Comb sort).
4. [Пирамидальная сортировка](http://ru.wikipedia.org/wiki/%D0%9F%D0%B8%D1%80%D0%B0%D0%BC%D0%B8%D0%B4%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%81%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0) (сортировка кучи, Heapsort).
5. [Быстрая сортировка](http://ru.wikipedia.org/wiki/%D0%91%D1%8B%D1%81%D1%82%D1%80%D0%B0%D1%8F_%D1%81%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0) (Quicksort).
6. [Introsort](http://ru.wikipedia.org/wiki/Introsort).
7. [Поразрядная сортировка](http://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%80%D0%B0%D0%B7%D1%80%D1%8F%D0%B4%D0%BD%D0%B0%D1%8F_%D1%81%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0) (цифровая сортировка).
8. [Глупая сортировка](http://ru.wikipedia.org/wiki/%D0%93%D0%BB%D1%83%D0%BF%D0%B0%D1%8F_%D1%81%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0) (Stupid sort).
9. [Сортировка подсчётом](http://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0_%D0%BF%D0%BE%D0%B4%D1%81%D1%87%D1%91%D1%82%D0%BE%D0%BC) (Counting sort).
10. [Алгоритм сортировки Timsort](http://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D1%81%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B8_Timsort) (*Timsort*).
11. [Гномья сортировка](http://ru.wikipedia.org/wiki/%D0%93%D0%BD%D0%BE%D0%BC%D1%8C%D1%8F_%D1%81%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0).