МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

“Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики”

(НИУ ИТМО)

_____

Факультет программной инженерии и компьютерной техники

Направление подготовки 09.04.04 Программная инженерия

Лабораторная работа № 1

“Автоматическое распараллеливание программ”

По дисциплине “Параллельные вычисления”

Студент группы P4114

Трофимова Полина Владимировна

Преподаватель:

Жданов Андрей Дмитриевич

Санкт-Петербург, 2024 г.

Оглавление

Описание решаемой задачи

На языке Си написать консольную программу lab1.c, решающую задачу, представленную ниже, в соответствии с вариантами:

A = 9* 6* 12 = 648; X = 1 + ((A mod 47) mod B);

Этап Map X = 1 + ((648 mod 47) mod 7) = 1 + (37 mod 7) = 3 (Гиперболический тангенс с последующим уменьшением на 1)

Этап Map X = 1 + ((648 mod 47) mod 8) = 1 + (37 mod 8) = 6 (Десятичный логарифм, возведенный в степень e)

Этап Merge X = 1 + ((648 mod 47) mod 6) = 1 + (37 mod 6) = 2 (Деление (т.е. M2[i] = M1[i]/M2[i]))

Этап Sort X = 1 + ((648 mod 47) mod 6) = 1 + (37 mod 6) = 2 (Сортировка расчёской (Comb sort))

В программе нельзя использовать библиотечные функции сортировки, выполнения матричных операций и расчёта статистических величин. В программе нельзя использовать библиотечные функции, отсутствующие в стандартных заголовочных файлах stdio.h, stdlib.h, sys/time.h, math.h. Задача должна решаться 100 раз с разными начальными значениями генератора случайных чисел (ГСЧ).

1. Этап Generate. Сформировать массив M1 размерностью N, заполнив его с помощью функции rand_r (нельзя использовать rand) случайными вещественными числами, имеющими равномерный закон распределения в диапазоне от 1 до A (включительно). Аналогично сформировать массив M2 размерностью N/2 со случайными вещественными числами в диапазоне от A до 10*A.
2. Этап Map. В массиве M1 к каждому элементу применить операцию: Гиперболический тангенс с последующим уменьшением на 1. Затем в массиве M2 каждый элемент поочерёдно сложить с предыдущим (для этого понадобится копия массива M2, из которого нужно будет брать операнды), а к результату сложения применить операцию (считать, что для начального элемента массива предыдущий элемент равен нулю): Десятичный логарифм, возведенный в степень e.
3. Этап Merge. В массивах M1 и M2 ко всем элементам с одинаковыми индексами попарно применить операцию (результат записать в M2): Деление (т.е. M2[i] = M1[i]/M2[i]).
4. Этап Sort. Полученный массив необходимо отсортировать методом (для этого нельзя использовать библиотечные функции; можно взять реализацию в виде свободно доступного исходного кода): Сортировка расчёской (Comb sort).

5. Этап Reduce. Рассчитать сумму синусов тех элементов массива M2, которые при делении на минимальный ненулевой элемент массива M2 дают чётное число (при определении чётности учитывать только целую часть числа). Результатом работы программы по окончании пятого этапа должно стать одно число X, которое следует использовать для верификации программы после внесения в неё изменений (например, до и после распараллеливания итоговое число X не должно измениться в пределах погрешности). Данное число необходимо выводить на каждой итерации на этапе верификации. Значение числа X следует привести в отчёте для различных значений N.

Краткая характеристика системы

Операционная система: Windows 10 Домашняя

Тип системы: 64-разрядная операционная система

Процессор: AMD Ryzen 7 5700U

Оперативная память: 8ГБ

Количество физических ядер: 8

Количество логических ядер: 16

gcc version 11.4.0 (Ubuntu 11.4.0-1ubuntu1~22.04)

Программа lab1.c

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#include <math.h>
#include <sys/time.h>

void combSort(double arr[], int n) {
    int gap = n;
    bool swapped = true;
    int temp;
    while (gap != 1 || swapped) {
        gap *= 10 / 13;
        if (gap < 1) {
            gap = 1;
        }
        swapped = false;
        for (int i = 0; i < n - gap; i++) {
            if (arr[i] > arr[i + gap]) {
                temp = arr[i];
                arr[i] = arr[i + gap];
                arr[i + gap] = temp;
                swapped = true;
            }
        }
    }
}

int main(int argc, char* argv[])
{
    int i, j, N;
    struct timeval T1, T2;
    double e = exp(1.0);
    double min = 1;
    double max = 648;
    long delta_ms;


    // N равен первому параметру командной строки
    N = atoi(argv[1]);

    double* restrict M1 = (double*)malloc(N * sizeof(double));
    double* restrict M2 = (double*)malloc(N / 2 * sizeof(double));
    double* restrict M2_copy = (double*)malloc(N / 2 * sizeof(double));

    unsigned int seed = 1;

    // запомнить текущее время T1
    gettimeofday(&T1, NULL);

    for (i = 0; i < 100; i++) // 100 экспериментов
    {
        seed = i; // инициализировать начальное значение ГСЧ

        for (j = 0; j < N; j++) // Заполнить массив исходных данных размером N
        {
            M1[j] = (((double)rand_r(&seed) / (RAND_MAX)) * (max - min) + min);
        }

        for (j = 0; j < N / 2; j++)
        {
            M2[j] = ((double)rand_r(&seed) / (RAND_MAX)) * (max * 10 - max) + max;
            M2_copy[j] = M2[j];
        }
```

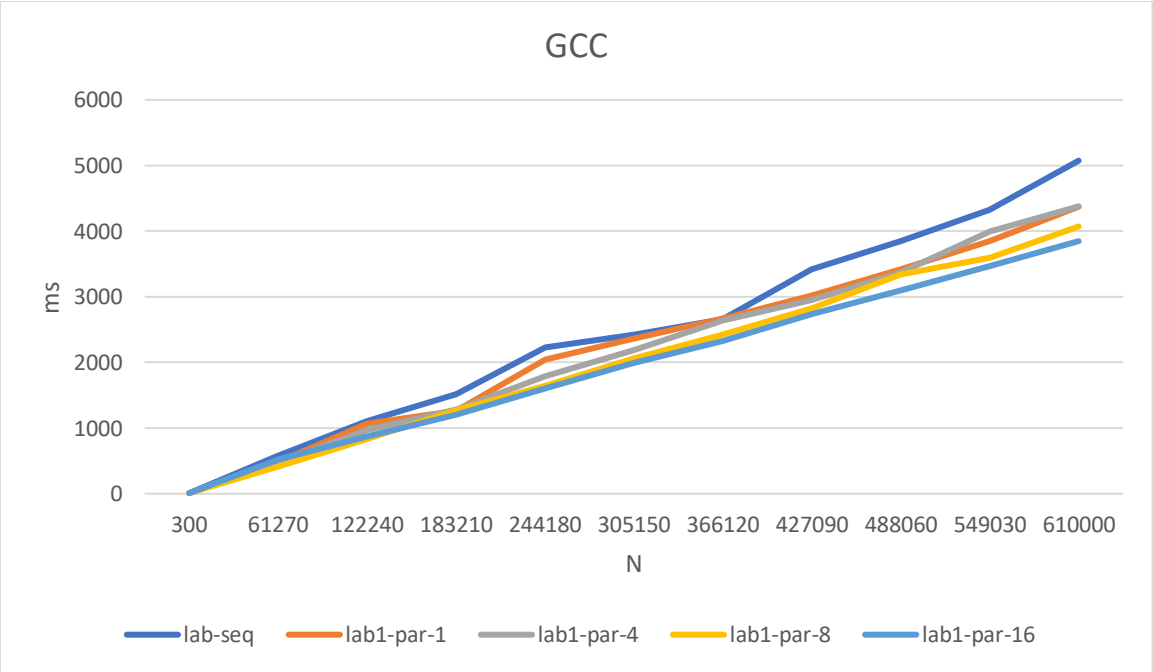
```
        //map
        for (j = 0; j < N; j++)
        {
            // Гиперболический тангенс с последующим уменьшением на 1
            M1[j] = ((tanh(M1[j])) - 1);
        }

        for (j = 1; j < N / 2; j++)
        {
            // Десятичный логарифм, возведенный в степень e
            M2[j] = pow((M2[j] + M2_copy[j - 1]), e);
        }
        M2[0] = pow(M2[0], e);

        for (j = 0; j < N / 2; j++) // Решить поставленную задачу, заполнить массив
с результатами
        {
            //Деление (т.е. M2[i] = M1[i]/M2[i])
            M2[j] = M1[j] / M2[j];
        }

        //combSort(M2, N / 2);
        // Сортировка расчёской (Comb sort) результатов

        //минимальные не нулевой на два фора
        double min_nonzero = M2[0];
        double sum_sin = 0.0;

        for (j = 0; j < N / 2; j++)
        {
            if ((M2[j] > 0) && (M2[j] < min_nonzero))
            {
                min_nonzero = M2[j];
            }
        }
        for (j = 0; j < N / 2; j++)
        {
            if ((int)(M2[j] / min_nonzero) % 2 == 0)
            {
                sum_sin += sin(M2[j]);
            }
        }
        printf("%.10lf\n", sum_sin);
    }

    gettimeofday(&T2, NULL); // запомнить текущее время T2
    delta_ms = (T2.tv_sec - T1.tv_sec) * 1000 + (T2.tv_usec - T1.tv_usec) / 1000;
    printf("\nN=%d. Milliseconds passed: %ld\n", N, delta_ms);

    free(M1);
    free(M2);
    free(M2_copy);

    return 0;
}
```

Результаты

Lab-seq
N1 = 300 8ms
N2 = 610000 5094ms

Delta 609700/10=60970

		N	lab-seq	lab1-par-1	lab1-par-4	lab1-par-8	lab1-par-16
1	N1	300	8	9	9	10	9
2	N1+delta	61270	581	482	472	418	519
3	N1+2delta	122240	1111	1063	970	834	873
4	N1+3delta	183210	1520	1263	1284	1277	1208
5	N1+4delta	244180	2226	2039	1793	1643	1603
6	N1+5delta	305150	2422	2365	2187	2067	1990
7	N1+6delta	366120	2661	2671	2640	2419	2328
8	N1+7delta	427090	3414	3014	2953	2820	2740
9	N1+8delta	488060	3847	3415	3365	3337	3097
10	N1+9delta	549030	4323	3846	3994	3597	3470
11	N2	610000	5074	4372	4377	4070	3848




```
test@LAPTOP-89IUHH64:/mnt/c/Users/ptrof/source/repos/Project3/Project3$ ./l
ab1-par-1 305150
-0.0000000039
-0.0000000039
-0.0000000041
-0.0000000033
```

```
-0.0000000043
-0.0000000029
-0.0000000034
-0.0000000037
-0.0000000041
-0.0000000032

N=305150. Milliseconds passed: 2365
```

Выводы

Явного прироста от параллелизма нет. Компилятор GCC не показывает заметного прироста, хотя видны потенциальные участки для распараллеливания.