Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Уфимский университет науки и технологий»
Кафедра ВМиК

Отчёт по лабораторной работе №1

на тему: «ПАРАЛЛЕЛЬНОЕ ВЫЧИСЛЕНИЕ СУММЫ РЯДА СРЕДСТВАМИ АВТОМАТИЧЕСКОГО РАСПАРАЛЛЕЛИВАНИЯ И OPENMP»

по дисциплине: «Параллельные вычисления»

Выполнили:

Студент группы: ПРО-428Б Мохаммед А. А.

Аль-Шаибани Е. Т.

Вахитов Т. Ф.

Проверила: Шерыхалина Н.М.

Уфа – 2023

1. Цель работы

Научиться использовать основные директивы, функции и переменные окружения интерфейса OpenMP на примерах написания простейших параллельных программ для многоядерных вычислительных систем.

2. Выполнение работы

1. Выполнить программную реализацию на языке C / C++ последовательного алгоритма суммирования функционального ряда согласно варианту. Предусмотреть вывод на экран времени работы программы.
2. Произвести распараллеливание программы путём добавления директив OpenMP в текст последовательной программы.
3. Отладить написанные программы при небольшом числе членов ряда. Убедиться, что результаты последовательной и параллельной программ корректны и стабильны при многократных запусках. Убедиться, что количество порождаемых потоков равно числу ядер вычислительной системы.
4. Добавить опцию задания числа порождаемых потоков и запустить программу при различном числе потоков.
5. Подключить автоматическое распараллеливание к последовательной версии программы (ключ /QParallel). Убедиться, что количество порождаемых потоков равно числу ядер вычислительной системы.
6. Для последовательной программы подобрать три значения N так, чтобы время ее работы на многоядерной системе составляло 10 $(N = N_1)$, 20 $(N = N_2)$ и 40 $(N = N_3)$ секунд. Запустить многопоточные программы (OpenMP и автораспараллеливание) при аналогичных значениях N. Количество потоков должно равняться числу ядер вычислительной системы. Время работы каждой программы занести в табл. 2.2.

Таблица 2.2

Программа	T_p, с		
	N_1	N_2	N_3
Послед.			
OpenMP			
Авторасп.			

7. Для параллельных программ (автоматическое распараллеливание и OpenMP) вычислить ускорение и эффективность по формулам (1.3) и (1.4) лабораторной работы №1 для каждого N, полученные значения занести в табл. 2.3. Объяснить полученные результаты.

Таблица 2.3

Программа	S_p^*			E_p^*		
	N_1	N_2	N_3	N_1	N_2	N_3
OpenMP						
Авторасп.						

3. Ход работы

Задание 1:

Дан ряд согласно варианту 2

2.	$\sum_{n=1}^{N} \frac{\sqrt[3]{n}}{(n+1)\sqrt{n}}$

Task 3: Sequential vs. Parallel Comparison

The sequential and parallel algorithms are compared for different values of N. The results demonstrate the correctness of the parallelization and provide insights into the performance improvement achieved.

Task 3

Label	Result	Time	Count Threads	Acceleration	Efficiency
Not parallel	2.92071	0	1	0	0
Using parallel	2.92071	0	8	0	0
Not parallel	3.80247	0	1	0	0
Using parallel	3.80247	0	8	0	0
Not parallel	4.40678	0.002	1	0	0
Using parallel	4.40678	0	8	0	0
Not parallel	4.81873	0.009	1	0	0
Using parallel	4.81873	0.001	8	9	1.125
Not parallel	5.09941	0.077	1	0	0
Using parallel	5.09941	0.014	8	5.5	0.6875
Not parallel	5.29064	0.757	1	0	0
Using parallel	5.29064	0.132	8	5.73485	0.716856

Task 4: Varying Thread Counts

The program is executed with varying thread counts to evaluate the scalability of the parallel algorithm. Results are presented, and the impact of different thread counts is analyzed.

Task 4

Label	Result	Time	Count Threads	Acceleration	Efficiency
Not parallel	3.80247	0	1	0	0
Using parallel	3.80247	0	5	0	0
Not parallel	4.40678	0.001	1	0	0
Using parallel	4.40678	0	6	0	0
Not parallel	4.81873	0.008	1	0	0
Using parallel	4.81873	0.002	7	4	0.571429
Not parallel	5.09941	0.082	1	0	0
Using parallel	5.09941	0.013	8	6.30769	0.788462
Not parallel	5.29064	0.768	1	0	0
Using parallel	5.29064	0.149	9	5.15436	0.572707
Not parallel	5.42092	7.676	1	0	0
Using parallel	5.42092	1.392	10	5.51437	0.551437
Not parallel	5.50968	76.575	1	0	0
Using parallel	5.50968	18.87	11	4.05803	0.368912

Task 6: Performance for Different N Values

The performance of both sequential and parallel algorithms is evaluated for different values of N. The results provide an understanding of how the algorithm scales with the size of the problem.

```
Task 6
-------------------------------------------------------------------------------------------------
| Label          | Result          | Time            | Count Threads   | Acceleration    | Efficiency  |
-------------------------------------------------------------------------------------------------

Time = 10
-------------------------------------------------------------------------------------------------
| Label          | Result          | Time            | Count Threads   | Acceleration    | Efficiency  |
-------------------------------------------------------------------------------------------------
Not parallel:
|                | 5.46971         | 26.739          | 1               | 0               | 0           |
-------------------------------------------------------------------------------------------------
Using parallel:
|                | 5.46971         | 4.771           | 8               | 5.60449         | 0.700561    |
-------------------------------------------------------------------------------------------------

Time = 20
-------------------------------------------------------------------------------------------------
| Label          | Result          | Time            | Count Threads   | Acceleration    | Efficiency  |
-------------------------------------------------------------------------------------------------
Not parallel:
|                | 5.49394         | 53.97           | 1               | 0               | 0           |
-------------------------------------------------------------------------------------------------
Using parallel:
|                | 5.49394         | 9.476           | 8               | 5.69544         | 0.71193     |
-------------------------------------------------------------------------------------------------

Time = 40
-------------------------------------------------------------------------------------------------
| Label          | Result          | Time            | Count Threads   | Acceleration    | Efficiency  |
-------------------------------------------------------------------------------------------------
Not parallel:
|                | 5.51893         | 113.769         | 1               | 0               | 0           |
-------------------------------------------------------------------------------------------------
Using parallel:
|                | 5.51893         | 20.735          | 8               | 5.48681         | 0.685851    |
-------------------------------------------------------------------------------------------------
```

программа	Tp, c		
	N1	N2	N3
Послед.	26.739	53.97	113.769
OpenMP	4.771	9.476	20.735

программа	Tp, c			Tp, c		
	N1	N2	N3	N1	N2	
Послед.	26.739	53.97	113.769	26.739	53.97	
OpenMP	4.771	9.476	20.735	4.771	9.476	

Task 8: Load Balancing Strategies

The program is executed with different load balancing strategies (static, dynamic, guided) and chunk sizes. The results are presented, and the impact on performance is discussed.

```
Task 8
-------------------------------------------------------------------------------------------------
| Label          | Result          | Time           | Count Threads   | Acceleration    | Efficiency  |
-------------------------------------------------------------------------------------------------

Schedule(static, 1):
|                | 5.46971         | 5.288          | 8               | 1               | 0.125       |
-------------------------------------------------------------------------------------------------

Schedule(static, 100):
|                | 5.46971         | 5.053          | 8               | 1               | 0.125       |
-------------------------------------------------------------------------------------------------

Schedule(static, 10000):
|                | 5.46971         | 5.023          | 8               | 1               | 0.125       |
-------------------------------------------------------------------------------------------------

Schedule(dynamic, 1):
|                | 5.46971         | 102.089        | 8               | 1               | 0.125       |
-------------------------------------------------------------------------------------------------

Schedule(dynamic, 100):
|                | 5.46971         | 5.498          | 8               | 1               | 0.125       |
-------------------------------------------------------------------------------------------------

Schedule(dynamic, 10000):
|                | 5.46971         | 5.337          | 8               | 1               | 0.125       |
-------------------------------------------------------------------------------------------------

Schedule(guided, 1):
|                | 5.46971         | 5.238          | 8               | 1               | 0.125       |
-------------------------------------------------------------------------------------------------

Schedule(guided, 100):
|                | 5.46971         | 5.405          | 8               | 1               | 0.125       |
-------------------------------------------------------------------------------------------------

Schedule(guided, 10000):
|                | 5.46971         | 5.261          | 8               | 1               | 0.125       |
-------------------------------------------------------------------------------------------------
```

Kind	T_P, c		
	C1	C2	C3
Static	5.288	5.053	5.023
Dynamic	102.089	5.498	5.337
guided	5.238	5.405	5.261

4. Выводы

В ходе выполнения расчетно-графической работы мы научились использовать основные директивы, функции и переменные окружения интерфейса OpenMP на примерах написания простейших параллельных программ для многоядерных вычислительных систем