*Лабораторная работа 2*

# Знакомство с возможностями OpenMP

**Цель работы.** *Получение практических навыков разработки параллельных программ с использованием OpenMP. Ознакомление с основными функциями и директивами OpenMP, позволяющими выполнять балансировку нагрузки при обработке данных.*

## Порядок выполнения работы

1. Ознакомиться с теоретическими сведениями.
2. Выполнить задания 1–3.
3. Подготовить отчет по лабораторной работе.
4. Защитить лабораторную работу перед преподавателем

## Задания

**Задание 2.1.** Разработайте консольное приложение, реализующее вычисление числа Pi разными алгоритмами:

– последовательная реализация;

– параллельные реализации с использованием возможностей по управлению нагрузки и синхронизации данных OpenMP;

Для каждой реализации функций вычисления числа Pi выполните расчет времени[1] и сформируйте сводную таблицу (табл. 2.1). В качестве параметра для наборов данных (НД) выступает количество итераций (**num_steps**), которое рекомендуется задавать в зависимости от мощности компьютера.

Таблица 2.1

| Функция (реализация) | Потоки | НД 1 | НД 2 | НД 3 | НД4 |
|---|---|---|---|---|---|
| Последовательная реализация | | *.**** мс. | | | |
| Параллельная реализация FOR (static) | 2 | | | | |
| | 3 | | | | |
| | 4 | | | | |
| Параллельная реализация FOR (dinamic) | 2 | | | | |
| | 3 | | | | |
| | 4 | | | | |
| Параллельная реализация FOR (guided) | 2 | | | | |
| | 3 | | | | |
| | 4 | | | | |
| Параллельная реализация Section | 2 | | | | |
| | 3 | | | | |
| | 4 | | | | |

[1] Расчет времени выполнять в доверительном интервале для не менее чем 100 запусков.

Пример последовательной реализации вычисления числа Pi:

```
long i, num_steps = 1000000;
double step, pi, x, sum = 0.0;
step = 1.0 / (double)num_steps;
for (i = 0; i < num_steps; i++)
{
   x = (i + 0.5) * step;
   sum = sum + 4.0 / (1.0 + x * x);
}
pi = step * sum;
std::cout << "Pi = " << pi;
```

**Задание 2.2.** Разработайте консольное приложение, реализующее заполнение и перемножение матриц **A** и **B** (с размерностью M×N и N×K соответственно). Для перемножения матриц использовать классический алгоритм и один быстрый алгоритм[2]. Каждый студент придумывает свои формулы заполнения для матриц **A** и **B** в зависимости от индексов. Необходимо осуществить следующие последовательную реализацию алгоритмов и параллельнык реализации с использованием различных видов распараллеливания.

При реализации перемножения матриц следует учитывать распределение элементов в памяти, и для одного алгоритма реализовать балансировку нагрузки. Также следует реализовать один из алгоритмов быстрого перемножения матриц.

**Задание 2.3.** Проведите экспериментального исследования по обработке данных с разными реализациями (выполнять не менее 20 запусков для каждой реализации). При проведении исследования используйте массивы вещественного типа размерностью от 800×800 до 2500×2500[3]; четыре разных набора данных (НД). Для реализованных функций на основе полученных данных исследования заполнить табл. 2.2.

Таблица 2.2

| **НД1** (Например, A = 1000×1000  B = 1000×1000) | | | | |
|---|---|---|---|---|
| Функция (реализация) | Поток | Время | $S_p(n)$ | $E_p(n)$ |
| Заполнение матриц (посл.) | | ***.**** мс. | | |
| Заполнение матриц (парал.) Вариант реализации 1 | 2 | | | |
| | 3 | | | |
| | 4 | | | |
| Заполнение матриц (парал.) Вариант реализации 2 | . . . . . | | | |

[2] Пример реализации последовательной версии быстрого алгоритма перемножения доступен на сервере Дистанционного образования СибГУ им. М.Ф. Решетнева

[3] В зависимости от оборудования (на котором будут выполняться эксперименты) и согласованию с преподавателем возможно изменения наборов данных в сторону уменьшения. При этом должно быть несколько наборов данных учитывающих прямоугольные матрицы.

Окончание табл. 2.2

| **НД1** (Например, A = 1000×1000  B = 1000×1000) | | | | |
|---|---|---|---|---|
| Функция (реализация) | Поток | Время | $S_p(n)$ | $E_p(n)$ |
| Перемножение матриц (посл.) | | | | |
| Перемножение матриц (парал.) Вариант реализации 1 | 2 | | | |
| | 3 | | | |
| | 4 | | | |
| Перемножение матриц (парал.) Вариант реализации 2 | . . . . . | | | |
| . . . . . | . . . . . | | | |
| Перемножение быстрый алгоритм (посл.) | | ***.**** мс. | | |
| Перемножение матриц быстрый алгоритм (парал.) | 2 | | | |
| | 3 | | | |
| | 4 | | | |
| **НД2** (Например, A = 1320×1500  B = 1500×1471) | | | | |
| Функция (реализация) | Поток | Время | $S_p(n)$ | $E_p(n)$ |
| Заполнение матриц (посл.) | | ***.**** мс. | | |
| Заполнение матриц (парал.) Вариант реализации 1 | . . . . . | | | |
| . . . . . | . . . . . | | | |
| Перемножение матриц (посл.) | | | | |
| Перемножение матриц (парал.) Вариант реализации 1 | 2 | | | |
| | 3 | | | |
| | 4 | | | |
| . . . . . | . . . . . | | | |

## Отчет

Отчет сдается преподавателю в электронном виде и должен содержать:

– титульный лист;

– цель работы;

– постановку задачи;

– исходный код программы задания 2.1;

– результаты экспериментального исследования программы в табличной форме (задание 2.1);

– формулы, используемые при заполнении матриц (задание 2.2);

– текст программы задания 2.2, с комментариями;

– конфигурация компьютера и параметры операционной системы, на которой производится выполнение задания 2.3;

– результаты выполнения задания 2.3 (экспериментального исследования) в табличной форме;

– выводы по результатам экспериментального исследования;

– выводы по лабораторной работе в целом.

## Контрольные вопросы и задания

1. Приведите примеры параллельных алгоритмов вычисления суммы последовательности числовых значений.
2. Каким образом происходит распределение работы между параллельными потоками?
3. Охарактеризуйте особенности организации параллельной обработки с использованием секций.
4. Раскройте особенности балансировки нагрузки вычислений в параллельных секциях.
5. Охарактеризуйте планировщик распределения итераций цикла между потоками.
6. Охарактеризуйте статическое распределение нагрузки.
7. Охарактеризуйте динамическое распределение нагрузки.
8. Охарактеризуйте управляемое распределение нагрузки.
9. Что нужно сделать для выделения упорядоченного блока в распараллеленном цикле?
10. Охарактеризуйте директивы синхронизации потоков atomic, barrier.
11. Как происходит согласование значений переменных между потоками?
12. Поясните подходы к реализации параллельно-последовательного вычисления числа Pi.
13. Приведите известные вам вариации реализаций классического алгоритма перемножения матриц.
14. Опишите известные вам быстрые алгоритмы перемножения матриц.
15. Приведите рекомендации по разработке параллельных программ.
16. Охарактеризуйте функции блокировки (Lock) в OpenMP.
17. Поясните управление простой блокировкой (omp_lock_t).
18. Дайте определение понятию «задачи в OpenMP» и охарактеризуйте директиву task.
19. Что означает понятие «взаимная блокировка» и когда она возникает?
20. Охарактеризуйте проблему параллельного программирования – неинициализированные переменные.
21. Дайте определение понятию «зависимость данных и гонки в циклах».
22. Приведите параллельные алгоритмы умножения матрицы на вектор.
23. Приведите возможные виды распараллеливания алгоритмов перемножение матриц.