6 Лабораторная работа №3. «Распараллеливание циклов с помощью технологии OpenMP»

6.1 Порядок выполнения работы

1. Добавить во все for-циклы в программе из ЛР №1 следующую директиву OpenMP:
   ”#pragma omp parallel for default(none) private(...) shared(...)”. Наличие всех перечисленных параметров в указанной директиве является обязательным.

2. Проверить все for-циклы на внутренние зависимости по данным между итерациями. Если зависимости обнаружились, использовать для защиты критических секций директиву ”#pragma omp critical” или ”#pragma omp atomic” (если операция атомарна), или параметр reduction (предпочтительнее) или вообще отказаться от распараллеливания цикла (свой выбор необходимо обосновать).

3. Убедиться, что получившаяся программа обладает свойством прямой совместимости с компиляторами, не поддерживающими OpenMP (для проверки этого можно скомпилировать программу без опции ”–fopenmp”, в результате не должно быть сообщений об ошибках, а программа должна корректно работать).

4. Провести эксперименты, замеряя параллельное ускорение. Привести сравнение графиков параллельного ускорения с ЛР №1 и ЛР №2.

5. Провести эксперименты, добавив параметр ”schedule” и варьируя в экспериментах тип расписания. Исследование нужно провести для всех возможных расписаний: static, dynamic, guided. Способ варьирования параметра chunk_size выбрать самостоятельно (но должно быть не менее 5 точек варьирования). Привести сравнение параллельного ускорения при различных расписаниях с результатами п.4.

6. Выбрать из рассмотренных в п.4 и п.5 наилучший вариант при различных N. Сформулировать условия, при которых наилучшие результаты получились бы при использовании других типов расписания.

7. Написать отчёт о проделанной работе.

8. Подготовиться к устным вопросам на защите.

9. **Необязательное задание №1** (для получения оценки «четыре»). Для иллюстрации того, что программа действительно распараллелилась, привести график загрузки процессора (ядер) от времени при выполнении программы при $N = N_1$ для лучшего варианта распараллеливания. Для получения графика можно как написать скрипт так и просто сделать скриншот диспетчера задач, указав на скриншоте моменты начала и окончания эксперимента (в отчёте нужно привести текст скрипта или название использованного диспетчера). Недостаточно привести однократное моментальное измерение загрузки утилитой htop, т.к. требуется привести график изменения загрузки за всё время выполнения программы.

10. **Необязательное задание №2** (для получения оценки «пять»). Построить график параллельного ускорения для точек $N < N_1$ и найти значения N, при которых накладные расходы на распараллеливание превышают выигрыш от распараллеливания (независимо для различных типов расписания).

6.2 Состав отчета

1. Титульный лист с названием вуза, ФИО студентов и названием работы.

2. Содержание отчета (с указанием номера страниц и т.п.).

3. Краткое описание решаемой задачи.

4. Характеристика использованного для проведения экспериментов процессора, операционной системы и компилятора GCC (точное название, номер версии/модели, разрядность, количество ядер и т.п.).

5. Полный текст программы с использованием параметра schedule.

6. Подробные выводы с анализом каждого из приведённых графиков.

7. Отчёт предоставляется в бумажном или электронном виде вместе с полным текстом программы. По требованию преподавателя нужно быть готовыми скомпилировать и запустить этот файл на компьютере в учебной аудитории (или своём ноутбуке).

6.3 Подготовка к защите

1. Уметь объяснить каждую строку программы, представленной в отчёте.

2. Уметь объяснить выводы, полученные в результате работы.

3. Знать назначение каждой директивы OpenMP, использованной в программе.

4. Повторить материал лекции №3, прочитать главу про OpenMP в методическом пособии

5. Прочитать материал о директиве #pragma_omp_for в книге Антонова «Параллельное программирование с использованием технологии OpenMP: Учебное пособие», знать ответы на вопросы в конце соответствующей главы.