Лабораторная работа 2.

Тема: Директива создания параллельной секций секции OpenMP и ее параметры.

Теоретический материал

Директива sections

Директива **sections** используется для реализации функционального распараллеливания. С директивой **sections** ассоциирован набор структурированных блоков, которые распределяются по потокам в группе.

Вспомогательная директива **section** задаёт участок кода внутри секции **sections** для выполнения одним потоком.

Каждый структурированный блок исполняется один раз одним из потоков в группе.

```
#pragma omp parallel sections опция[[[,] опция] ...]
{
#pragma omp section
{структурированный блок}
#pragma omp section
{структурированный блок}
...
}
```

При выполнении этого кода OpenMP сначала создает группу потоков, а затем распределяет между ними выполнение отдельных потоков команд, оформленных в виде структурированных блоков отдельных секций.

Какие именно нити будут задействованы для выполнения какой секции, не определено. Если количество нитей больше количества секций, то часть нитей для выполнения данного блока секций не будет задействована. Если количество нитей меньше количества секций, то некоторым (или всем) нитям достанется более одной секции.

Возможные опции:

private (список) – задаёт список переменных, для которых порождается локальная копия в каждой нити; начальное значение локальных копий переменных из списка не определено;

firstprivate (список) – задаёт список переменных, для которых порождается локальная копия в каждой нити; локальные копии переменных инициализируются значениями этих переменных в нити-мастере;

lastprivate (список) – переменным, перечисленным в списке, присваивается результат, полученный в последней секции;

reduction (оператор:список) – задаёт оператор и список общих переменных; для каждой переменной создаются локальные копии в каждой нити; локальные копии инициализируются соответственно типу оператора (для аддитивных операций – 0 или его аналоги, для мультипликативных операций – 1 или её аналоги); над локальными копиями переменных после завершения всех секций выполняется заданный оператор; оператор это: +, *, -, &, |, ^, &&, ||; порядок выполнения операторов не определён, поэтому результат может отличаться от запуска к запуску;

nowait – в конце блока секций происходит неявная барьерная синхронизация параллельно работающих нитей: их дальнейшее выполнение происходит только тогда, когда все они достигнут данной точки; если в подобной задержке нет необходимости, опция nowait позволяет нитям, уже дошедшим до конца своих секций, продолжить выполнение без синхронизации с остальными.

Рекомендуемые функции библиотеки Open MP:

omp_get_num_threads();

Возвращает количество потоков в параллельной области.

omp_set_num_threads();

Задает значение переменной среды исполнения OMP_NUM_THREADS.

omp_get_thread_num ();

Возвращает номер вызывающего потока.

omp_get_wtime ();

Возвращает в вызвавшем потоке астрономическое время в секундах (вещественное число двойной точности — double), прошедшее с некоторого момента в прошлом. Если некоторый участок программы окружить вызовами данной функции, то разность возвращаемых значений покажет время работы данного участка.

omp_get_wtick()

Возвращает в вызвавшем потоке разрешающую способность таймера в секундах, то есть точность таймера.

Описание лабораторной работы 2

Порядок выполнения

1. Ознакомиться с описанием лабораторной работы. Вариант задания совпадает с вариантом задания на лабораторную работу 1.
2. Проанализировать программу ЛР1 и определить, какие фрагменты программы целесообразно реализовать в виде секции секций.
3. Распараллелить программу с использованием директивы **#pragma omp parallel sections.** Кроме указанной директивы допускается использование функций библиотеки Open MP для работы с переменными окружения.
4. Убедиться, что прежняя и новая версии выдают одинаковые результаты.
5. Зафиксировать время выполнения программы при различных размерах обрабатываемых массивов.

Содержание отчета

Полный отчет предоставляется в электронном виде и/или твердой копии (по требованию преподавателя). Он должен содержать:

1. Титульный лист.
2. Описание полученного задания;
3. Фрагмент кода, в котором проведено распараллеливание.
4. Скриншоты, демонстрирующие результат работы программы и показатели загрузки системы (используется диспетчер задач или его аналог);
5. Таблица времени исполнения последовательной и параллельной программы при различных размерах массивов.
6. Выводы (ОБЯЗАТЕЛЬНО!).