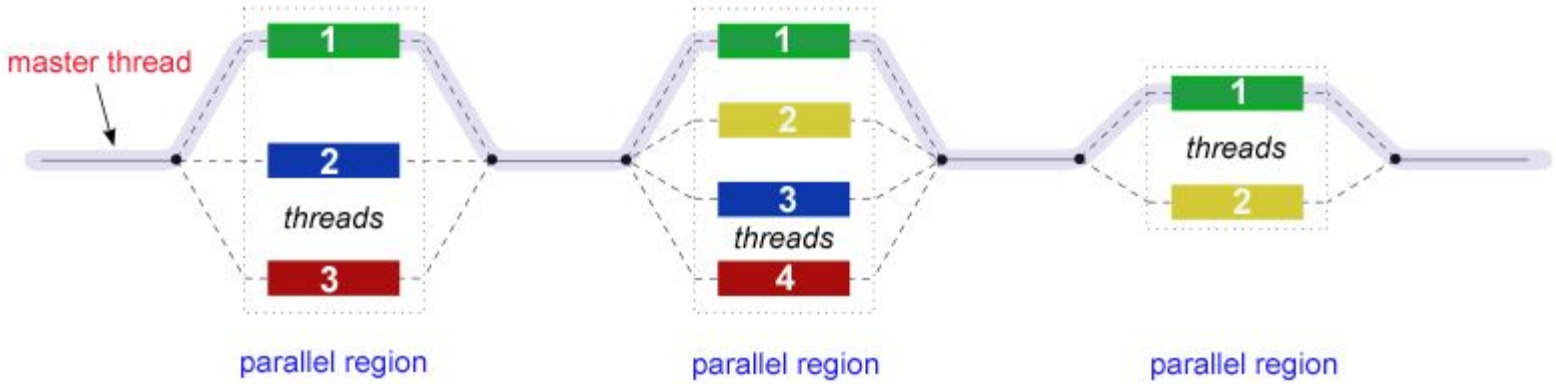
Средства и системы параллельного программирования

Семинар #5
Основы OpenMP

Fork-join model



Директивы в OpenMP

#pragma omp directive-name [clause[[,] clause] ...] **new-line**

Виды директив:

- parallel Construct
- Loop-Related Directives
- Combined Constructs
- Worksharing Constructs
- Synchronization Constructs and Clauses

to be continued……..

Сборка и запуск

gcc -fopenmp hello.c

./a.out

parallel Construct

#pragma omp parallel [clause[[,] clause] ...] new-line

structured-block

полезные clause:

num_threads(integer-expression)
default(firstprivate, none, private, shared)
private(list)
firstprivate(list)
shared(list)
proc_bind(master | close | spread)

Число потоков в OpenMP-программе

- `export OMP_NUM_THREADS=4` - *выставление переменной окружения (до окончания сессии в терминале)*
- `OMP_NUM_THREADS=4 ./a.out` - *выставление переменной окружения для конкретного запуска*

- `void omp_set_num_threads(int num_threads);` - функция для вызова внутри кода, до параллельной области
- num_threads в clause у директивы parallel

omp_get_thread_num() - идентификация номера потока

Single (worksharing construct)

#pragma omp single [clause[[,] clause] ...] new-line

structured-block

полезные clause:

private(list)

firstprivate(list)

copyprivate(list)

Master (work-sharing construct)

#pragma omp master new-line

structured-block

Loop-Related Directives

#pragma omp for [clause[[,] clause] ...] new-line

for-loops

полезные clause:
private(list)

firstprivate(list)

lastprivate([lastprivate-modifier:] list)

reduction([reduction-modifier,]reduction-identifier : list)

schedule([modifier [, modifier]:]kind[, chunk_size])

nowait

Измерение времени

`double omp_get_wtime(void);` - потокобезопасная функция, ей можно измерять время выполнения одного потока

```
double start;
double end;
start = omp_get_wtime();
... work to be timed ...
end = omp_get_wtime();
```

Combined parallel-loop directive

#pragma omp parallel for [clause[[,] clause] ...] new-line

for-loops

В качестве clause можно использовать любые clause из parallel и for

Вложенный параллелизм

```
void omp_set_dynamic(int dynamic_threads);

void omp_set_nested(int nested);
```

Задание

На отрезке [a, b] задана точка x, $a < x < b$; a, x, b - целые числа

Задана вероятность p перехода точки вправо. В момент времени i точка совершает переход **с шагом 1** направо или налево (с вероятностью p или $1 - p$, соответственно).

Процесс останавливается, когда точка достигает точки a или точки b.

Рассмотрим N частиц, совершающих случайные блуждания, начиная с точки x. Интересующие нас результаты модели случайных блужданий - частоты попадания в каждое из поглощающих состояний и среднее время блужданий частиц.

Подобный эксперимент относится к классу методов Монте-Карло.

Задание (продолжение)

Задача: реализовать параллельный алгоритм для модели случайных блужданий с использованием OpenMP

Программа должна работать при любых значениях *a*, *b*, *p*, *x*, *N, P* (число потоков) и выдавать в качестве результата

- вероятность достижения *b*,
- среднее время жизни одной частицы
- время работы основного цикла (*for (i = 0, i < N, ...)*)

Для отчёта: (запуски выполняем на Polus)

Составить график зависимости *T*(*N*), *S*(*N*), *E*(*N*) при фиксированном значении *P* ***!=*** *1*

Составить график зависимости *T*(*P*), *S*(*P*), *E*(*P*) при фиксированном **большом** значении *N*. Для значений *P* достаточно брать {1, 2, 4, 8, 16}.

Дедлайн: 28.10, 4.11

T - время работы программы (основного цикла)

S - ускорение

E - **Эффективность распараллеливания** (parallel efficiency) $E=S/p$ (где S - полученное ускорение работы программы)