Технологии параллельного программирования на C++

Семинар 6

OpenMP

Замки(locks)

- Один из вариантов синхронизации
- Два типа замков:
  - Простые
    omp_init_lock(omp_lock_t *lock) – инициальзация замка
    omp_destroy_lock(omp_lock_t *lock) – освобождение переменной замка
    omp_set_lock(omp_lock_t *lock) – захвать замка
    omp_unset_lock(omp_lock_t *lock) – освобождение замка

  - Множественные (Могут неоднократно захватываться одной нитью, устанавливается счетчик количества захватов. Когда он становится равным нулю, замок освобождается)
    omp_init_nest_lock(omp_lock_t *lock) – инициальзация замка
    omp_destroy_nest_lock(omp_lock_t *lock) – освобождение переменной замка
    omp_set_nest_lock(omp_lock_t *lock) – захвать замка
    omp_unset_nest_lock(omp_lock_t *lock) – освобождение замка

Пример

```
#include <iostream>
#include <omp.h>
#include <unistd.h>
using namespace std;
omp_lock_t lock;
int main()
{
   int n;
   omp_init_lock(&lock);
#pragma omp parallel private (n)
   {
      n=omp_get_thread_num();
      omp_set_lock(&lock);
      cout<<n<<endl;
      sleep(2);
      cout<<n<<endl;
      omp_unset_lock(&lock);
   }
   omp_destroy_lock(&lock);
}
```

Директива **atomic**

- Частный случай использования критических секций в случае обновления общей переменной

- **#pragma omp atomic**

```
int a = 0;
#pragma omp parallel
{
        #pragma omp atomic
        a++;
}
```

Задача 1
Поиск определителя

- Найти определитель матрицы с большой размерностью
- Построить графики ускорения и эффективности

Задача 2
Решение СЛАУ методом Гаусса

- $Ax = b$

- Задать матрицу A большой размерности
- Проверить, что матрица невырожденная
- Если матрица невырожденная, то решаем систему линейных алгебраических уравнений с помощью метода Гаусса
- Построить графики ускорения и эффективности