Introduction to Parallel & Distributed Programming

Lec 04—Shared Memory Programming

Subodh Sharma | Jan 12, 2026

Recap

- Parallelism in S/w
  - Models of Parallelism (SIMD, MIMD)
  - Models of Communication (Synchronisation) — Shared Memory, Message Passing
  - Example — Data Parallel Program (using OpenMP)
  - Data races and Atomicity violations (will revisit again - later)

Data Parallelism Example: Parallel Version

```
static inline double heavy(double x) {
    for (int k = 0; k < 50; k++) {
        x = sin(x) + cos(x) + sqrt(x*x + 1.0);
    }
    return x;
}
```

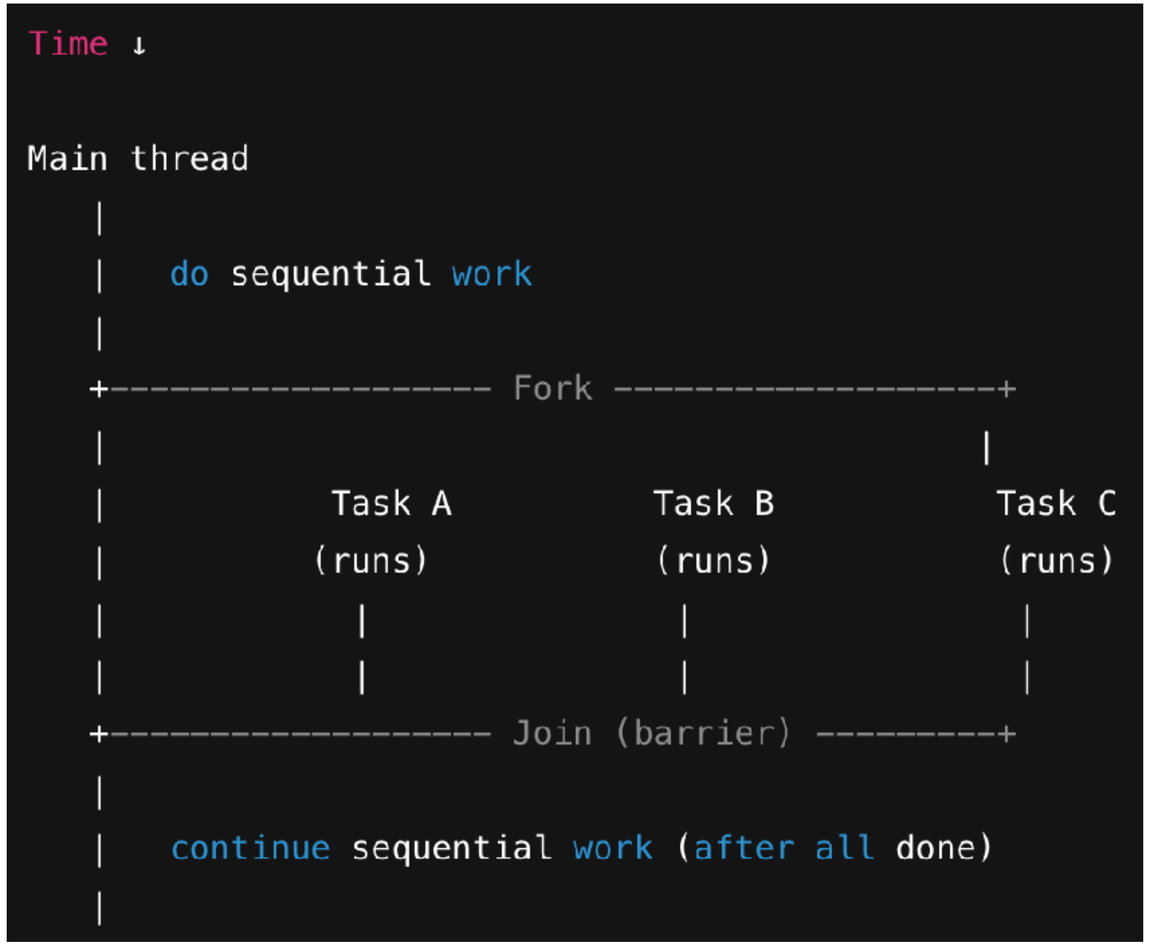
Function

```
int main() {

  const int N = 1000 * 1000;    // 1 million

   t1 = omp_get_wtime();
#pragma omp parallel for
   for (int i = 0; i < N; i++) B2[i] = heavy(A[i]);
   t2 = omp_get_wtime();
   double parallel = t2 - t1;
```

OMP pragma to parallelise loop iterations

OpenMP: A Quick Introduction

- An API for C/C++ and FORTRAN
- Mostly captures **Fork-Join Parallelism**
  - But from OpenMP 3.0, **task-based parallelism** is also supported
    - Supports highly unstructured and dynamic parallelism through the **tasking model (will be covered later)**
  - The example on data parallelism represented fork-join parallelim
    - **`#pragma omp parallel`** and **`#pragma omp parallel for`** are a classic fork–join: threads start together (fork), split loop iterations, then implicitly synchronize at the end of the `for` (join) unless `nowait` is used.



Fork-Join parallelism

OpenMP: A Quick Introduction

#pragma omp parallel

```c
#include <omp.h> // required header to write OpenMP code
#include <stdio.h>

int main (){

  // sentinel directi

#pragma omp parallel
  {
    int tid = omp_get
    printf("hello wor

  } // implicit barri
  return 0;
}
```

```
Subodhs-MacBook-Pro:openmp-prgs svs$ emacs -nw omp_parallel.c
Subodhs-MacBook-Pro:openmp-prgs svs$ gcc-15 -fopenmp omp_parallel.c
Subodhs-MacBook-Pro:openmp-prgs svs$ ./a.out
hello world 3
hello world 2
hello world 1
hello world 5
hello world 4
hello world 6
hello world 0
hello world 7
```

Compiling OpenMP Programs

- **`gcc-15 -fopenmp <prg_name>.cpp`**
- To change the number of threads
  - In bash: **`export OMP_NUM_THREADS=8`**
  - Number of threads can also be specified in the program as a clause in the OMP directive

OpenMP: A Quick Introduction

#pragma omp parallel

```c
#include <omp.h> // required header to write OpenMP code
#include <stdio.h>

int main (){

  // sentinel directi

#pragma omp parallel
  {
    int tid = omp_get
    printf("hello wor

  } // implicit barri
  return 0;
}
```

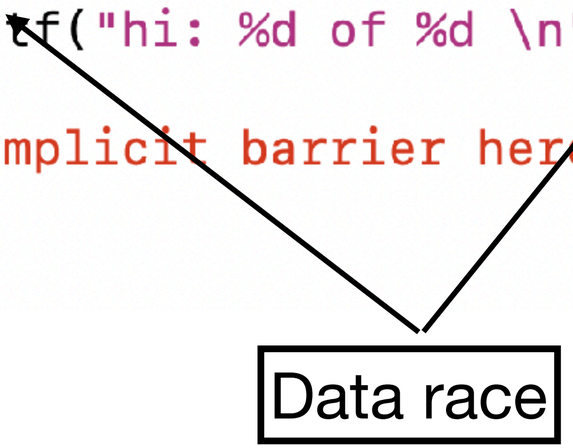
```
Subodhs-MacBook-Pro:openmp-prgs svs$ emacs -nw omp_parallel.c
Subodhs-MacBook-Pro:openmp-prgs svs$ gcc-15 -fopenmp omp_parallel.c
Subodhs-MacBook-Pro:openmp-prgs svs$ ./a.out
hello world 3
hello world 2
hello world 1
hello world 5
hello world 4
hello world 6
hello world 0
hello world 7
```

OpenMP: A Quick Introduction

DATA RACES

- Operations are not instantaneous
- Such operations become visible to different threads at different times
- **Data Race**: When two concurrent operations are accessing the same memory location and at least one of them is a write: R-W, W-W
- **Race condition:** When concurrent requests to access a common resource are made

```
int main (){

  int numt, tid ;

#pragma omp parallel num_threads (4)
  {
    numt = omp_get_num_threads();
    tid = omp_get_thread_num();
    printf("hi: %d of %d \n", tid, numt);

  } // implicit barrier here!

}
```



OpenMP: A Quick Introduction

Fixing the DATA RACE

- Observe that `tid` need not be shared
- `tid` can be made thread-local
- OMP provides a declaration to make specific set of variables **`private`** and **`shared`**

Execution Model