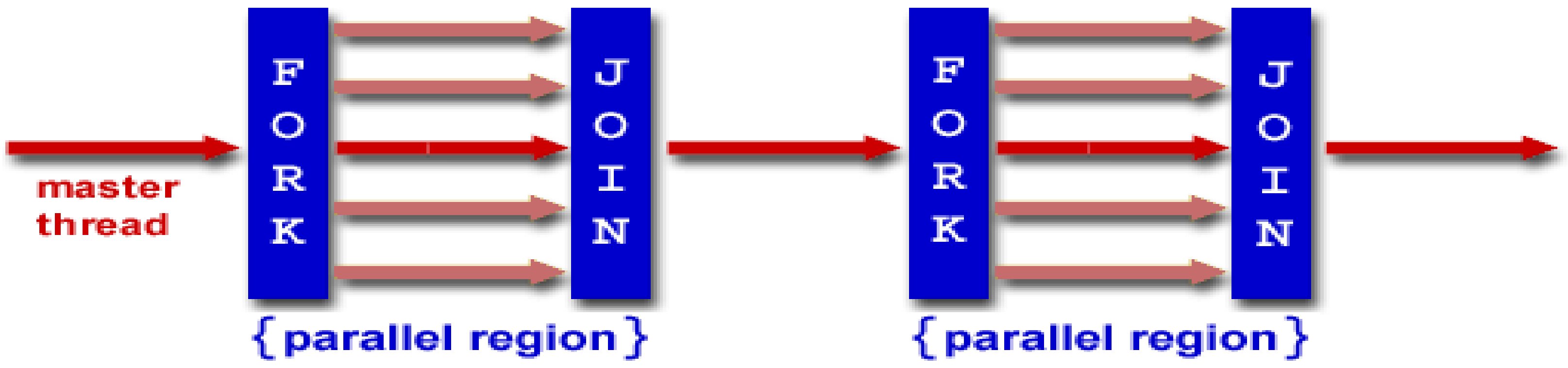
OpenMP

OpenMP

- OpenMP (Open Multi-Processing) is a C/C++ and Fortran Application Programming Interface for shared memory architectures.

- OpenMP is based on the Fork and Join model:

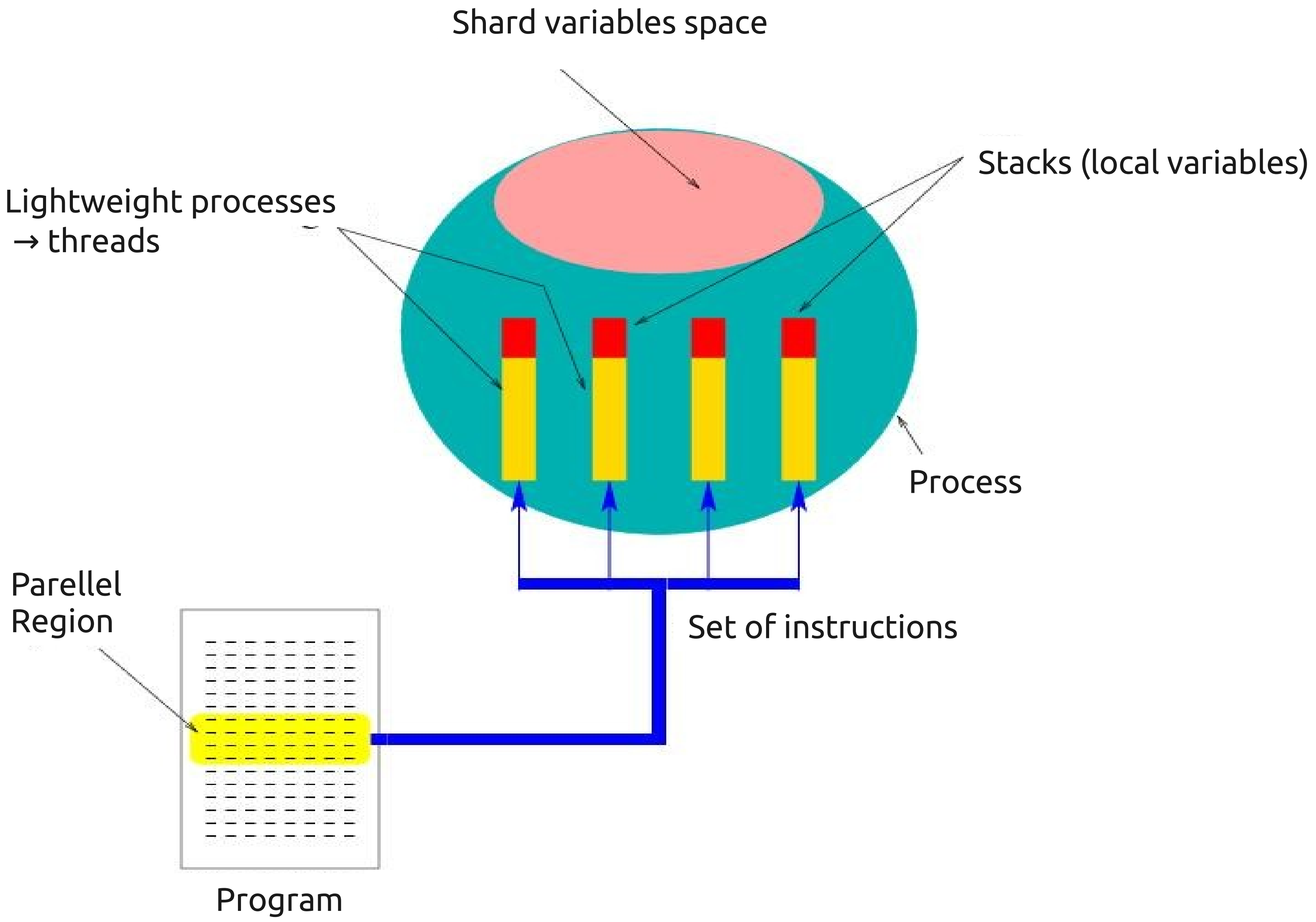


- OpenMP consists in:
  - ✓ A set of compiler directives.
  - ✓ Library functions calls
  - ✓ Environment variables

OpenMP

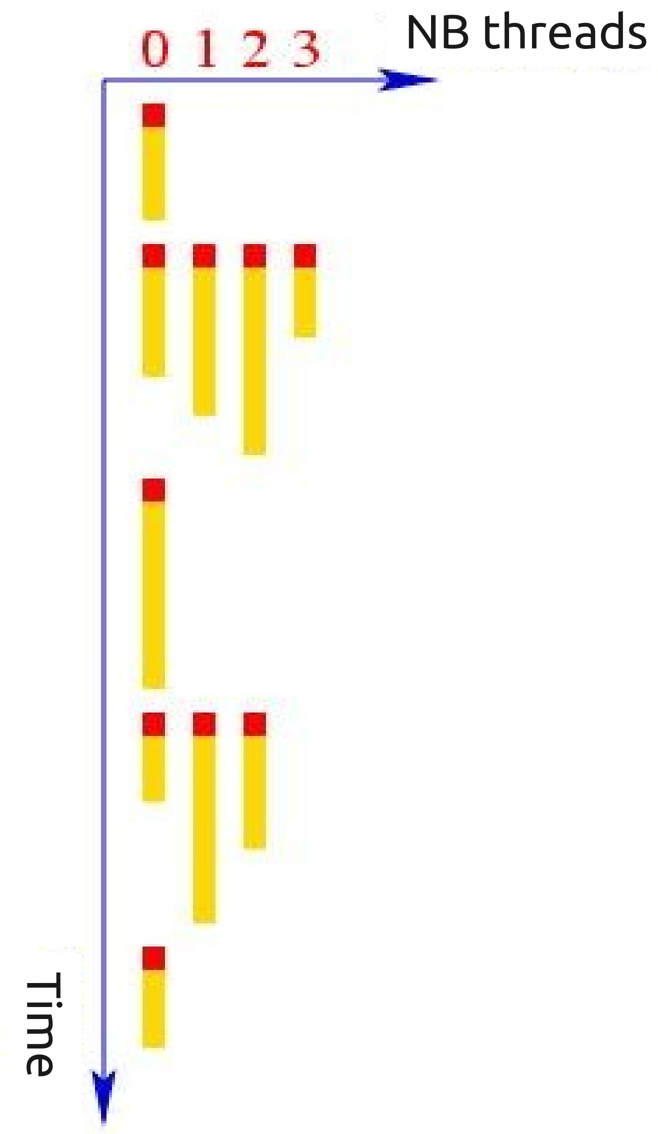
- An OpenMP program is executed by one process → thread master.
- The process activates several lightweight processes (ie. threads) when a parallel region starts → Fork.
- The code of the parallel region is duplicated and each thread executes that code.
- Different threads executes the code at the same time.
- At the end of a parallel region (ie. join), only the master thread continues execution.
- During the execution a the threads, a variable can be read/written:
  - ✓ If the variable is in the thread's stack → private variable
  - ✓ If the variable defined in a shared memory space → shared variable

OpenMP


Shard variables space
Stacks (local variables)
Lightweight processes
→ threads
Process
Parellel
Region
Set of instructions
Program

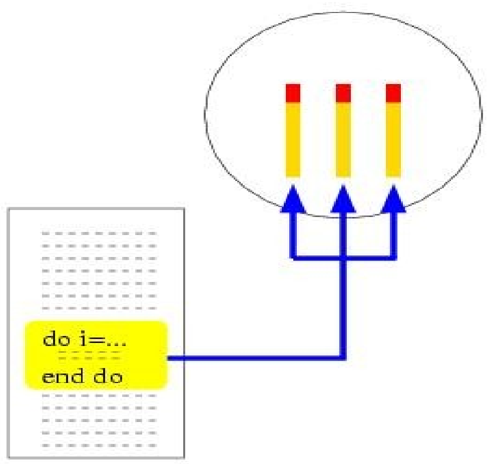
OpenMP

- An OpenMP program alternates sequential and parallel regions.
- The sequential region is always executed by the thread master → thread 0.
- A parallel region is executed by different threads at the same time.
- The threads may share the work inside the parallel region.

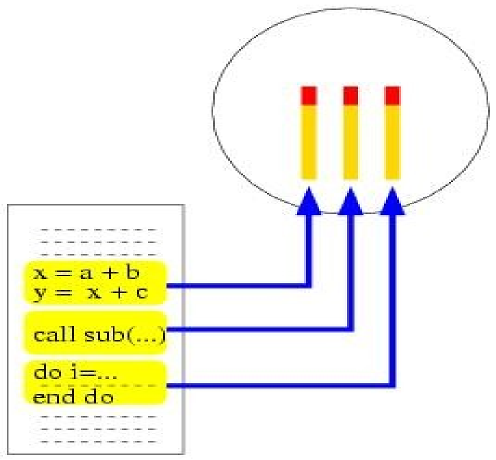


OpenMP

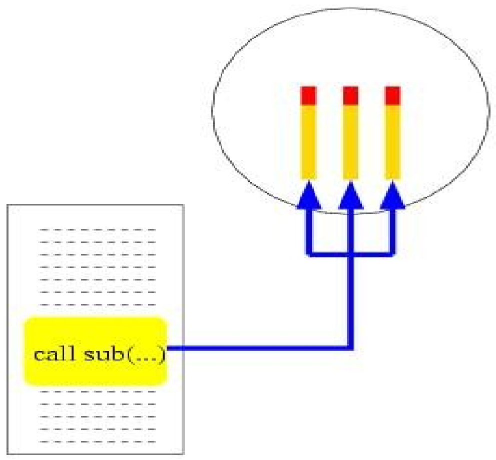
- What is the work sharing between threads?
  - ✓ Share the iteration space of a loop between threads.
  - ✓ Execute different sections of a program but one section per thread.
  - ✓ Execute different occurrences of a procedure by different threads.



Loop Level Parallelism



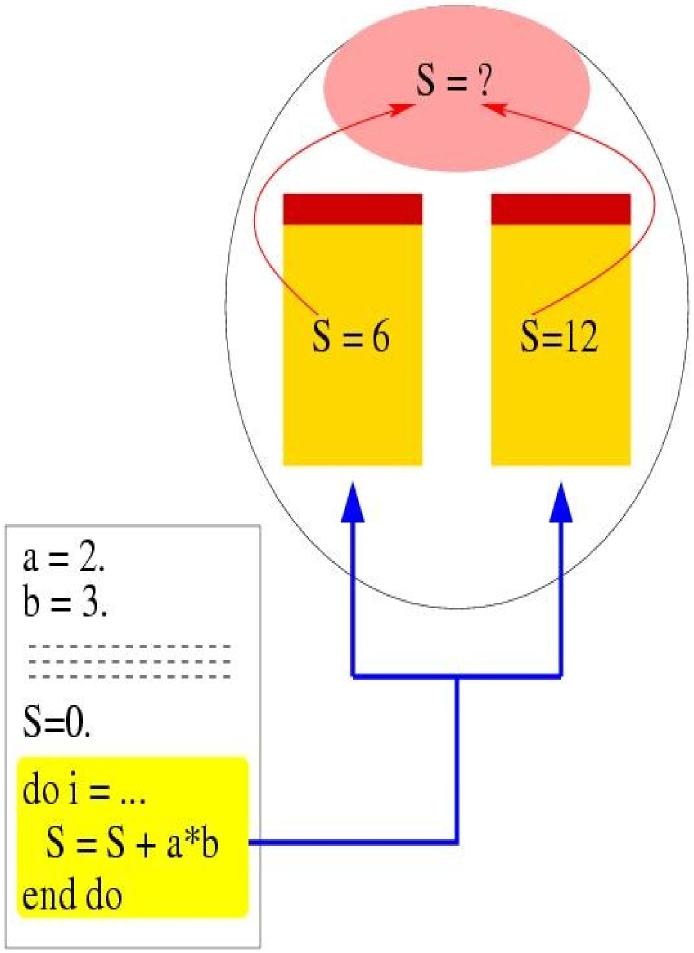
Parallel Sections



Parallel Procedures

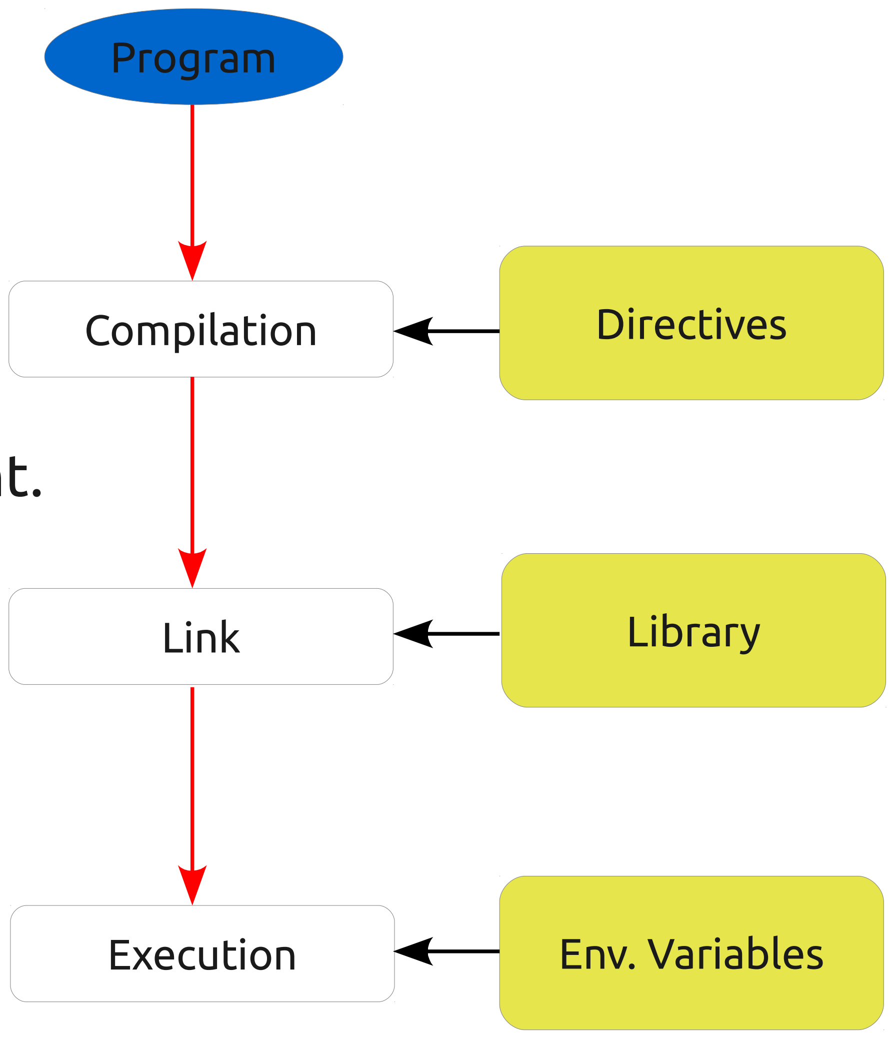
OpenMP

- What is data race?
  - ✓ It is the access to a shared variable by different threads and at least one access is a write.
- In case of data race, synchronization between threads is mandatory.
- For example, in case of a reduction, a synchronization is needed to avoid the modification, of the value of the shared variable, in an incorrect order.



OpenMP

- Compilation directives:
  - ✓ Define the work sharing.
  - ✓ Synchronization.
  - ✓ Privacy of variables.
  - ✓ If correct flag not set, the compiler consider the directive as a comment.
- Library functions calls:
  - ✓ It is loaded at link.
- Environment variables:
  - ✓ When set up, their values are considered at execution time.



OpenMP - Syntax

- The OpenMP directives are:
  - ✓ Inserted in the source code by the programmer
    OR
  - ✓ Inserted automatically in the source code (ie. automatic parallelization)

- An OpenMP directive has the following shape:

  #pragma omp *directive [clause[clause]...]* *for C/C++*

  sentinel directive [clause[clause]...] *for Fortran*

- There is an include file "*omp.h*":
  - ✓ It defines all OpenMP functions.
  - ✓ It should be included in each OpenMP program to be able to use the functions.

OpenMP - Syntax

```
#include <omp.h>

…

#pragma omp parallel private(a,b) \
              shared(d,c)
{

…

}
```