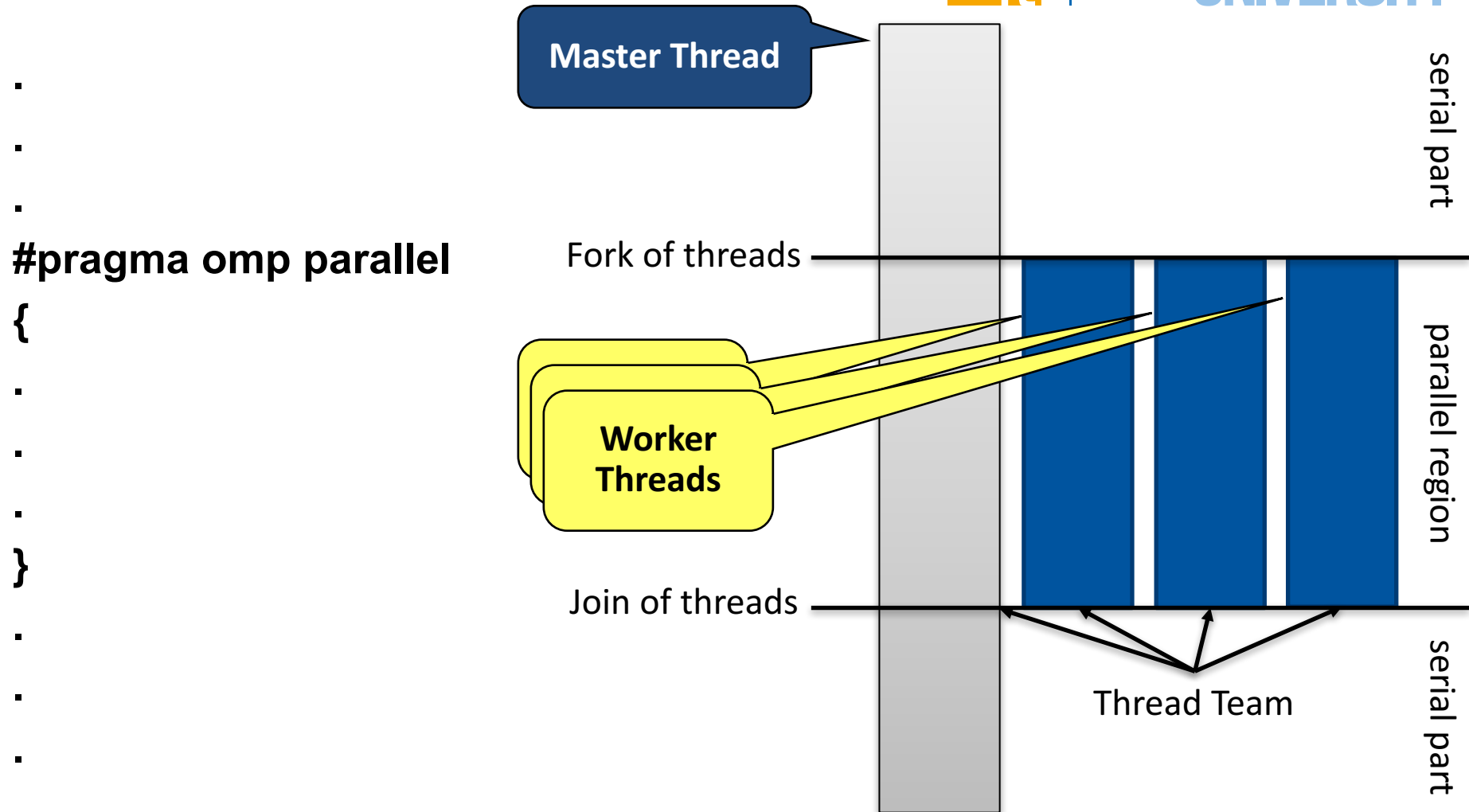


# Summary: Introduction to OpenMP

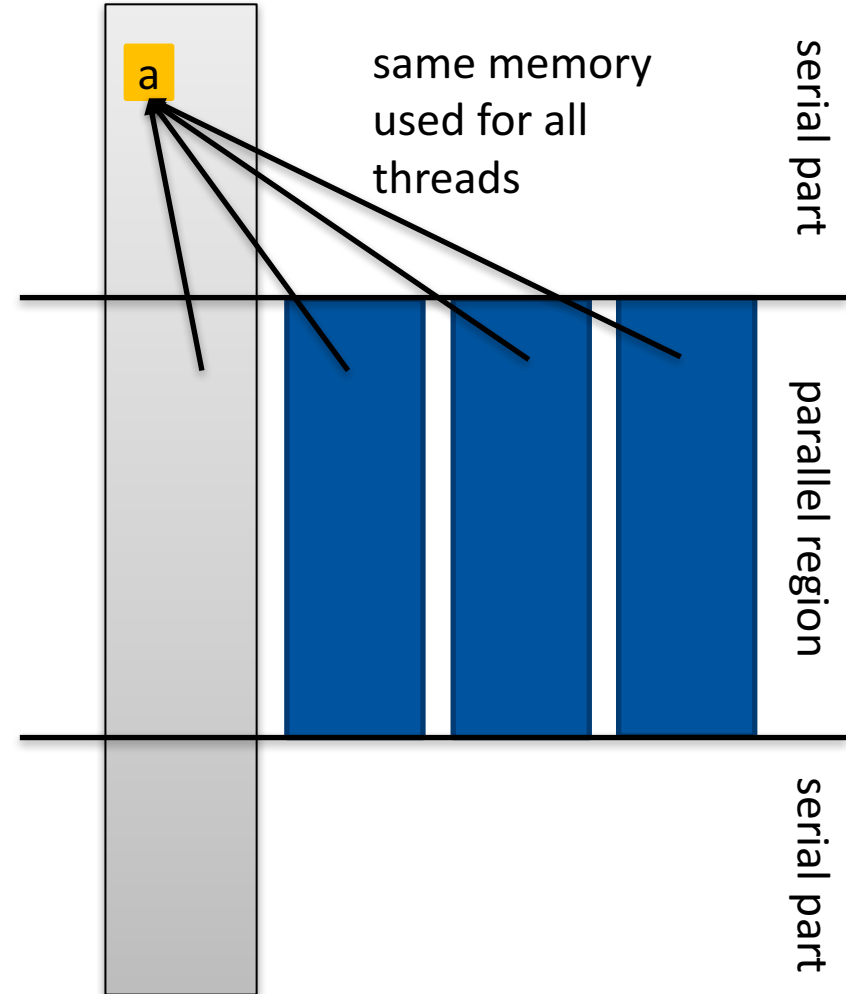
# Fork-Join Execution Model



# Data Sharing Attributes



```
int a;  
.  
.  
.  
#pragma omp parallel shared(a)  
{  
.  
.  
.  
.  
}  
.  
.  
.
```

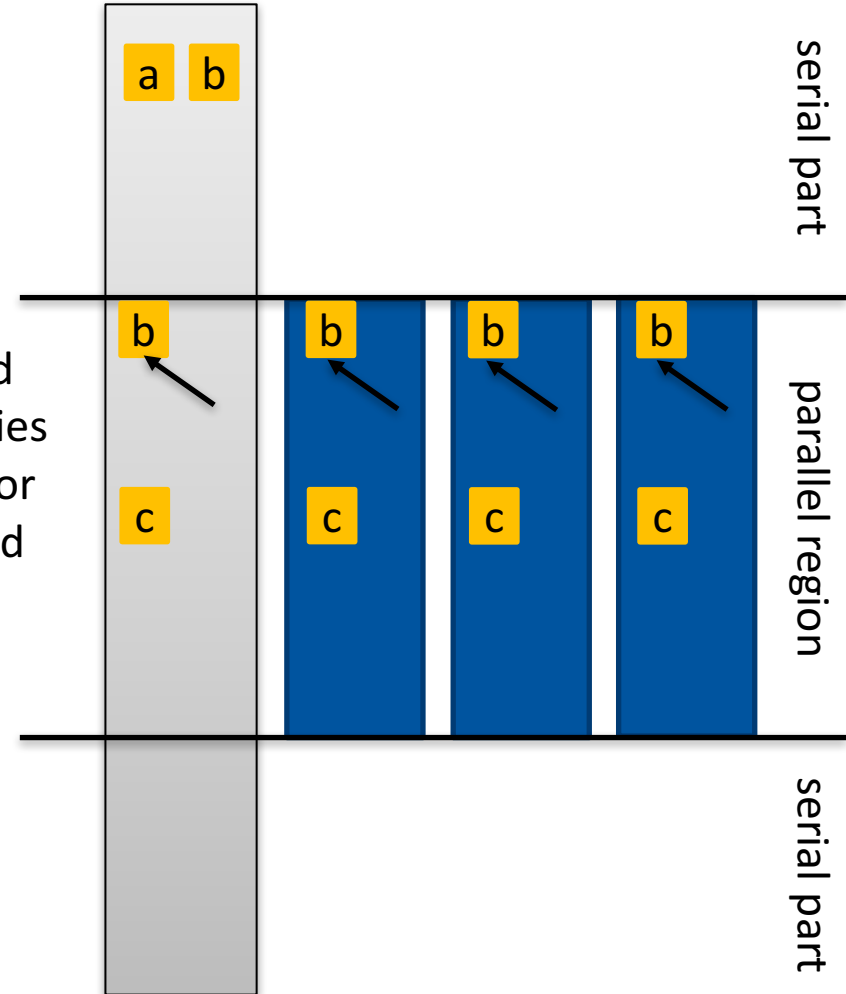


# Data Sharing Attributes



```
int a,b;  
.  
#pragma omp parallel shared(a) //  
  private(b)  
{  
.  
int c;  
.  
}  
.  
.  
.
```

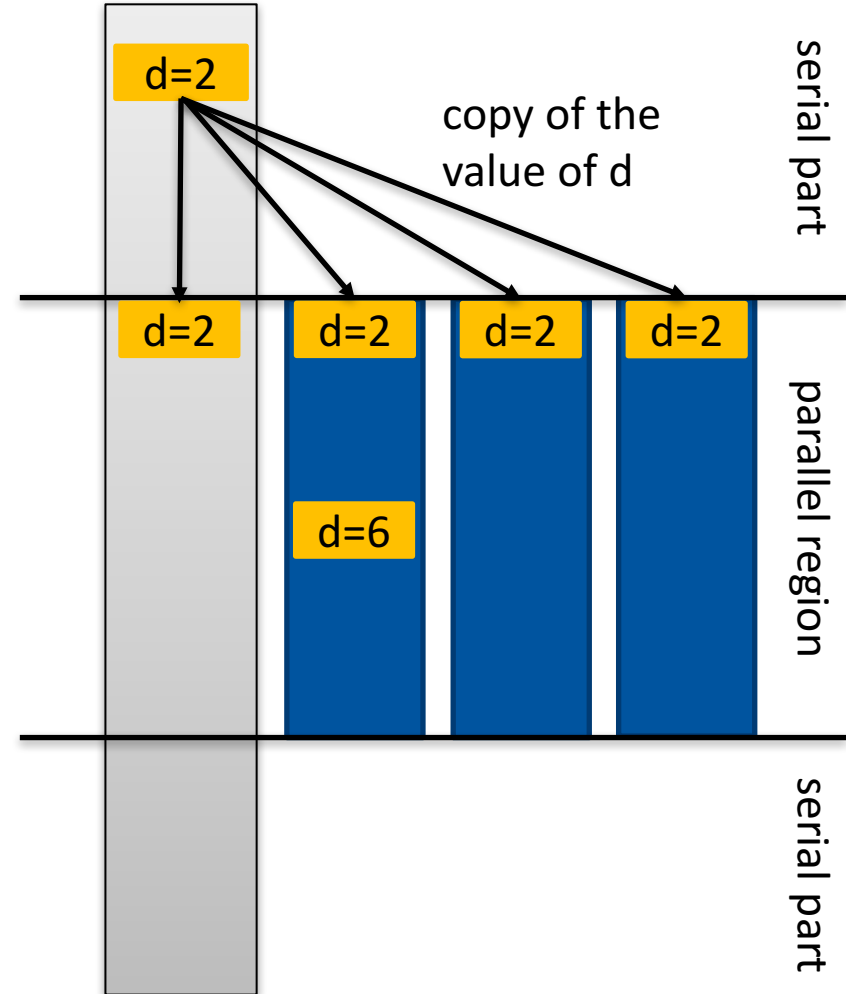
uninitialized  
private copies  
of b and c for  
every thread



# Data Sharing Attributes



```
int d=2;
.
.
#pragma omp parallel firstprivate(d)
{
  #pragma omp single
  {d=6;}
.
.
}
```

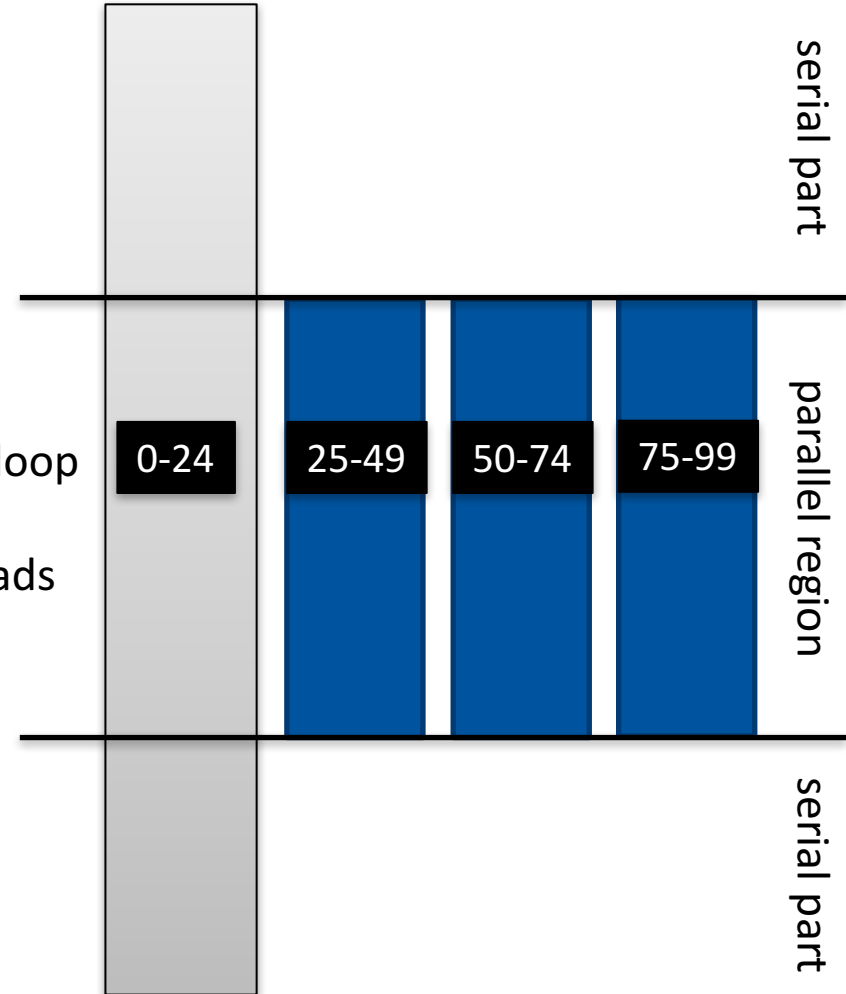


# For Worksharing



```
.  
. .  
. .  
#pragma omp parallel  
#pragma omp for  
for (int i=0; i<100; i++){  
. .  
. .  
. .  
}  
. .  
. .
```

distributes loop iterations across threads



- **Loop iterations must be independent to parallelize a loop!**

No loop dependencies => parallelizable

```
#pragma omp parallel for
for ( i=0 ; i<100 ; i++ ){
    a[i] = b[i] + c[i];
}
```

Loop dependencies => **not** parallelizable

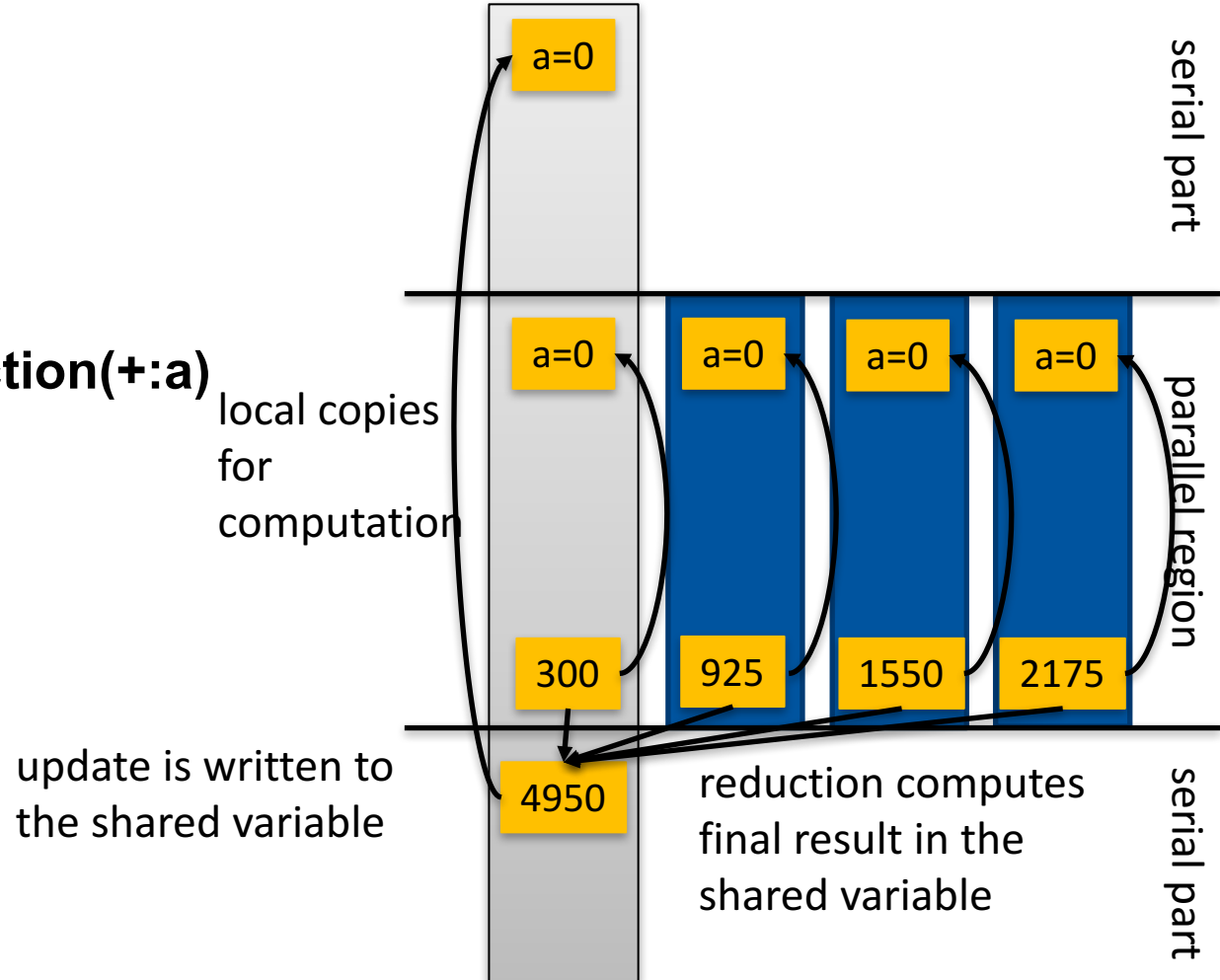
```
#pragma omp parallel for
for ( i=1 ; i<100 ; i++ ){
    a[i] = a[i] + a[i-1];
}
```

- **Simple test: If the results differ when the code is executed backwards, the loop iterations are not independent.**  
**BUT: This test alone is not sufficient**

# Reduction Operations



```
int a=0;
.  
.  
#pragma omp parallel  
#pragma omp for reduction(+:a)  
for (int i=0; i<100; i++)  
{  
    a+=i;  
}
```



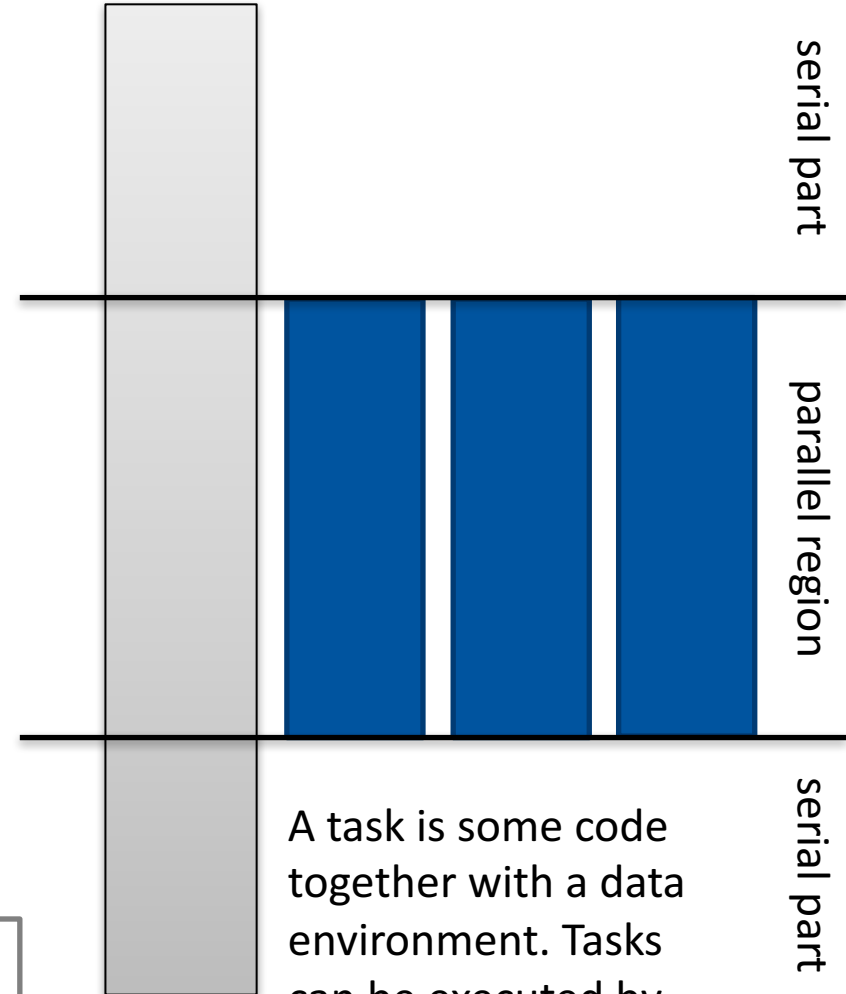
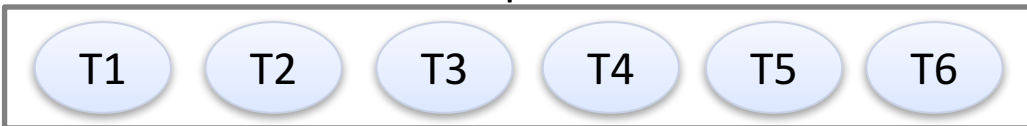


# Tasks



```
#pragma omp parallel
#pragma omp single
while (work()){
    #pragma omp task
    {
        ...
    }
} // implicit barrier here
```

Taskqueue



A task is some code together with a data environment. Tasks can be executed by any thread in any order.

## ■ OpenMP `barrier` (implicit or explicit)

→ All tasks created by any thread of the current *Team* are guaranteed to be completed at barrier exit

```
C/C++
```

```
#pragma omp barrier
```

## ■ Task barrier: `taskwait`

→ Encountering Task suspends until child tasks are complete

→ Only direct childs, not descendants!

```
C/C++
```

```
#pragma omp taskwait
```



# Questions?