Taming Wild Threads

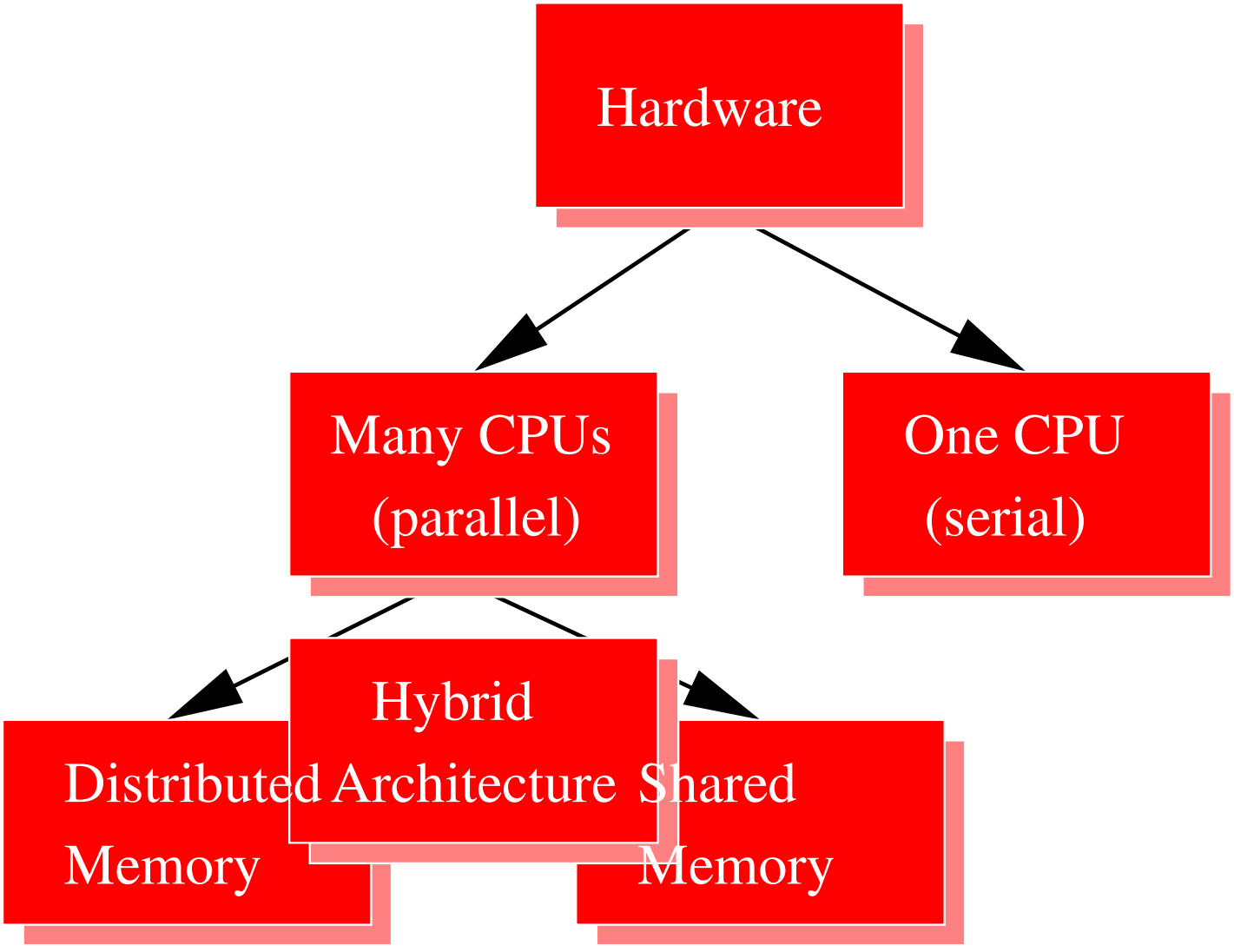
Tips and Pitfalls in Hybrid Programming

Christoph Pospiech

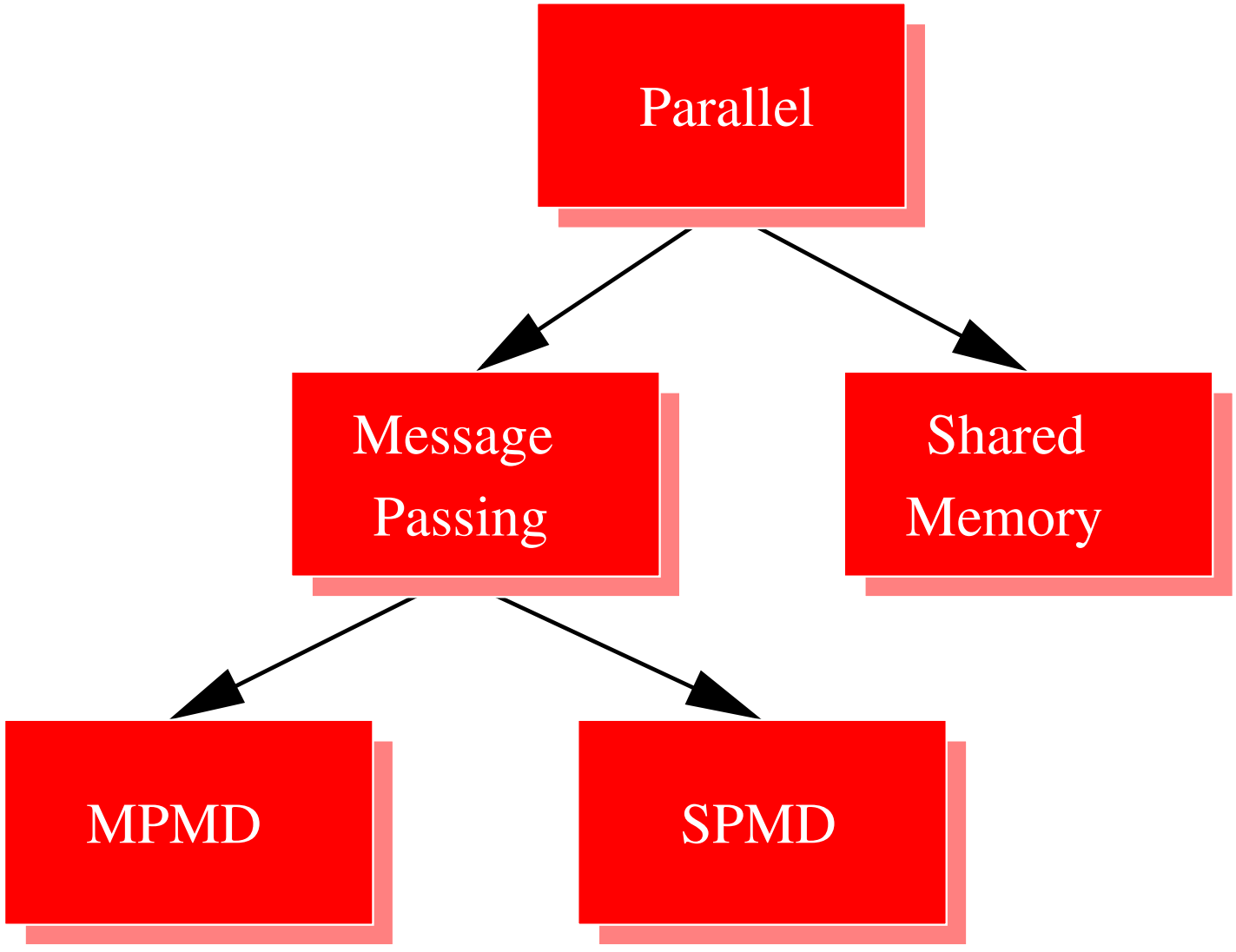
HPC Performance Engineering
Lenovo Global Technology Germany GmbH
Dresden

November 24, 2017

Parallel Computer Basics



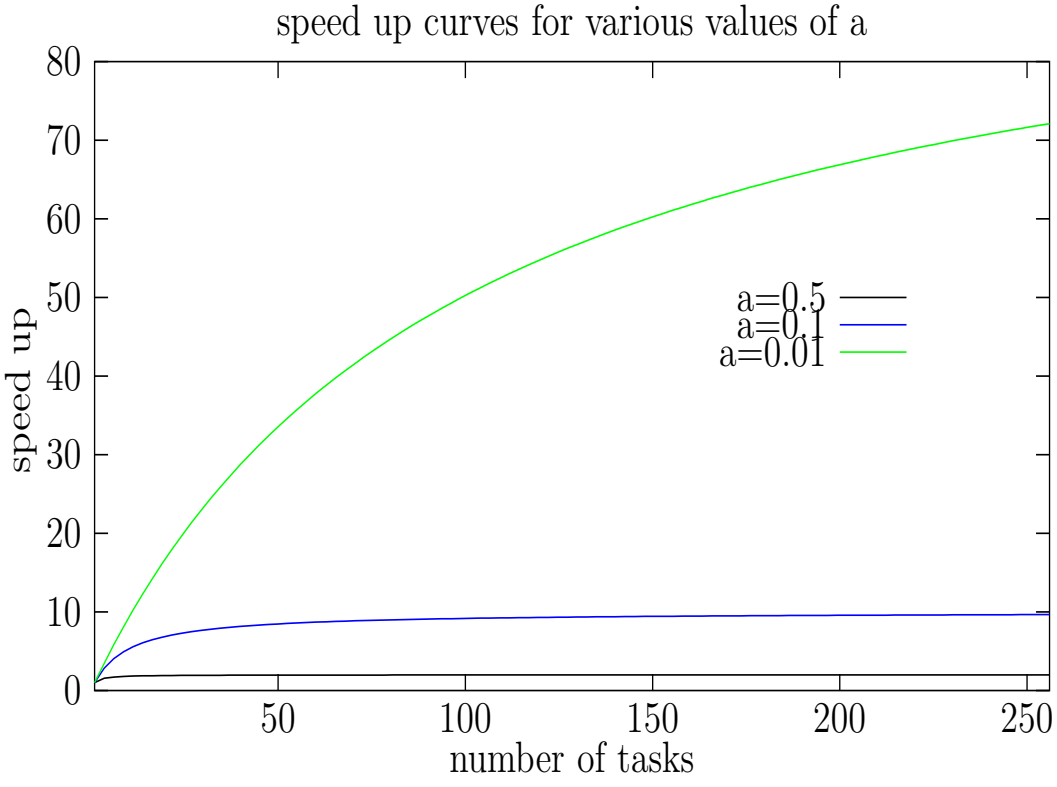
Basic Programming Paradigms



Amdahls Law

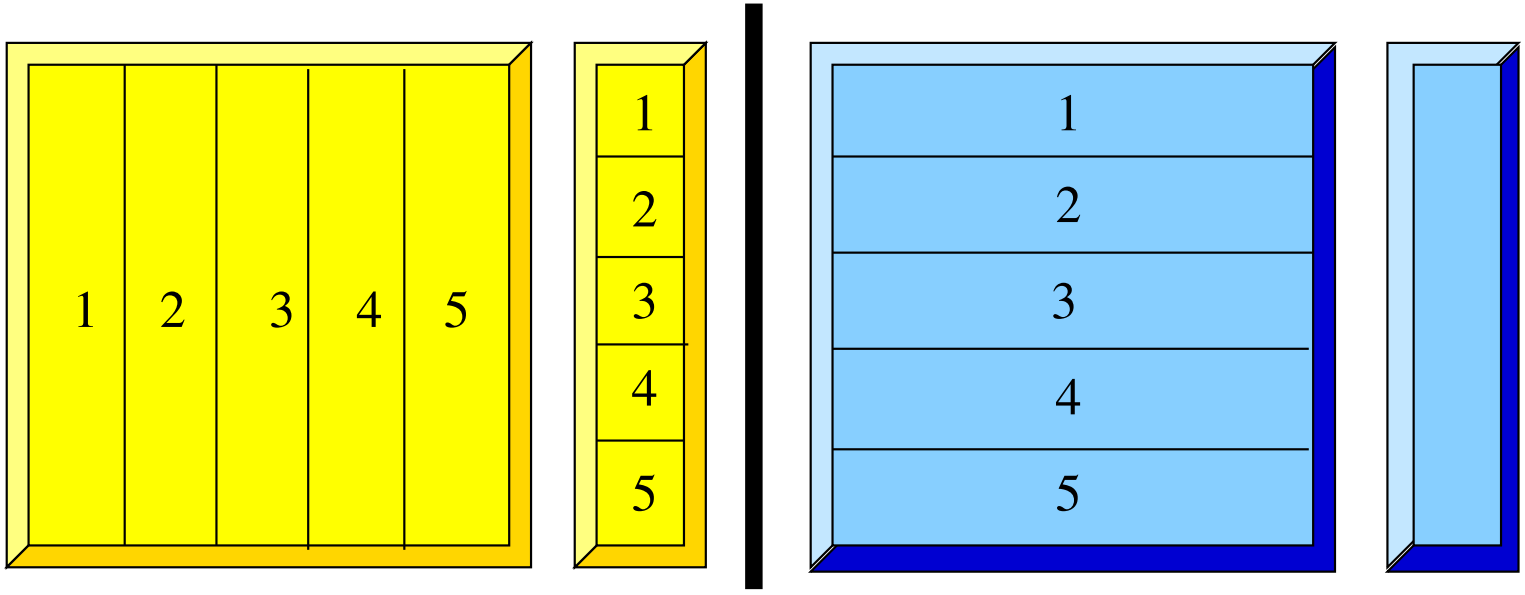
- $t(1) = t_{ser} + t_{par}$
- $t(p) = t_{ser} + t_{par}/p$
- $a = t_{ser}/t(1)$
- Speed up:

$$s = t(1)/t(p)$$
$$= 1/(a + \frac{1-a}{p})$$
$$= 1/a \quad p \to \infty$$



Choose Data Distribution

Matrix Vector Multiplication



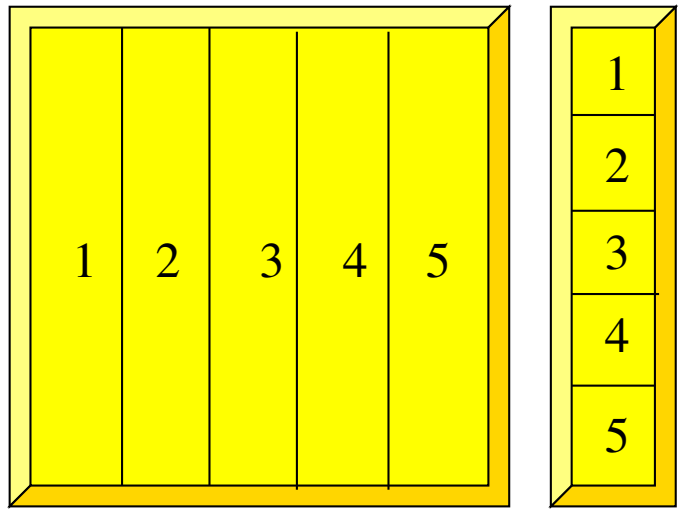
Stride 1 access

Access with large stride

Check Loop Boundaries

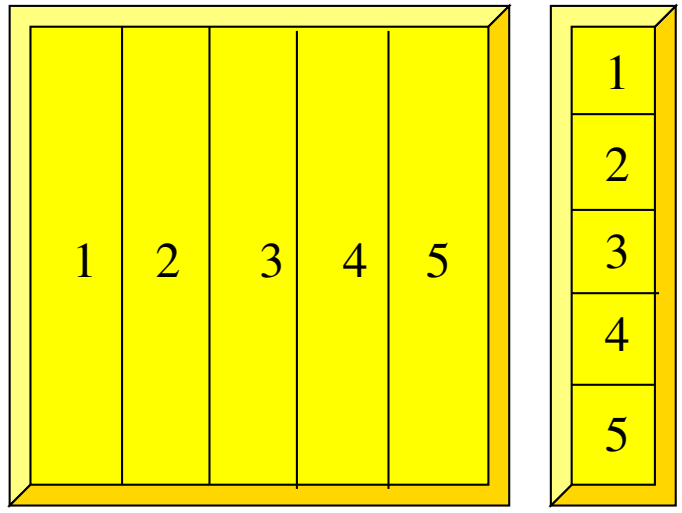
Do $j = 1, n\_loc$
  Do $i = 1, n$
    $c(i) = c(i) + a(i,j) * b(j)$
  end Do
end Do

$n\_loc = ncols / nprocs$

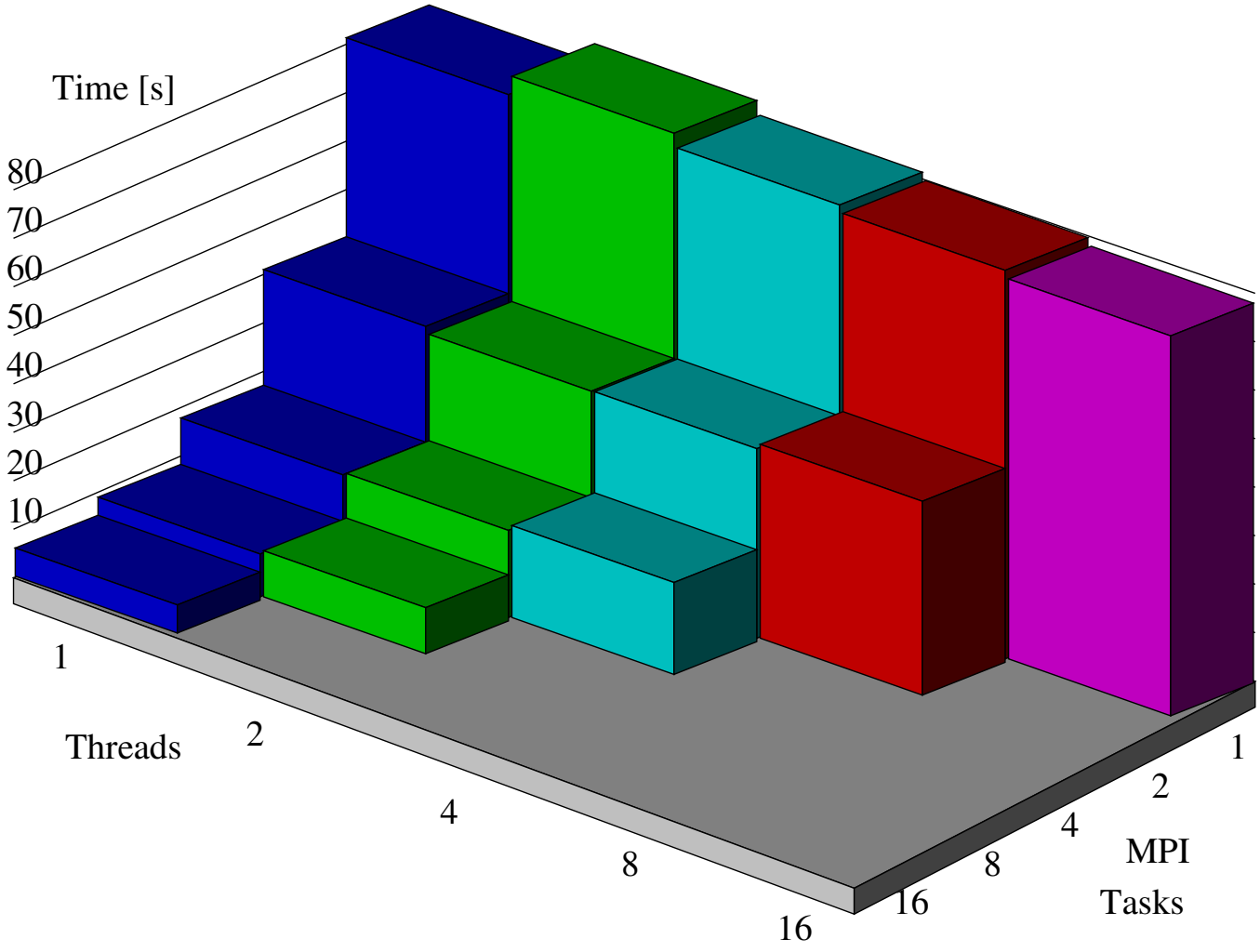


Set Up MPI Communication

Do $j = 1, n\_loc$
  Do $i = 1, n$
    $c(i) = c(i) + a(i,j) * b(j)$
  end Do
end Do
call mpi_reduce_scatter



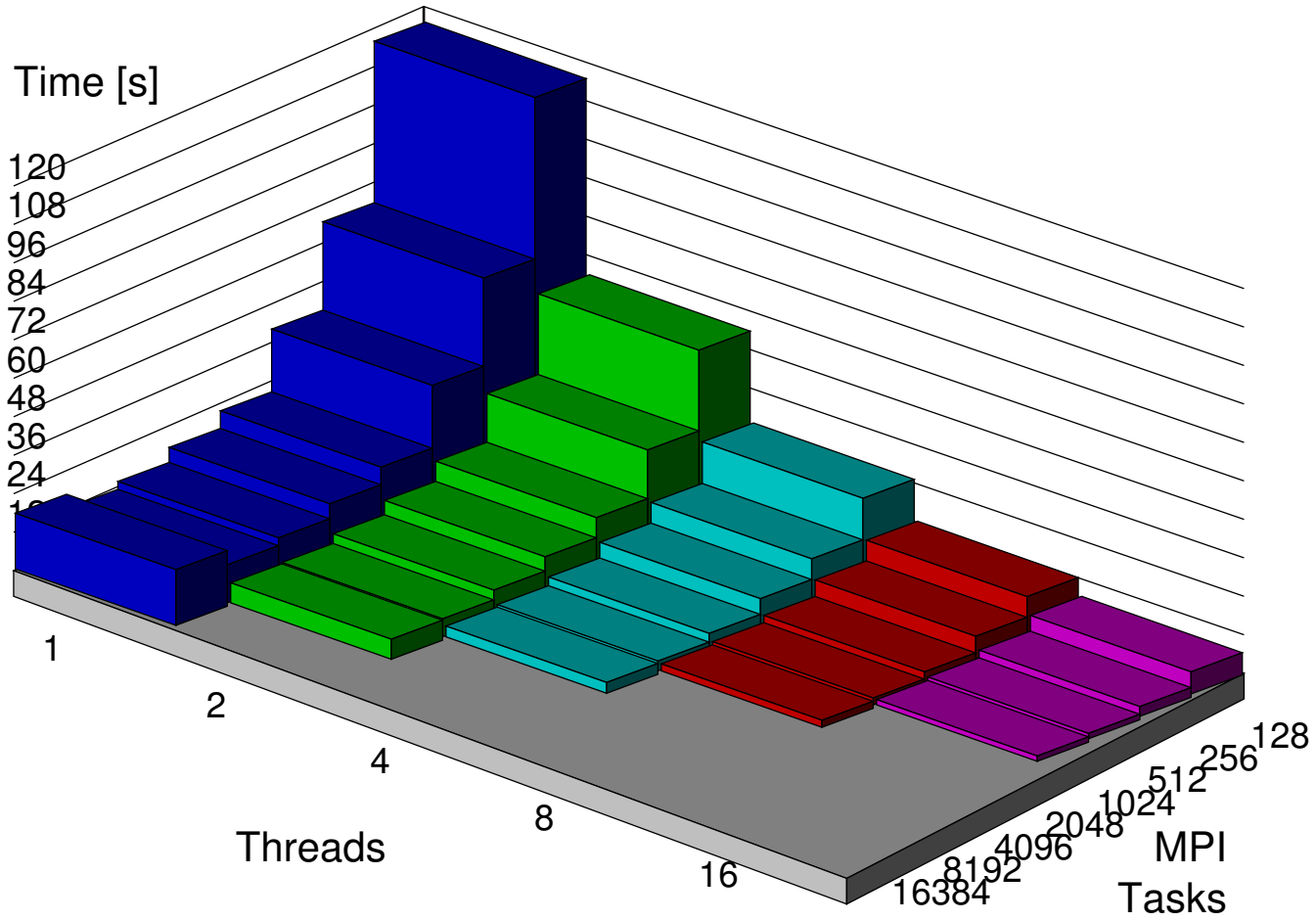
Timings on p690 (-qsmp=auto)



Conditions to make this happen

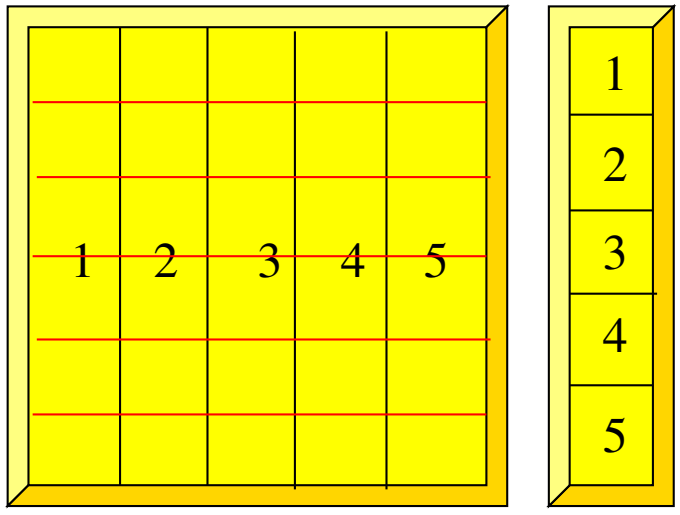
- Incomplete compiler options -
  - Use MPI wrapper for the compiler.
  - Forget to set the OpenMP option.
- OMP_NUM_THREADS not set. Possible defaults -
  - OMP_NUM_THREADS=1
  - OMP_NUM_THREADS="all node"
- OMP_NUM_THREADS only limits number of threads.

BG/Q Timings (-qsmp=auto)



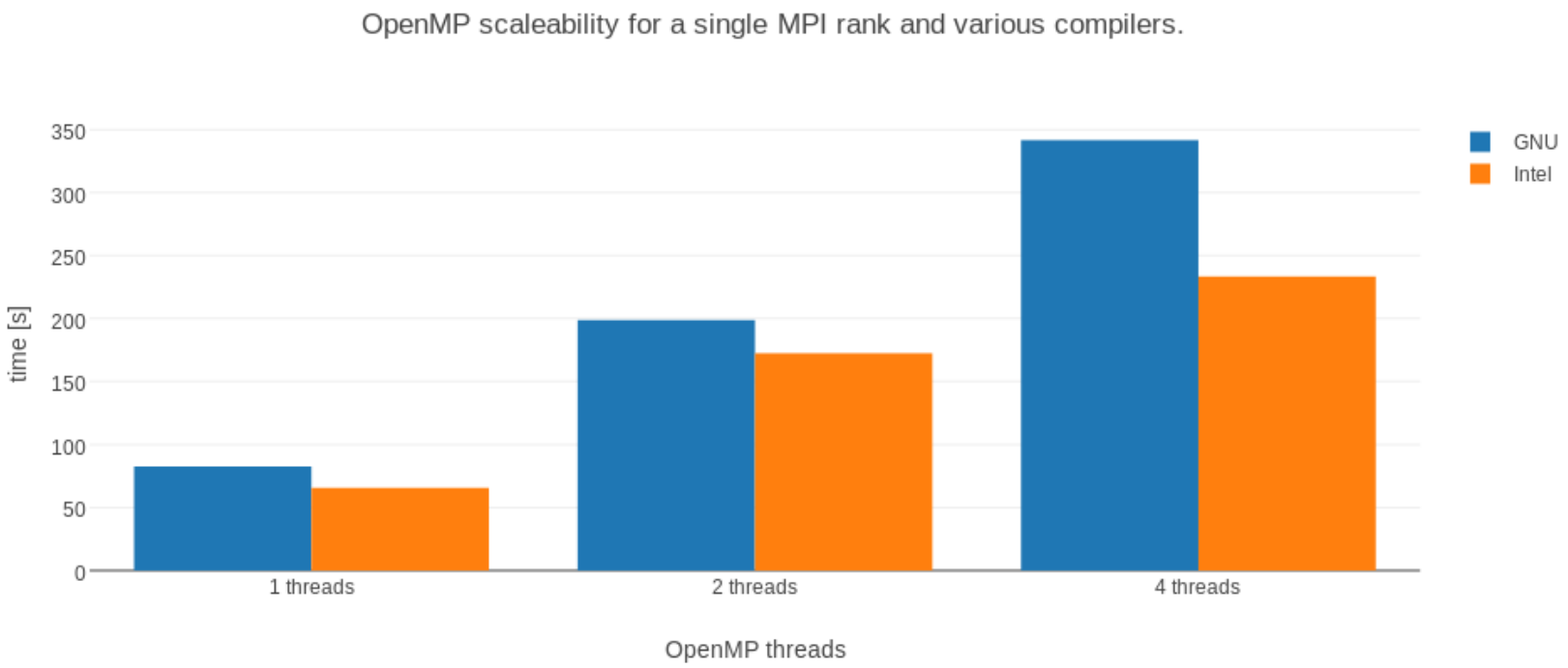
Choose OpenMP Work Distribution

```
Do j = 1, n_loc
$OMP do parallel reduction(+:c)
   Do i = 1,n
     c(i) = c(i) + a(i,j) * b(j)
   end Do
$OMP end do parallel
end Do
call mpi_reduce_scatter
```



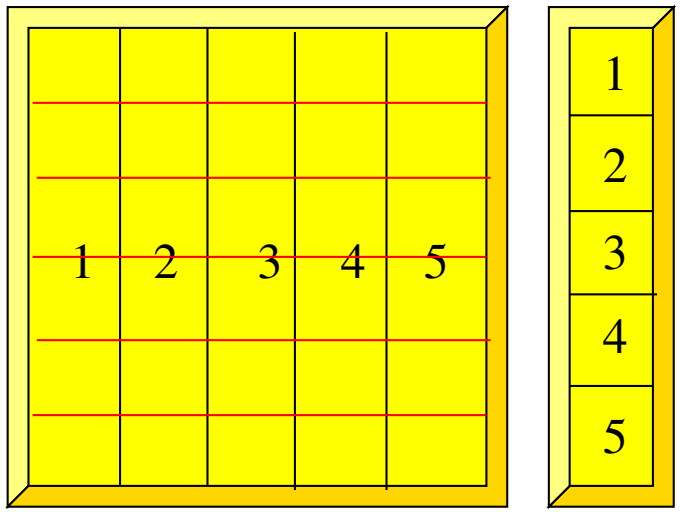
Overhead !!!

Skylake 8168 Timings

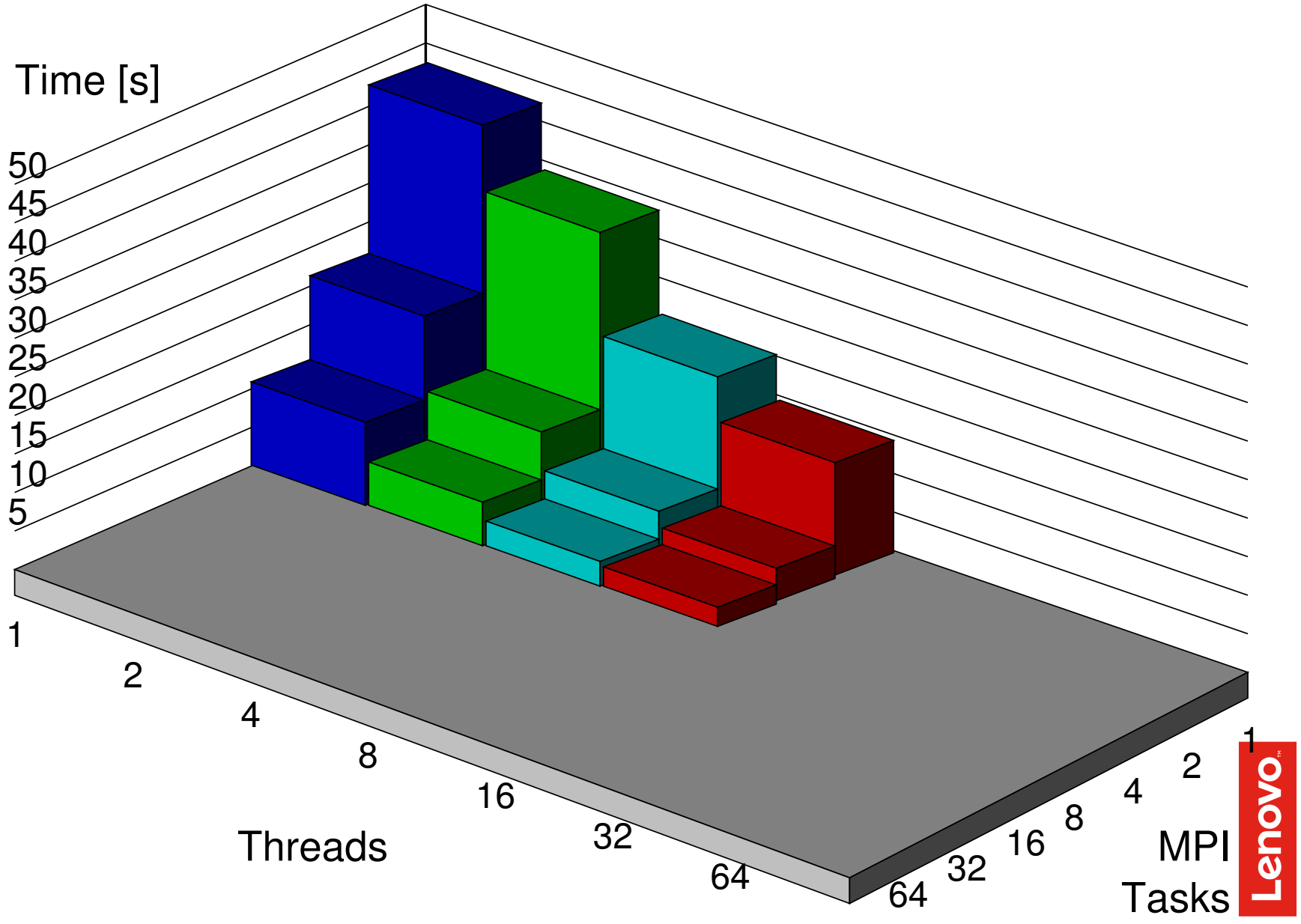


Do we need the Reduction Clause ?

```
$OMP parallel shared(c)
 c = 0
 Do j = 1,n_loc
$OMP do parallel
    Do i = 1,n
      c(i) = c(i) + a(i,j) * b(j)
    end Do
$OMP end do
 end Do
$OMP end parallel
 call mpi_reduce_scatter
```



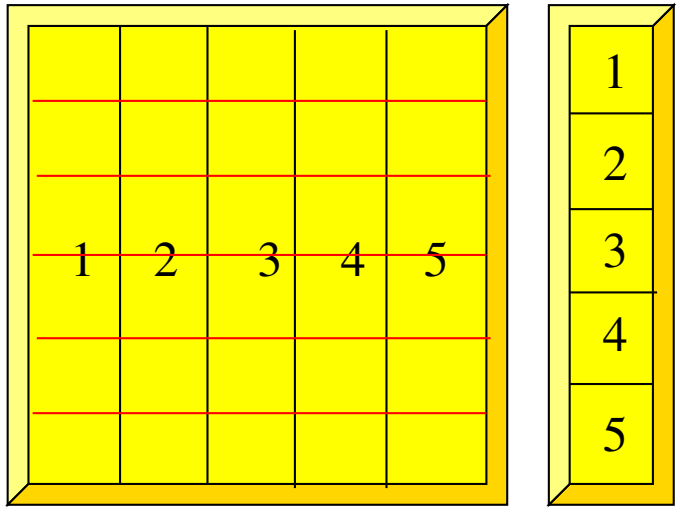
Skylake 8168 Timings for Intel Compiler



Enlarging parallel Region

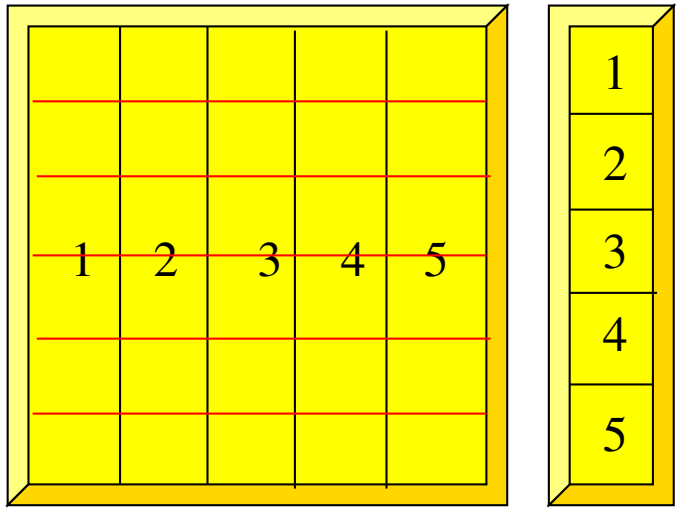
```
$OMP parallel reduction(+:c)
c = 0
Do j = 1,n_loc
$OMP do parallel
   Do i = 1,n
     c(i) = c(i) + a(i,j) * b(j)
   end Do
$OMP end do
end Do
$OMP end parallel
call mpi_reduce_scatter
```

Wrong results !!!



The Correct OpenMP Program

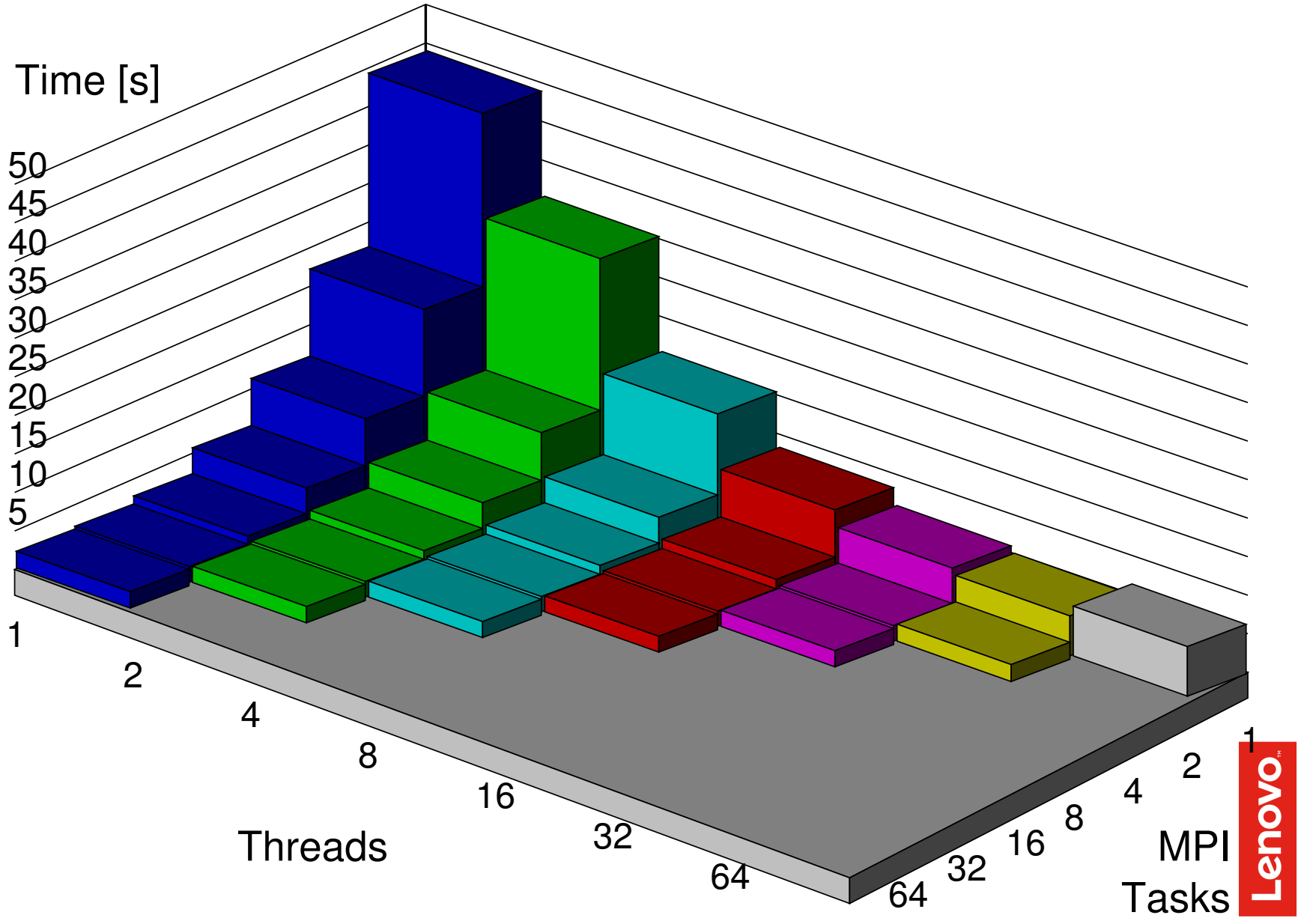
```
$OMP parallel reduction(+:c)
c = 0

c = 0
$OMP parallel reduction(+:c)
Do j = 1,n_loc
$OMP do parallel
   Do i = 1,n
     c(i) = c(i) + a(i,j) * b(j)
   end Do
$OMP end do
end Do
$OMP end parallel
call mpi_reduce_scatter
```



Initialization Move Explained

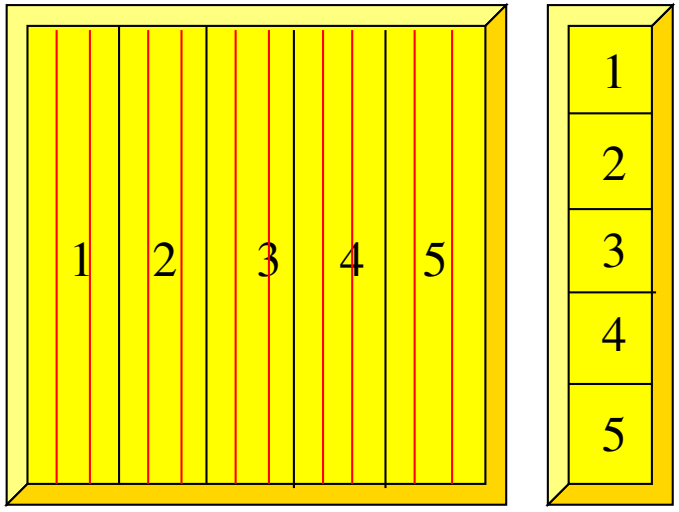
- Inside the parallel region the shared variable c is hidden behind private c's
- All private c's are initialized with 0
- All private c's are added to the shared c at the end of the parallel region
- Shared c needs initialization

Skylake 8168 Timings for GNU Compiler



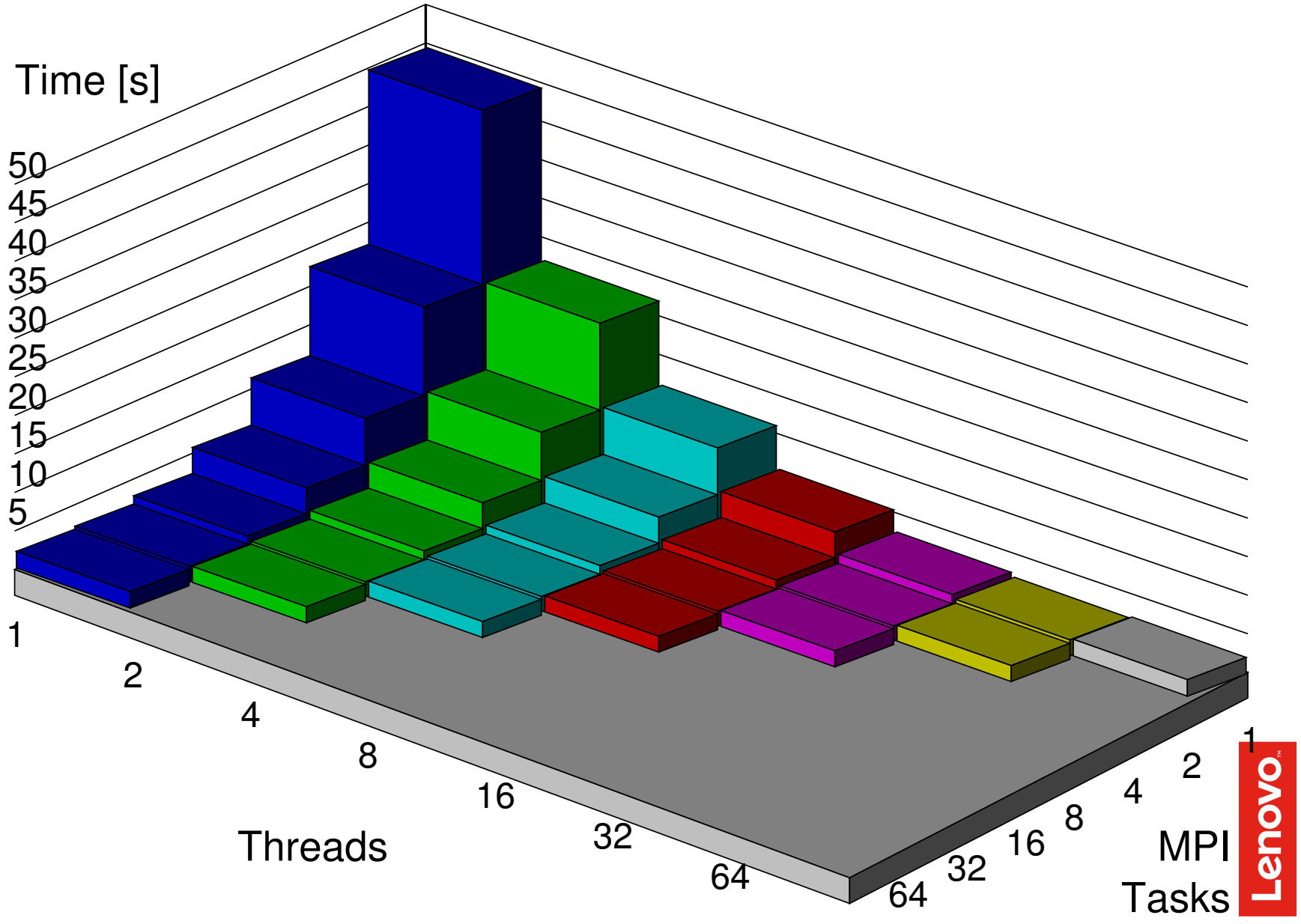
Redistribute OpenMP Work

```
$OMP do parallel
$OMP reduction(+:c)
 Do j = 1,n_loc
    Do i = 1,n
      c(i) = c(i) + a(i,j) * b(j)
    end Do
 end Do
$OMP end do parallel
 call mpi_reduce_scatter
```

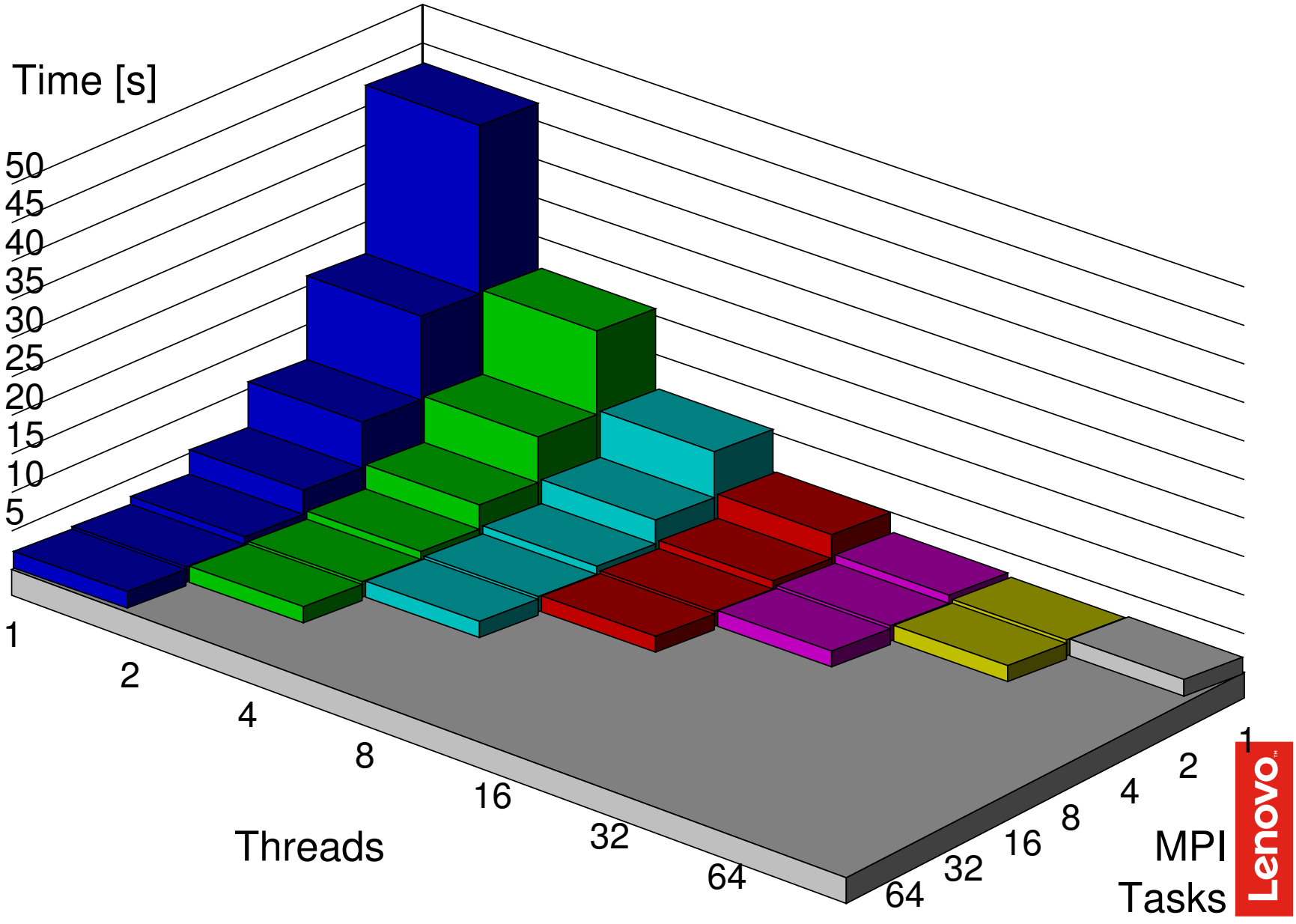


Choose the same loop for OpenMP and MPI

Skylake 8168 Timings for GNU Compiler



Skylake 8168 Timings for Intel Compiler



Conditions for Mixed Mode Programming

- Hybrid mode includes an MPI program
- SMP on each node has to pay off
  - Easy and effective SMP parallelization
  - MPI communication overhead explodes
  - Escape from Adapter bottle necks
  - Local dynamical load balancing

Hybrid mode does not invalidate or escape from Amdahls Law !