HP-SEE

Programming with OpenMP

www.hp-see.eu

HP-SEE

High-Performance Computing Infrastructure
for South East Europe's Research Communities

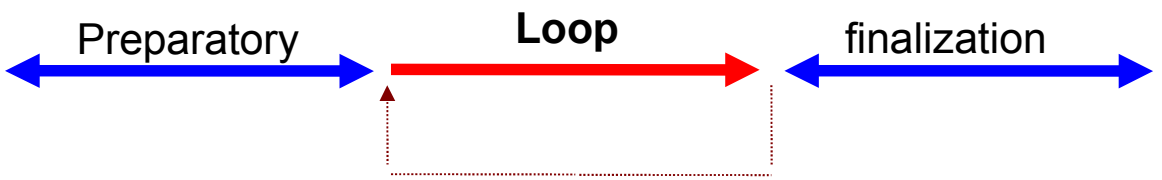
N. Frasheri, B. Cico, D. Dardeli

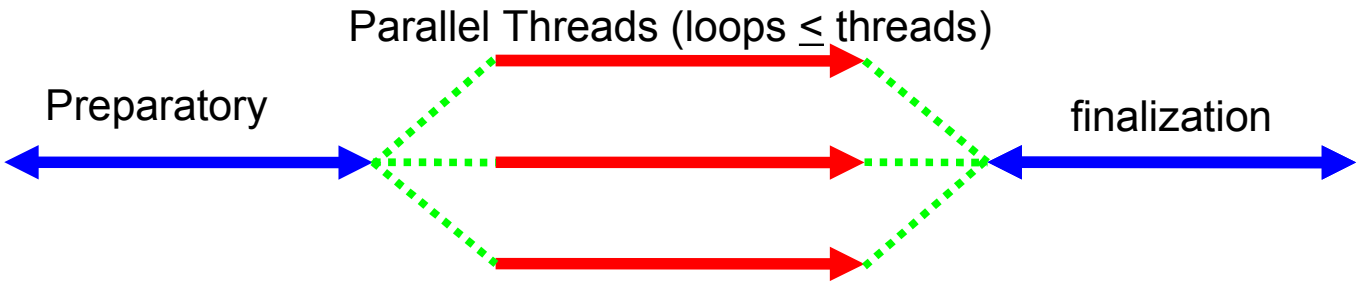
The HP-SEE initiative is co-funded by the European Commission under the FP7 Research Infrastructures contract no. 261499

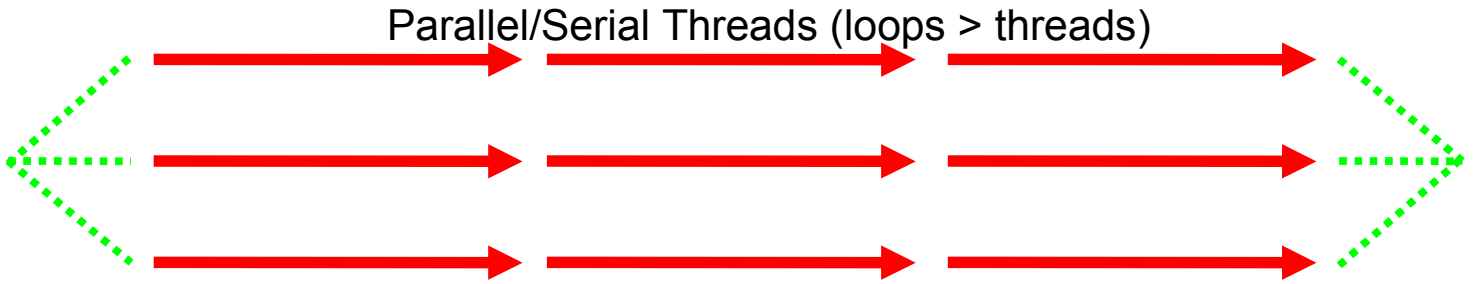
Principle of OpenMP

HP-SEE
High-Performance Computing Infrastructure for South East Europe's Research Communities

Working with THREADS







Test OpenMP Source Code

HP-SEE
High-Performance Computing Infrastructure for South East Europe's Research Communities

```
#include <omp.h>
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
int main (int argc, char *argv[])
{      typedef struct
       { long int thred;
         double start; double stop;
       } metad; metad   * array;
       long int i,j,k, Nthread, Niter, Nruntime;
       double time_start, time_stop, seconds,
            sinsin;
       time_start=omp_get_wtime();
       Nthread=1; Niter=16;
            Nruntime=9999999;
       if (argc>1) Nthread=atoi(argv[1]);
       if (argc>2) Niter  =atoi(argv[2]);
       array = malloc(Niter*sizeof(metad));
```

```
#pragma omp parallel for \
      num_threads(Nthread) private(j,k,sinsin)
for (i=0; i<Niter; i++)
{    array[i].start = omp_get_wtime();
     for (j=0; j<16/Niter; j++)
          for (k=0; k<Nruntime  ; k++)
               { sinsin=sin(k); }
     array[i].thred= omp_get_thread_num();
     array[i].stop = omp_get_wtime();
} // end pragma
time_stop=omp_get_wtime();
seconds=time_stop - time_start;
printf("walltime %f \n", seconds);
for (i=0; i<Niter; i++)
     printf("%ld Thread %ld Start %f Stop %f \n",
     i,array[i].thred,array[i].start, array[i].stop);
return 0;
}
```

Comments on the Source

HP-SEE
High-Performance Computing Infrastructure for South East Europe's Research Communities

OpenMP based on shared memory by all threads

- Risk of two threads touching the same memory word
- Leading to eventual errors or runtime delays

Number of threads declared in *pragma* directive

Execution of loop's code must be independent

- Declare as ***private*** all in-loop variables
- Avoid dependencies between iterations

Preparation of Test

HP-SEE
High-Performance Computing Infrastructure for South East Europe's Research Communities

Compilation

```
gcc -fopenmp test.c /usr/lib/libm.a -o test
```

Script

```
run.sh = (/usr/bin/time $1 $2 $3)
```

Execution

```
./run.sh test <nr-threads> <nr-iter>
```

Tested hardware

Dell Inspiron ~ 1 processor 2 cores

OpenMP Test Example

HP-SEE

High-Performance Computing Infrastructure for South East Europe's Research Communities

```
$ ./run.sh test2 4 8
                walltime 6.503488
0 Thread 0 Start 2171.904544 Stop 2175.616247 Run 3.711703
1 Thread 0 Start 2175.616249 Stop 2178.385400 Run 2.769152
2 Thread 1 Start 2171.893791 Stop 2175.608792 Run 3.715001
3 Thread 1 Start 2175.608793 Stop 2178.396882 Run 2.788088
4 Thread 2 Start 2171.901047 Stop 2175.698175 Run 3.797128
5 Thread 2 Start 2175.698176 Stop 2177.961336 Run 2.263160
6 Thread 3 Start 2171.893766 Stop 2173.640498 Run 1.746732
7 Thread 3 Start 2173.640500 Stop 2176.935941 Run 3.295441
```

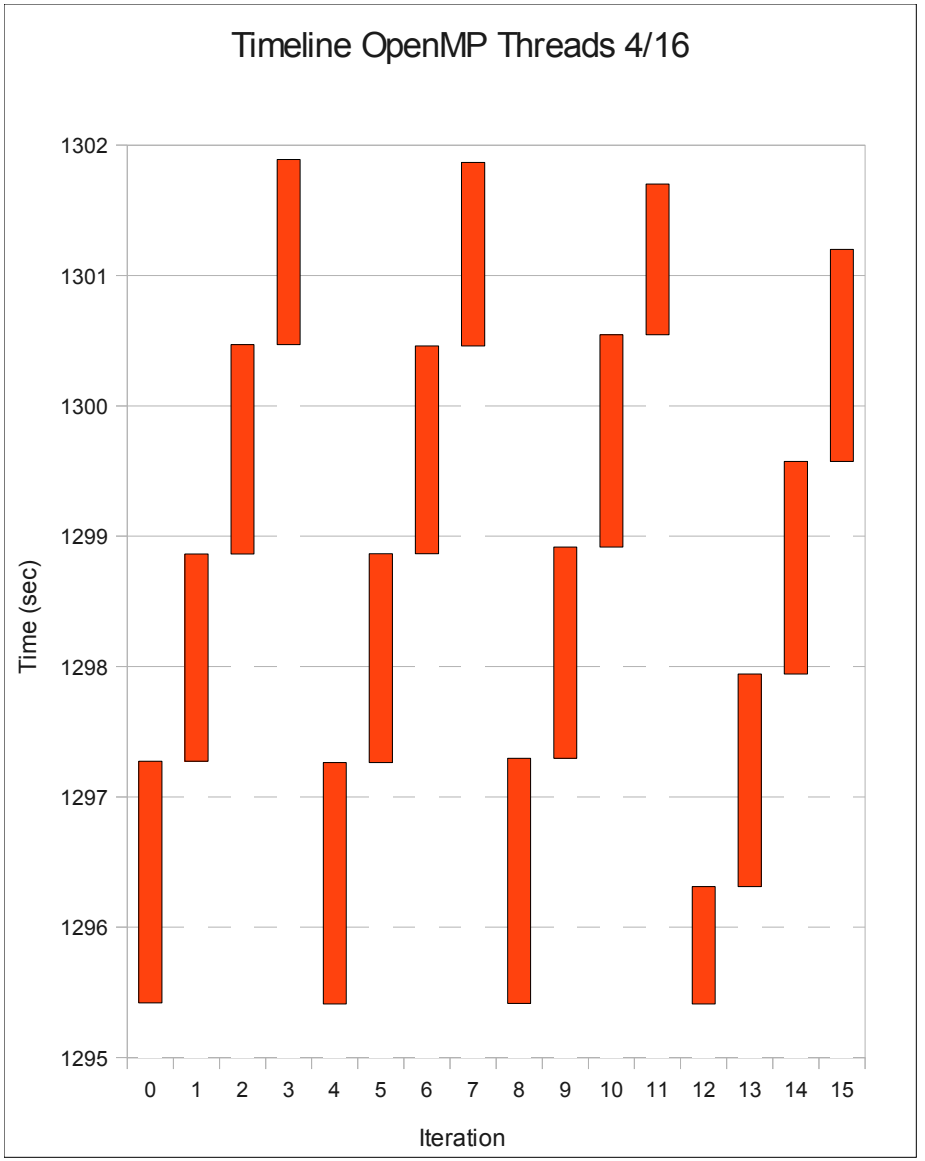
time_stop - time_start

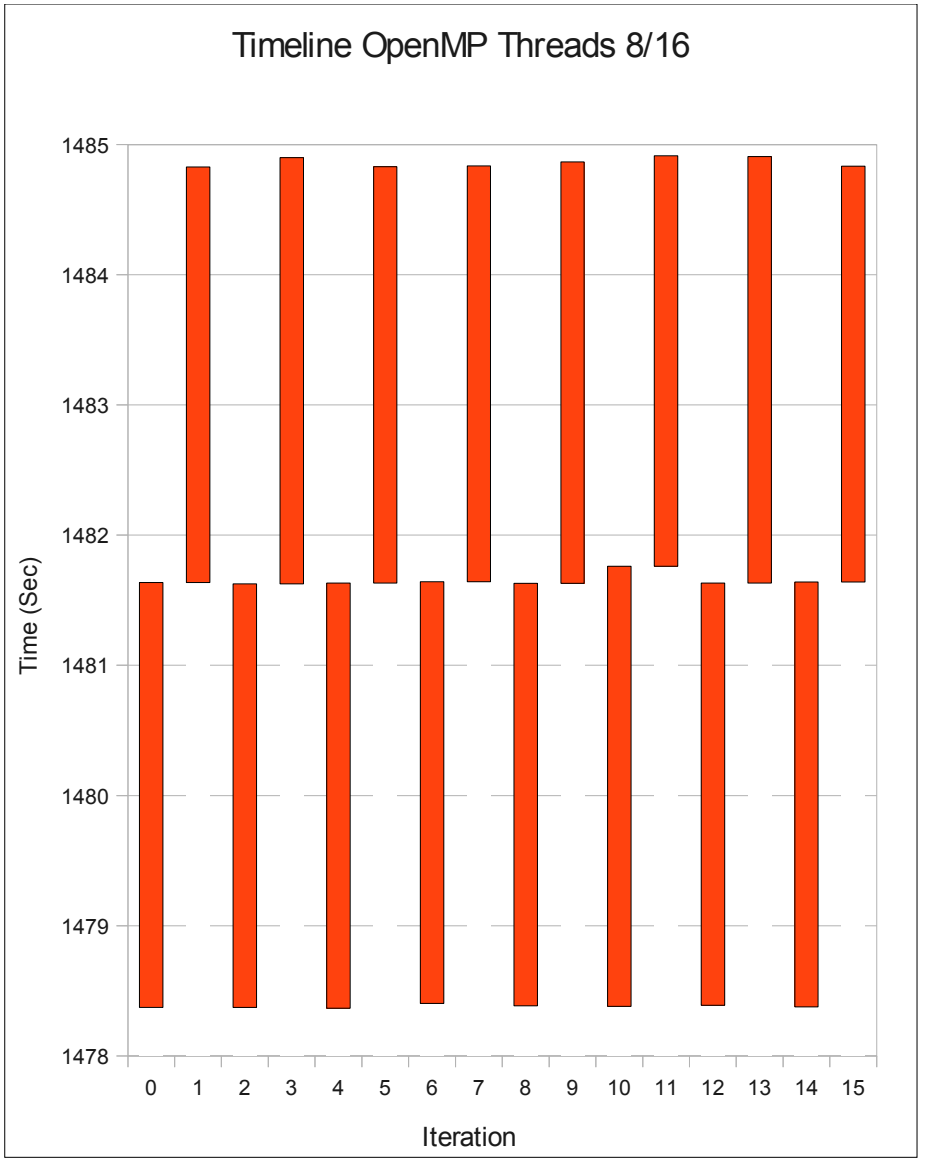
ITER

THREAD

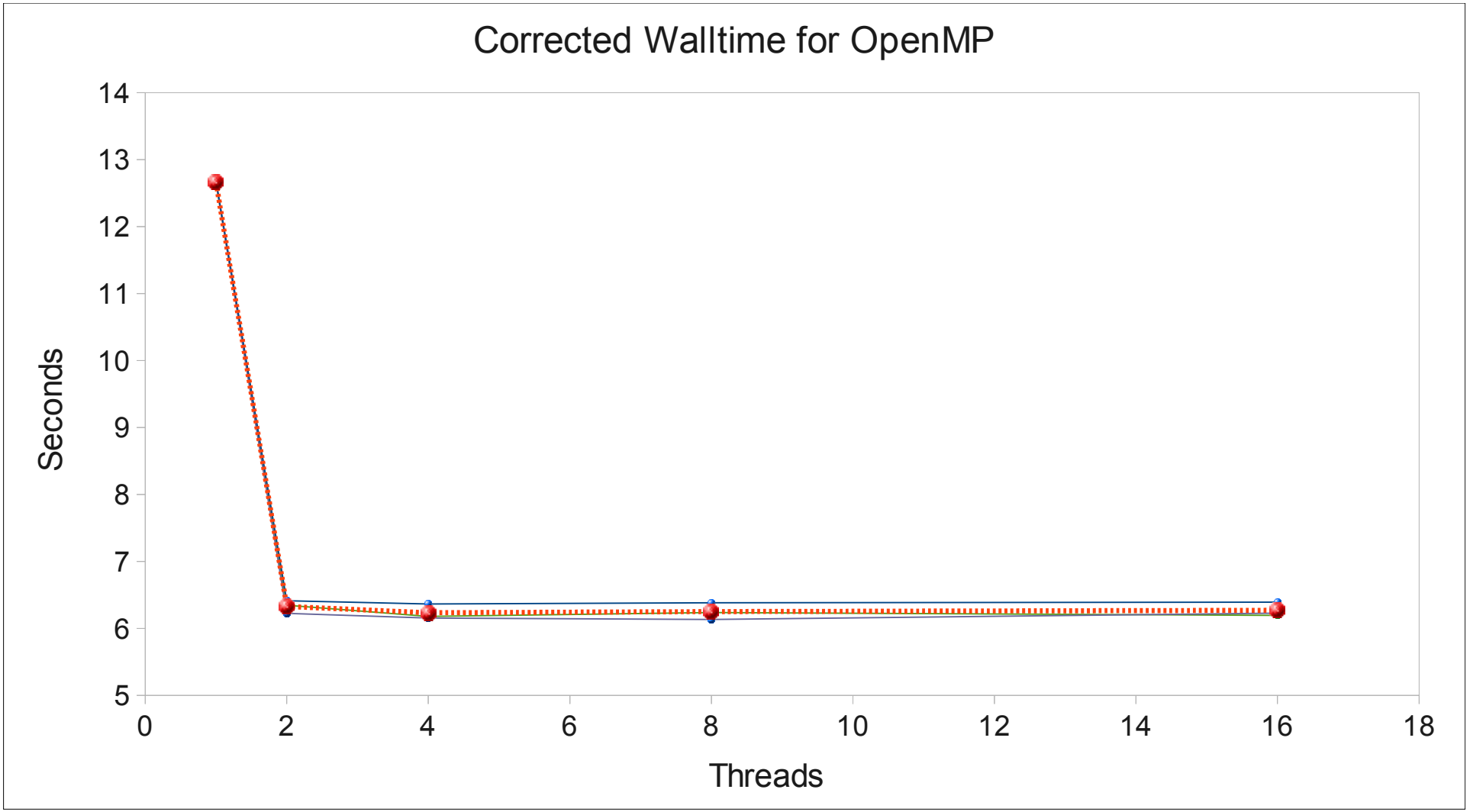
```
12.75 user 0.01 system 0:06.50 elapsed 196% CPU
   (0avgtext+0avgdata 2688maxresident)k
0inputs+0outputs (0major+218minor)pagefaults 0swaps
```

Timesharing OpenMP Threads

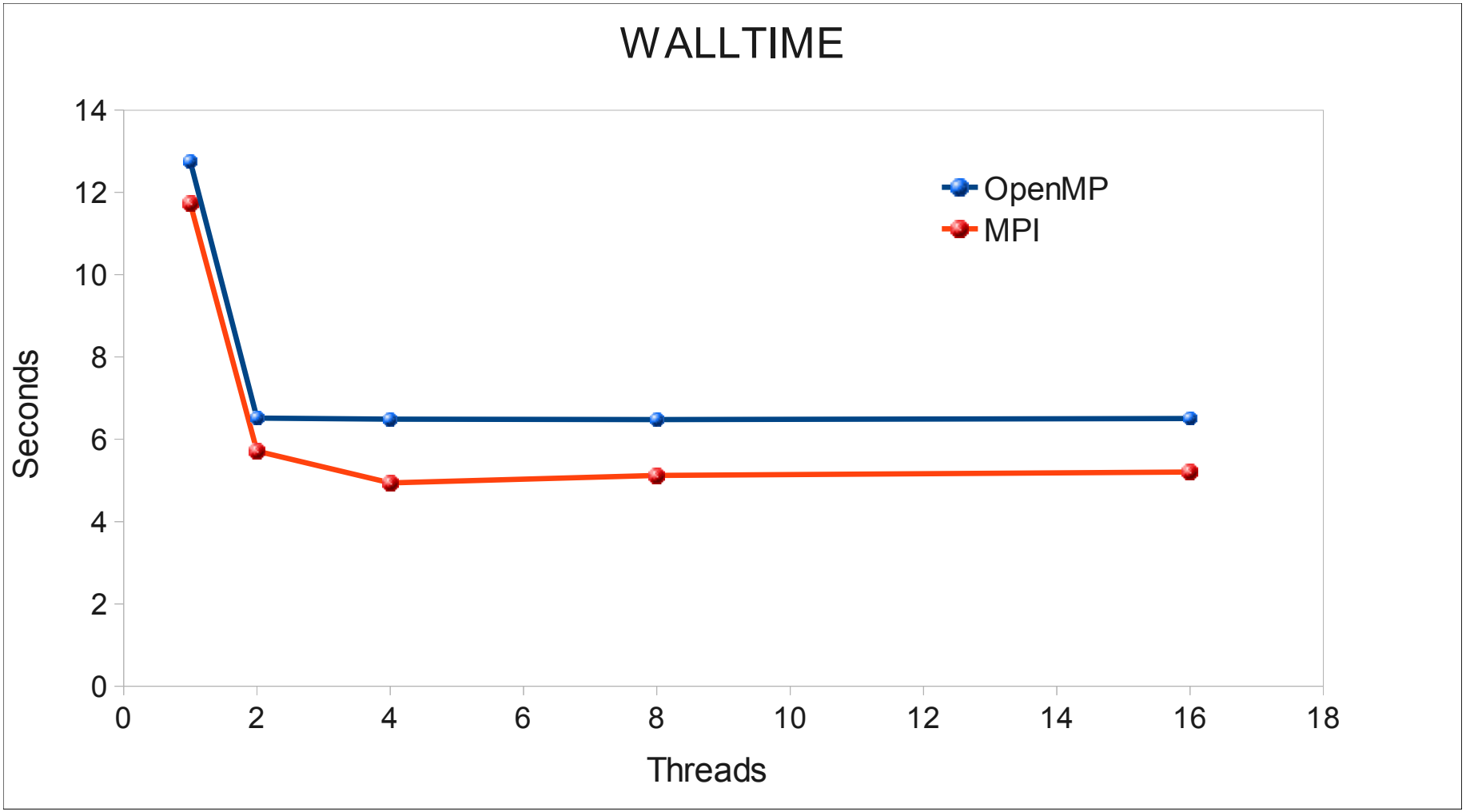




Test OpenMP Walltime

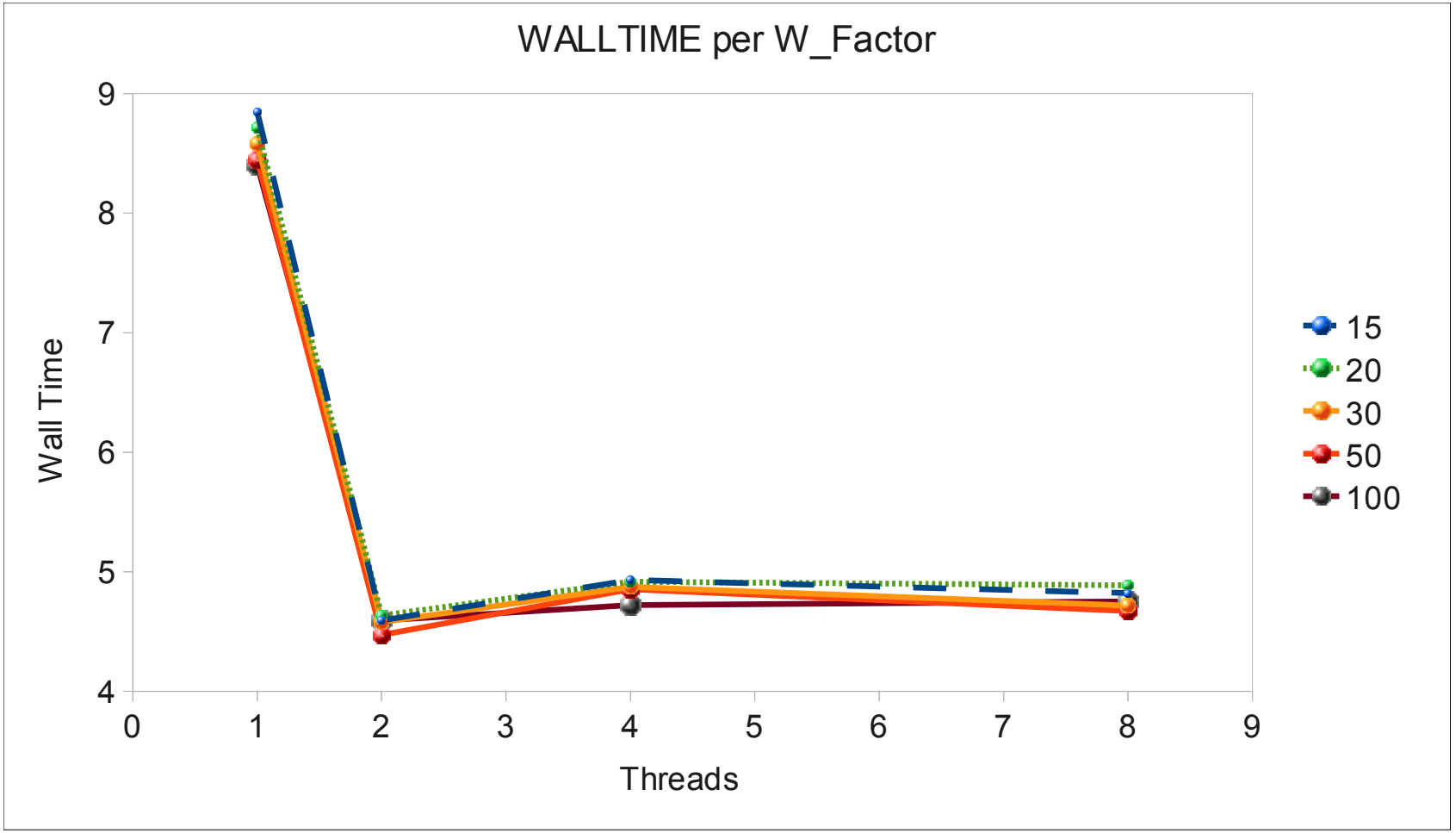


When Compared with MPI



Another Real Example

HP-SEE
High-Performance Computing Infrastructure for South East Europe's Research Communities



Better Performance ?

HP-SEE

High-Performance Computing Infrastructure for South East Europe's Research Communities

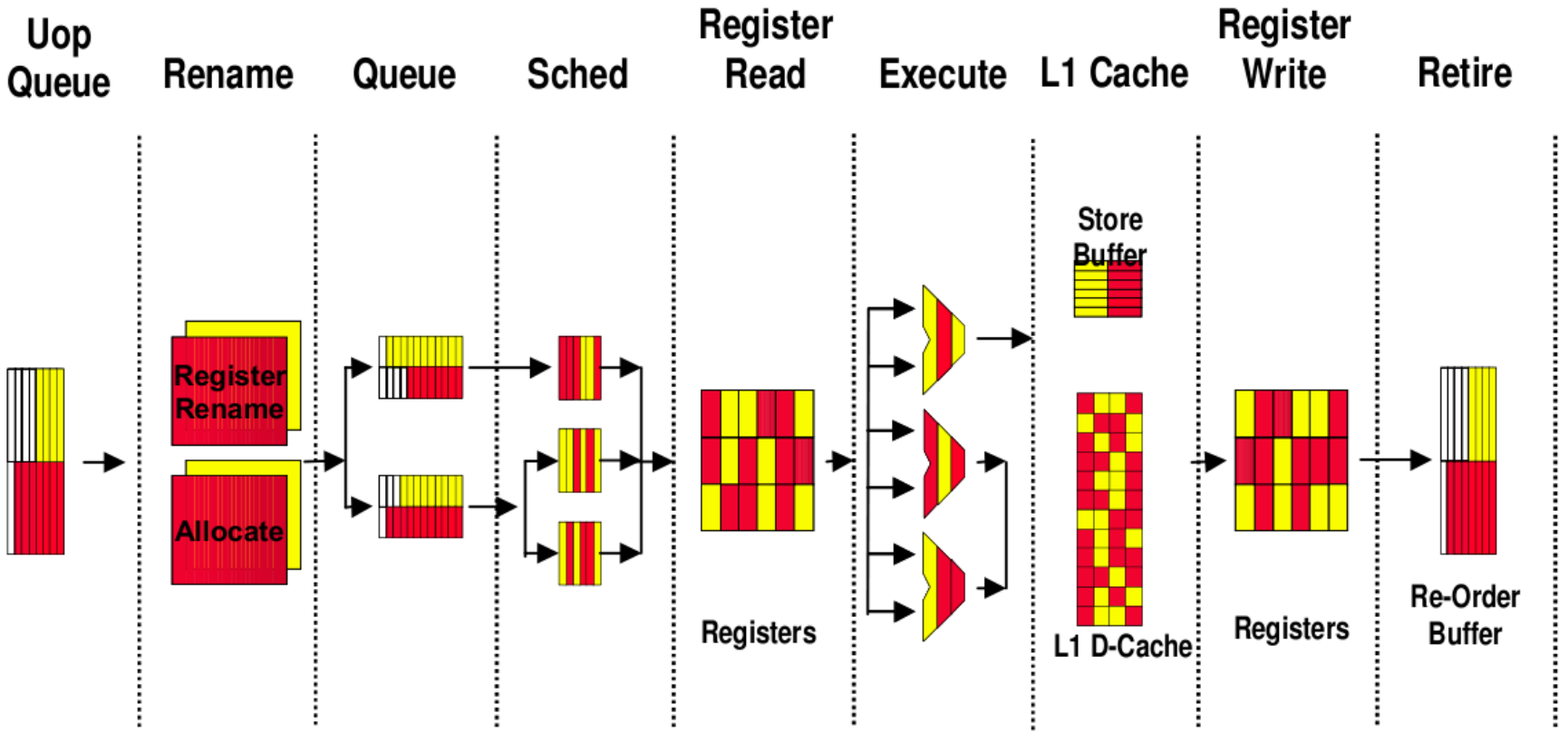
Performance ~ number of cores

Possible optimization ways:

- Threads share the same central memory
- Less overhead with swapping and cache
- Need for better use of hardware architecture
  - Pipeline
  - Hyper-thread

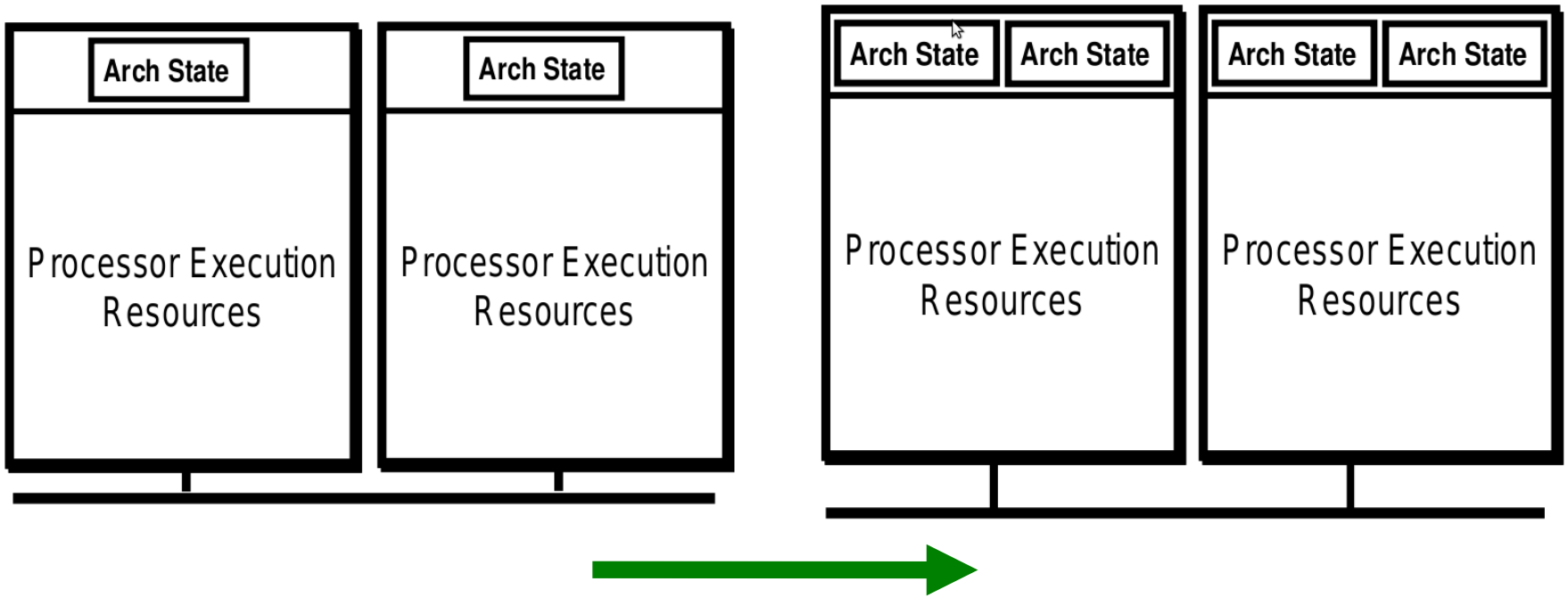
See the Intel Compiler of C Language

Intel Pipeline



Intel Hyperthreads

Deborah T. Marr et al. Hyper-Threading Technology Architecture and Microarchitecture. Intel Technology Journal Volume 06 Issue 01, Feb 14, 2002



Conclusions on OpenMP

HP-SEE
High-Performance Computing Infrastructure for South East Europe's Research Communities

Easy to use in today's multicore computers

- Download & install OpenMP package
  - **`http://openmp.org/`**
- Do minimal modifications in source loops
- Include directive PRAGMA before loops
- Compile adding "fopenmp"
- Run …