HP-SEE

Programming with MPI

www.hp-see.eu

HP-SEE

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

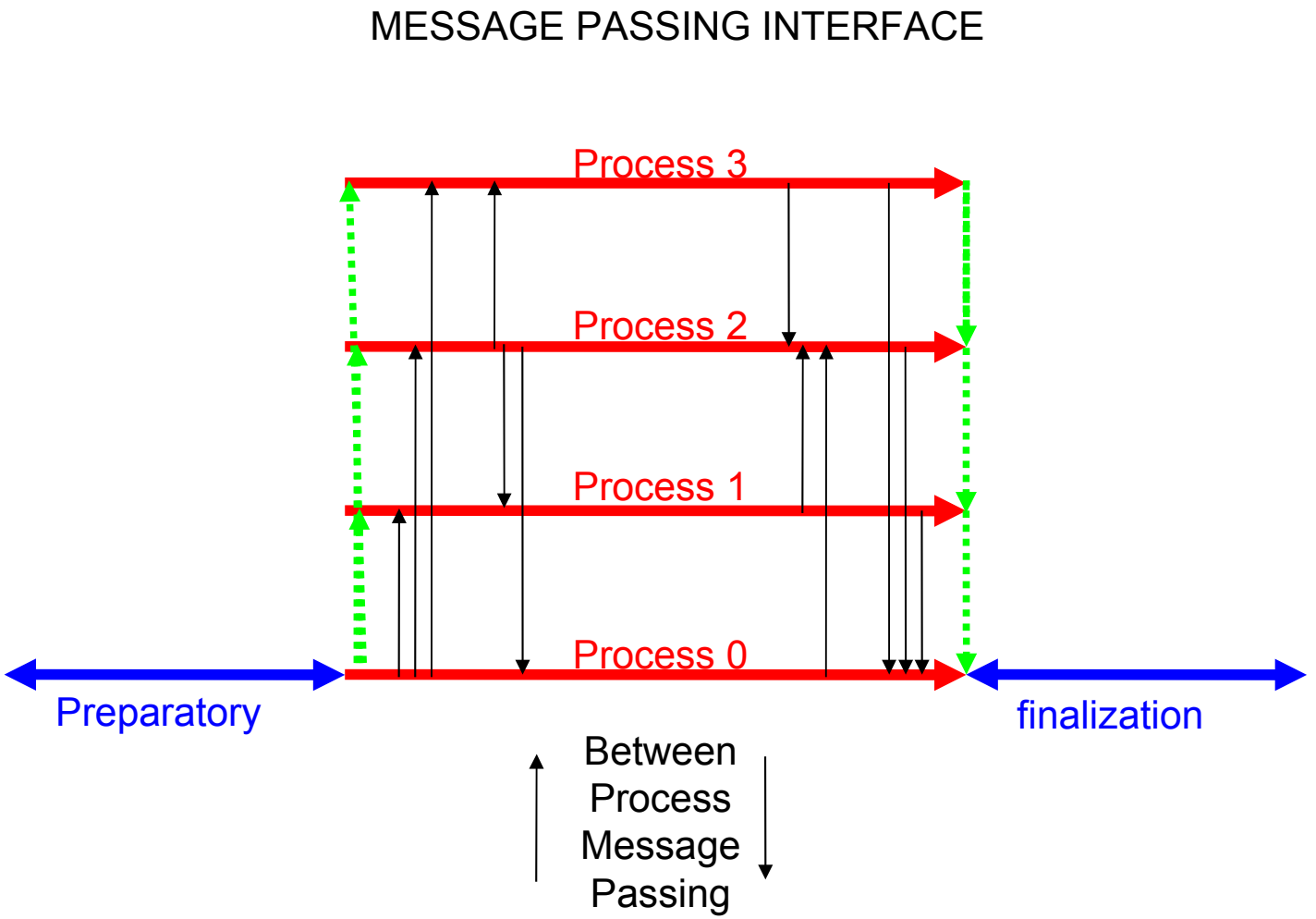
N. Frasheri, B. Cico, A. Shehu

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

Principle of MPI

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

Working with PROCESSES



Test MPI Source Code

HP-SEE

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

```
#include <mpi.h>
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
typedef struct
{ double thred, start, stop;
} metad;  metad   * array, procs;
int main(int argc, char **argv)
{   int rank, size, i, j, k;
    double sinsin, startime, stoptime;
    long int li, Nruntime; Nruntime=9999999;
  MPI_Init(&argc, &argv);  // init MPI
  MPI_Comm_rank(MPI_COMM_WORLD, &rank);
  MPI_Comm_size(MPI_COMM_WORLD, &size);
// print inputs & take time in mother process
  if (rank==0)
  {    startime = MPI_Wtime();
       array = malloc(size*sizeof(metad));
       for (i=0; i<size; i++)
       {    array[i].thred=0;
            array[i].start=0; array[i].stop =0;
  }   }
MPI_Scatter(&array[rank],3,MPI_DOUBLE,&procs,3,MPI
    _DOUBLE,0,MPI_COMM_WORLD);
// run processes inner loops
  procs.start = MPI_Wtime();
  for (j=0; j<16/size; j++)
       for (k=0; k<Nruntime; k++)
            { sinsin=sin(k); }
  procs.thred= rank;
  procs.stop = MPI_Wtime();
MPI_Gather(&procs,3,MPI_DOUBLE,array,3,MPI_DOUB
    LE,0,MPI_COMM_WORLD);
// print process runtime vector in mother process
  if(rank==0)
  {  for (i=0; i<size; i++)
            printf("Iter %d process %f Runtime %f \n",
            i, array[i].thred, array[i].stop - array[i].start);
     stoptime = MPI_Wtime();
     printf("time = %f \n",stoptime-startime);
  } MPI_Finalize(); // close MPI
  return 0; }
```

Comments on the Source

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

Working with processes

Each process manages its own central memory area

Loop on processes is missing ~ automatically via mpirun

Part of preparation & finalization in mother process

Procedures to distribute and collect data

- MPI_Scatter and MPI_Gather

  distribute & collect slots of data

- MPI_Bcast for single values

  distribute the same data with synchronization

- MPI_Send and MPI_Recv

  process-to-process etc.

Preparation of Test

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

Compilation

build.sh = mpicc test.c -o test

Script

run.sh = (/usr/bin/time mpirun -np $2 ./$1)

Execution

./run.sh test <no_process>

Tested in hardware

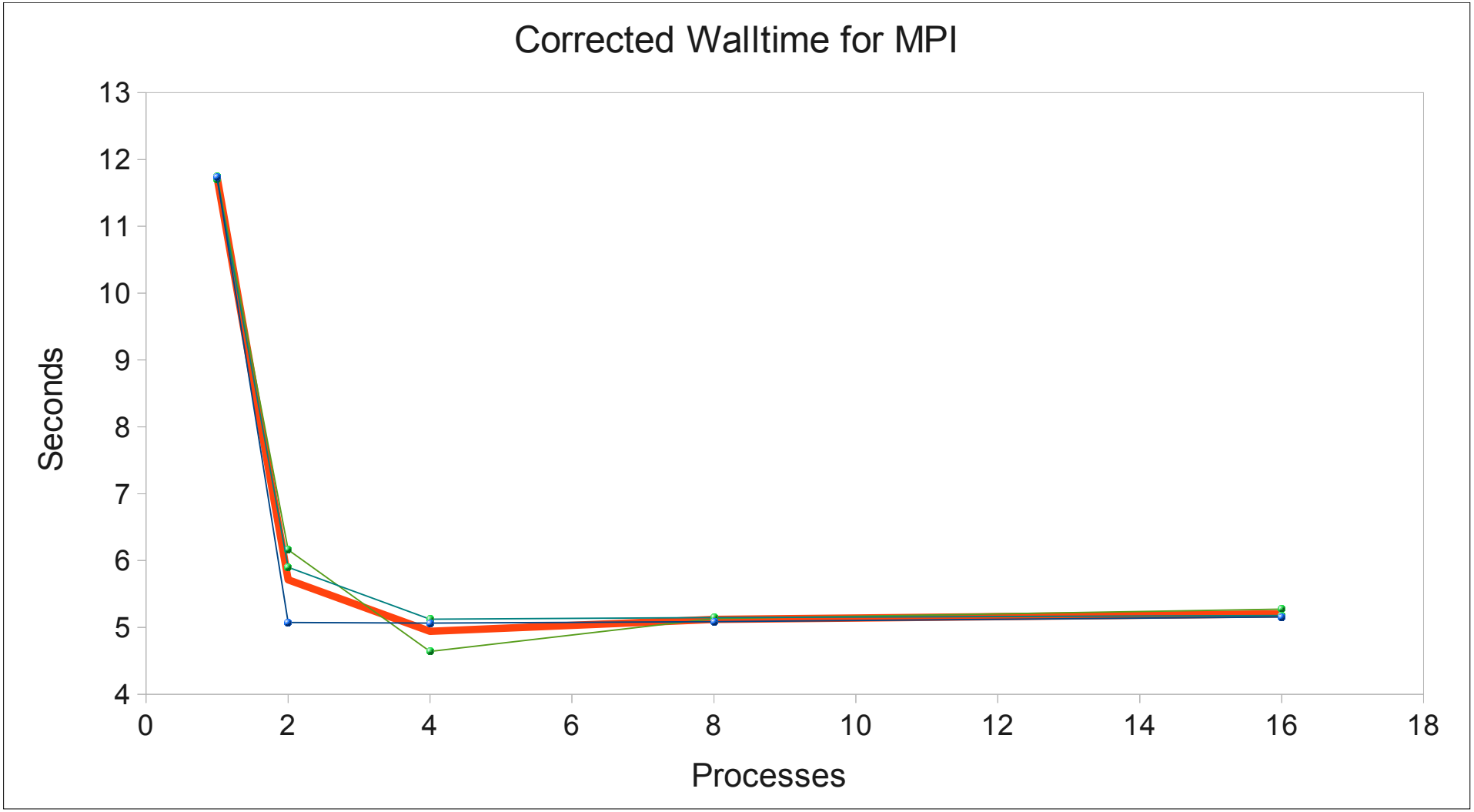
Dell Inspiron ~ 1 processor 2 cores

MPI Test Example

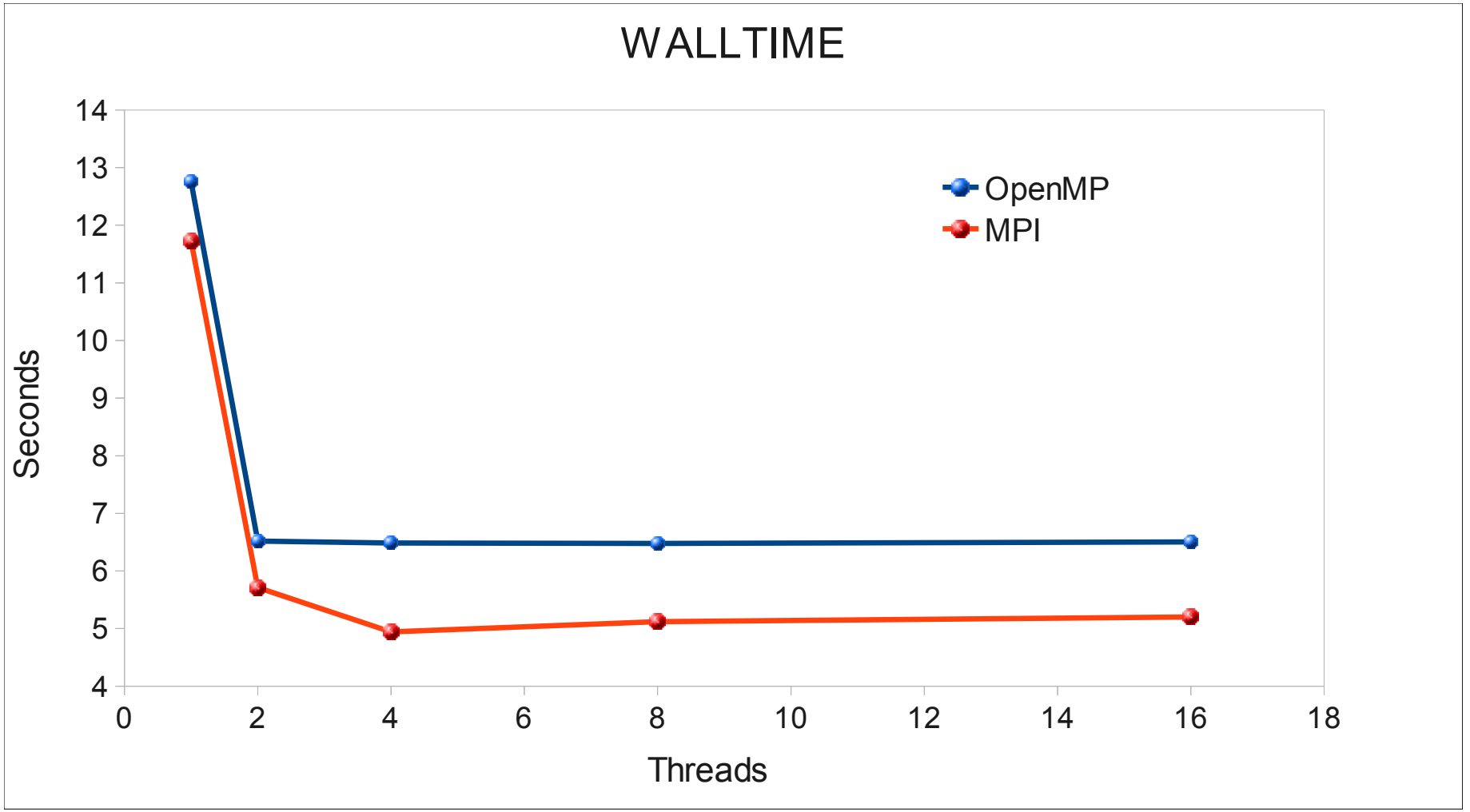
HP-SEE
High-Performance Computing Infrastructure for South East Europe's Research Communities

```
$ ./run.sh  test 4
input nopr=4
Iter 0 process 0 Runtime 3.508056
Iter 1 process 1 Runtime 3.685100
Iter 2 process 2 Runtime 3.748083
Iter 3 process 3 Runtime 3.768022
time = 3.768220
7.26 user 0.13 system 0:04.83 elapsed 152% CPU
    (0 avgtext + 0 avgdata 14448 maxresident)k
    103 inputs + 680 outputs (19 major + 5707 minor) pagefaults 0 swaps
```

Test MPI Walltime



Compared with OpenMP



Comments on Performance

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

The case of four processes in two cores

- Started simultaneously
- Shared the time of CPU (total of 152%)
- Similar runtime for all processes
- Performance ~ number of cores
- Compared with OpenMP
  - OpenMP requires shared memory for all cores
  - MPI may run in distributed hardware …

Conclusions on MPI

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

More difficult to be implemented

- Need for radical modifications in the code
  - replacement of loops
  - use of message passing routines
  - dealing with distributed data files

But … More easy to run

- Does not require shared memory architecture
- May run in distributed hardware (cluster / grid)

References

HP-SEE

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

http://www.open-mpi.org/

http://software.intel.com/

http://www.openchannelsoftware.com/