Winter term 2020/2021

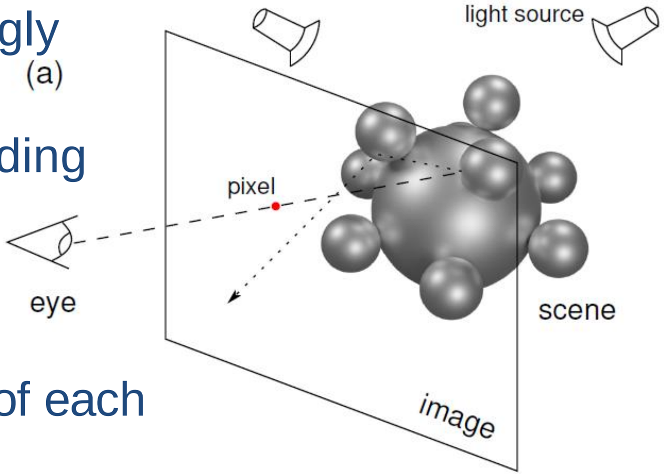
Parallel Programming with OpenMP and MPI

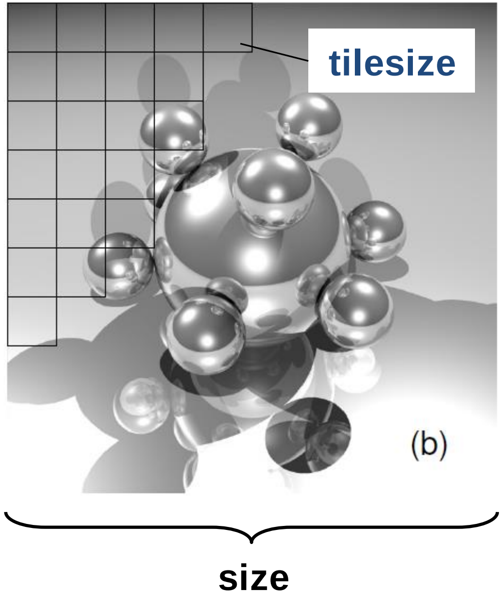
Dr. Georg Hager
Erlangen Regional Computing Center (RRZE) at Friedrich-Alexander-Universität Erlangen-Nürnberg
Institute of Physics, Universität Greifswald

Assignment 8 discussion

Assignment 8, Task 1 – MPI-parallel ray tracer

- Raytracing is “embarrassingly parallel”
- Each pixel is drawn by sending a “beam” through the scene and calculating its color value
- All pixels are independent of each other
- Picture is divided into tiles which are distributed dynamically among the MPI processes
- “Master-Worker” scheme: One process collects data and sends out new tile coordinates





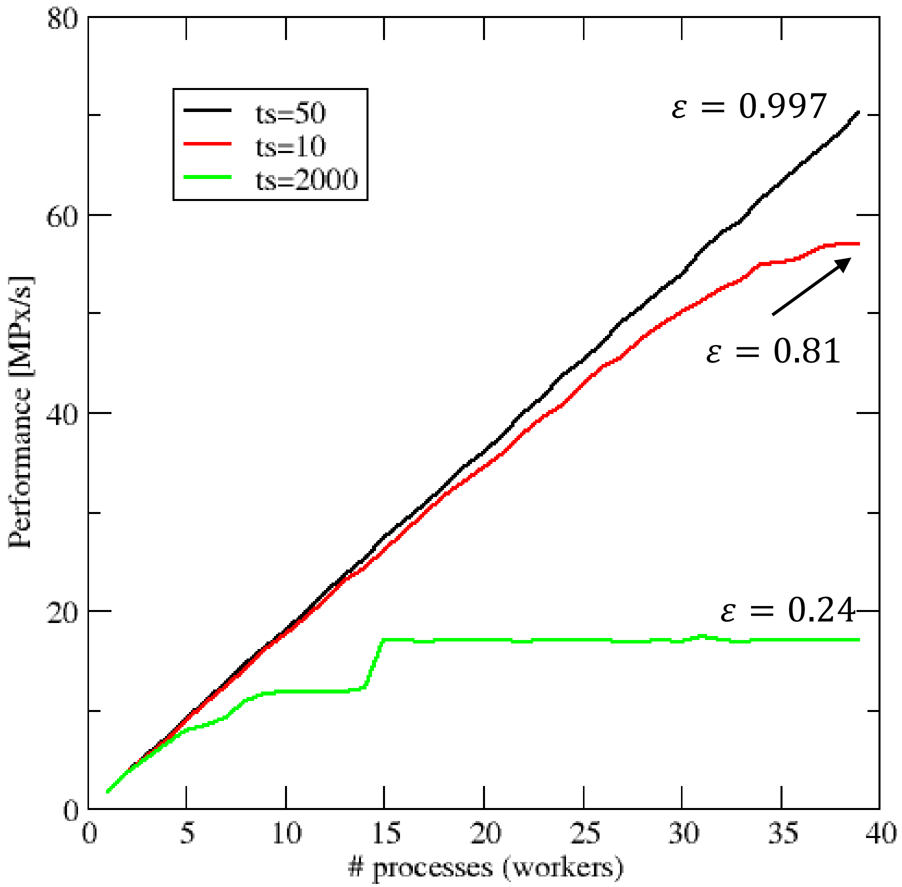
MPI ray tracer pseudocode

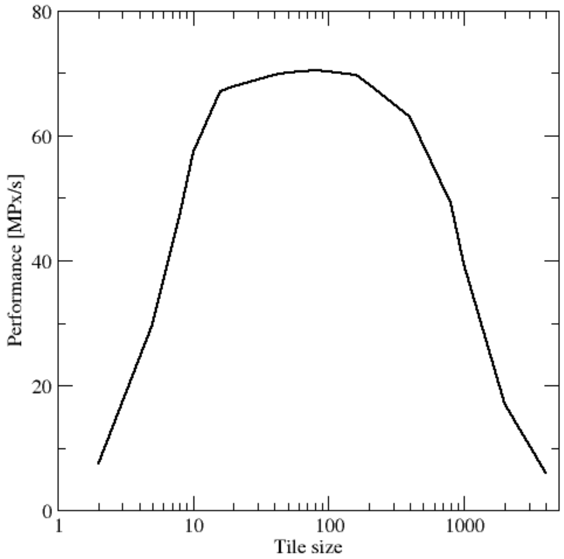
```
mpi_comm_rank(MPI_COMM_WORLD, &id);
if(id==0) {    // I am the master
  while(tiles_to_receive != 0) {
    … wait for anyone to send “ready” message …
    … store finished tile (if any) && tiles_to_receive-- …
    if(tiles_to_send != 0)
      … send new tile coordinates to worker …
      tiles_to_send--
    else
      … send “finish” message to worker …
  }
} else {       // I am a worker
  … send tile request to master …
  while(1) {
    … receive tile coordinates …
    if(finish_received) break
    calculate_tile()
    … send tile data to master …
  }
}
```

Master code

Worker code

Performance on Emmy, 8000x8000 px

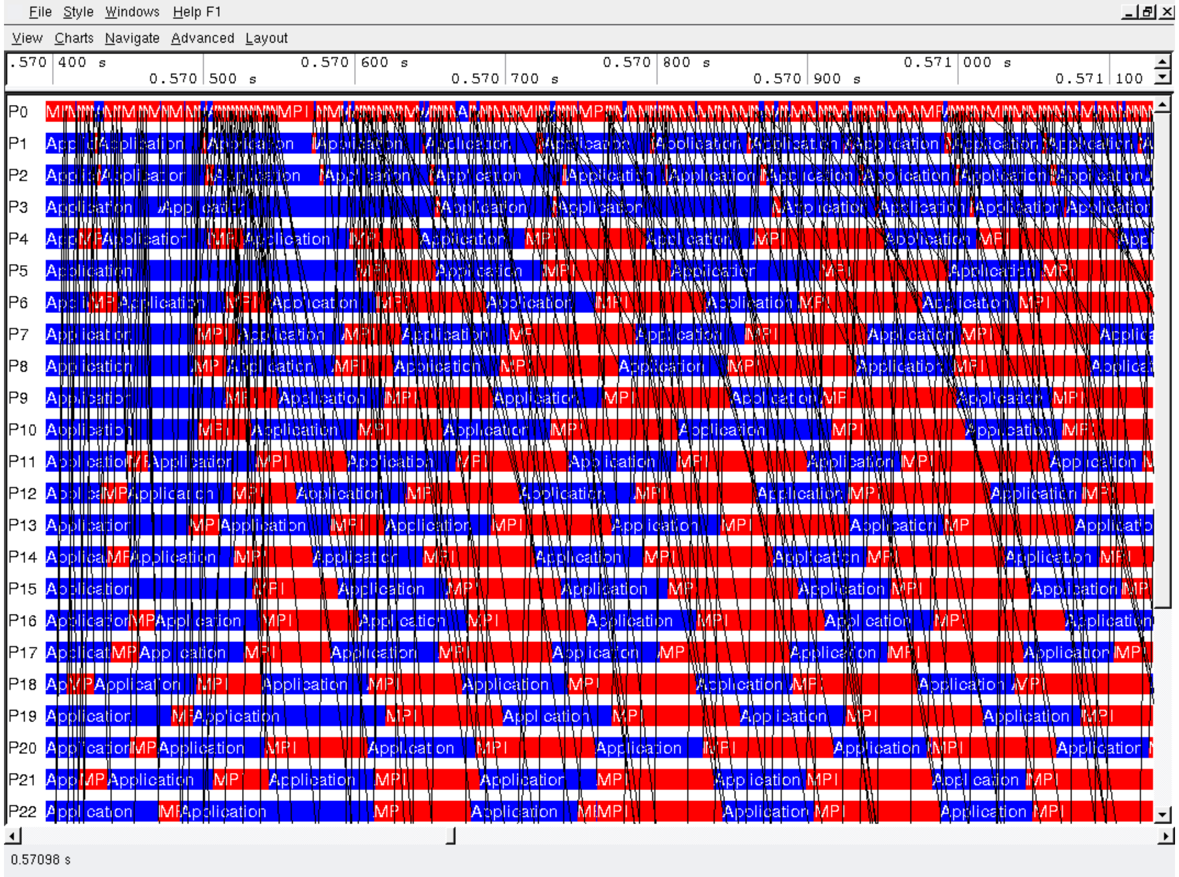




- Small tiles → communication overhead?
- Large tiles → load imbalance?

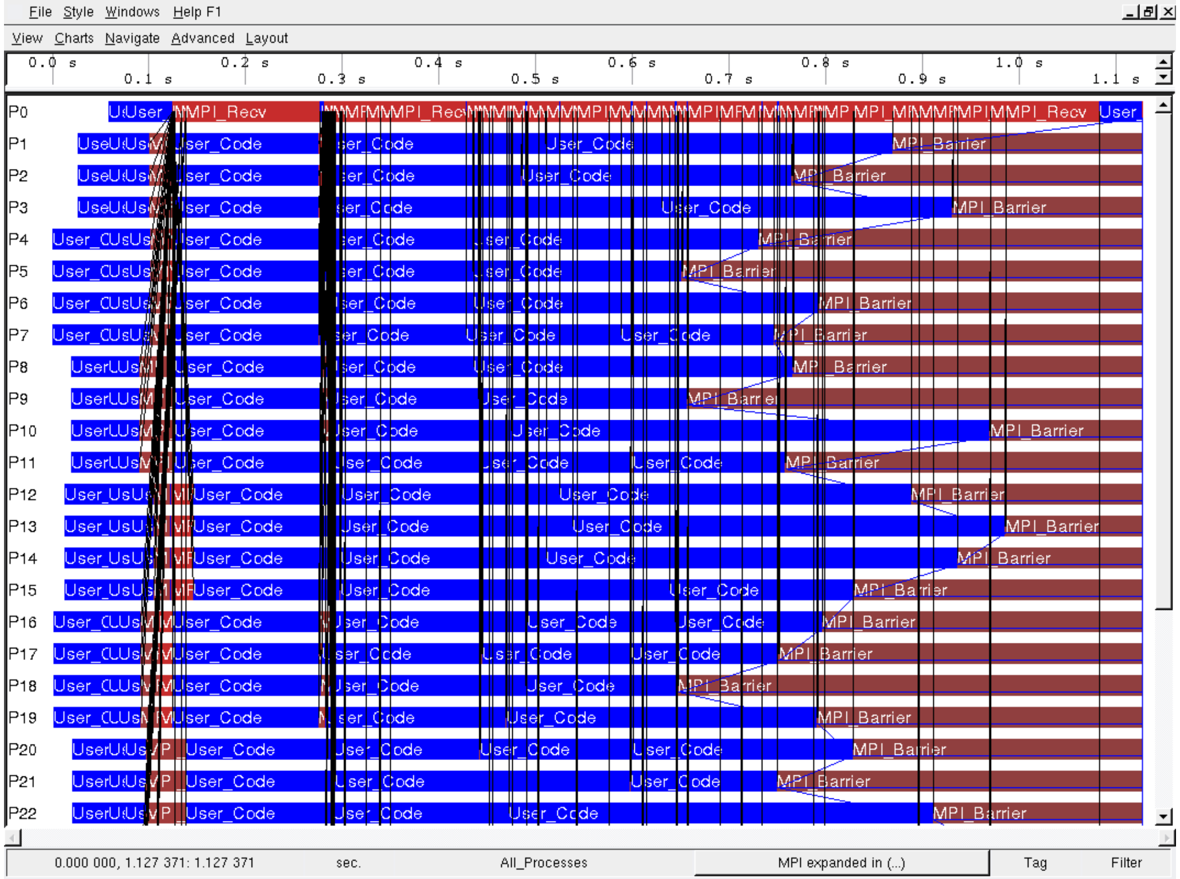
MPI event timeline (Intel Trace Analyzer)

- tilesize=10



MPI event timeline (Intel Trace Analyzer)

- tilesize=500



Average gray value (result: 196.93)

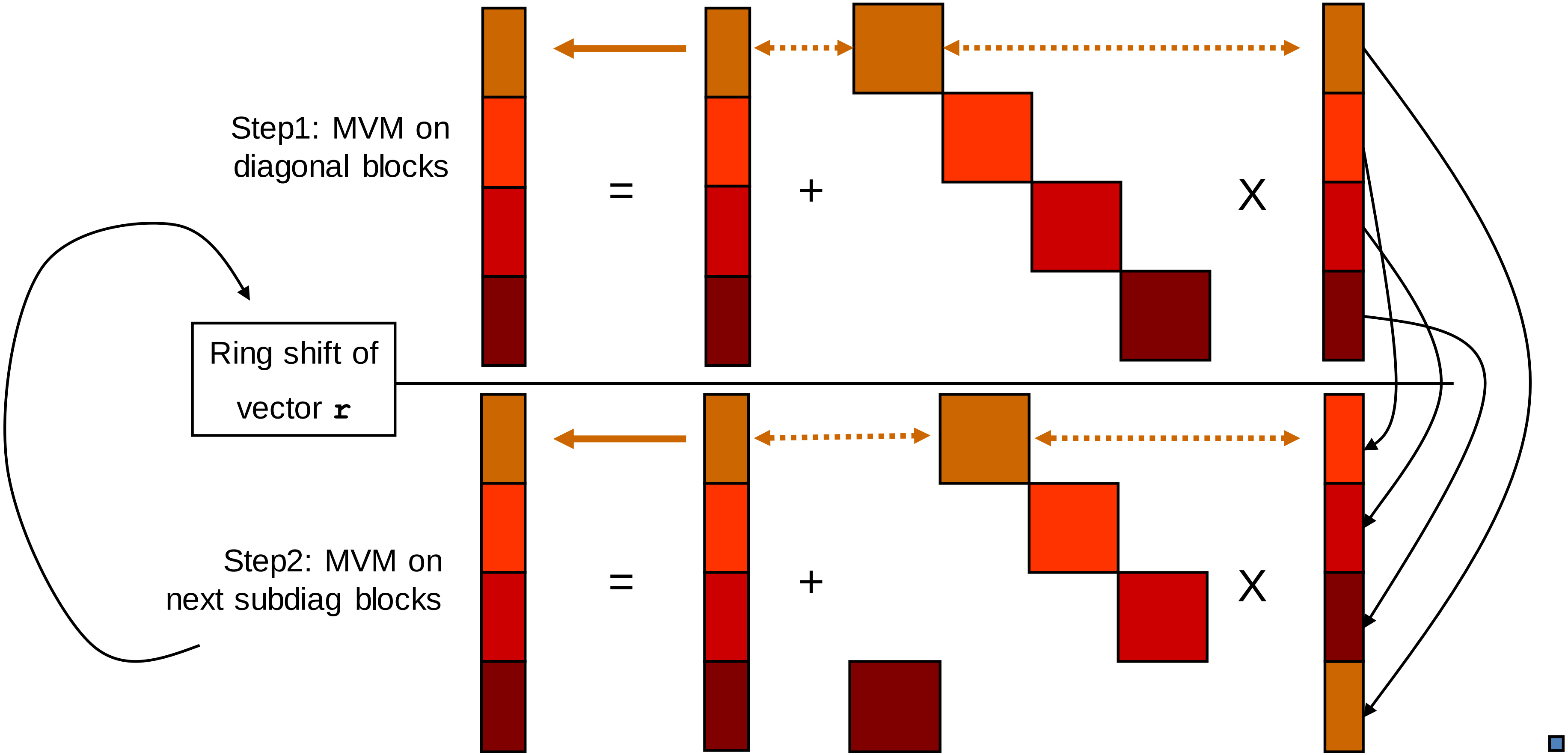
in calc_tile()

```
double calc_tile([...])
{
[...]
#pragma omp parallel for private(x,dx,dy,dz,c,r) schedule(static,1)
collapse(2)
  for (y = ystart; y < ystart+tilesize; y++)
    for (x = xstart; x < xstart+tilesize; x++)
      {
        [...]
        tile[(y-ystart)*tilesize+(x-xstart)]=(unsigned char)c;
        sum += c;
      }
    return sum;
}
```

in main()

```
if(0==my_rank) {
   MPI_Reduce(MPI_IN_PLACE,&sum,1,
              MPI_DOUBLE,MPI_SUM,0,MPI_COMM_WORLD);
   printf("Avg gray = %.2lf\n", sum/((double)size*size));
} else {
   MPI_Reduce(&sum,&sum,1,
              MPI_DOUBLE,MPI_SUM,0,MPI_COMM_WORLD);
}
```

Assignment 8, Task 2: MPI dense MVM



Step1: MVM on diagonal blocks

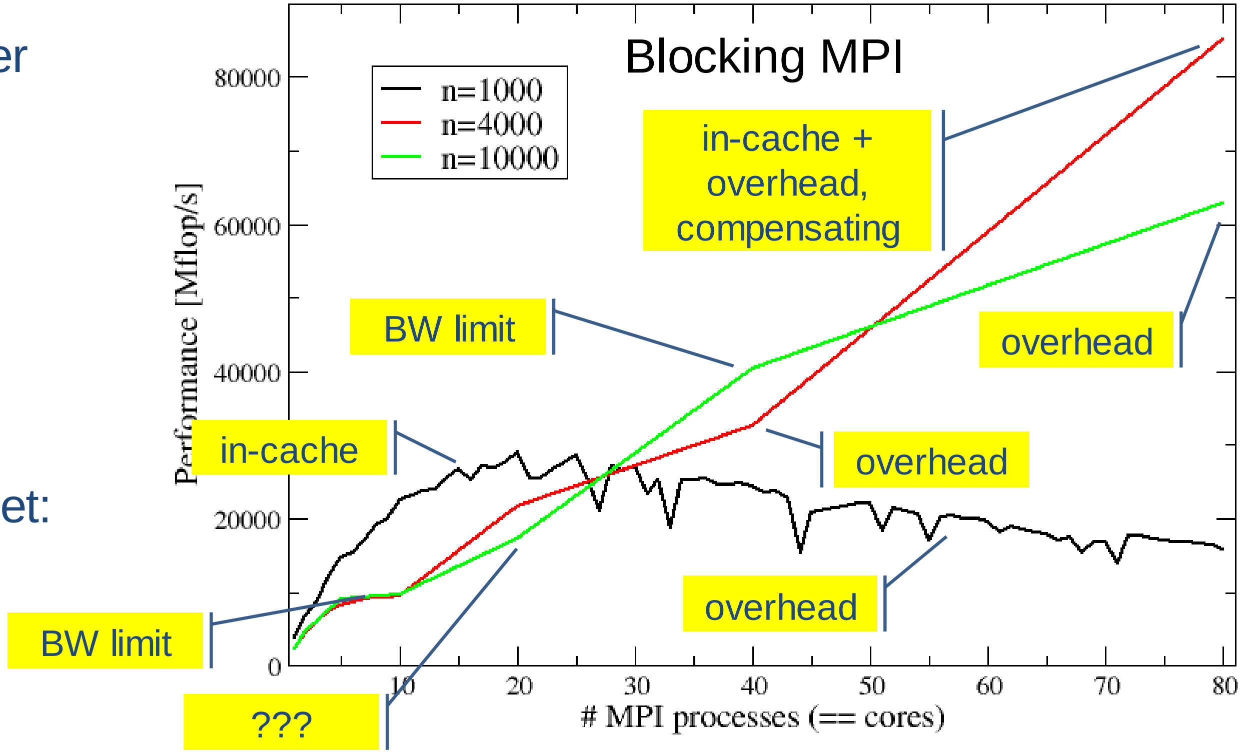
Ring shift of vector **r**

Step2: MVM on next subdiag blocks

Assignment 8, Task 2: MPI dense MVM

```
$ mpirun_rrze -np $p -pin \
0_1_2_3_4_5_6_7_8_9_10_11_12_13_14_15_16_17_18_19 ./a.out
```

- Cache size: 25 MiB per socket (10 cores)
- Working sets
  - 1000^2: 8 MB
  - 4000^2: 128 MB
  - 10000^2: 800 MB
- Roofline limit per socket: $\approx 10\ \mathrm{Gflop/s}$



in-cache + overhead, compensating

BW limit

overhead

in-cache

overhead

overhead

BW limit

???

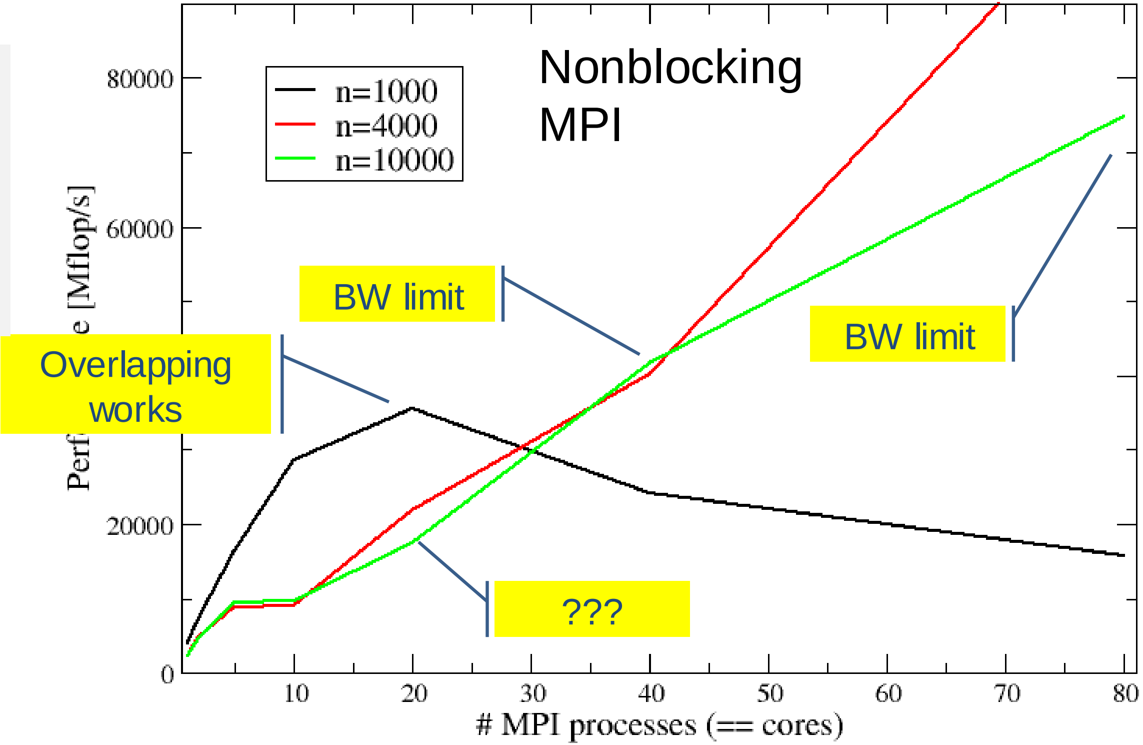
Assignment 8, Task 2: MPI dense MVM

Pseudo-code for nonblocking MPI: Use **`MPI_I{send/recv}()`** instead of **`MPI_Sendrecv()`**

```
MPI_Isend(buf1, to_left,…);
MPI_Irecv(buf2, from_right,…);

do_local_mvm();

MPI_Waitall(…);
```



Overlapping works

BW limit

BW limit

???

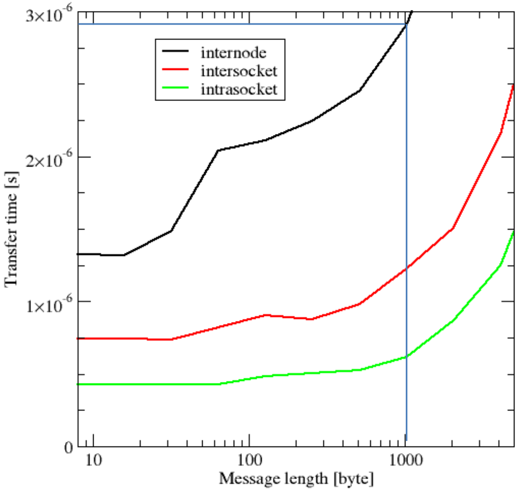
→ Overlapping seems to work on this system, with these msg sizes, and with this particular MPI

Assignment 8, Task 2: MPI dense MVM

- Back-of-the-envelope overhead estimate for 80 processes and n=10,000
  - Time for actual MVM execution:

$$T_{exec} = \frac{8 \times 10^{8}\ \mathrm{B}}{8 \times 40 \times 10^{9}\ \mathrm{B}/s} = 2.5 \times 10^{-3} s$$

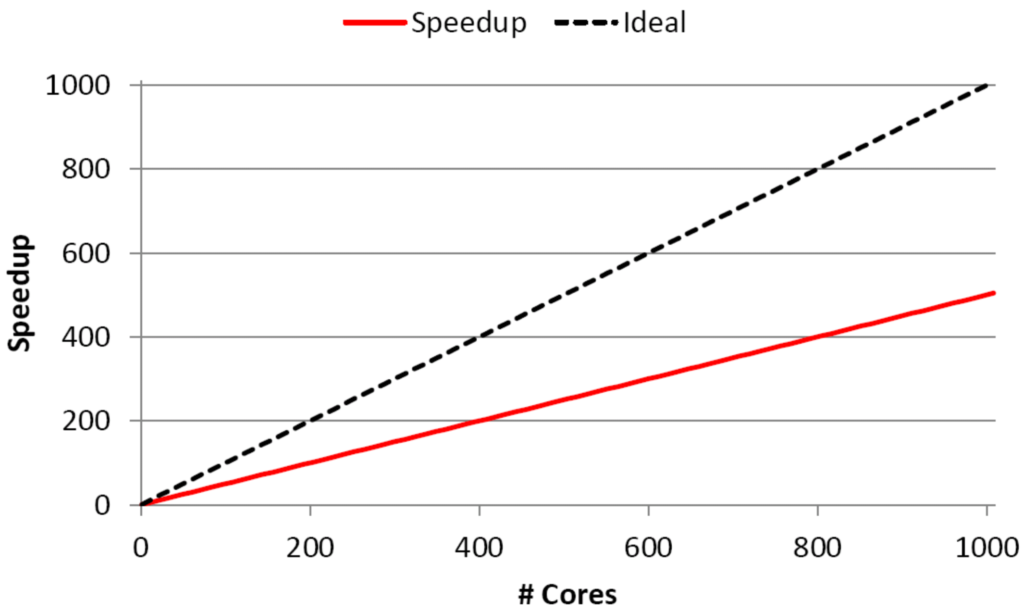
  - Time for communication:
    Need 80 ring shifts with 10000/80 = 125 elements, i.e. 1000 byte
    → $\approx 3\ \mu s$ per shift → $\approx 240\ \mu s$ overhead
    → 10% communication overhead in this case



- n=4,000 → 6.25 times faster computation.
  You do the math.

Assignment 8, Task 3

- Speedup is shown w.r.t. cores
- Modern multicore systems have different bottlenecks on different system hierarchy levels
- The node-level behavior is not visible in the plot
- Scaling baselines should be separated



- Question: “What is the scaling behavior on the cores of a single node?”
- 2nd question: “Why are you reporting only the speedup and not the performance?” (remember “slow computing” – Assignment 2)