Programming Techniques for Supercomputers Tutorial

Erlangen National High Performance Computing Center

Department of Computer Science

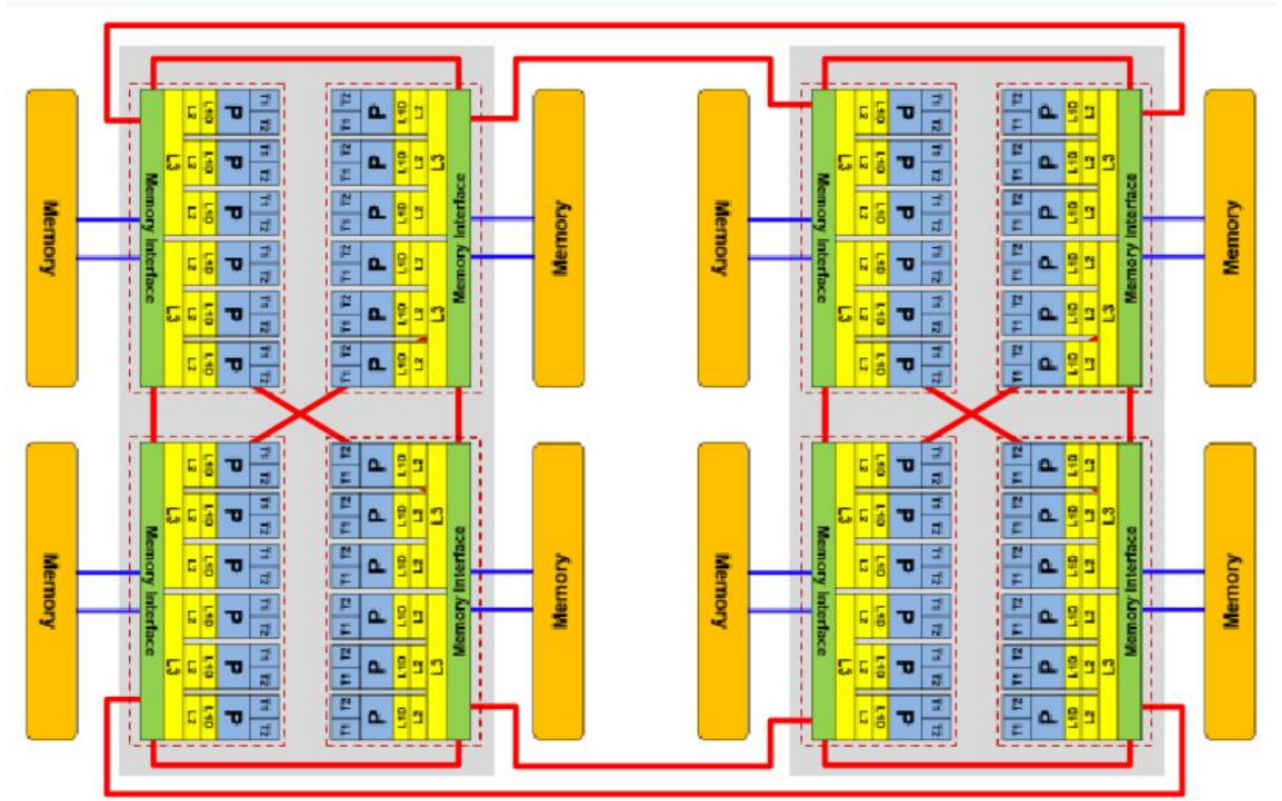
FAU Erlangen-Nürnberg

Sommersemester 2025

Assignment 10 – Task 1

Scaling behavior a) Topology

- 2 sockets
- 4 ccNUMA domains per socket
- 8 ccNUMA domains total



Assignment 10 – Task 1

Scaling behavior b) Memory Bandwidth

- We reach $\sim 260\ [GB/s]$ with full node

To include write-allocates

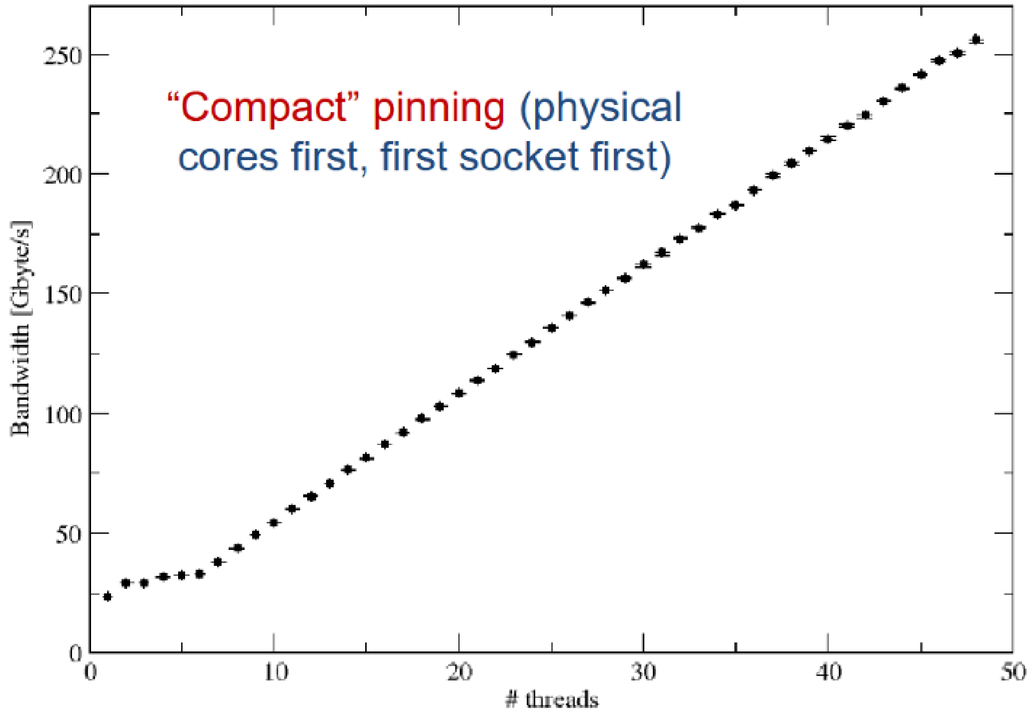
- Our hardware capable of achieving

$$\boldsymbol{b}_{\mathbf{s},\ \mathbf{Total}} = 260 * \left(\frac{4}{3}\right)\left[\frac{GB}{s}\right] \approx \mathbf{347}\left[\frac{\boldsymbol{GB}}{\boldsymbol{s}}\right]$$

- Each ccNUMA domain can achieve

$$\boldsymbol{b}_{\mathbf{s},\ \boldsymbol{ccNUMA}} = \frac{347}{8}\left[\frac{GB}{s}\right] \approx \mathbf{43.4}\left[\frac{\boldsymbol{GB}}{\boldsymbol{s}}\right]$$

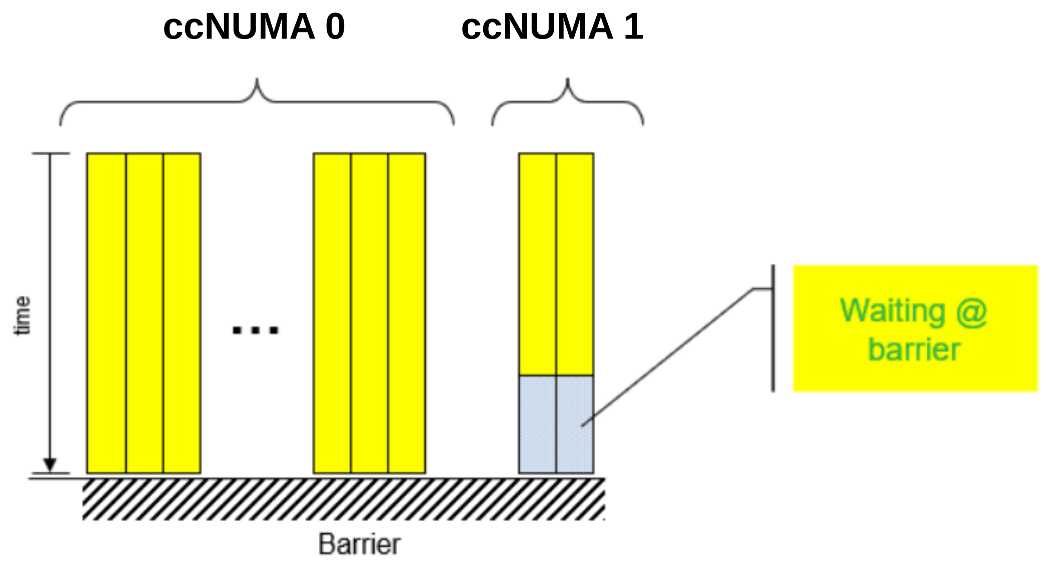
```
#pragma omp parallel for
for(int i=0; i<N; ++i)
   a[i] = b[i] + s * c[i];
```



Assignment 10 – Task 1

Scaling behavior c) Linear Scalaing

1. Every thread has the same workload
2. Left ccNUMA domain performance is saturated
3. Barrier enforces "speeders" to wait
   - Faster due to more available bandwidth

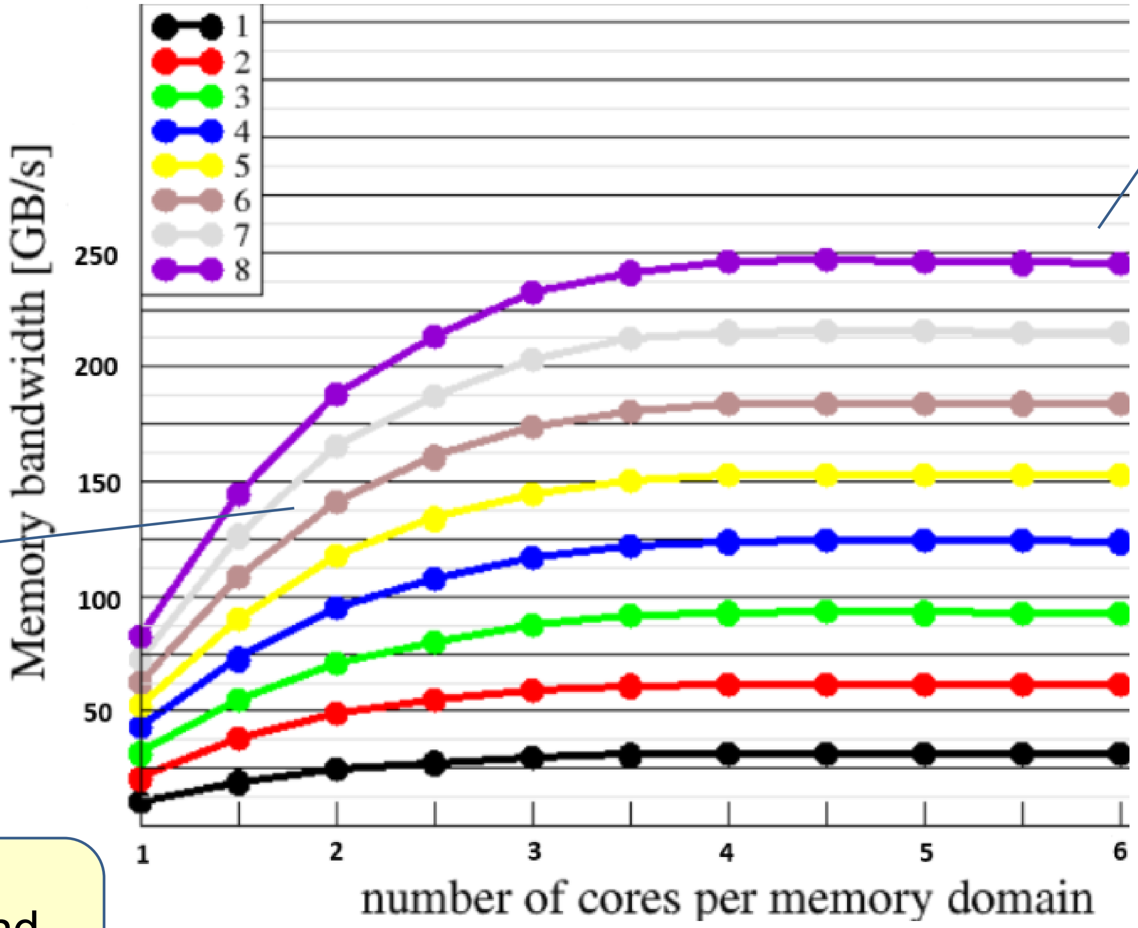


- Each new speeder contributes a "per-core performance" equal to the average per-core performance of the already-filled domain

- Constant contributions => linear scalability

Assignment 10 – Task 1

Scaling behavior d) Scattered Pinning



Achieve the same node-wide BW

Pattern resembles single ccNUMA domain saturation

Image taken from “TheBandwidthBenchmark” and modified for illustration. Only integer data points are relevant.

Assignment 10 – Task 2

Triangular parallel MVM

Parallelize outer loop to reduce OpenMP overhead

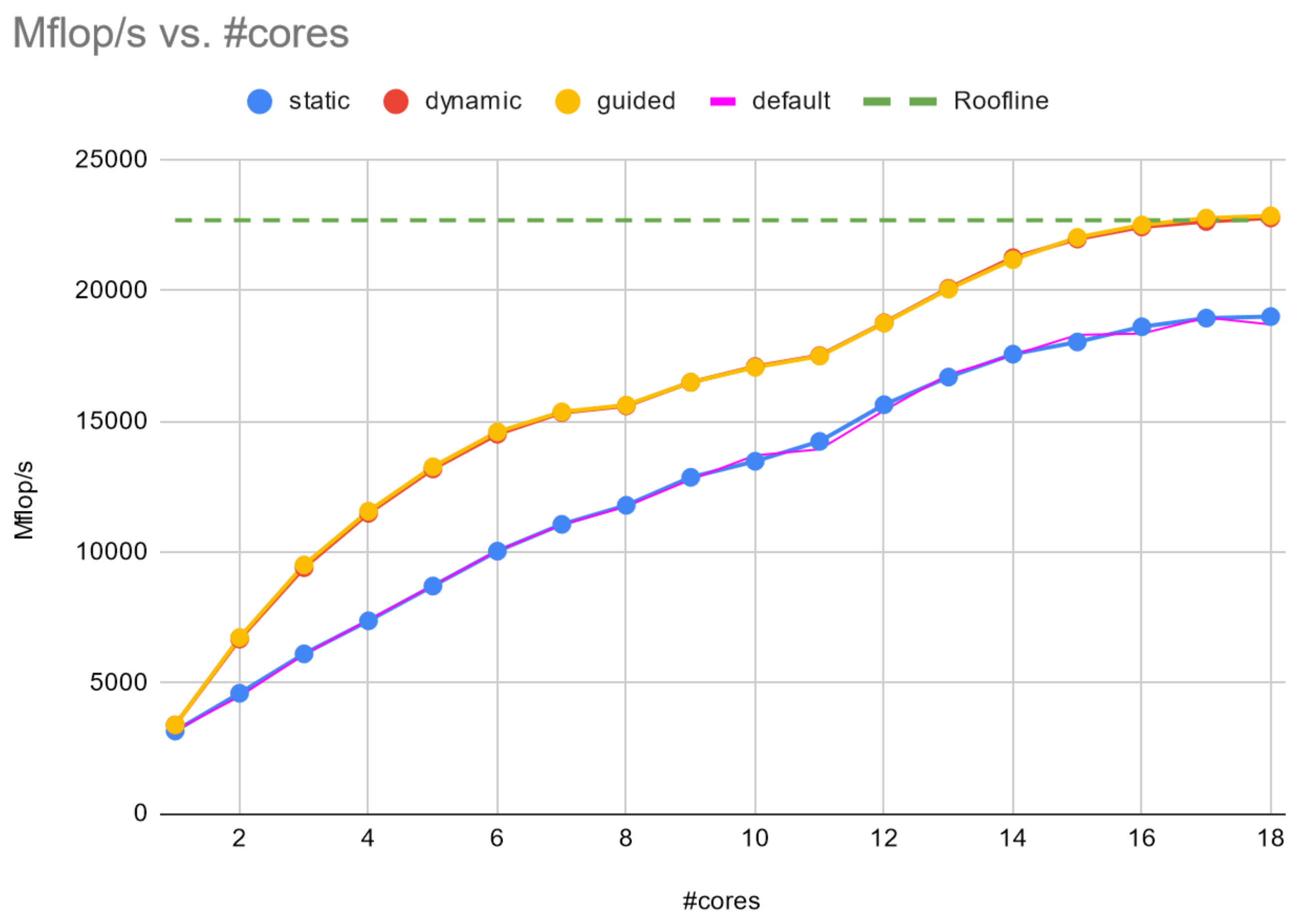
```
#pragma omp parallel private(i,k)
{
  for(k=0; k<niter; k++){
  #pragma omp for schedule(runtime)
  for(j=0; j<size; ++j)
      for(i=0; i<=j; ++i)
        c[j]=c[j]+a[i+size*j]*b[i];
      if(c[size >> 1]<0.0) whatever();
  }
}
```

Non-default schedule to improve load balancing

Assignment 10 – Task 2

Triangular parallel MVM

```
-Ofast -xHost -fno-inline -fno-alias -qopenmp
```



Assignment 10 – Task 3

Performance modeling of a 3D Jacobi smoother

```
#pragma omp parallel private(iter)
{
  for(iter=0; iter<maxiter; ++iter) {
    #pragma omp for schedule(runtime) private(j,i)
    for(k=1; k<N-1; ++k) {
      for(j=1; j<N-1; ++j) {
        for(i=1; i<N-1; ++i) {
          f[t1][k][j][i] = 1./6. * (f[t0][k-1][j][i]+ f[t0][k+1][j][i]+
                                    f[t0][k][j-1][i]+ f[t0][k][j+1][i]+
                                    f[t0][k][j][i-1]+ f[t0][k][j][i+1]);
        }
      }
    }
    #pragma omp single
      swap(t0,t1);
  }
}
```

Assignment 10 – Task 3

Performance modeling of a 3D Jacobi smoother b) Data Layout

- Swap from `f[t][k][j][i] -> f[k][j][i][t]`?

- Problem 1: Read and write streams are interleaved
  - ALL cache lines are modified
  - RHS data must be read, and then written to memory again!

- Problem 2: Poor SIMD utilization
  - Requires shuffling around data in SIMD registers

Assignment 10 – Task 3

Performance modeling of a 3D Jacobi smoother c) Performance

- $b_s = 41 \left[\frac{GB}{s}\right], (L3\$)\ CS = 25\ [MiB]$

L2 not considered on IVB (inclusive $)

- $N = 350 \rightarrow (2 \times 350^3 \times 8)[B] = 686\ [MB] \rightarrow$ out-of-cache working set
- **`OMP_SCHEDULE = static`**
  - LC: $10 \times 350^2 \times 3 \times 8\ [B] = 29.4\ [MB] > CS/2$
  - LC broken in L3 → $B_c = 40\ [B/LUP]$
  - $P = b_s/B_c = 1025\ [MLUP/s]$

Each thread needs 3 layers

- **`OMP_SCHEDULE = static,1`**
  - L3$-LC: $(10 + 2) \times 350^2 \times 8[B] = 11.8[\mathrm{MB}] < CS/2$
  - LC holds in L3 → $B_c = 24\ [B/LUP]$
  - $P = b_s/B_c = 1708\ [MLUP/s]$

Only hold 10+2 layers total!