Technische Universität Kaiserslautern
Fachbereich Informatik
RHRK
Priv.–Doz. Dr. Josef Schüle

# Introduction in High Performance Computing
# Sheet 5

---

**Performance Monitoring** Your next task is the performance monitoring of a reduction. Write a short example and inspect the performance with help of vampir (module add vampir/latest). Use calls to function sleep to analyse the tree spanned by the MPI installation.
Vampir visualization may be done on the head nodes. To collect data to be visualized you may use vampirtrace (module add vampirtrace) to instrument your code via:

```
vtcc -vt:cc mpicc file.c -o executable
```

Run now the instrumented code in `executable` for 8 (1 and 2 nodes), 16 (1 and 2 nodes) and 32 (1 nodes) tasks on nodes of the same type and compare the results.
If you use the MPI-Distribution OpenMPI you may use mpicc-vt to compile your files.

**MPI Parallelization of CG Algorithm** Your task is to parallelize the code for the CG algorithm given in the last exercise with MPI. Consider first a corresponding distribution of data. Use this distribution starting in line 40 of `ssolo.c` (function `zeros`) and distribute matrix $A$ and vector $b$ as necessary.
Test the MPI version for 4 and 8 cores on one node. Use 16 and 32 cores with $n = 20,000$. Does the program scale (going from 4 to 16 and 8 to 32 cores) according the $O(n^2)$ complexity?
If you are not satisfied with the performance of the program – Vampir may help you.
Now that you have the MPI version - could it be used to write an effectiver OpenMP code?

**One sided communication** Use the given sources and implement a one sided get between 2 tasks. Run the benchmark with 2 tasks on the same node and on 2 different nodes.