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

Introduction in High Performance Computing

Sheet 2

Hints The following exercises may be executed on elwe.rhrk with the intel compiler. Think of proper optimization options for the compiler. On elwe we have different nodes which may be selected via the batch system LSF (keyword `model`). Please specify hardware characteristics for the nodes choosen, especially the speed of the CPU, the size of the cache and, if available, the speed of the DRAM and the cache topology (1st, 2nd, 3rd,....). See task to the topology of the system below.

Determine the topology of the used systems
Find a way (*apropos topology* in a terminal window may help) to determine the memory and processor topology of the system used.

First experience with benchmarks
Benchmarking is essential for High Performance Computing (HPC). This exercise gives a slight idea of writing your own benchmark and practice in the usage of an existing benchmark.
Stream Benchmark
Have a look at the WWW-Site `http://www.cs.virginia.edu/stream/`. Download the stream-benchmark, compile, and run it in double precision.
Bonus
Experiment with the command `numactl`. Run your programs with memory bound to the same CPU and to a different CPU.

Theory - SPM
Consider the nested loops

```
 for (i=0;i<n;i++) {
   for (j=0;j<m;j++) {
  x[i] += A[i*m+j] *y[j];
}
}
```

Calculate the estimated execution time for this nested loops in SPM

- for a single processing core
- for several (p) cores.

For several cores outline first a proper algorithm. Assume that the data is already distributed on the cores accordingly.

Benchmarking again
Write your own benchmark with help of the supplementary file `Benchmark.h` as header file. Find out, how the time measurement is called. Benchmark the loop

```
for(i=0;i<N;i++) y[i]=2.3*x[i]+y[i];
```

after initializing the array `y` and `x` with random numbers. Both arrays are of type double precision. The size of the arrays should vary with `N` in the range of 10 to 10 Million. Choose the steps inbetween properly. Use the number of FLOPS as measure for the output instead of time. Depict your results graphically and choose a proper scaling of axis. You might think of 2 plots - one for small N and one for larger ones. In some cases the naive way of writing this exercise in a single function in a single file gives unrealistic results, especially with higher optimization levels to the compiler. Write the above for loop in a file by

its own and perform the compilation in two steps (think about a `Makefile`. This will help you for future exercises and programming tasks).
Now that you have a working environment for benchmarks, exchange the above loop against

```
for(INC=1;INC<17;INC++)
for(i=0,s=0.;i<N;i+=INC) s+=y[i];
```

for `N=1,600,000`.
Depict your results graphically using memory bandwidth (Bytes per second) as measure. Plot Bandwidth (y) against INC (x).