I Parallelisation of self-consistent calculations of electronic-structure properties

I.0.1 First scaling tests

The first step in analysing the scaling of QUANTUM ESPRESSO is to perform a baseline scaling test without any optimisations yet appplied. In Fig. **??** two scaling tests on the earlier mentioned benchmarking systems Si and TaS2 are pictured. The QUANTUM ESPRESSO version used is compiled using OpenMPI 4.1.0 , without any further compilation or runtime optimisation parameters used.

What exactly? Which compiler?

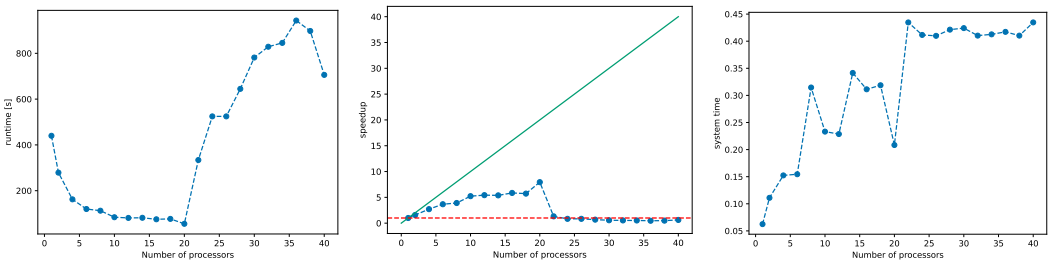

Abbildung I.1: *XXX*

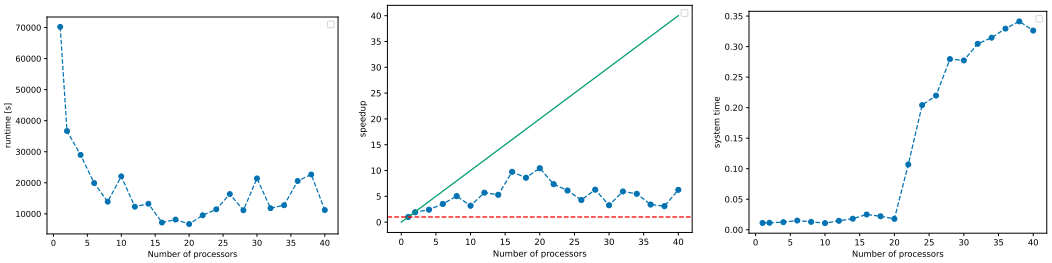

Abbildung I.2: *XXX*

caption

Three different metrics of scalability are pictured in **??**.

- runtime: absolute runtime of the compute job
- speedup: runtime divided by runtime of the job on a single core

- system time: percentage of wall time used by system tasks, e.g. writing to disk, etc. (calculated as walltime - cputime)

For further analysis mainly speedup will be used as a metric of scalability, because it lends itself to easy interpretation: optimal scalability is achieved when the speedup scales linearly with the number of processors (with a slope of one), as discussed in ch. **??**.

On a single node, both the Si and TaS2 calculations show good, but not perfect scaling behavior: the speedup does approximately scale linearly with the number of processors, but the slope is round about $\frac{1}{2}$. Even though the scaling behavior is not perfect, there is just a small, almost constant amount of runtime used by system calls, this speaks for good parallelisation: as discussed in ch. **??**, startup time is part of every

missing

When using more than one node, not only does the scaling get worse, the execution needs longer than on a single core for the Si system, with a marginally better performance for the TaS2 system. This is also seen in the plots of system time. The percentage of time used for tasks not directly related to calculations (mostly exchange of data in this case, which induces long waiting time when the connection between processors is not as fast as on one motherboard) goes from a near constant value for under 20 processors to 50% of the execution time for the Si system and 35% for the TaS2 system.

These scaling tests pose now two questions to be answered:

- Is better scaling on a single node possible?
- How can scaling over more than one node be achieved?

I.0.2 Testing different compilers and mathematical libraries

A first strategy for solving issues with parallelization is trying different compilers and mathematical libraries. In the PHYSnet cluster a variety of software packages is available. For the compilation of Quantum ESPRESSO a

missing

For testing Quantum ESPRESSO will be compiled using the following software combinations:

- OpenMPI 4.1.0 and OpenBLAS
- OpenMPI 4.1.0 and Scalapack
- Intel oneAPI 2021.4 (includes Intel MPI, Fortran and C compilers as well as Intel MKL, a scalable mathematical library)

caption

I.0.3 Using the parallelisation parameters of Quantum ESPRESSO

As detailed in section **??**, Quantum ESPRESSO offers ways to manage how the workload is distributed among the processors. In `pw.x` the default plane wave parallelization, k-point-parallelization and linear-algebra parallelization are implemented.

The benchmark pictured in I.4 is set up as follows: for a given number of processors N_p, the parameter N_k splits the N_p processors into N_k processors pools. As the number of processors in one pool has to be a whole number, only certain combinations of N_p and N_k are possible, for example $N_p = 32$ could be split into processor pools of size 2 with $N_k = 16$, size 8 with $N_k = 4$ or size 16 with $N_k = 2$. This leads to choosing the size of the processor pools as a variable, not

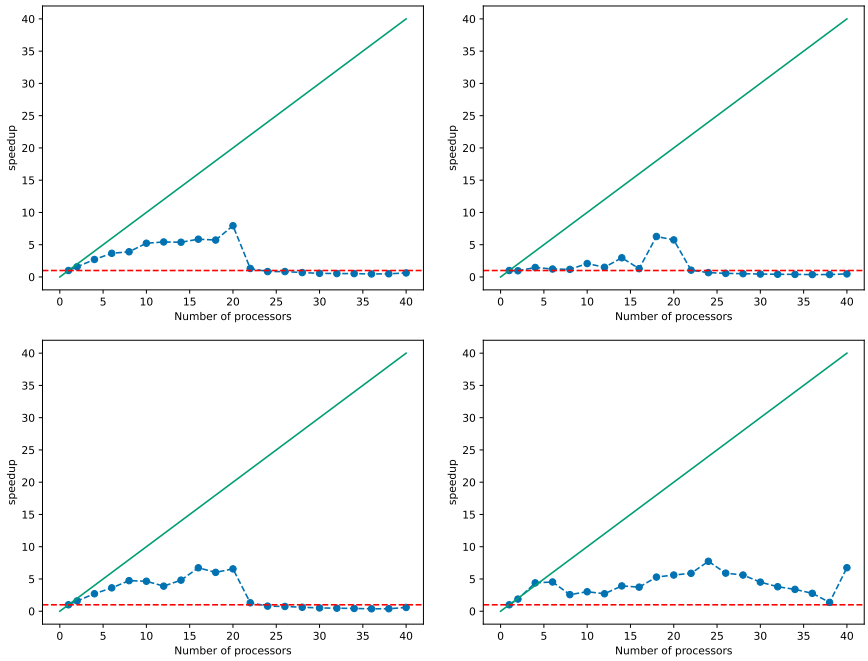

Abbildung I.3: *XXX*

the parameter `nk`. Fig. I.4 shows the scaling for poolsizes 2, 8 and 16 for QUANTUM ESPRESSO being compiled with OpenMPI/Scalapack and Intel oneAPI. This choice of pool sizes showcases the smallest pool size possibly (namely 2), as well as a bigger pool size with 16, that still gives rise to a few data points over the chosen range of processors.

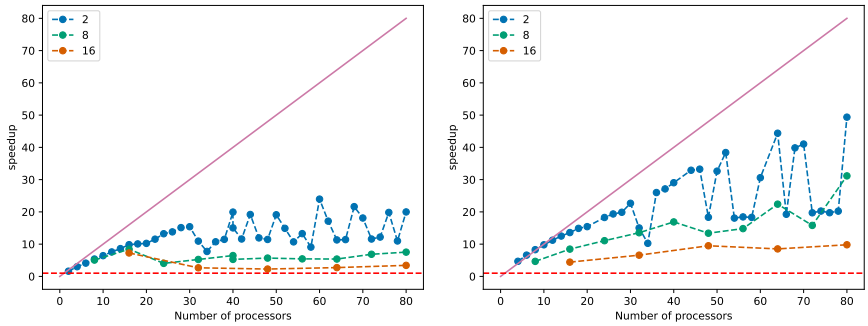

Abbildung I.4: *XXX*

Fig. I.4 shows

The same scaling test is applied to the TaS2 system in fig. I.5, with the same list of pool sizes, but over a wider range of processors.

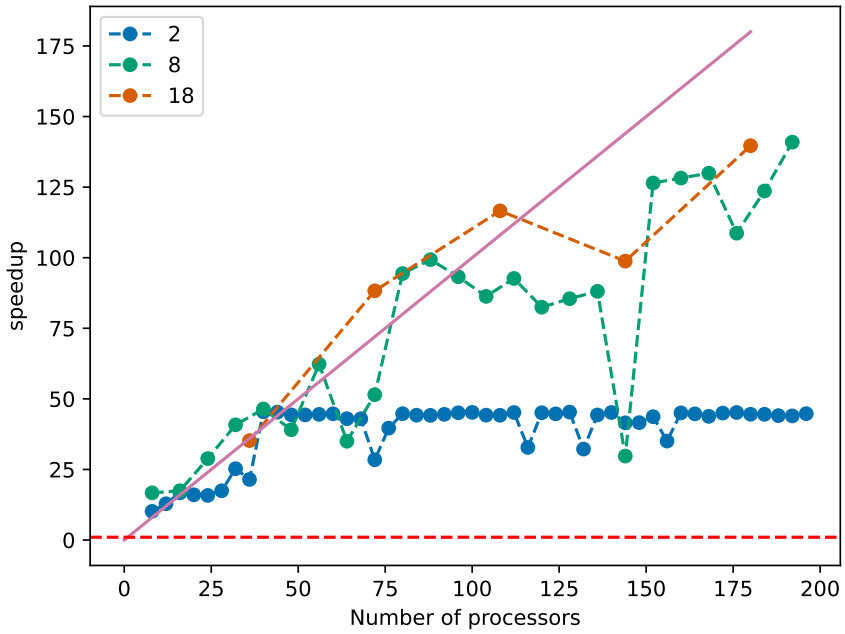

Abbildung I.5: *XXX*

Remarkably, the scaling behavior is swapped in comparison to I.4, as the pool size 2 saturates fast and the bigger pool sizes showing way better scaling behavior.

It can also be instructive to look at the idle time for this benchmark to judge the quality of parallelization.

Fig. I.6 shows a distribution of idle times between 1 % and 4 % of the whole wall time, without any kind of systemic increase over any range of processors. This means the parallelization is as good as possible for these

I.0.4 Comparison with calculations on the HLRN cluster

I.0.5 Conclusion: Parameters for optimal scaling

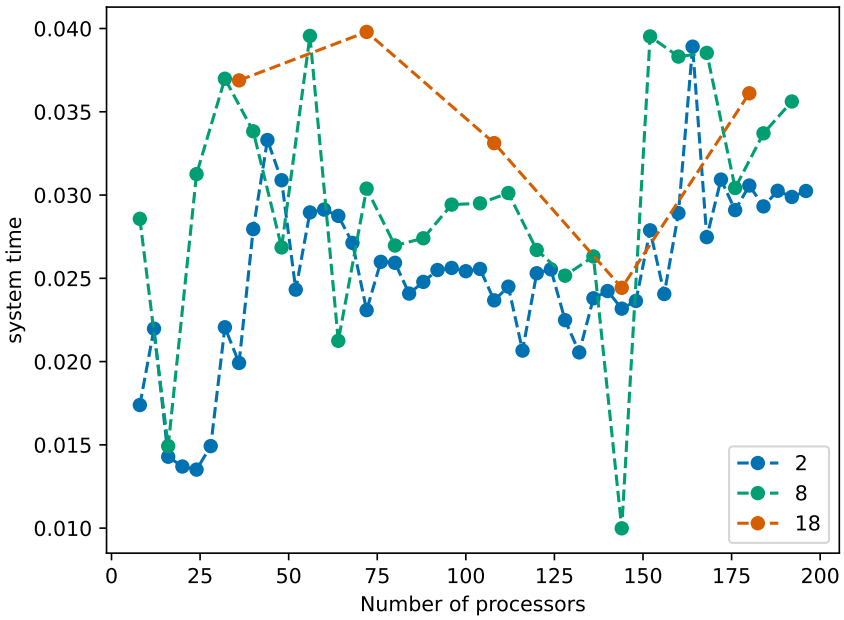

Abbildung I.6: *XXX*