Performance/Scaling of the tRecX Code

Scaling is strongly application-dependent, mostly driven by the locality properties of the system's representation. Performance in tRecX is achieved by a highly structured representation of the operators. Most of the time is spent in time-propagation with the repeated application of block-sparse operators, where each block in itself is structured, e.g., as tensor products or singular-value representations. The linear dimension of the blocks varies from 1 0 to 400. The blocks determine the code's granularity and are distributed based on self-measured CPU load. The distribution algorithm also attempts a trade off between load-balance and communication.

On the block level LAPACK and EIGEN routines are used. Expansion of the full operator into a matrix in order to use parallel linear-algebra software, while possibly improving formal scaling, leads to the loss of tensor structure and dramatically increases the flop count. Scaling is shown in Fig. 1 for calculations where most of the load is used in the electron-electron interaction. The scaling data are for actual production-type runs including processing and output of intermediate results after each timestep, which involves non-local operations. It does not include setup times, as setup data can be reused among the runs of a parameter study. Results are for *strong-scaling*, i.e., a problem of fixed size is distributed over an increasing number of nodes with a corresponding increase of the relative role of communication.

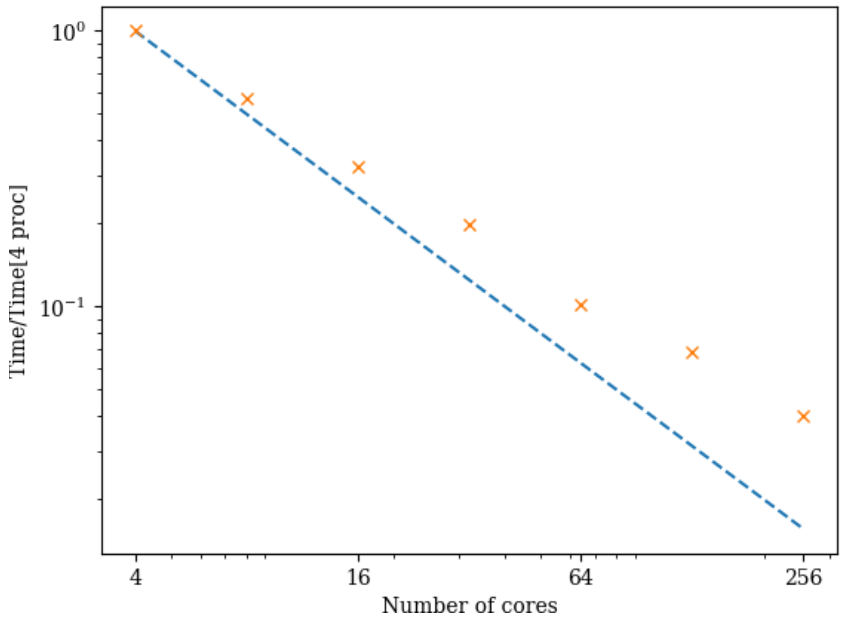

FIG. 1. Scaling of tRecX for double-ionization of Helium at laser wave length 400 nm. The results were obtained on the Munich Cool-MUC cluster. The specific section of Cool-MUC has nodes composed of two sockets with 16 cores each of an Intel(R) Xeon(R) Gold 6130 CPU @ 2.10GHz with 192GB RAM DDR4@2666MHz. The dashed line is ideal scaling.