Code Performance and Scaling

The antibody and docking protocols are embarrassingly parallel, *i.e.*, they compute multiple trajectories where each core calculates one trajectory without need to communicate to other cores besides job and initial seed assignment. Therefore these jobs will use multiple CPU cores for its computations with extremely little or no penalties.

Typical antibody jobs require between 300-350 CPU-hours. On Stampede, we will assign appropriate job sizes for each protocol, using MPI to manage the job across cores. The figure below shows the scaling performance of the antibody CDR H3 modeling application. As the number of CPUs are increased from 16 to 512, there is very little penalty in efficiency. At 1024 CPUs, the efficiency drops off to about 80%. This drop in efficiency can be easily explained and gives insight into the best practice for choosing the number of cores for each job. The antibody job creates 1000 candidate structures using 1000 independent trajectories. The 1000 trajectories are each assigned to one core; in the case where 1024 cores are requested, 24 cores are idle immediately. As each trajectory finishes, since there are no remaining trajectories to compute, the assigned core will become idle. Under MPI, all cores wait to exit until all trajectories are complete. For jobs with smaller numbers of cores, there will be less time wasted waiting for all jobs to complete. Thus, the number of CPUs should be smaller than the number of trajectories, at least by a factor of two and higher if rapid results are not needed. The final decision is a trade-off between computational efficiency and rapid job completion time. We will typically choose cores equal to about one quarter of the number of independent trajectories to compute. For the antibody application, which averages 361 CPU-h per job, this corresponds to 97% efficiency and job completion in around 1.5 hours on 256 cores.

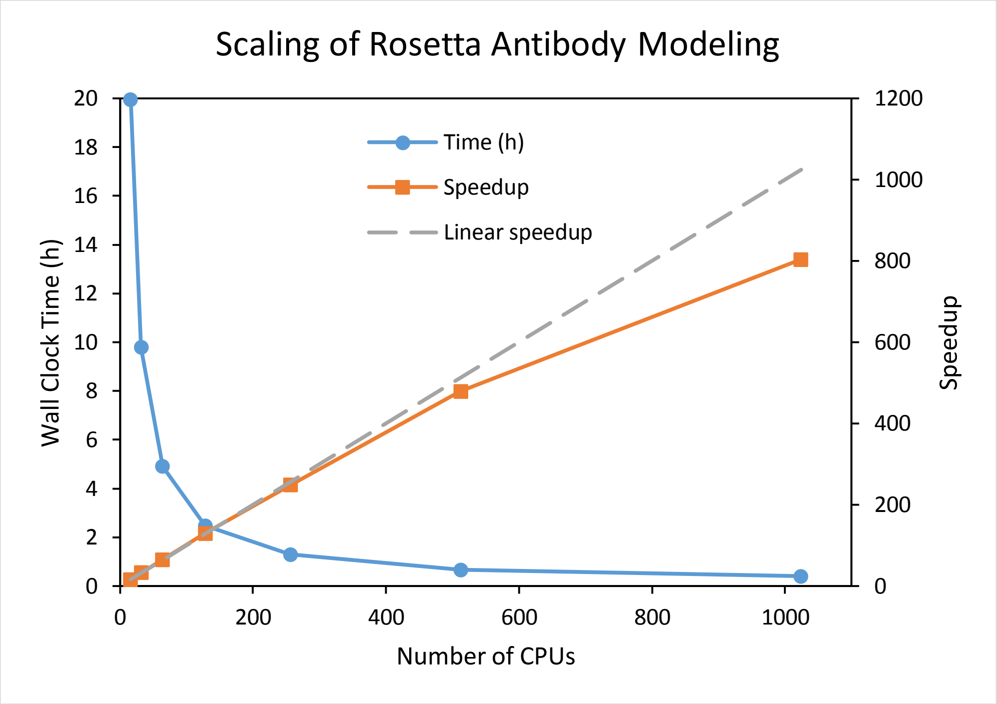

Figure: Scaling of Rosetta's antibody modeling application. Test case is a naïve repertoire antibody (Ab ND2-1005) with a CDR H3 loop of 12 residues. Calculation performed on Stampede using standard run options. Code was compiled with Intel C++ compilers using MKL libraries for automatic offloading of appropriate computations to the Intel Xeon Phi co-processors.