How to address pre-Exascale systems? Is it sufficient to tune existing applications?

Guillaume Latu

Disruptive transitions in HPC landscape have already occurred when the fundamental strategy for organizing data and computations changed. For example, back in the 1990's, many programmers had to switch from vector processing to massively parallel computers. There was a disruptive ramp-up phase during which developers took care both of the old vector code and the new massively parallel processing code. We are facing the same kind of situation with the petascale-exascale transition currently on-going. So many applications are already bound by memory bandwidth and have sustained performance below 2% of CPU peak. The capabilities of networks, both in term of latency and bandwidth can hardly compete with the high CPU speed. Then, applications will need to be agile in evaluating and adopting technologies and software tools that are most promising along the way. It will even require exploration of new computing paradigms as we move to extreme parallelism and heterogeneity. One key component to help for this issue is the concept of mini-application. Mini-applications attempt to capture meaningful aspects of a fraction of a large application. The goal is to provide a sufficiently small set of code lines (e.g. less than 2000 lines of code) in order to try different strategies, various algorithms or schemes quickly and at a relative low human cost. Indeed, this is much important to converge towards good solutions satisfying numerous constraints at a reasonable cost, but also to have means to keep the pace of change in software/hardware.

Gysela is a simulation code that investigates plasma turbulence within tokamak devices. In this parallel code, the semi-Lagrangian scheme is used to solve a set of 5-dimensional gyrokinetic equations which are self-consistently coupled to a Poisson equation The relative efficiency of this MPI+OpenMP application, for a weak scaling starting from 8k cores, is higher than 95% at 65k cores on several supercomputers. Producing physics results with this tool already necessitates large CPU resources. Moreover, it is expected that the needs will increase in the near future due to higher mesh resolution, and core plasma-edge plasma interaction. In that respect, adapting the code to upcoming parallel architectures is a key issue, and designing a new code that departs from the existing one is required. In this talk, the bottlenecks and some of the possible solutions to follow the path of Exascale will be given. Modelling the interplay between edge and core turbulent transport with such a gyrokinetic code is for a large part an unexplored territory, mainly because of numerical and physics bottlenecks. These stem from some characteristics of the edge physics, all requiring a dramatic increase of numerical resources: large variations of equilibrium temperature by 1 to 2 orders of magnitude from the core to the scrape-off layer region, and complex geometry including plasma-wall interaction and X-point. Several additional difficulties come from the necessary use of more complex mathematical and HPC methods: develop specific multi-scale methods and use multi-patch techniques, bring the scaling of algorithms at the exascale level while reducing prohibitive communication costs.