

## **Innovation for Exascale computing**

Exascale computing is a goal in itself for some basic science. It is also a way to push to the limit existing practices. This way, new tools will be developed that will be very useful for large scale (not necessarily exascale) computing. Thus I will address scientific computing not only from the exascale point of view. What is behind exascale computing is that the number of cores is not the limit but data transfer between cores and thus the way we use all these cores. This raises several issues. I will start with two topics that, to my opinion, are underestimated. Then, I will finish with the computer science aspect that I guess other authors will address as well.

### **Books are fundamental as a basis for paradigm changes. More than scientific papers.**

Exascale challenge can only be solved with breakthroughs. Incremental solutions are not enough. Innovations are first published as papers in journals. Then from these seminal papers, many other papers are published that develop and refine the original idea. At some point, this literature becomes very numerous. Due to space constraints, technical but important analysis can not be given or are referred to other papers which refer to other papers and so on. When such a point is reached, it becomes very difficult to have a clear view of what is going on. Limitations of the original approach get blurred. In short, informations are too much scattered among papers and are difficult to analyze. Then, a book written by one (or two or three) author that gives a unified self-contained view to this specific area is a real bonus. The book structures the results of the literature. Based on this material, a critical analysis is made possible and it becomes possible to propose new paradigms. From my own experience on coarse space for parallel solvers, I can testify that the book by O. Widlund on domain decomposition methods was a key element in our capacity to propose new coarse space constructions that are adaptive to heterogeneities. Books take time to write and one way or another should be found to encourage scientists to write them since they are the basis for paradigm changes.

### **Numerical simulations in the hands of end users**

Complexity of simulations has greatly increased and that at the same time the end user of simulation codes is less and less aware of the methods that are used. First computations were made on quite small structured two-dimensional grids with a simple physics where it was easy to have everything under the control of one scientist. Now, computations are made on large unstructured three-dimensional grids with complex multiphysics. Somehow, this is a measure of the success of scientific computing/computational physics. But the consequence is that it is virtually impossible for one person to control the whole thing. The need for automating as much as possible the whole simulation process can only rely on mathematical analysis. An example of such successful automation is the time step adaptation used classically in the numerical solving of ordinary differential equations

(ODE), see Matlab routines ode45 or ode23. Although partly heuristic (and as such not yet perfect), these algorithms are essentially based on rigorous analysis of numerical schemes for ODEs. A key feature is that they apply to arbitrary ODEs. All efforts pointing to this direction for solving partial differential equation (PDE) based models are the future of scientific computing: a posteriori error control, adaptive mesh refinement, adaptive time step for PDE, automatic monitoring of a given simulation, adaptive iterative solvers, ... . All these methods should have the strongest mathematical foundation since it is the only way to ensure both robustness and generality. Many efforts already exist in this direction but what is sometimes lacking is to have tools that apply to arbitrary PDEs and not to a specific PDE.

Another important aspect of having non specialists of numerical analysis performing more and more complex simulations is to give them the possibility to express their own physics without a priori limitation and at the same time give them powerful automated tools. A kind of Matlab for PDE is needed. Domain specific language (DSL) for PDE based models, such as FreeFem++, Feel++ or Fenics, enable end users to do such things. The development of such tools is a very difficult mix of software engineering and mathematical tools to abstract numerical schemes. Such projects should receive long term funding as PETSC has for instance.

### **Wishlist addressed to computer scientist for exascale**

As any applied mathematician involved in high performance computing, I have a wishlist which is shared by many others I guess. I see the need for genuinely parallel operating systems. This is only way to really suppress bottlenecks in large scale simulations. Programming languages must give a way to express data locality. Task based programming will also help a lot. As for fault tolerance, it is important to develop algorithms and programming practices (systematic use of try-catch) that allow for hardware failures. Anyway, even if a small part of a program is not fault tolerant, it is enough to have the whole program not fault tolerant. Thus, the practical solution to this problem will come from the development of **RAIC** (redundant array of independent cores) analogous what has been done for storage with RAID. The machines we have access to are somehow flat and create artificial bottlenecks even for basic operations such as scalar product of two vectors. Processors have to be hierarchical from their initial design. Of course, all algorithms must be rethought in terms of parallelism and a modern "parallel Knuth" is needed.