Magret du canard, foie gras, cassoulet et beaucoup d'autres choses

ir. Jacko Koster

Jacko Koster was OOTI from December 1990 to December 1992. Currently, he is a three-year Ph.D. student at CERFACS in Toulouse, France. In this article he writes about his current work environment, the research world of parallel numerical linear algebra in which he is doing his Ph.D. and he looks back at his life at OOTI.

CERFACS

CERFACS (Centre Europeen de Recherche et de Formation Avancee en Calcul Scientifique) is located just outside of Toulouse, in the south-west of France, about one hour drive from the Pyrenees. CERFACS was founded November 9, 1988. The primary goal was to create a 'Centre d'Excellence' which expertise is the use of high-performance computers to solve problems in applied science and engineering and to be a focal point for research and training in this area. Although CERFACS has expanded and developed over the years, it still retains many of its original objectives and programmes. Currently, there are about 60 researchers working at CERFACS, mainly Ph.D. students and Post-Docs, working in various fields: Computational Fluid Dynamics, Electromagnetism, Visualization & Post-processing, Climate Modeling & Global Change, Shape Optimization, Computational Chemistry, and Parallel Algorithms.

Parallel Algorithm Project

The primary objectives of the Parallel Algorithm Project are to develop mathematical techniques, algorithms, software, and tools to support large-scale scientific computation on modern computers. The people in the project are involved in many training programmes both in support of these objectives and to spread the gospel in general about the use of modern computers in scientific and industrial computing. A major focus of the activities is to develop techniques to exploit parallelism and to train other scientists how to use the tools developed by ourselves and others. Since probably the most important building block for scientific computing is numerical linear algebra, this has been and still is a central activity in the research programme. Some examples of research topics are eigenvalue computations, block iterative methods, domain decomposition, preconditioning techniques, and computational kernels.

A subgroup of the Parallel Algorithm Project is concerned in the numerical soundness of the applied techniques. For an engineer it is of crucial importance to have confidence in the results of large codes and in the robustness of the models used. Indeed, the emergence of supercomputers has allowed intensive use of numerical simulation to replace physical experiments, even for problems at the frontiers of instability. However, tools which provide information on the quality and validity of computer results are very rarely used in industry. In order to fill this gap, the subgroups' objectives are to

- explore problems at the edge of computability,
- provide analysis tools, both quantitative and qualitative, that help in extracting the appropriate information from results seemingly wrong, and
- solve industrial problems which have the following characteristics: unstable models, pathological numerical behaviours, very large scale.

Sparse computations

Several people in the project including myself are involved in sparse matrix computations. Sparse matrices arise in diverse fields as management science, oil reservoir simulation, power systems analysis, circuit theory, structural analysis, and fluid dynamics. Mathematical models in all of these areas give rise to very large systems of linear equations that can be solved only because the matrices

contain only relatively few nonzeros (often much less than 5%).

The exploitation of sparsity (by avoiding operations on the many zero elements of the matrix) can lead to enormous computational savings that make the solution of very large systems feasible. To give an idea of the complexity of sparse matrix computations a non-exhaustive list of the sub-areas of research that underpin the basis of sparse matrix research is given below.

- Data structure design.
- Graph theory.
- Design and analysis of algorithms.
- Numerical round-off error analysis.
- Influence of computer architecture.

One of the most important operations in scientific computing is the solution of (large) sparse systems of linear equations $Ax = b$, where A is a non-singular $n \times n$ matrix, b is a vector of length n and x is the unknown solution vector. Methods for solving such systems can roughly be divided into two classes: *direct* methods and *iterative* methods.

Direct methods for solving a system $Ax = b$ try to compute the solution vector x exactly. A widely known method is based on LU decomposition. In its simplest form, the LU decomposition of A produces a lower triangular matrix L and an upper triangular matrix U such that $A = LU$. By solving $Ly = b$ by forward substitution (y is a temporary vector) and $Ux = y$ by backward substitution, the solution vector x can be obtained. In general, the resulting factors L and U contain more nonzeros than A.

Iterative methods, on the other hand, try to approximate the exact solution of a system in a number of steps. For example, a large class of iterative methods is based on a single splitting of A into matrices M and N such that $A = M - N$. The iteration scheme

$$Mx^{(k+1)} = Nx^{(k)} + b$$

produces a sequence of iterates $x^{(k)}$, $k \geq 1$, that (hopefully) converges to the solution vector x, starting from an initial guess $x^{(0)}$. Indeed, convergence is not always guaranteed, but if it is, then the method should converge as fast as possible. Many iterative methods exist, each having its own specific use for certain problems characteristics. Iterative solvers are an alternative to direct solvers either when there are too many nonzeros in the factors (L and U) of the decomposition to fit into the memory of the computer or when the amount of floating point arithmetic for the decomposition becomes prohibitive.

Of course, hybrid methods that use direct as well as iterative techniques also exist.

When the sparse systems that are to be solved become really huge (i.e., systems of 1,000 upto 1,000,000 unknowns), the use of parallel machines comes into being. Many techniques that exploit parallelism in solving linear systems of equations exist. An obvious kind of parallelism comes from (in the case of a distributed-memory machine) distributing the data over the several processors and letting the processors simultaneously perform the necessary operations on the (local) data. An additional kind of parallelism comes from the sparsity that is present in the problem. The order in which many of the operations have to be performed is irrelevant in sparse computations, whereas in dense computations (that contain only relatively few zeros in the matrix) they have to be performed in sequence. The highest achievable level of parallelism, however, is very problem dependent. In general, one cannot say that the sparser the problem, the more parallelism can be obtained.

A simple but very effective idea in sparse computations is to decompose a large system of equations into several smaller and highly independent subsystems that can be solved efficiently on multiprocessor machines.

OOTI and I

My final project during OOTI dealt with the LU decomposition of general sparse matrices on a distributed-memory multiprocessor. The project was carried out at Koninklijke/Shell-Laboratorium, Amsterdam and the machine used for experiments was a Parsytec FT-400 consisting of 400 T800-20 transputers. During this project I came into contact with the Parallel Algorithm Project at CERFACS and they granted me a three-year Ph.D. position.

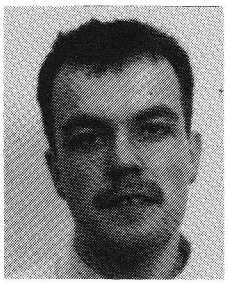
My Ph.D. subject is related to my final OOTI project so that I fully benefit from my expertise acquired at Shell. It concerns reordering techniques

of sparse matrices that are conducive to the convergence of iterative methods and/or increase the numerical stability of direct methods. Such reordering techniques often have a combinatorial and a graph-theoretical nature. Optimal reorderings are often hard to compute (NP-complete), and that is why many of the existing techniques are based on heuristics. There already exists a vast amount of literature on this subject, but since reorderings can have a really dramatic impact on the efficiency and numerical stability of the applied methods, it still remains an active field of research.

There are not that many parts of my OOTI knowledge that I put into practice nowadays (apart from my Shell experience). Tools that are being taught in the OOTI programme are not being used: specification languages are not even known, object-oriented languages are not being used (although I think they can be quite useful in this area of research). Everything is implemented straight into FORTRAN77 and a little in C. In fact, these two languages are the only 'existing' languages in the linear algebra world. Even FORTRAN90, the successor of FORTRAN77, that is already on the market for a couple of years, has a hard time of conquering a place in this field of reaearch.

When I think back of my OOTI period, then, among other things, I remember that the relevance of some of the OOTI courses was not always clear to me. Sometimes I had the feeling that the people were telling about their own favourite subject and other times I thought that the course, although quite interesting, was not really appropriate for the OOTI curriculum, but would fit better in a normal Masters programme for computing science.

Nevertheless, I think that OOTI helped and helps me a lot. If I would be in a similar situation now, as if I was in December 1990, then I would probably choose for OOTI again. OOTI gave me the chance to put theory into practice and to look at computing science problems in other research areas like mechanical engeering and electrical engineering. For me, the real value of OOTI was not in learning new facts and figures, but more in understanding how to organize things, how to work in a group of people, how descent project management looks like, etc., things of which you are not fully aware when you finish your Masters. And without any doubt there are other things that I picked up during my OOTI life of which I do not realize that I benefit from them now. □



Ir. Jacko Koster completed the post-masters programme Software Technology in 1992 and is currently working towards his Ph.D. at CERFACS (Centre Europeen de Recherche et de Formation Avancee en Calcul Scientifique) in Toulouse, France. He is a member of XOOTIC.