

Florin Dobrian - Professional Experience and Interests

My basic education is in computer engineering. I began my professional training in 1984, studying both computer hardware and software at the Bucharest Polytechnic Institute, in Bucharest, Romania. During those first years I developed a strong interest in algorithms and software engineering. After receiving the M.S. degree in 1989 I worked in the software industry for five years, until 1994. The projects I worked on focused mainly on networking. This period of time gave me the opportunity to improve my software engineering skills and also exposed me more to algorithms, in particular to algorithms of combinatorial nature. After five years in the industry I decided to join the academic environment again in order to obtain a Ph.D. degree.

I started my doctoral studies at Old Dominion University (ODU), where I had the opportunity to work with researchers with a strong expertise in scientific computing. This allowed me to study challenging combinatorial algorithms but also to contribute with my software engineering expertise. My doctoral advisor was Alex Pothen.

The particular scientific computing field I belong to, since I began my doctoral studies, is known as combinatorial scientific computing. This field is concerned with the design, analysis and implementation of algorithms for combinatorial problems that arise in scientific computing. My work spans both theory and practice. While I am interested in the design and analysis of various combinatorial algorithms for scientific computing, I am even more interested in high quality software implementations of them. It is my strong belief that, while such algorithms can be highly technical and sophisticated, they should be easy to deploy in applications, mainly because their users should not be expected to make the effort to understand their internal mechanisms.

As a graduate student I focused on the solution of sparse linear systems by factorization (direct) methods. In order to investigate various algorithmic aspects of the direct solution of sparse symmetric linear systems I developed my own solver library, called Oblio. Oblio is basically a framework for the quick prototyping and testing of algorithmic ideas and it was designed around two goals: flexibility and efficiency. In addition to being able to achieve good performance I needed a code that is easy to understand, maintain and modify, features that are not traditionally characteristic of scientific computing software.

For my doctoral dissertation I investigated two problems that arise in an out-of-core context. In order to solve very large linear systems one may have to use both the internal and the external storage of a computing platform. In the first problem the question is how small the core (the internal storage) can be such that every data item is read from or written to the external storage exactly once. In the second one the core is no longer considered large enough and data items may be read or written many times, the question being what kind of traffic to expect. My dissertation answers these questions by providing both analytical characterizations and simulations, the latter using Oblio.

After receiving my Ph.D. degree I obtained a postdoctoral position in a project funded by the Department of Energy. The project is known as Terascale Optimal Partial Differential Equation Solvers (TOPS) and it is part of a larger initiative called Scientific Discovery through Advanced Computing. TOPS focuses on parallel linear and nonlinear solvers for applications described by partial differential equations. My postdoctoral advisor was David Keyes, who was originally at ODU but later moved to Columbia University. I am currently still employed by the TOPS project but as an assistant research scientist at Columbia University since 2004.

My tasks in TOPS involved working both on applications and infrastructure. In terms of the former I assisted application people with my expertise in linear solvers and I developed codes for model problems in order to evaluate various parallel linear solver techniques in specific application areas. During the five year time frame of TOPS I collaborated with applied physicists and mathematicians from Princeton Plasma Physics Laboratory, University of New Hampshire, Argonne National Laboratory, Lawrence Livermore National Laboratory, Lawrence Berkeley National Laboratory, Los Alamos National Laboratory and University of Colorado at Boulder.

One of my contributions at the application level is in the area of implicit methods, which can be proved to be more effective than the previously used explicit ones. All application work that I did for TOPS was parallel, using PETSc, the scalable solver library.

In terms of infrastructure I had the opportunity to continue to maintain and upgrade Oblio, which became a robust package. A recent study by scientists from Rutherford Appleton Laboratory and Wolfram Research places Oblio among the best sparse linear solvers. Oblio is currently used by a machine learning application developed by the Australian National University and the National Information and Communications Technology Australia, as a solver for interior point methods for linear and nonlinear programming. A package from the automotive industry also relies on Oblio as a linear solver. In addition, I am currently in the final stages of deploying Oblio into Mathematica (from Wolfram Research).

Several of my ongoing infrastructural interests concern matching in graphs, a fundamental combinatorial problem in computer science with various applications in scientific computing. Among them, preprocessing sparse linear systems for numerical reasons. All current models that address this issue are based on matching in bipartite graphs. Such models are natural for unsymmetric systems but symmetric systems, such as those that arise in linear and nonlinear programming contexts, need to employ nonbipartite graph models, which are technically more challenging. Another application of interest for graph matching is the computation of sparse bases. Although this can be modeled as an edge-weighted matching computation in bipartite graphs, a more natural model is based on a vertex-weighted matching computation in bipartite graphs.

Because of the need for good graph matching algorithms in various areas of scientific computing I decided to build a graph matching library for the following matching problems, both bipartite and nonbipartite: maximum cardinality, maximum vertex-weight and maximum edge-weight. The package, called, Milan, was implemented by Mahantesh Halappanavar, a doctoral student of Alex Pothen at ODU and a close collaborator of mine, and contains algorithms that range from the easiest to implement ones to those that have the best known computational complexity, employing techniques such as augmentation along multiple paths, optimal disjoint set operations and binary heaps for priority queues. We are currently maintaining and upgrading Milan and we use it in various scientific computing experiments where matching computations are required.

Recent progress in the use of graph matching for scientific computing includes encouraging preliminary results in using a new nonbipartite model for preprocessing symmetric linear systems, developed by me and my collaborators at ODU, as well as a new theoretical result for the computation of approximate vertex-weighted matchings.

To summarize, the skills I acquired over the past twenty years include the analysis and design of combinatorial algorithms, linear and nonlinear algebra, the design of mathematical software and parallel computing.

In the future I intend to continue to seek challenging professional environments in which I can use and extend my expertise. In terms of infrastructure I would like to continue to contribute to various areas of scientific computing, through effective algorithm design as well as through robust and modern software implementations.

Any context that involves combinatorial algorithms and sparse linear algebra would be natural for me. In particular I would like to be able to contribute with new and better techniques for solving sparse linear systems. Among them, the use of nonbipartite graph models for preprocessing sparse symmetric linear systems for numerical reasons but also a new approach for performing structural computations based on paths in graphs rather than on edges in order to solve sparse unsymmetric linear systems.

Of even greater interest would be to develop and implement such techniques in parallel and distributed environments. The world of computing is evolving very fast into this direction. Our personal computers are in the process of getting multiple cores while the most powerful supercomputers are currently migrating from thousands of processors to tens and hundreds of thousands of processors. In addition, large data sets can be stored only in a distributed fashion. The biggest challenge is to design algorithms and software that address the difficulties of parallel and distributed contexts as effectively as possible.

From the application perspective, I enjoyed working with real applications over the past five years and I would like to have this opportunity in the future as well. One possibility that I consider is data mining. My current expertise covers various combinatorial algorithms such as matching, ordering, and partitioning. Because of this I strongly consider clustering, especially multilevel clustering, as a project of interest. Another alternative is optimization. I am attracted to this area by my collaborations on linear and nonlinear programming projects and by the challenges posed by the problems I encountered during these collaborations.