E.R. Jessup

User-centered Interfaces to Matrix Algebra Software

University of Colorado
Department of Computer Science
430 UCB
Boulder
CO 80309
jessup@cs.colorado.edu
B. Norris

Linear algebra calculations constitute the most time-consuming part of simulations in diverse fields ranging from atmospheric science to quantum physics to structural engineering. Reducing the costs of those computations can have a significant impact on overall routine performance especially as the size and complexity of scientific computations push the limits of processor technology. Programming scientific applications is hard, however, and optimizing them for high performance is even harder. There is often a large gap between the achieved performance of applications and the peak available performance, with many applications achieving 10% or less of the peak.

The process of converting matrix algebra from algorithm to high-quality implementation is a complex one. The code developer must create efficient implementations from scratch or select the appropriate numerical routines and then devise ways to make these routines run efficiently on the architecture at hand. Once the numerical routines have been identified, the process of including them into a larger application can often be tedious and difficult. The tuning of the application itself then presents a myriad of options generally centered around one or more of the following three approaches: manually optimizing code fragments; using tuned libraries for key numerical algorithms; and, less frequently, using compiler-based source transformation tools for loop-level optimizations. At each step, the code developer is thus confronted with many possibilities, generally requiring expertise in numerical computation, mathematical software, compilers, and computer architecture.

We will present our work in progress on a taxonomy of software that can be used to build highly-optimized matrix algebra software. The taxonomy will provide an organized anthology of software components and programming tools needed for that task. The taxonomy will serve as a guide to practitioners seeking to learn what is available for their programming tasks, how to use it, and how the various parts fit together. It will build upon and improve existing collections of numerical software, adding tools for the tuning of matrix algebra computations. Its interface will take a high level description of a matrix algebra computation

and produce a customizable optimized template using the software in the taxonomy. That template will aid the developer at all steps of the process–from the initial construction of Basic Linear Algebra Subprogram (BLAS)-based codes through the full optimization of that code. Initially, the tools will accept a MATLAB prototype and produce optimized Fortran or C.

A number of taxonomies exist to aid the code developer in translation of matrix algebra algorithms to numerical software. Perhaps the oldest one is the Netlib Mathematical Software Repository, started in 1985, which contains freely available software, documents, and databases pertaining to numerical computing including linear algebra. Contents are provided as lists of packages or routines, with or without some explanatory words. In newer work, the Linear Algebra Software Survey lists over sixty items categorized as support routines, dense direct solvers, sparse direct solvers, preconditioners, sparse iterative solvers, and sparse eigenvalue solvers together with a checklist specifying problem types for each entry. A third example is NIST's Guide to Available Mathematical Software (GAMS) which includes even more basic matrix algebra software along with software for a variety of other numerical applications.

Numerical software taxonomies such as those just mentioned are general and allow relatively stand-alone algorithms to be found, downloaded, and compiled, or used through a domain-specific Web interface. Operations for which no library implementation exists or more complex software packages, however, cannot be accommodated by this function-level indexing and query capability. They may also not have sufficiently simple common interfaces that enable Web-based services as is possible for some optimization problems. For example, the functionality of large toolkits, such as Trilinos and PETSc, is difficult or impossible to represent and maintain in existing taxonomies, which at present simply point the user to the toolkits' home pages. In many cases this idiosyncrasy results in a taxonomy user's being directed to simpler implementations, rather than to potentially better options, both in terms of suitability and performance.

One goal of our work is to make a taxonomy that is easy to use. While GAMS and Netlib are extensive and valuable resources, it is not always easy to find the proper results from them. In addition, the taxonomies provide little information about the routines they return. Our objective is to build a taxonomy that will provide all of the software needed to take a matrix algebra problem from algorithm description to a high-performance implementation. The taxonomy will thus include not only numerical routines but also interfaces to tools for code tuning. It will not comprise all available software but rather a collection of routines that allows for efficient programming of a wide variety of matrix algebra operations. We will pay special attention to identifying combinations of numerical software and tuning tools that can be successfully used together. Members of the taxonomy will be annotated with enough information to explain the purpose of the routines and how they are used in the code construction process. This practical taxonomy will thus serve as an educational tool for the

computational scientist.

Existing taxonomies serve primarily as user interfaces to numerical software. While such functionality is also one of our goals, we will create a richer representation and a set of interfaces that allows software tools to access the taxonomy information. For example, the same Netlib server that hosts the numerical software taxonomy also hosts a performance database for a variety of benchmark codes on an extensive list of platforms. There are no links, however, from the code of some of these benchmarks in the taxonomy to the performance results for the corresponding library or function in the performance database. A natural extension to the information stored for entries of the taxonomy is to provide references to existing performance data for benchmarks of that operation or for applications that use it.