CISC 822 Algebraic Algorithms Course Project
October 16, 2014

Each student is to complete a project in this course that involves coding and a writeup. The programming aspect may be done in a high level environment such as Sage or Maple, or be library based C/C++ (LinBox based, ...or FLINT or fflas-ffpack or Givaro). The purpose may be to develop or improve an algebraic algorithm or to use algebriac methods in an application.

The following are some project ideas. You may choose one of these or propose one of your own.

1. Extend LinBox capability for computation with dense symmetric matrices. For starters design a representation that stores only half the matrix (since $a_{i,j} = a_{j,i}$). Then make an algorithm for Cholesky decomposition (LDL^T). It should use a divide and conquer strategy so as to employ (fast) matrix multiplication.

2. Extend LinBox capability for computation with sparse symmetric matrices. Possibly use a representation that stores only half the matrix (since $a_{i,j} = a_{j,i}$), but surely use sparse storage format. This can be done specifically with the goal of supporting blackbox algorithms (optimize for multiplying this matrix by a vector or dense matrix). Then tune a blackbox or block blackbox algorithm for a linear algebra problem such as rank, determinant, or system solving.

3. Same as item above (sparse symmetric matrices), but make code for Cholesky factorization (symmetric version of LU factorization), and use a sparse storage format optimized for it.

4. Design a hybrid algorithm for multiplication of polynomial matrices, $A \in \mathbb{F}[x]^{n\times n}$. Such objects can be viewed as *matrix polynomials* (polynomials with coefficients in $\mathbb{F}^{n\times n}$) or as *polynomial matrices* (matrices whose entries are in $\mathbb{F}[x]$). Use a combination of fast methods for matrix multiplication (such as Strassen's algorithm) and fast methods for polynomial multiplication (such as FFT or Karatsuba's). Suppose the matrices are $n \times n$ with polynomial entries of degree d. Design for $d \ll n, n \ll d$, and $n \sim d$.

5. Build a system of "pipes" for LinBox. A pipe is a program that works with data on the standard input and output streams. Thus pipes work well with shell pipe, tee, and redirection commands. Of course the representation of numbers, vectors, polynomials, and matrices on these streams must be well specified. For example, pipe one might read in a dense matrix and write a (LU) factorization of it. Then pipe two takes in a factored matrix and writes it's determinant (or rank, or nullspace basis, or ...). Other pipes are for sparse matrices. Each pipe is designed for integer matrices or for a specified range of finite fields (such as GF2, or wordsize prime field, or ...). A code generator for pipes is a distinct possibility: Input: specify field or

ring ($\mathbb{Z}, \mathbb{F}(p), \mathbb{F}(p^e)$, etc.), problem (rank, det, solve, minpoly, etc.), matrix type (dense, sparse); Output: code for that problem on that matrix type over that ring/field.

6. Explore the problem of determining the spectral radius of an integer matrix. The spectral radius is the magnitude of the eigenvalue farthest from zero in the complex plane. LinBox can determine the minimal polynomial of a matrix. Sage, Flint, or Maple can factor it (perhaps – depending on degree). Can you determine a number that provably separates the largest occuring eigenvalue magnitude from the second largest magnitude? Probably it would be good to restrict to symmetric integer matrices so that the eigenvalues are real. This would be a tool for representing the largest eigenvalue exactly by it's minimal polynomial together with an inverval separating it from other roots of that polynomial.

7. Document fflas. Fflas is a package for dense linear algebra using the numerical BLAS, basic linear algebra subroutines. BLAS provides matrix muitiplication and other basic matrix nad vector arithmetic over floats, doubles, complex, and double complex types. Fflas extends that to finite fields, specifically prime fields with word size primes. It does so by adapting numeric blas to this exact-computation purpose. Fflas is used by LinBox and maintained in conjunction with LinBox. The current documentation is minimal. This project would involve building and using benchmarks to demonstrate the range of fields and field representations supported by fflas as well as expressing the capabillities clearly in documentation.

8. Also on the symmetric matrix theme: Can anything be made of Raleigh quotient iteration for symmetric matrix eigenvalue/vector computation in exact arithmetic? This is a thought inspired by the talk of Richard Tapia, a seminar on Oct 15.

Schedule

1. October 23 submit plan

2. November 20 submit progress report

3. December: present project results, submit code and written report.