The mathematical connections between graph theory and linear algebra are intimate and well known. The computational links between the two fields are also deep, extending all the way to the design of basic data structures and fundamental algorithms. In the first 50 years of this computational relationship, graphs served numerical linear algebra by enabling efficient sparse matrix computation. Recently, matrix computation has been returning the favor, particularly in the domain of parallel and high-performance algorithms.

I will describe two software systems for computing with large graphs and networks on parallel computers. The key to their performance and scaling is sparse matrix computation.

The Knowledge Discovery Toolbox (KDT) is a flexible open-source toolkit for implementing complex graph algorithms and executing them on high-performance machines. Analysts can use KDT by invoking existing KDT routines from Python; developers can write Python code using KDT’s computational primitives.

Those primitives are supplied by a parallel backend called CombBLAS, which is written in C++ with MPI for high performance and portability. I will briefly describe the algorithmic foundations of CombBLAS, which views graph algorithms as sparse matrix computations on semirings. A key primitive is “SpGEMM,” or generalized sparse matrix-sparse matrix multiplication. Our parallel SpGEMM algorithm, which uses a two-dimensional block data distribution with serial hypersparse kernels, scales to thousands of processors.

I will conclude with an advertisement for the Graph BLAS Forum, an open effort to standardize a set of middleware graph-algorithms primitives, which builds on many groups’ work on graph algorithms in the language of linear algebra.