Introduction to GraphBLAS
Tools and resources
Gábor Szárnyas
szarnyas@mit.bme.hu
Further reading and libraries
RESOURCES

Presentations and tutorials for learning GraphBLAS along with a link to a collection of pointers.

J.R. Gilbert:
*GraphBLAS: Graph Algorithms in the Language of Linear Algebra*, Seminar talk since 2014

S. McMillan and T.G. Mattson:
*A Hands-On Introduction to the GraphBLAS*, Tutorial at HPEC since 2018

A. Buluç:
*GraphBLAS: Concepts, algorithms, and applications*, Scheduling Workshop, 2019

M. Kumar, J.E. Moreira, P. Pattnaik:
*GraphBLAS: Handling performance concerns in large graph analytics*, Computing Frontiers 2018

List of GraphBLAS-related books, papers, presentations, posters, software, etc. [szarnyasg/graphblas-pointers](https://szarnyasg/graphblas-pointers)
THE LAGRAPHER LIBRARY

- Similar to the LAPACK library for BLAS
- Uses SuiteSparse:GraphBLAS
- Implementations of common algorithms
  - BFS, SSSP, LCC, PageRank, Boruvka
  - Triangle count, $k$-truss
  - CDLP (community detection using label propagation)
  - Weakly connected components, Strongly Conn. Comps (draft)
  - Betweenness centrality
  - Deep neural network

T.G. Mattson et al.: LAGraph: A Community Effort to Collect Graph Algorithms Built on Top of the GraphBLAS, GrAPL@IPDPS 2019
REQUIREMENTS BY GRAPH COMPUTATIONS

Libraries for linear-algebra based graph processing support the following features (prioritized):

1. Sparse matrices  
   For reasonable performance
2. Arbitrary semirings  
   For expressive power
3. Masking  
   A big reduction in complexity for some algos
4. Parallel execution  
   Constant speedup, ideally by #threads

Most libraries only satisfy requirement #1: Intel MKL, Eigen, Boost uBLAS, MTL4, Armadillo, NIST Sparse BLAS, GMM++, CUSP, Numpy

Exceptions are the Efficient Java Matrix Library (EJML) and Julia’s SparseArrays library, where arbitrary semirings can be used.
GRAPHBLAS PAPERS AND BOOKS

- **Standards for Graph Algorithm Primitives**
  - Position paper by 19 authors @ IEEE HPEC 2013

- **Novel Algebras for Advanced Analytics in Julia**
  - Technical paper on semirings in Julia @ IEEE HPEC 2013

- **Mathematical Foundations of the GraphBLAS**
  - Theory paper by 16 authors @ IEEE HPEC 2016

- **Design of the GraphBLAS C API**
  - Design decisions and overview of the C API @ GABB@IPDPS 2017

- **Algorithm 1000: SuiteSparse:GraphBLAS: graph algorithms in the language of sparse linear algebra**
  - Algorithms in the SuiteSparse implementation @ ACM TOMS 2019
BOOKS

- **Mathematics of Big Data**
  - Authored by Jananthan & Kepner, published by MIT Press in 2018
  - Generalizes the semiring-based approach for associative arrays
  - Contains reprints of papers, including the HPEC’16 paper above
  - Discusses D4M (Dynamic Distributed Dimensional Data Model)

- **Graph Algorithms in the Language of Linear Algebra**
  - Algorithms for connected components, shortest paths, max-flow, betweenness centrality, spanning tree, graph generation, etc.
  - Algorithms and data structure for fast matrix multiplication
  - Predates GraphBLAS: preliminary notation, no API usage
Annual events:

- **May**: IEEE IPDPS conference
  - GrAPL workshop (Graphs, Architectures, Programming and Learning), a merger of
    - GABB (Graph Algorithms Building Blocks)
    - GraML (Graph Algorithms and Machine Learning)
  - See [graphanalysis.org](http://graphanalysis.org) for previous editions
- **Sep**: IEEE HPEC conference
  - GraphBLAS BoF meeting
- **Nov**: IEEE/ACM Supercomputing conference
  - GraphBLAS Working Group
  - IA³ workshop (Workshop on Irregular Applications: Architectures and Algorithms)

**Blog**: [AldenMath](http://aldenmath.com) by Timothy Alden Davis
RedisGraph

- Graph database built on top of Redis with partial (but extending) support for the Cypher language
- Uses SuiteSparse:GraphBLAS for graph operations
- Preliminary benchmark results show good performance on traversal-heavy workloads

R. Lipman, T.A. Davis:
*Graph Algebra – Graph operations in the language of linear algebra*
RedisConf 2018

R. Lipman:
*RedisGraph internals*
RedisConf 2019
GRAPHBLAS IMPLEMENTATIONS

- SuiteSparse:GraphBLAS
  - v1.0.0: Nov 2017 – sequential
  - v3.0.1: July 2019 – parallel
  - v4.0.0: WIP – bitmap representation

- IBM GraphBLAS
  - Complete implementation in C++, released in May 2018
  - Concise but sequential

- GBTL (GraphBLAS Template Library): C++
  - v1.0: parallel but no longer maintained
  - v2.0, v3.0: sequential

- GraphBLAST: GPU implementation, based on GBTL
GRAPHULO

- Build on top of the Accumulo distributed key-value store
- Written in Java
- Focus on scalability

V. Gadepally et al.: Graphulo: Linear Algebra Graph Kernels for NoSQL Databases, GABB@IPDPS 2015
COMBBLAS: COMBINATORIAL BLAS

- “an extensible distributed memory parallel graph library offering a small but powerful set of linear algebra primitives”
- Not a GraphBLAS implementation but serves as an incubator for new ideas that may later find their way into GraphBLAS
- Scales to 250k+ CPU cores
  - Used on supercomputers such as Cray

Learning GraphBLAS
LEARNING GRAPHBLAS

- Challenging to get started – need to have an understanding of:
  - Linear algebra (semirings, matrix multiplication, etc.)
  - Connection between linear algebra and graph operations (intuition)
  - Sizeable C API

- 200+ slides presenting algorithms in GraphBLAS
  - BFS variants, PageRank, Triangle count, $k$-truss
  - Community detection using label propagation
  - Luby’s maximal independent set algorithm
  - Computing connected components on an overlay graph
  - etc.
PYGRAPHBLAS: PYTHON WRAPPER #1

- A Python wrapper
- See example code for SSSP and triangle count
- Close to pseudo-code
- Jupyter support

Added benefit: both Python and GraphBLAS use 0-based indexing

```python
def sssp(matrix, start):
    v = Vector.from_type(matrix.gb_type, matrix.nrows)
    v[start] = 0

    with min_plus, Accum(min_int64):
        for _ in range(matrix.nrows):
            w = Vector.dup(v)
            v @= matrix
            if w == v:
                break
    return v

def sandia(A, L):
    return L.mxm(L, mask=L).reduce_int()

sandia(M, M.tril())
```

>michelp/pygraphblas
GRBLAS: PYTHON WRAPPER #2

- Goal: one-to-one mapping of the GraphBLAS API
  - less Pythonic
  - comes with a Conda package
  - compiles user-defined functions to C
Tradeoffs
TRADEOFFS OF GRAPHBLAS

- Challenging to get started – need to have an understanding of:
  - Linear algebra (semirings, matrix multiplication, etc.)
  - Connection between linear algebra and graph operations
  - The GraphBLAS C API

- Some additional building blocks required
  - Sorting
  - Permutation
  - Dynamically-sized values (e.g. to store paths)

- Currently the only high-performance CPU implementation is SuiteSparse:GraphBLAS
  - Written in C11
  - Some features (e.g. the _Generic macro) are incompatible with C++.
TRADEOFFS OF GRAPHBLAS

- Updating CSR/CSC matrices is slow
  - SuiteSparse mitigates this to some extent by allowing incremental build before using the matrix but it still needs to rebuild the matrix before major operations such as mxv, mxm
  - But: GraphBLAS objects are opaque and alternative formats can be used
    - Hornet [HPEC’18]
    - faimGraph [SC’18]
    - STINGER [HPEC’12]