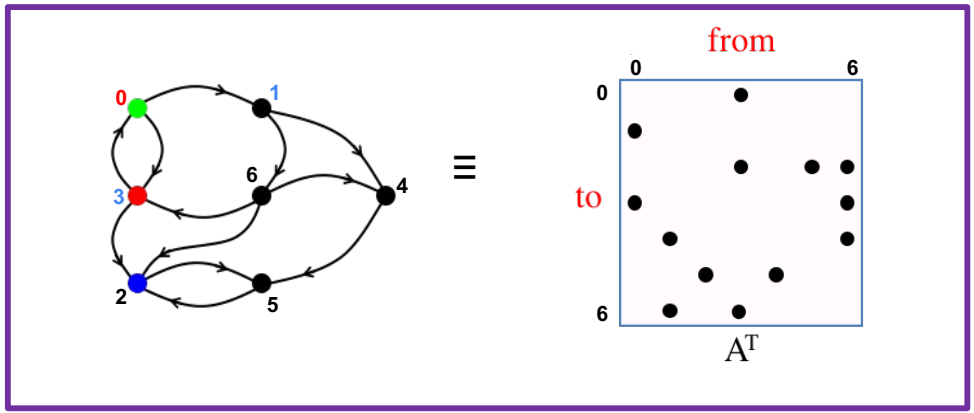
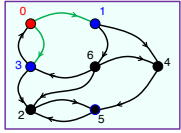


GraphBLAS: Updates

Jose Moreira and Tim Mattson,
with help from the rest of the Math Specification team
(Scott McMillan, Ben Brock, Hayden Jananthan, and Raye Kimmerer)

A copy of these slides will appear on our website … www.graphlas.org

The GraphBLAS Community



- **GraphBLAS Forum mailing list:** Graphblas@lists.lbl.gov
  - Hosted by LBL (mailto:abuluc@lbl.gov)
  - Join the GraphBLAS Forum by joining the list **Current membership is 133 (as of Nov 2025)**

- **GraphBLAS forum website:** http://graphblas.org
  - Lists workshops and conferences
  - Link to the latest C API Specification and the original “Math Document”
  - Lists teams developing implementations

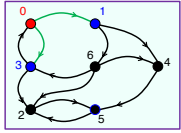
- **Monthly GraphBLAS Forum teleconference:**
  - Second Friday of every month, 12pm Eastern Time
  - Send email to Jeremy Kepner to receive the calendar invite.

- **Monthly LAGraph+SuiteSparse/GraphBLAS teleconference:**
  - Second Wednesday of every month, 2pm Central Time
  - Send email to Tim Davis(davis@tamu.edu) to receive the calendar invite.

Updates from the GraphBLAS Ecosystem

GraphBLAS Implementations

This is our standard list of Implementations



C Libraries:

- SuiteSparse:GraphBLAS library (Texas A&M): First fully conforming GraphBLAS release
  - http://faculty.cse.tamu.edu/davis/GraphBLAS.html
- GraphBLAS C (IBM): the second fully conforming release
  - https://github.com/IBM/ibmgraphblas

C++ Libraries:

- RGRI: C++ GraphBLAS Reference implementation (under development)
  - https://github.com/GraphBLAS/rgri
- ALP/GraphBLAS (Huawei): Non-blocking C++ GraphBLAS with a distributed-memory/hybrid parallel backend
  - https://github.com/Algebraic-Programming/ALP
- GBTL: C++ GraphBLAS Template Library (CMU/SEI/IU/PNNL):
  - https://github.com/cmu-sei/gbtl
- GraphBLAST: C++ GraphBLAS for GPUs (UC Davis)
  - https://github.com/gunrock/graphblast

Python bindings:

- PyGB: A python wrapper around GBTL (UW/PNNL/CMU)
  - https://github.com/jessecoleman/gbtl-python-bindings
- pygraphblas: A python wrapper around SuiteSparse
  - https://github.com/michelp/pygraphblas
- python-graphblas: Anaconda's python wrapper around SuiteSparse
  - https://github.com/python-graphblas/python-graphblas
- PyD4M: Python wrapper around D4M … includes GraphBLAS (under development)
  - https://github.com/Accla/D4M.py

forGraphBLASGo: A Go binding for SuiteSparse

- https://pkg.go.dev/github.com/intel/forGraphBLASGo

Julia wrapper around SuiteSparse

- SuiteSparseGraphBLAS.jl

Matlab wrappers around SuiteSparse

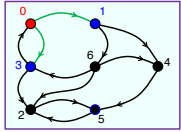
- https://github.com/DrTimothyAldenDavis/GraphBLAS

pggraphblas: A PostgreSQL wrapper around SuiteSparse

- https://github.com/OneSparse/OneSparse.git

GraphBLAS Implementations

Let's partition the list to show a bit more about their status and relationships



Notable Active Research Projects

RGRI: C++ GraphBLAS Reference implementation

- https://github.com/GraphBLAS/rgri

ALP/GraphBLAS (Huawei): Non-blocking C++ GraphBLAS with a distributed-memory/hybrid parallel backend

- https://github.com/Algebraic-Programming/ALP

Nonblocking Julia GraphBLAS with Multistage Programming

- https://arxiv.org/abs/2509.14211

Inactive Projects*

PyGB: A python wrapper around GBTL (UW/PNNL/CMU)

- https://github.com/jessecoleman/gbtl-python-bindings

pygraphblas: A python wrapper around SuiteSparse

- https://github.com/michelp/pygraphblas

GraphBLAS C (IBM): the second fully conforming release

- https://github.com/IBM/ibmgraphblas

GBTL: C++ GraphBLAS Template Library (CMU/SEI/IU/PNNL):

- https://github.com/cmu-sei/gbtl

GraphBLAST: C++ GraphBLAS for GPUs (UC Davis)

- https://github.com/gunrock/graphblast

Matlab wrappers around SuiteSparse

- https://github.com/DrTimothyAldenDavis/GraphBLAS

pggraphblas: A PostgreSQL wrapper around SuiteSparse

- https://github.com/OneSparse/OneSparse.git

python-graphblas: Anaconda's python wrapper around SuiteSparse

- https://github.com/python-graphblas/python-graphblas

forGraphBLASGo: A Go binding for SuiteSparse

- https://pkg.go.dev/github.com/intel/forGraphBLASGo

Julia wrapper around SuiteSparse

- SuiteSparseGraphBLAS.jl

SuiteSparse supports a wide cross-language GraphBLAS ecosystem

SuiteSparse:GraphBLAS C library (Texas A&M):

- http://faculty.cse.tamu.edu/davis/GraphBLAS.html

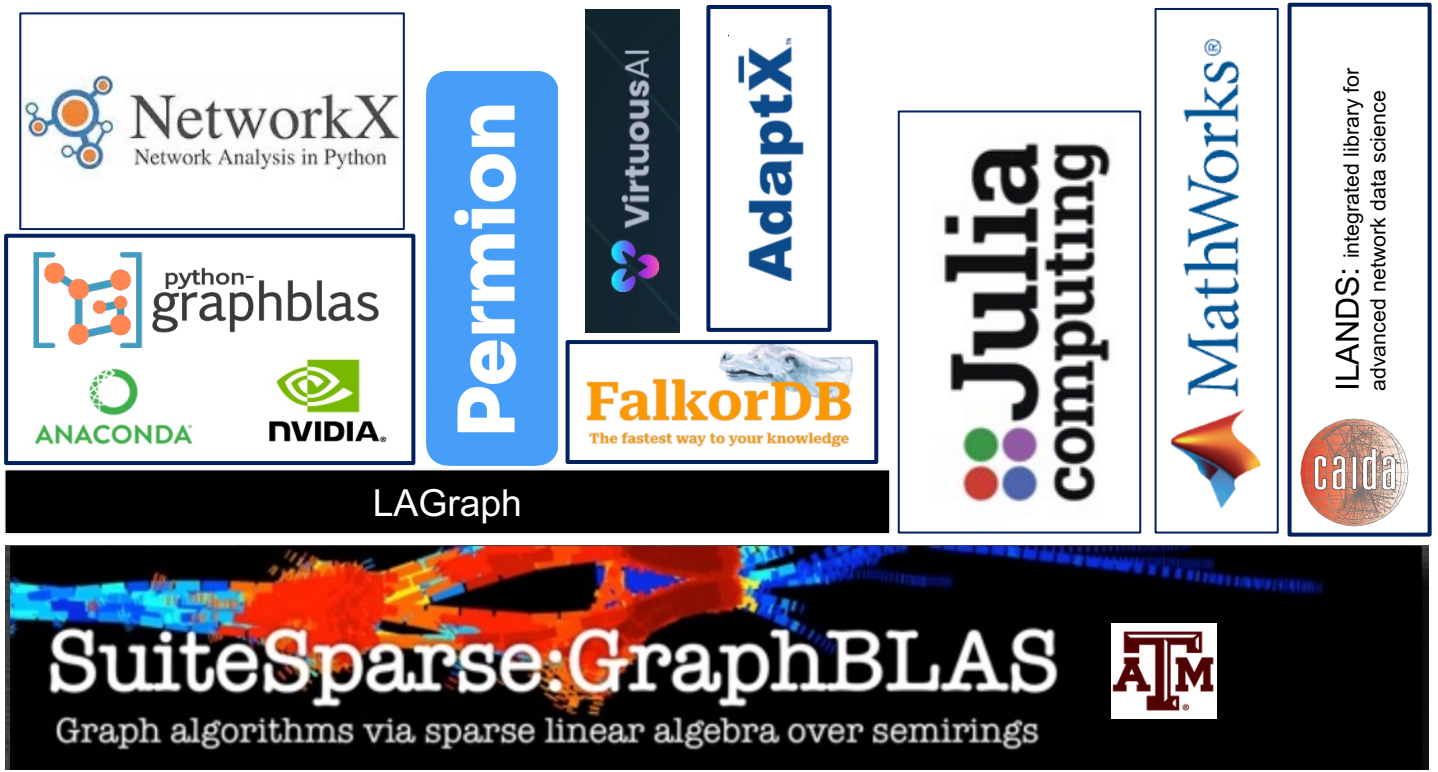
PyD4M: Python wrapper around D4M … includes GraphBLAS

- https://github.com/Accla/D4M.py

*A project is inactive if its git repository has not been updated in 4 or more years.

The GraphBLAS is more than a set of specifications

It appears inside Applications, SW-frameworks, and Products



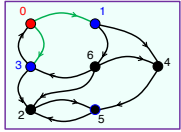
The GraphBLAS C API

https://www.falkordb.com https://people.engr.tamu.edu/davis/GraphBLAS.html (Julia, Mathworks) https://www.caida.org/funding/ccri-ilands/

https://www.youtube.com/watch?v=FCwTMQLH2Kc (NetworkX, python-graphblas) https://ieeexplore.ieee.org/document/9622790

GraphBLAS Specifications

GraphBLAS Implementations



C Libraries:

- SuiteSparse:GraphBLAS library (Texas A&M): First fully conforming GraphBLAS release
  - http://faculty.cse.tamu.edu/davis/GraphBLAS.html
- GraphBLAS C (IBM): the second fully conforming release
  - https://github.com/IBM/ibmgraphblas

C++ Libraries:

- RGRI: C++ GraphBLAS Reference implementation (under development)
  - https://github.com/GraphBLAS/rgri
- ALP/GraphBLAS (Huawei): Non-blocking C++ GraphBLAS with a distributed-memory/hybrid parallel backend
  - https://github.com/Algebraic-Programming/ALP
- GBTL: C++ GraphBLAS Template Library (CMU/SEI/IU/PNNL):
  - https://github.com/cmu-sei/gbtl
- GraphBLAST: C++ GraphBLAS for GPUs (UC Davis)
  - https://github.com/gunrock/graphblast

Python bindings:

- PyGB: A python wrapper around GBTL (UW/PNNL/CMU)
  - https://github.com/jessecoleman/gbtl-python-bindings
- pygraphblas: A python wrapper around SuiteSparse
  - https://github.com/michelp/pygraphblas
- python-graphblas: Anaconda's python wrapper around SuiteSparse
  - https://github.com/python-graphblas/python-graphblas
- PyD4M: Python wrapper around D4M … includes GraphBLAS (under development)
  - https://github.com/Accla/D4M.py

forGraphBLASGo: A Go binding for SuiteSparse

- https://pkg.go.dev/github.com/intel/forGraphBLASGo

Julia wrapper around SuiteSparse

- SuiteSparseGraphBLAS.jl

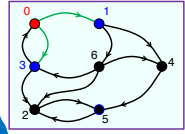
Matlab wrappers around SuiteSparse

- https://github.com/DrTimothyAldenDavis/GraphBLAS

pggraphblas: A PostgreSQL wrapper around SuiteSparse

- https://github.com/OneSparse/OneSparse.git

GraphBLAS Implementations



These followed a specification (C GraphBLAS spec)

C Libraries:

- SuiteSparse:GraphBLAS library (Texas A&M): First fully conforming GraphBLAS release
  - http://faculty.cse.tamu.edu/davis/GraphBLAS.html
- GraphBLAS C (IBM): the second fully conforming release
  - https://github.com/IBM/ibmgraphblas

C++ Libraries:

- RGRI: C++ GraphBLAS Reference implementation (under development)
  - https://github.com/GraphBLAS/rgri
- ALP/GraphBLAS (Huawei): Non-blocking C++ GraphBLAS with a distributed-memory/hybrid parallel backend
  - https://github.com/Algebraic-Programming/ALP
- GBTL: C++ GraphBLAS Template Library (CMU/SEI/IU/PNNL):
  - https://github.com/cmu-sei/gbtl
- GraphBLAST: C++ GraphBLAS for GPUs (UC Davis)
  - https://github.com/gunrock/graphblast

These did not follow a specification

Python bindings:

- PyGB: A python wrapper around GBTL (UW/PNNL/CMU)
  - https://github.com/jessecoleman/gbtl-python-bindings
- pygraphblas: A python wrapper around SuiteSparse
  - https://github.com/michelp/pygraphblas
- python-graphblas: Anaconda's python wrapper around SuiteSparse
  - https://github.com/python-graphblas/python-graphblas
- PyD4M: Python wrapper around D4M … includes GraphBLAS (under development)
  - https://github.com/Accla/D4M.py

forGraphBLASGo: A Go binding for SuiteSparse

- https://pkg.go.dev/github.com/intel/forGraphBLASGo

Julia wrapper around SuiteSparse

- SuiteSparseGraphBLAS.jl

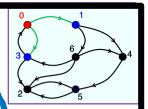
Matlab wrappers around SuiteSparse

- https://github.com/DrTimothyAldenDavis/GraphBLAS

pggraphblas: A PostgreSQL wrapper around SuiteSparse

- https://github.com/OneSparse/OneSparse.git

GraphBLAS Implementations



These followed a specification (C GraphBLAS spec)

C Libraries:

- SuiteSparse:GraphBLAS library (Texas A&M): First fully conforming GraphBLAS release
  - http://faculty.cse.tamu.edu/davis/GraphBLAS.html
- GraphBLAS C (IBM): the second fully conforming release
  - https://github.com/IBM/ibmgraphblas

C++ Libraries:

- RGRI: C++ GraphBLAS Reference implementation (under development)
  - https://github.com/GraphBLAS/rgri
- ALP/GraphBLAS (Huawei): Non-blocking C++ GraphBLAS with a distributed-memory/hybrid parallel backend
  - https://github.com/Algebraic-Programming/ALP
- GBTL: C++ GraphBLAS Template Library (CMU/SEI/IU/PNNL):
  - https://github.com/cmu-sei/gbtl
- GraphBLAST: C++ GraphBLAS for GPUs (UC Davis)
  - https://github.com/gunrock/graphblast

These did not follow a specification

Python bindings:

- PyGB: A python wrapper around GBTL (UW/PNNL/CMU)
  - https://github.com/jessecoleman/gbtl-python-bindings
- pygraphblas: A python wrapper around SuiteSparse
  - https://github.com/michelp/pygraphblas
- python-graphblas: Anaconda's python wrapper around SuiteSparse
  - https://github.com/python-graphblas/python-graphblas
- PyD4M: Python wrapper around D4M … includes GraphBLAS (under development)
  - https://github.com/Accla/D4M.py

forGraphBLASGo: A Go binding for SuiteSparse

- https://pkg.go.dev/github.com/intel/forGraphBLASGo

Julia wrapper around SuiteSparse

- SuiteSparseGraphBLAS.jl

Matlab wrappers around SuiteSparse

- https://github.com/DrTimothyAldenDavis/GraphBLAS

pggraphblas: A PostgreSQL wrapper around SuiteSparse

- https://github.com/OneSparse/OneSparse.git

Perhaps there's a lesson here for us to learn?

Most developers don't care about specifications these days ….. The implementation **IS** the specification.

This approach will work as long as all the various implementations across languages support the same mathematics.

We need a mathematical definition of the GraphBLAS … so all the implementations of GraphBLAS will hopefully support the same mathematics and algorithms can move across implementations/languages

We need to create a Math Spec

GraphBLAS Math Spec

- The math spec project started back in 2023. It has proven much more difficult than we anticipated.
- Our goal was to finish by SC'25. We didn't make it. **We will be done before SC'26.**

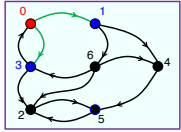
Mathematical specification of GraphBLAS

BENJAMIN BROCK, Intel Corporation, USA
HAYDEN JANANTHAN
RAYE KIMMERER
TIM MATTSON, University of Bristol, USA
SCOTT MCMILLAN, Software Engineering Institute, Carnegie Mellon University, USA
JOSÉ MOREIRA, IBM Research, USA

- Our major challenge … separating math from implementation.
  - Much of the GraphBLAS was only nailed down as we developed the C specification (e.g. masks, replace vs merge semantics, and more). How do these C implementation details map back onto the underlying Math?
  - It is difficult to take concepts well grounded mathematically (real arithmetic and associativity) and project them into types associated with an implementation (IEEE-754 arithmetic is not associative).

- A key result from our Math Spec work will be common notation algorithm developers can use … confident their algorithms will map across GraphBLAS language bindings.
  - We are starting from the notation developed by Gabor Szarnyas for his work on LAGraph.

Math Spec goals and remaining challenges



- Different programming languages, different platforms, and different application domains have good reasons to NOT implement certain features of the Math spec.
  - We need to define a more nuanced understanding of what it means to “conform to the Math Spec” … perhaps by defining a core set of features all implementations must support and then different levels of conformance for additional sets of features.

- In addition to traditional binary and unary operators, we sometimes need operators that work with the indices of the values in a sparse matrix or vector. How should we handle this in the Math Spec?
  - We will add extended domains that replaces values with tuples (values and their indices). A casting operator will transform a domain into an extended domain (with indices). These extended domains will let us utilize indexed operators anywhere an operator appears in the Math Spec.

- The Math Spec must define a readable and intuitive notation for algorithm designers. Past work on LAGraph has given us a great starting point, but it is still very much a work in progress.

- As we consider a wider range of applications, we’ve found that we need to go beyond Semirings. How do we add additional algebras, such as fields, to the GraphBLAS?

Conclusion

- The GraphBLAS ecosystem is in solid shape … though frankly, without SuiteSparse GraphBLAS we'd be in trouble.
- The Math Spec will support mathematical consistency for a world where the implementation is the specification.
  - It is proving much more difficult to create than we anticipated.
- … But just wait for SC'26. We'll have it done before then and have a detailed presentation about it at SC'26.

Tim's Greenlandic skin-on-frame kayak in the middle of Budd Inlet during a negative tide