STING: Software for Analysis of Spatio-Temporal Interaction Networks and Graphs

http://www.cc.gatech.edu/stinger/

Dr. David A. Bader
Dr. Jason Riedy
Dr. Henning Meyerhenke[a]
David Ediger
Rob McColl
Oded Green
Anita Zakrzewska
Rohit Banga

[a]Karlsruhe Institute of Technology

1. Motivation

Many interesting problems today can be formulated as dynamic spatio-temporal graph problems:

- monitoring when previously separate groups merge,
- tracking communities within social networks as interactions are added or relationships removed, and
- identifying bridges between communities or those who switch allegiances over time.

The development of algorithms and codes for large-scale, dynamic graph problems proceeds without a common core of data structures or software frameworks. Cross-evaluation and adoption of new techniques is time-consuming both for programmers modifying algorithms and systems transforming data structures. STING's design aims to reduce inefficiencies in time, space, and productivity with a canonical graph representation and pluggable analysis kernels.

2. STINGER: Common Data Structure

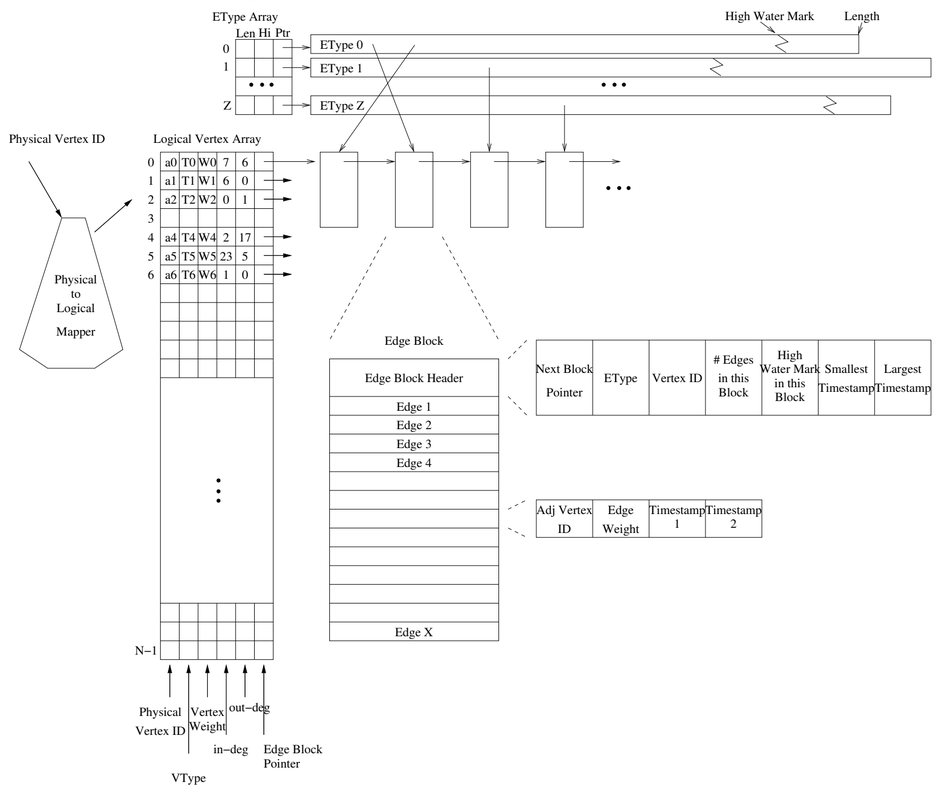
STINGER provides the central graph store accessed by multiple analysis kernels.

Portability Algorithms written for STINGER can be ported to multiple architectures and translated between multiple languages.

Productivity STINGER provides a common abstract data structure that the large graph community can use to quickly evaluate and extend each others' research developments, much like common dense and sparse array structures in linear algebra.

Performance No single data structure is optimal for every graph algorithm. STINGER provides a sensible data structure that can run most algorithms well.

There should be little performance reduction using STINGER when compared with other general data structures across a broad set of typical graph algorithms. STINGER assumes a globally-addressable memory space and supports both sequential and parallel algorithms.



3. STING: Software Framework

Currently targeting both OpenMP on Intel multi-cores and the Cray XMT environments, STING is a multi-platform software framework for developing both static and dynamic graph analysis kernels.
Example analysis kernels include the following:

Connected components Maintain a component label for each vertex as edges are inserted and removed. Heuristics tuned for common social network structure quickly rule out nearly all edge removals that do not alter the structure.

Clustering coefficients Compute and update per-vertex clustering coefficients and the graph's global clustering coefficient. The clustering coefficient is the ratio of the number of triangles (three connected edges) to the number of pairs (two connected edges).

Modularity clustering Refine a global partitioning to maintain a large modularity within each partition. A modular partition is one with more edges inside each partition than expected by random chance given the degree distribution. (in progress)

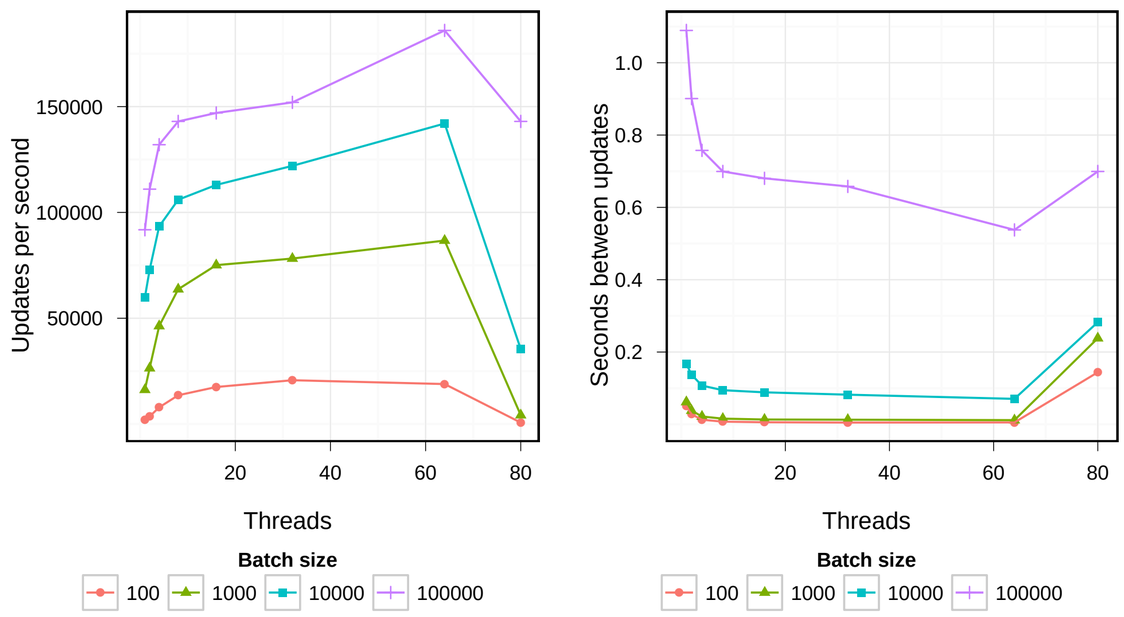
Ongoing work is evaluating a STING framework for PGAS programming environments like UPC and X10 and also on GPGPU accelerators using NVIDIA's CUDA.

4. Example Performance: Connected Components

Task Monitor the connected components while inserting and removing edges in batches.

Graph small R-MAT graph with 16.7 million vertices and over 250 million edges.

Platform Intel-based server platform located at Georgia Tech with four ten-core Intel Xeon E7-8870 processors running at 2.40GHz with 30MiB of L3 cache per processor. The processors support HyperThreading, so the 40 physical cores appear as 80 logical cores. This server, mirasol, is ranked #17 in the November 2011 Graph 500 list and is equipped with 256 GiB of 1 067 MHz DDR3 RAM.



- Batching the edge insertions and removals provides opportunities for parallel execution and reduce overhead.
- The small graph and relatively small batch sizes limit total performance.
- Increasing the batch size improves total performance but limits the rate at which we update the connected components. Which is more appropriate depends on a particular application.
- We plan on supporting kernel cooperation so other kernels can use the connected component mapping.

5. Acknowldegements

This work was supported in part by the Pacific Northwest National Lab (PNNL) Center for Adaptive Supercomputing Software for MultiThreaded Architectures (CASS-MT), NSF Grant CNS-0708307, and the Intel Labs Academic Research Office for the Parallel Algorithms for Non-Numeric Computing Program. We thank PNNL and the Swiss National Supercomputing Centre for providing access to Cray XMT systems.

6. References

- D.A. Bader, J. Berry, A. Amos-Binks, D. Chavarria-Miranda, C. Hastings, K. Madduri, and S.C. Poulos, "STINGER: Spatio-Temporal Interaction Networks and Graphs (STING) Extensible Representation," Technical Report, May 8, 2009.
- D. Ediger, K. Jiang, J. Riedy, and D.A. Bader, "Massive Streaming Data Analytics: A Case Study with Clustering Coefficients," 4th Workshop on Multithreaded Architectures and Applications (MTAAP), Atlanta, GA, April 23, 2010.
- D. Ediger, J. Riedy, H. Meyerhenke, and D.A. Bader, "Tracking Structure of Streaming Social Networks," 5th Workshop on Multithreaded Architectures and Applications (MTAAP), Anchorage, AK, May 20, 2011.

15th SIAM Conference on Parallel Processing for Scientific Computing

Georgia Institute of Technology