1. STINGER: Graph analysis for a dynamic world

STINGER focuses on updating analyses from streaming data:
- Updating reduces data access and motion,
- batching trades off between throughput and latency, and
- aggressive parallelism maintains high rates of change.

STINGER’s two primary pieces:
- a scalable, high performance in-memory dynamic graph data structure, and
- a software framework (sometimes called STING) for building and executing graph analysis kernels.

2. What can STINGER represent?

- Nearly any set of relationships (semantic graphs)
  - Healthcare, social networks, computer security, business intelligence, systems biology, power grid, travel networks...
- Example: Twitter
  - Vertex types: Users, hash tags, tweets as vertex types
  - Edge types: Authorship, retweet, mentions, follower/ee
- Example: Computer network
  - Vertex types: Users, PCs, printers, emails, URLs, files, ...
  - Edge types: Email alias/from/to, logon/off, print, IM, ...

3. What are (some of) STINGER’s capabilities?

- Trading high throughput for lower latency with batch sizes.
- Maintaining connected components in a graph of half a billion edges:
  - Up to 1.26 million updates per second,
  - 137× faster than recomputing.
- Monitoring locally optimal communities at real-world rates:
  - Built on scalable agglomerative modularity maximization.
  - Over one million updates per second on tens of millions of edges, or
  - Latency near 10^{-3}s with one to ten thousand updates per second.
- From 4× to 3500× speed up over recomputation.

4. Data structure performance

![Graph showing data structure performance](image1.png)

5. STINGER analysis kernels

- Streaming clustering coefficients, local and global
- Streaming connected components
- Streaming community monitoring
- Streaming betweenness centrality
- Agglomerative clustering / community detection
- Seed-set expansion (growing a local community)
- K-core extraction
- Breadth-first search
  - Edge deletions currently produce un-intuitive results.
  - For relatively small graphs / subgraphs.

6. Dynamic communities: Giving a better view

Taking a community view is better than a simple component / BFS view. Consider a limited view around @youtube in a March, 2013 twitter who-mentions-whom graph. Expanding / maintaining the community provides more structure in the same number of vertices.

![Graph showing community structure](image2.png)

7. STINGER’s future

- Convert the workflow to separate processes for reliability.
- Support composition of kernels and subgraph selection.
- Optimize power as well as performance.
- Study the effects of errors, and learn what to forget.
  - At some point, memory fills. Which edges and vertices can be over-written? Which analysis kernels care?

8. Authors and Contributors

- David A. Bader,
- Jason Riedy,
- David Ediger,
- Robert McColl,
- Oded Green,
- Anita Zakrzewska,
- James Fairbanks,
- Jason Poovey,
- Karl Jiang, and
- feedback from users in industry, government, academia...

See [http://stingergraph.com](http://stingergraph.com) for software and publications.

Funding acknowledgments: NSF, DARPA, DoD, Intel, IBM