1. GRATEFUL's goals

- GRATEFUL providing mission-critical analysis of semantic data where needed:
  - as close to the data and data use as possible
  - using the least power necessary to enable rapid decisions with reliable analysis.
- new algorithms providing new, fundamental capabilities under run-time power and environmental constraints, while coping with real-world uncertainties
- in the graph-structured data that occurs in many industrial and scientific application areas:
  - computer security,
  - intelligence analysis,
  - logistic planning

2. PPA Baseline Kernels

- GRATEFUL will demonstrate improvement over the kernels, which are baseline, state-of-the-art algorithms.
  - Single source shortest paths
  - Community detection
  - Seed set expansion

3. 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.

4. Dynamic PageRank

- PageRank is a well-established metric that gives each vertex a score based on visit frequencies during a random walk on the graph.
- Dynamic PageRank incrementally updates when the graph changes resulting in 30%-90% less data movement.
- Leaving out contributions from low-ranked vertices further reduces edge traversals.

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

- Can represent nearly any set of relationships (semantic graphs)
  - Healthcare, social networks, computer security, business intelligence, systems biology, power grid...
- Scalable from hand-held devices to multi-terabyte servers.
- 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:
  - Over one million updates per second on tens of millions of edges
  - latency near $10^{-3}$
  - From 4× to 3500× speed up over recomputation.
- Future:
  - Convert the workflow to separate processes for reliability.
  - Support composition of kernels and subgraph selection.
  - Tunable power optimizations as well as performance.
  - Study the effects of errors, and learn what to forget.

6. Dynamic Betweenness Centrality on the GPU

- Importance of work efficiency

- Speedup over GPU recomputation and CPU streaming

Adam McLaughlin and David A. Bader: ‘A Work Efficient Implementation of Dynamic Betweenness Centrality on the GPU.” [H]”

7. Tracking Dynamic Communities

- A new algorithm tracks community changes over time and detects community-switching vertices. The additional running time on top of Louvain community detection is less than 10% on average.
- The set of interesting vertices is reduced on average to 11%-30%.

8. Personnel

- David A. Bader
- Jason Riedy
- Anita Zakrzewska
- Adam McLaughlin

See http://www.cc.gatech.edu/stinger/ for software and publications.