Carnegie Mellon

Electrical & Computer ENGINEERING

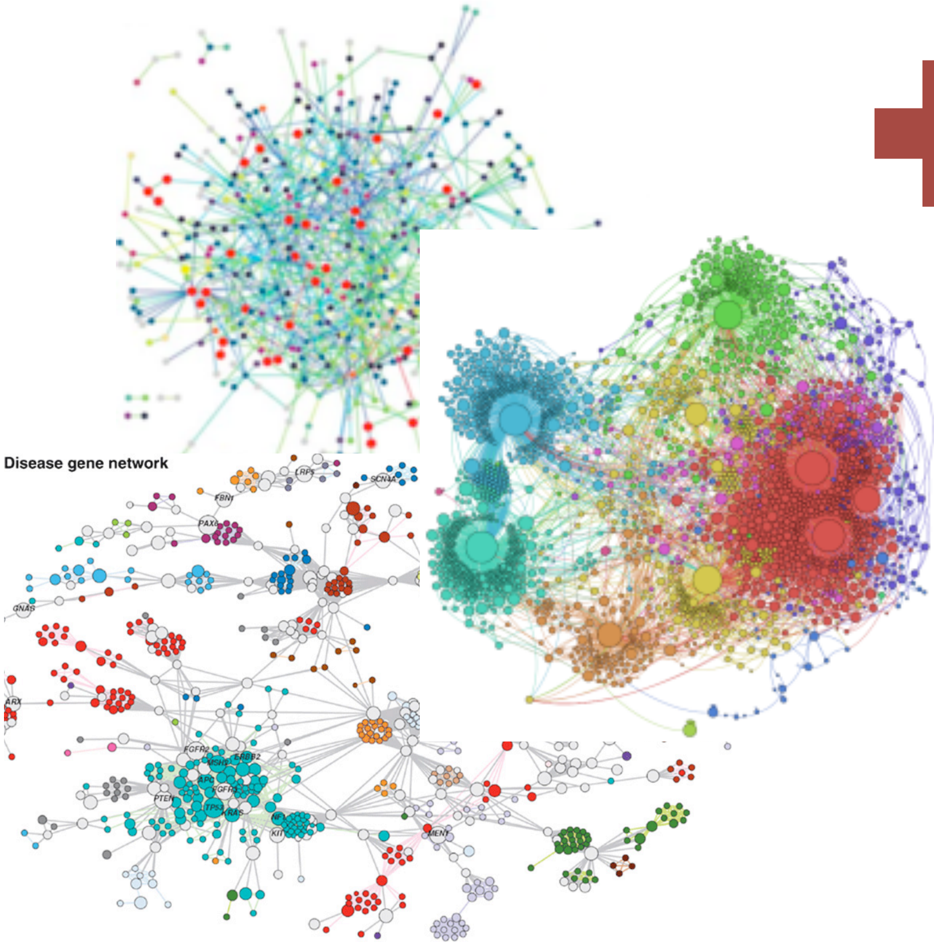
# A Family of Provably Correct Algorithms for Exact Triangle Counting

Matthew Lee, Tze Meng Low

Correctness 2017

# Motivation

## Graphs are everywhere




https://en.wikipedia.org/wiki/Bioinformatics
http://www.mkbergman.com/968/a-new-best-friend-gephi-for-large-scale-networks/
https://www.cs.umd.edu/research/projects/16672

## Correctness in HPC

$$C = AA^T + C$$

# FLAME

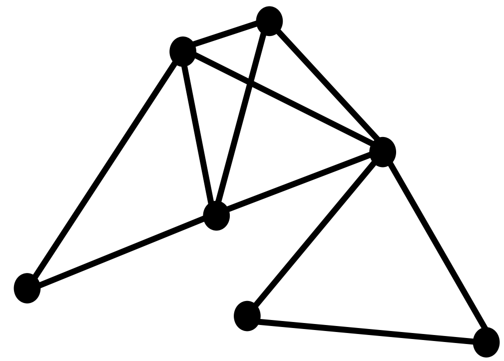
- Formal Linear Algebra Methods Environment
- Main components
  - 8-step algorithm derivation methodology
    - Requires loop invariants as input
    - Produces both algorithm and proof of correctness
  - Index-free APIs for implementing derived algorithms
- libFLAME
  - Formally derived common LAPACK functionality
  - High performance

**Key Idea: Find Loop Invariants for Graph Algorithms**

# Specifying the problem

- Number of triangles in a graph

$$\Delta = \frac{1}{6}\Gamma(A^3) \quad A \rightarrow \begin{pmatrix} A_{TL} & A_{TR} \\ A_{TR}^T & A_{BR} \end{pmatrix}$$

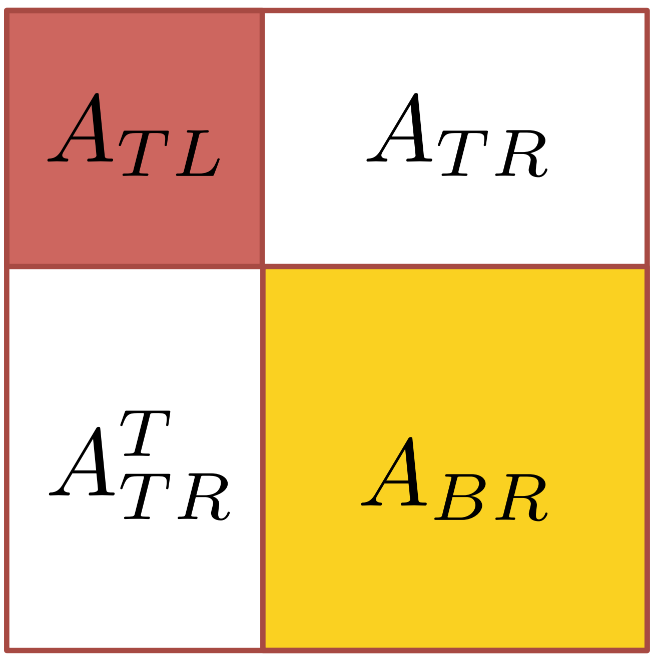


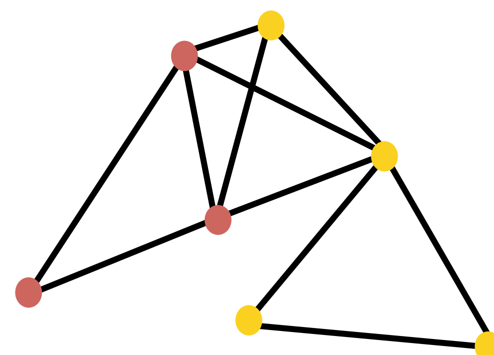
$$\Delta = \frac{1}{6}\Gamma(A_{TL}^3) + \frac{1}{2}\Gamma(A_{TR}^T A_{TL} A_{TR}) + \frac{1}{2}\Gamma(A_{TR} A_{BR} A_{TR}^T) + \frac{1}{6}\Gamma(A_{BR}^3)$$

**Partitioned Matrix Expression (PME)**

# Partitioned Matrix Expression (PME)

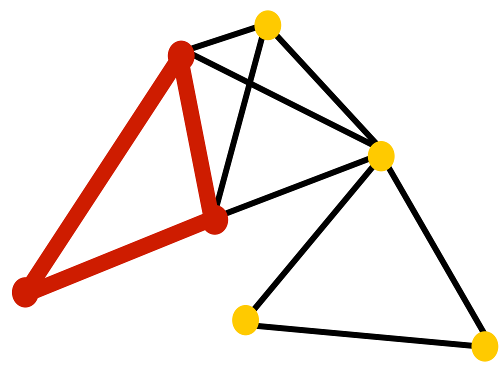
$$\Delta = \frac{1}{6}\Gamma(A_{TL}^3) + \frac{1}{2}\Gamma(A_{TR}^T A_{TL} A_{TR}) + \frac{1}{2}\Gamma(A_{TR} A_{BR} A_{TR}^T) + \frac{1}{6}\Gamma(A_{BR}^3)$$



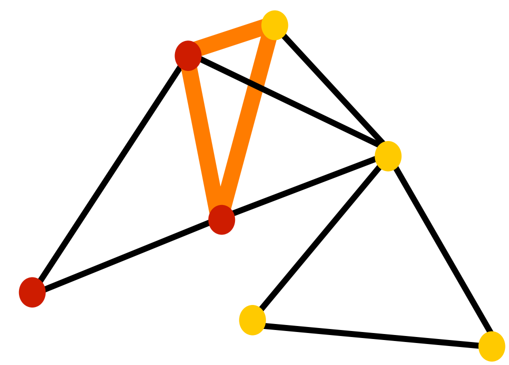



# Different Types of Triangles

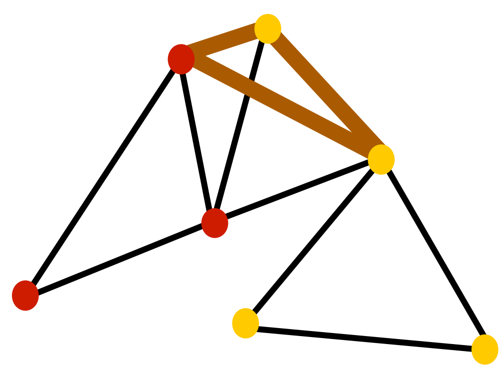
$$\Delta = \frac{1}{6}\Gamma(A_{TL}^3) + \frac{1}{2}\Gamma(A_{TR}^T A_{TL} A_{TR}) + \frac{1}{2}\Gamma(A_{TR} A_{BR} A_{TR}^T) + \frac{1}{6}\Gamma(A_{BR}^3)$$



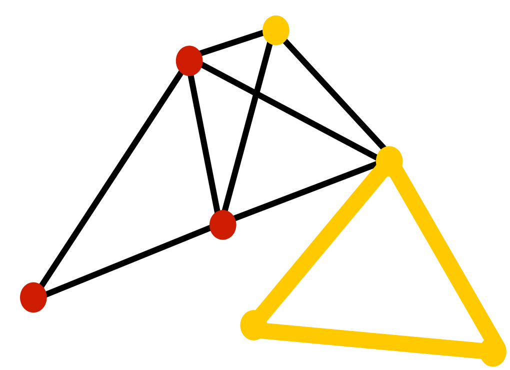
Category I



Category II



Category III



Category IV

# Finding Loop Invariants

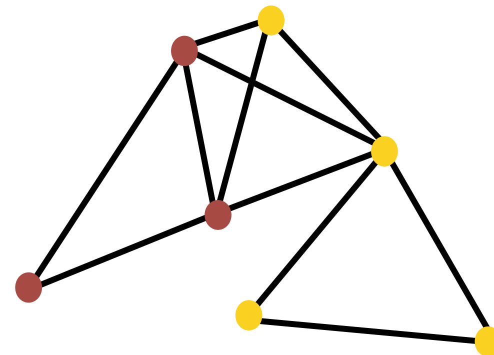
- Assertion that must be true at start and end of every iteration

```
while ( G ) {



}
```

# Finding Loop Invariants

- Assertion that must be true at start and end of every iteration

```
while ( G ) {

}
```



**Loop Invariant:**
**# of triangles that have been computed**

# Loop Invariants from PME

$$\Delta = \underbrace{\frac{1}{6}\Gamma(A_{TL}^3)}_{\text{I}} + \underbrace{\frac{1}{2}\Gamma(A_{TR}^T A_{TL} A_{TR})}_{\text{II}} + \underbrace{\frac{1}{2}\Gamma(A_{TR} A_{BR} A_{TR}^T)}_{\text{III}} + \underbrace{\frac{1}{6}\Gamma(A_{BR}^3)}_{\text{IV}}$$

$$\Delta = \frac{1}{6}\Gamma(A_{TL}^3) + {\color{lightgray}\frac{1}{2}\Gamma(A_{TR}^T A_{TL} A_{TR}) + \frac{1}{2}\Gamma(A_{TR} A_{BR} A_{TR}^T) + \frac{1}{6}\Gamma(A_{BR}^3)}$$

$$\Delta = \frac{1}{6}\Gamma(A_{TL}^3) + {\color{lightgray}\frac{1}{2}\Gamma(A_{TR}^T A_{TL} A_{TR}) +} \frac{1}{2}\Gamma(A_{TR} A_{BR} A_{TR}^T) {\color{lightgray}+ \frac{1}{6}\Gamma(A_{BR}^3)}$$

$$\Delta = \frac{1}{6}\Gamma(A_{TL}^3) + \frac{1}{2}\Gamma(A_{TR}^T A_{TL} A_{TR}) {\color{lightgray}+ \frac{1}{2}\Gamma(A_{TR} A_{BR} A_{TR}^T) + \frac{1}{6}\Gamma(A_{BR}^3)}$$

$$\Delta = \frac{1}{6}\Gamma(A_{TL}^3) + \frac{1}{2}\Gamma(A_{TR}^T A_{TL} A_{TR}) + \frac{1}{2}\Gamma(A_{TR} A_{BR} A_{TR}^T) {\color{lightgray}+ \frac{1}{6}\Gamma(A_{BR}^3)}$$

$$\Delta = {\color{lightgray}\frac{1}{6}\Gamma(A_{TL}^3) + \frac{1}{2}\Gamma(A_{TR}^T A_{TL} A_{TR}) + \frac{1}{2}\Gamma(A_{TR} A_{BR} A_{TR}^T)} + \frac{1}{6}\Gamma(A_{BR}^3)$$

$$\Delta = {\color{lightgray}\frac{1}{6}\Gamma(A_{TL}^3) + \frac{1}{2}\Gamma(A_{TR}^T A_{TL} A_{TR}) +} \frac{1}{2}\Gamma(A_{TR} A_{BR} A_{TR}^T) + \frac{1}{6}\Gamma(A_{BR}^3)$$

$$\Delta = {\color{lightgray}\frac{1}{6}\Gamma(A_{TL}^3) +} \frac{1}{2}\Gamma(A_{TR}^T A_{TL} A_{TR}) {\color{lightgray}+ \frac{1}{2}\Gamma(A_{TR} A_{BR} A_{TR}^T)} + \frac{1}{6}\Gamma(A_{BR}^3)$$

$$\Delta = {\color{lightgray}\frac{1}{6}\Gamma(A_{TL}^3) +} \frac{1}{2}\Gamma(A_{TR}^T A_{TL} A_{TR}) + \frac{1}{2}\Gamma(A_{TR} A_{BR} A_{TR}^T) + \frac{1}{6}\Gamma(A_{BR}^3)$$

# Derived Algorithms

Algorithm: $\hat{\Delta} := \frac{1}{6}\Gamma(\hat{A}^3)$

$$A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{TR}^T & A_{BR} \end{array}\right)$$

where $A_{TL}$ is a $0 \times 0$ matrix

while $m(A_{TL}) < m(A)$ do

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{TR}^T & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{01}^T & \alpha_{11} & a_{12}^T \\ \hline A_{02}^T & a_{12} & A_{22} \end{array}\right)$$

where $\alpha_{11}$ is a $1 \times 1$ matrix

| **Algorithm 1** | **Algorithm 2** |
|---|---|
| $\Delta := \Delta + \frac{1}{2}a_{01}^T A_{00} a_{01}$ | $\Delta := \Delta + a_{01}^T A_{02} a_{21}$ |
| **Algorithm 3** | **Algorithm 4** |
| $\Delta := \Delta + \frac{1}{2}a_{01}^T A_{00} a_{01}$<br>$\Delta := \Delta + \frac{1}{2}a_{12}^T A_{22} a_{12}$<br>$\Delta := \Delta - a_{01}^T A_{02} a_{21}$ | $\Delta := \Delta + \frac{1}{2}a_{12}^T A_{22} a_{12}$ |

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{TR}^T & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{01}^T & \alpha_{11} & a_{12}^T \\ \hline A_{02}^T & a_{12} & A_{22} \end{array}\right)$$

endwhile

Algorithm: $\hat{t} := \frac{1}{6}\Gamma(A^3)$

$$A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{TR}^T & A_{BR} \end{array}\right)$$

where $A_{BR}$ is a $0 \times 0$ matrix

while $m(A_{TL}) < m(A)$ do

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{TR}^T & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{01}^T & \alpha_{11} & a_{12}^T \\ \hline A_{02}^T & a_{12} & A_{22} \end{array}\right)$$

where $\alpha_{11}$ is a $1 \times 1$ matrix

| **Algorithm 5** | **Algorithm 6** |
|---|---|
| $\Delta := \Delta + \frac{1}{2}a_{01}^T A_{00} a_{01}$ | $\Delta := \Delta + a_{01}^T A_{02} a_{21}$ |
| **Algorithm 7** | **Algorithm 8** |
| $\Delta := \Delta + \frac{1}{2}a_{01}^T A_{00} a_{01}$<br>$\Delta := \Delta + \frac{1}{2}a_{12}^T A_{22} a_{12}$<br>$\Delta := \Delta - a_{01}^T A_{02} a_{21}$ | $\Delta := \Delta + \frac{1}{2}a_{12}^T A_{22} a_{12}$ |

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{TR}^T & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{01}^T & \alpha_{11} & a_{21}^T \\ \hline A_{02}^T & a_{21} & A_{22} \end{array}\right)$$

endwhile

# Derived Algorithms

**Algorithm:** $\tilde{\Delta} := \frac{1}{6}\Gamma(\hat{A}^3)$

$A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{TR}^T & A_{BR} \end{array}\right)$

where $A_{TL}$ is a $0 \times 0$ matrix

**while** $m(A_{TL}) < m(A)$ **do**

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{TR}^T & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{01}^T & \alpha_{11} & a_{12}^T \\ A_{02}^T & a_{12} & A_{22} \end{array}\right)$$

where $\alpha_{11}$ is a $1 \times 1$ matrix

**Algorithm 1**

$\Delta := \Delta + \frac{1}{2} a_{01}^T A_{00} a_{01}$

**Algorithm 2**

$\Delta := \Delta + a_{01}^T A_{02} a_{21}$

**Algorithm 3**

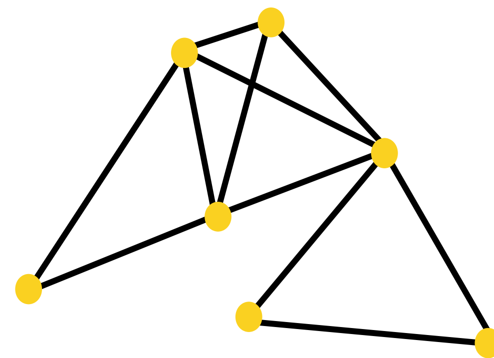
$\Delta := \Delta + \frac{1}{2} a_{01}^T A_{00} a_{01}$
$\Delta := \Delta + \frac{1}{2} a_{12}^T A_{22} a_{12}$
$\Delta := \Delta - a_{01}^T A_{02} a_{21}$

**Algorithm 4**

$\Delta := \Delta + \frac{1}{2} a_{12}^T A_{22} a_{12}$

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{TR}^T & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ a_{01}^T & \alpha_{11} & a_{12}^T \\ \hline A_{02}^T & a_{12} & A_{22} \end{array}\right)$$

**endwhile**



# Derived Algorithms

Algorithm: $\tilde{\Delta} := \frac{1}{6}\Gamma(\hat{A}^3)$

$A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{TR}^T & A_{BR} \end{array}\right)$

where $A_{TL}$ is a $0 \times 0$ matrix

while $m(A_{TL}) < m(A)$ do

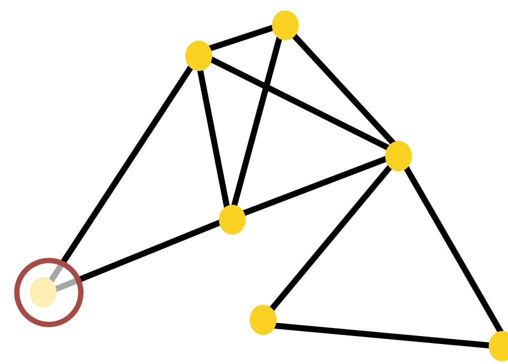
$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{TR}^T & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{01}^T & \alpha_{11} & a_{12}^T \\ A_{02}^T & a_{12} & A_{22} \end{array}\right)$

where $\alpha_{11}$ is a $1 \times 1$ matrix

| **Algorithm 1** | **Algorithm 2** |
|---|---|
| $\Delta := \Delta + \frac{1}{2} a_{01}^T A_{00} a_{01}$ | $\Delta := \Delta + a_{01}^T A_{02} a_{21}$ |
| **Algorithm 3** | **Algorithm 4** |
| $\Delta := \Delta + \frac{1}{2} a_{01}^T A_{00} a_{01}$<br>$\Delta := \Delta + \frac{1}{2} a_{12}^T A_{22} a_{12}$<br>$\Delta := \Delta - a_{01}^T A_{02} a_{21}$ | $\Delta := \Delta + \frac{1}{2} a_{12}^T A_{22} a_{12}$ |

$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{TR}^T & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ a_{01}^T & \alpha_{11} & a_{12}^T \\ \hline A_{02}^T & a_{12} & A_{22} \end{array}\right)$

endwhile



# Derived Algorithms

**Algorithm:** $\tilde{\Delta} := \frac{1}{6}\Gamma(\tilde{A}^3)$

$$A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{TR}^T & A_{BR} \end{array}\right)$$

where $A_{TL}$ is a $0 \times 0$ matrix

**while** $m(A_{TL}) < m(A)$ **do**

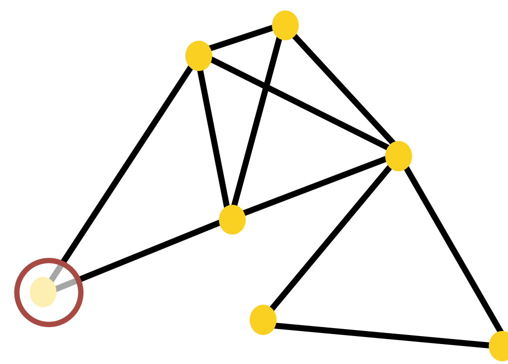
$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{TR}^T & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{01}^T & \alpha_{11} & a_{12}^T \\ A_{02}^T & a_{12} & A_{22} \end{array}\right)$$

where $\alpha_{11}$ is a $1 \times 1$ matrix

| Algorithm 1 | Algorithm 2 |
| --- | --- |
| $\Delta := \Delta + \frac{1}{2} a_{01}^T A_{00} a_{01}$ | $\Delta := \Delta + a_{01}^T A_{02} a_{21}$ |
| **Algorithm 3** | **Algorithm 4** |
| $\Delta := \Delta + \frac{1}{2} a_{01}^T A_{00} a_{01}$<br>$\Delta := \Delta + \frac{1}{2} a_{12}^T A_{22} a_{12}$<br>$\Delta := \Delta - a_{01}^T A_{02} a_{21}$ | $\Delta := \Delta + \frac{1}{2} a_{12}^T A_{22} a_{12}$ |

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{TR}^T & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ a_{01}^T & \alpha_{11} & a_{12}^T \\ \hline A_{02}^T & a_{12} & A_{22} \end{array}\right)$$

**endwhile**



# Derived Algorithms

Algorithm: $\tilde{\Delta} := \frac{1}{6}\Gamma(\hat{A}^3)$

$A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{TR}^T & A_{BR} \end{array}\right)$

where $A_{TL}$ is a $0 \times 0$ matrix

while $m(A_{TL}) < m(A)$ do

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{TR}^T & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{01}^T & \alpha_{11} & a_{12}^T \\ \hline A_{02}^T & a_{12} & A_{22} \end{array}\right)$$

where $\alpha_{11}$ is a $1 \times 1$ matrix

**Algorithm 1**

$\Delta := \Delta + \frac{1}{2} a_{01}^T A_{00} a_{01}$

**Algorithm 2**

$\Delta := \Delta + a_{01}^T A_{02} a_{21}$

**Algorithm 3**

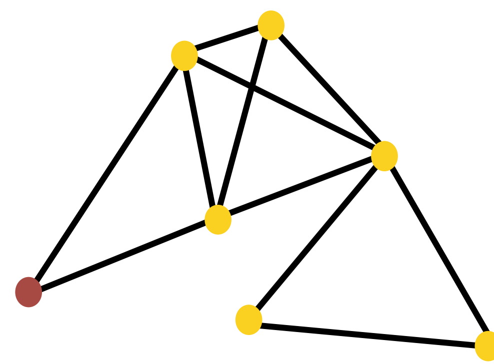
$\Delta := \Delta + \frac{1}{2} a_{01}^T A_{00} a_{01}$
$\Delta := \Delta + \frac{1}{2} a_{12}^T A_{22} a_{12}$
$\Delta := \Delta - a_{01}^T A_{02} a_{21}$

**Algorithm 4**

$\Delta := \Delta + \frac{1}{2} a_{12}^T A_{22} a_{12}$

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{TR}^T & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{01}^T & \alpha_{11} & a_{12}^T \\ \hline A_{02}^T & a_{12} & A_{22} \end{array}\right)$$

endwhile



# Derived Algorithms

Algorithm: $\tilde{\Delta} := \frac{1}{6}\Gamma(\hat{A}^3)$

$A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{TR}^T & A_{BR} \end{array}\right)$

where $A_{TL}$ is a $0 \times 0$ matrix

while $m(A_{TL}) < m(A)$ do

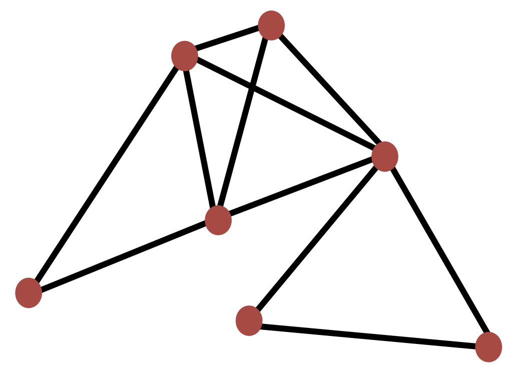
$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{TR}^T & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{01}^T & \alpha_{11} & a_{12}^T \\ \hline A_{02}^T & a_{12} & A_{22} \end{array}\right)$

where $\alpha_{11}$ is a $1 \times 1$ matrix

| | |
|---|---|
| **Algorithm 1**<br>$\Delta := \Delta + \frac{1}{2} a_{01}^T A_{00} a_{01}$ | **Algorithm 2**<br>$\Delta := \Delta + a_{01}^T A_{02} a_{21}$ |
| **Algorithm 3**<br>$\Delta := \Delta + \frac{1}{2} a_{01}^T A_{00} a_{01}$<br>$\Delta := \Delta + \frac{1}{2} a_{12}^T A_{22} a_{12}$<br>$\Delta := \Delta - a_{01}^T A_{02} a_{21}$ | **Algorithm 4**<br>$\Delta := \Delta + \frac{1}{2} a_{12}^T A_{22} a_{12}$ |

$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{TR}^T & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{01}^T & \alpha_{11} & a_{12}^T \\ \hline A_{02}^T & a_{12} & A_{22} \end{array}\right)$

endwhile



# FLAME API

- Index-free API for implementing derived algorithms

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{TR}^T & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{01}^T & \alpha_{11} & a_{12}^T \\ \hline A_{02}^T & a_{12} & A_{22} \end{array}\right)$$

where $\alpha_{11}$ is a $1 \times 1$ matrix

```
FLA_Repart_2x2_to_3x3( ATL, /**/ ATR,       &A00, /**/ &a01, &A02,
                    /* ************* */ /* ************************** */
                                           &a10t, /**/ &alpha11, &a12t,
                      ABL, /**/ ABR,        &A20, /**/ &a21, &A22,
                      1, 1, FLA_BR );
```

- Separation of implementation and algorithm concerns

# Extension to the FLAME API

- Existing API supports only dense matrices
- Introduced
  - Support for sparse matrices (CSR)
  - Additional function, `dist_to_nonzero_column`
    - Conceptually treat sparse matrices as dense
    - Each “dense” block has only one non zero column
    - Returns blocking parameter to the next non zero


Actual Layout

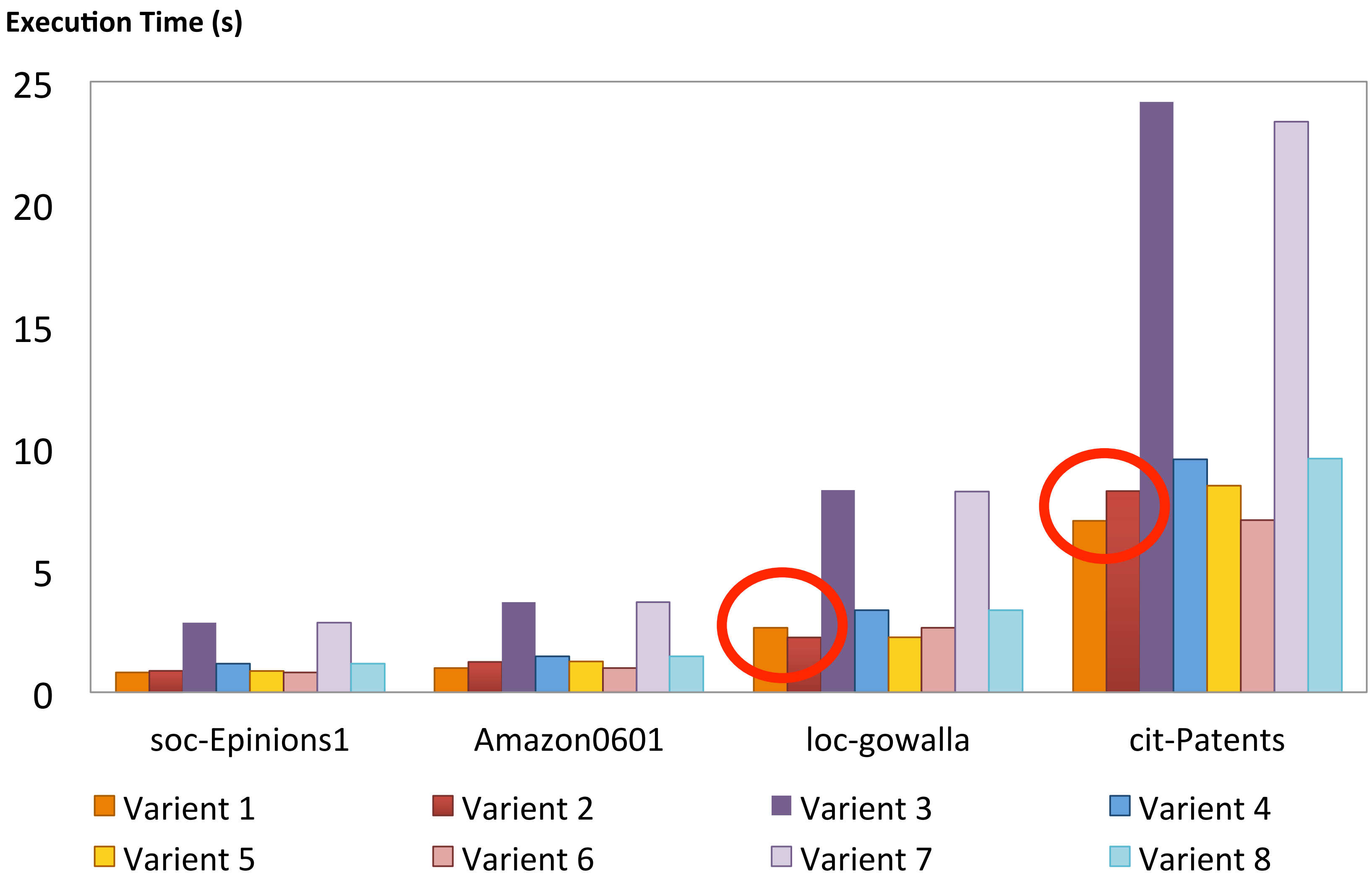
Conceptual Layout

# Datasets

- Datasets
  - Graph Challenge website
  - Stanford Large Network Dataset (SNAP)
  - Directed graphs were made undirected

| Dataset | Nodes | Edges | Triangles |
|---|---|---|---|
| soc-Epinions1 | 75,879 | 508,837 | 1,624,481 |
| Amazon0601 | 403,394 | 3,387,388 | 3,986,507 |
| loc-gowalla | 196,591 | 950,327 | 2,273,138 |
| Cit-Patents | 3,774,768 | 16,518,948 | 7,515,023 |
| Com-Friendster | 65,608,366 | 1,806,067,135 | 4,173,724,142 |

# Performance




Sequential Performance on Intel i7 E5-2667 v 3 Haswell , 3.2GHz

# Summary

- A family of formally derived algorithms for computing triangles in a graph
- First extension of the FLAME methodology beyond DLA
- API to support CSR format
- To do:
  - Analyze graph features that determine performance of algorithm
  - Reduce overhead of indexing functions

Carnegie Mellon

Electrical & Computer ENGINEERING

# Questions?

**Acknowledgement**
This research was supported in part by NSF Award ACI 1550486. Any opinions, findings conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.