

# Complex Network Analysis using Parallel Approximate Motif Counting

George M. Slota   Kamesh Madduri

Computer Science and Engineering  
The Pennsylvania State University  
gmslota@psu.edu, madduri@cse.psu.edu  
fascia-psu.sf.net

IPDPS 2014

May 20, 2014

# Our Contributions

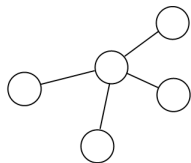
- New distributed-memory parallelization of a dynamic programming-based subgraph counting scheme
  - memory-efficient
  - complementary to prior shared-memory parallelism
- Comparative network analysis using relative subgraph counts as graph signatures
  - find motifs in networks
  - detect local structure in network snapshots
  - cluster networks into categories
- Open-source tool: [fascia-psu.sf.net](http://fascia-psu.sf.net)

# Talk Outline

- **Introduction:** Color-coding for subgraph counting
- **Background:** Why fast subgraph counting?
- **Algorithms:** New distributed-memory parallelization
- **Results:** Parallel performance and scalability
- **Analysis:** Large network collection using subgraph counts

# Introduction

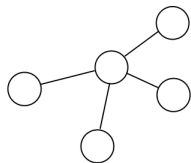
## Subgraph Counting



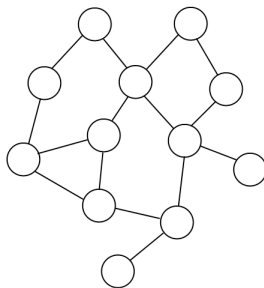
Template

# Introduction

## Subgraph Counting



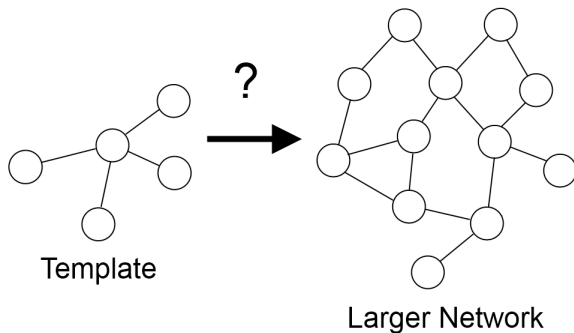
Template



Larger Network

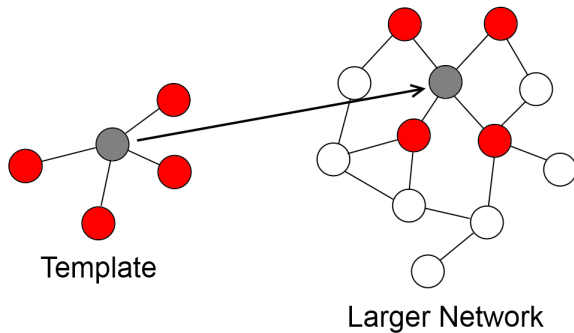
# Introduction

## Subgraph Counting



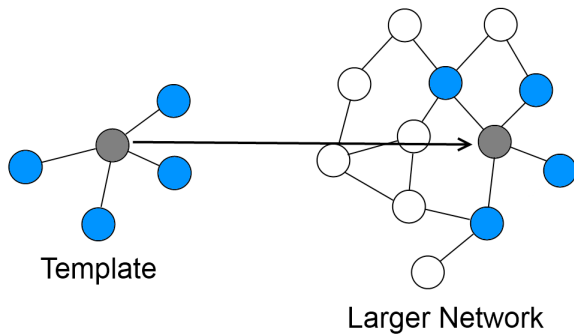
# Introduction

## Subgraph Counting



# Introduction

## Subgraph Counting

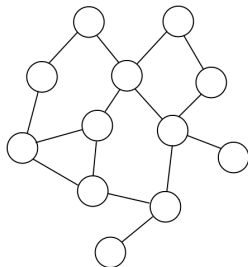




# Introduction

Color-coding [Alon et al., 1995] for Approximate Subgraph Counting

- Color-coding: randomized method to get approximate counts of *tree-structured non-induced subgraphs*, termed as **treelets**

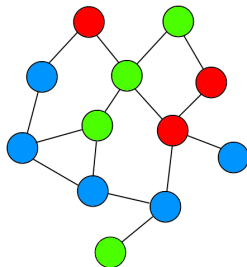
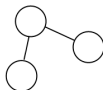


# Introduction

Color-coding [Alon et al., 1995] for Approximate Subgraph Counting

- Color-coding: randomized method to get approximate counts of *tree-structured non-induced subgraphs*, termed as **treelets**

Template:

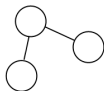


# Introduction

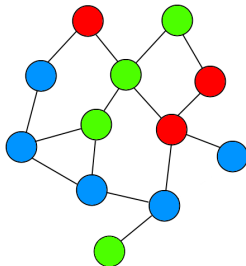
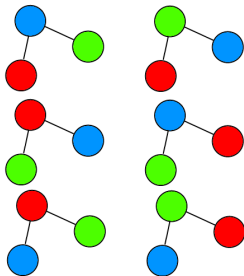
## Color-coding [Alon et al., 1995] for Approximate Subgraph Counting

- Color-coding: randomized method to get approximate counts of *tree-structured non-induced subgraphs*, termed as **treelets**

Template:



Possible Colorful Embeddings:

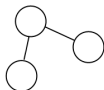


# Introduction

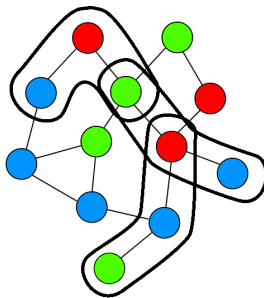
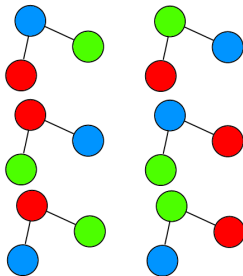
## Color-coding [Alon et al., 1995] for Approximate Subgraph Counting

- Color-coding: randomized method to get approximate counts of *tree-structured non-induced subgraphs*, termed as **treelets**

Template:



Possible Colorful Embeddings:

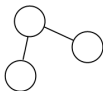


# Introduction

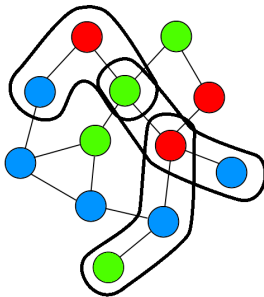
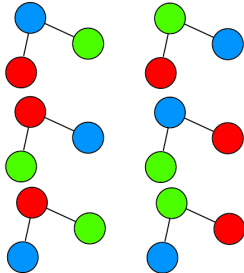
## Color-coding [Alon et al., 1995] for Approximate Subgraph Counting

- Color-coding: randomized method to get approximate counts of *tree-structured non-induced subgraphs*, termed as **treelets**
- $cnt_{colorful} = 3$ ,  $C_{total} = 3^3$ ,  $C_{colorful} = 3!$ ,  $P = \frac{3!}{3^3}$
- $cnt_{estimate} = \frac{cnt_{colorful}}{P} = 13.5$

Template:



Possible Colorful Embeddings:

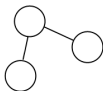


# Introduction

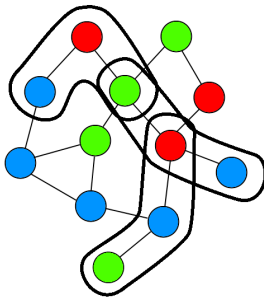
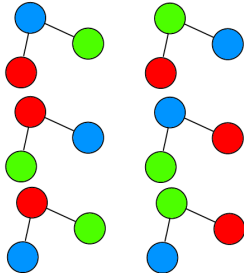
## Color-coding [Alon et al., 1995] for Approximate Subgraph Counting

- Color-coding: randomized method to get approximate counts of *tree-structured non-induced subgraphs*, termed as **treelets**
- $cnt_{colorful} = 3$ ,  $C_{total} = 3^3$ ,  $C_{colorful} = 3!$ ,  $P = \frac{3!}{3^3}$
- $cnt_{estimate} = \frac{cnt_{colorful}}{P} = \mathbf{13.5}$
- Use multiple coloring iterations. Each iteration is  $O(m2^k)$  work.

Template:



Possible Colorful Embeddings:



# Motivation

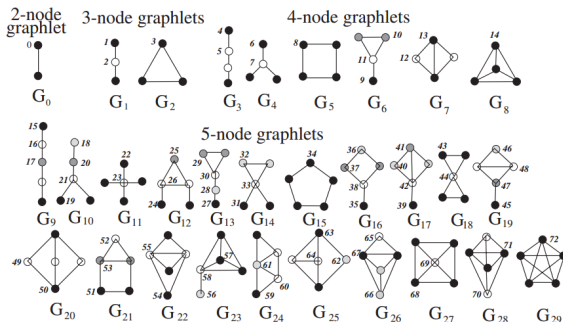
Why do we want fast algorithms for subgraph counting?

- Important uses in bioinformatics, chemoinformatics, social network analysis, communication network analysis, etc.
- Counts form basis for more complex analyses:
  - Motif finding
  - Graphlet frequency distances (GFD)
  - Graphlet degree distributions (GDD) and agreements (GDDA)
  - Graphlet degree signatures (GDS)
- Counting and enumeration on large networks is very expensive,  $O(n^k)$  complexity for naïve algorithm. Color-coding reduces this to  $O(m2^k n_{\text{iter}})$ .

# Background

## Network analysis using graphlet counts

- Graphlets: all possible 2-5 vertex undirected subgraphs





# Background

## Network analysis using graphlet counts

- Graphlets: all possible 2-5 vertex undirected subgraphs
- Graphlet frequency distance

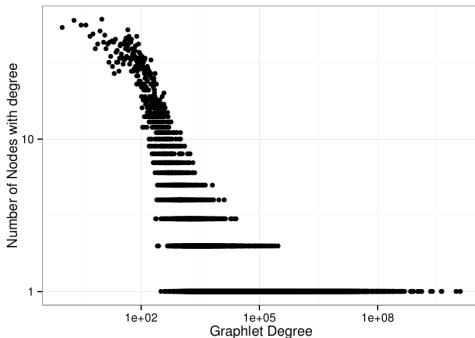
$$S_i(G) = -\log\left(\frac{C_i(G)}{\sum_{i=1}^n C_i(G)}\right)$$

$$D(G, H) = \sum_{i=1}^n |S_i(G) - S_i(H)|$$

# Background

## Network analysis using graphlet counts

- Graphlets: all possible 2-5 vertex undirected subgraphs
- Graphlet frequency distance
- Graphlet degree distribution and agreement



# Background

## Network analysis using graphlet counts

- Graphlets: all possible 2-5 vertex undirected subgraphs
- Graphlet frequency distance
- Graphlet degree distribution and agreement

$$S_G^j(k) = \frac{d_G^j(k)}{k}$$

$$N_G^j(k) = \frac{S_G^j(k)}{\sum_{k=1}^{\infty} S_G^j(k)}$$

$$A^j(G, H) = 1 - \frac{1}{\sqrt{2}} \left( \sum_{k=1}^{\infty} [N_G^j(k) - N_H^j(k)]^2 \right)$$

# Background

## Network analysis using graphlet counts

- Graphlets: all possible 2-5 vertex undirected subgraphs
- Graphlet frequency distance
- Graphlet degree distribution and agreement
- Graphlet degree signature

$$S_i(u, v) = 1 - w_i \times \frac{|\log(u_i + 1) - \log(v_i + 1)|}{\log(\max\{u_i, v_i\} + 2)}$$

[Milo et al., 2002, Alon et al., 2008, Pržulj, 2004, 2007, Milenković and Pržulj, 2008]

# Background

## Network analysis using treelet counts

- Can we use treelets instead of graphlets? Are they more/less powerful?
- Goals of this work:
  - Create distributed-memory subgraph counting program to produce counts on larger networks, and faster counts on smaller networks.
  - Quantitative analyses using GFD and GDD with treelets to evaluate efficacy.
  - Evaluate effect of noise on treelet counts by deleting vertices and edges, as well as rewiring edges in various networks.

# FASCIA

Fast Approximate Subgraph Counting (for In-memory/Insightful/I\* Analytics)

- Previous work: FASCIA for shared-memory color-coding treelet counting [Slota and Madduri, 2013].
  - Memory reduction through efficient table and color set representations
  - Work reduction through template partitioning
  - Multilevel algorithm parallelization
- Present work: FASCIA for distributed memory
  - Further memory and communication reductions through CSR-like representation of table
  - Partitioned counting allowing counts for larger networks
  - Distributed counting for faster counts on smaller networks

# FASCIA

## Color-coding algorithm overview

- 1: Partition input template  $T$  ( $k$  vertices) into subtemplates  $S_i$  using *single edge cuts*.
- 2: Determine  $Niter \approx \frac{e^k \log 1/\delta}{\epsilon^2}$ , the number of iterations to execute.  
 $\delta$  and  $\epsilon$  are input parameters that control approximation quality.
- 3: **for**  $it = 1$  to  $Niter$  **do**
- 4:     Randomly assign to each vertex  $v$  in graph  $G$  a color between 0 and  $k - 1$ .
- 5:     Use a dynamic programming scheme to count *colorful non-induced occurrences of  $T$* .
- 6: Take average of all  $Niter$  counts to be final count.

# FASCIA

## New partitioned counting approach

- Each task gets a subset of  $v \in G$ , counts for this subset are further computed in parallel for each task
- Semi-partitioned: Each task holds full table for child subtemplates

**for**  $it = 1$  to  $Niter$  **do**

Color  $G(V, E)$  with  $k$  colors

**for all** subtemplates  $S_i$  in reverse order of partitioning **do**

Init  $Table_{i,d}$  for  $V_d$  (vertex partition on task  $d$ )

**for all**  $v \in V_d$  **do in parallel** ▷ Multithreaded parallelism

**for all**  $c \in C_i$  **do**

Compute all  $Count_{S_i,c,v}$

$N_d, I_d, B_d \leftarrow \text{Compress}(Table_{i,d})$

**for all**  $d = 1$  to  $NumTasks$  **do**

$N_i, I_i, B_i \leftarrow Bcast(N_d, I_d, B_d)$

$$Count_d += \sum_v^{V_d} \sum_c^{C_T} Count_{T,c,v}$$

$Count \leftarrow \text{Reduce}(Count_d)$

Scale  $Count$  based on  $Niter$  and colorful embed prob.



# FASCIA

Compressed Sparse Row (CSR)-like representation of dynamic program table

- Dynamic programming table corresponding to each subtemplate can be represented as a  $n \times C_i$  rectangular matrix ( $n$ : number of graph vertices,  $C_i$ : number of possible color sets of subtemplate  $i$ ).
- We represent the table in a CSR-like format using three arrays:
  - $N$ : count values in table
  - $I$ : color set indexes
  - $B$ : offsets for each vertex

# FASCIA

Distributed counting (assign multiple iterations to each process)

**for all**  $it = 1$  to  $Niter$  **in parallel do**

Color  $G(V, E)$  with  $k$  colors

Initialize 3D count table

**for all**  $S_i$  in reverse order of partitioning **do**

**for all**  $v \in V$  **in parallel do**

Update count table for template  $S_i$   
using child subtemplate counts

▷ MPI task-level

▷ multithreaded

# Results

## Experimental setup - systems and graphs

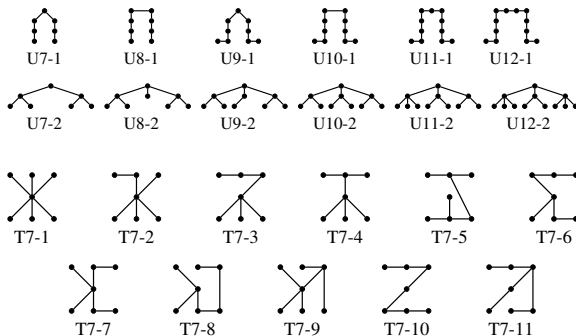
- Performance results on *Compton*, a Sandia cluster with dual-socket Intel Xeon E5-2670 (Sandy Bridge) nodes and 64 GB main memory per node.
- Studies also performed on Cyberstar and Hammer clusters at Penn State.
- Networks from SNAP, Konect, DIMACS, UF sparse matrix, and Virginia Tech NDSSL collections; GTgraph and igraph generators.

Network Type	Count	$n (\times 10^3)$		$m (\times 10^3)$	
		min	max	min	max
Collaboration	6	26	425	14	1050
Communication	4	30	63	87	855
$G(n, p)$	4	10	100	100	1000
Peer-to-peer	9	6	63	9.7	77
Bio PPI	4	0.7	22	1.3	22
Road	5	440	1970	530	2800
Scale-free	4	10	100	100	1000
Social	6	60	150	214	5400
Small-world	4	10	100	100	1000
Web Crawl	4	280	875	761	3900
Orkut	-	3100	-	117000	-
Portland	-	1620	-	31000	-

# Results

## Experimental setup - templates used

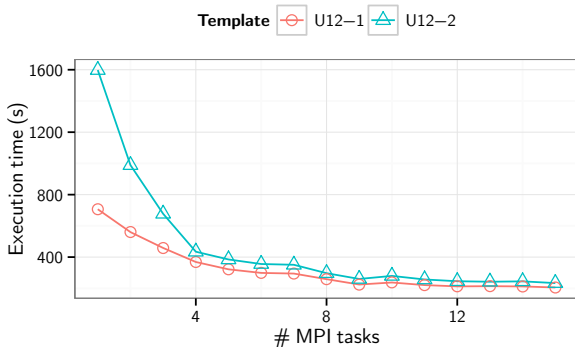
- UX-1 indicates chain or simple path, UX-2 indicates more complex tree
- T7-X indicates all 7 vertex treelets, used for evaluating effects of network noise on relative counts



# Results

## Running times and parallel scaling

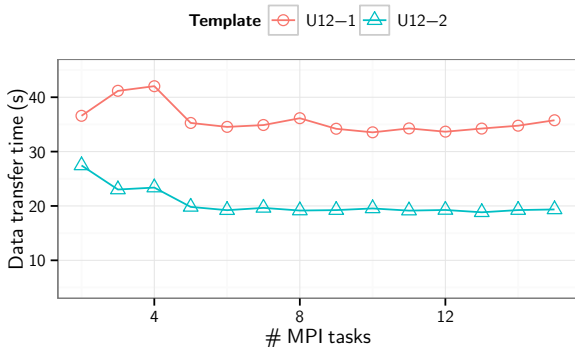
- Parallel speedup on 15 nodes;  $3.5\times$  (U12-1) and  $7\times$  (U12-1) for Orkut crawl



# Results

## Running times and parallel scaling

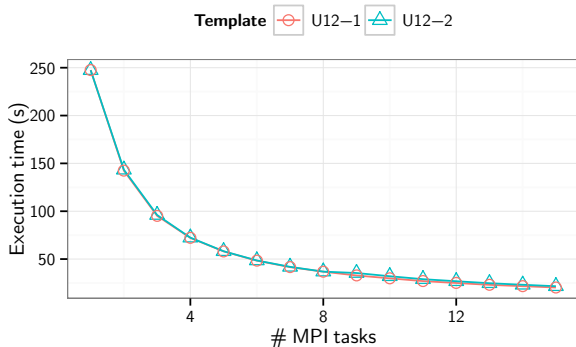
- Parallel speedup on 15 nodes;  $3.5\times$  (U12-1) and  $7\times$  (U12-1) for Orkut crawl
- Communication time is about 15-25% of total time



# Results

## Running times and parallel scaling

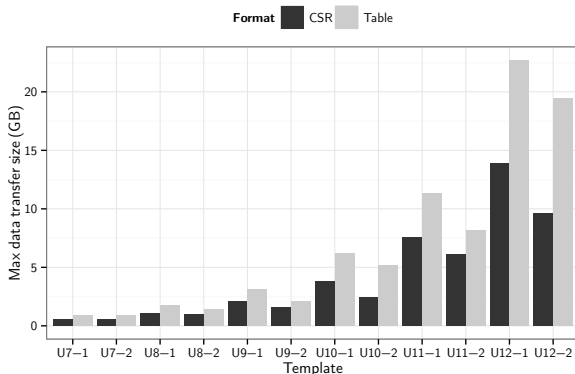
- Parallel speedup on 15 nodes;  $3.5\times$  (U12-1) and  $7\times$  (U12-1) for Orkut crawl
- Communication time is about 15-25% of total time
- Near-linear speedup for distributed counting on C.elegans PPI network



# Results

## Communication volume reduction

- Maximal data transfer during counting on Orkut for templates from 7 to 12 vertices
- About 35% mean reduction in transfer costs with CSR compression

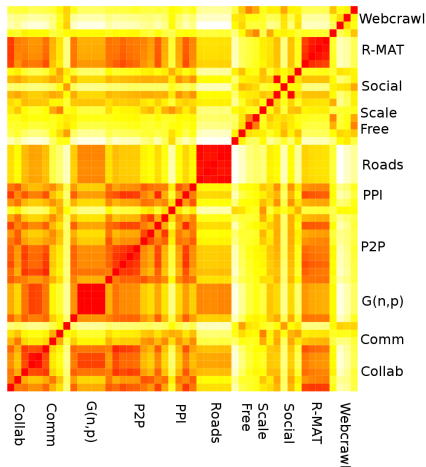




# Results

## Graphlet frequency distance analysis

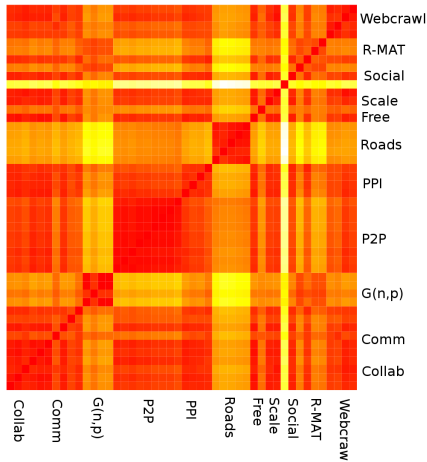
- Treelet frequency distance, GFD calculated with counts of all 4-9 vertex treelets (92 total)
- Intra-class mean agreements highest for 5/10 classes



# Results

## Graphlet degree distribution agreement analysis

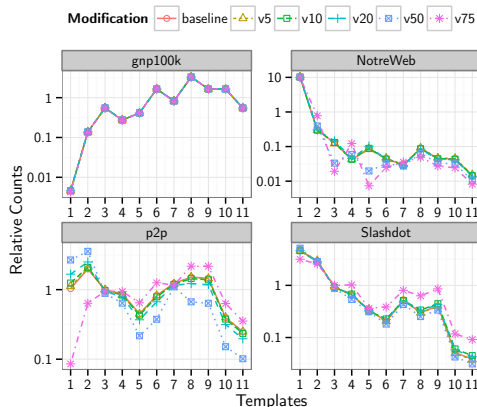
- Treelet degree distribution agreements, GDDA calculated with counts of orbits of all 3-7 vertex treelets (83 total)
- Similar observations as with GFD, intra-class mean highest 6/10 classes



# Results

## Vertex deletion

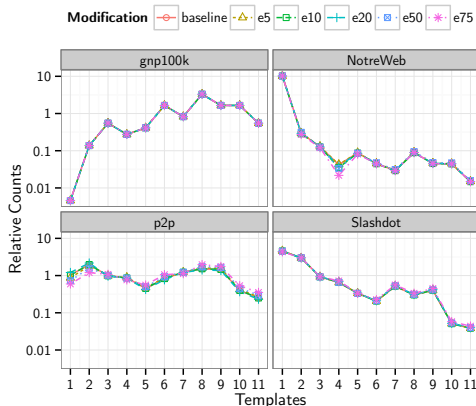
- 5, 10, 20, 50, and 75% vertices deleted, evaluated based on GFD for **only** 7 vertex templates, relative counts shown
- Largest disagreement for Notre Dame webcrawl, value of 4.1
- Mean disagreement between networks is 9.2



# Results

## Edge deletion

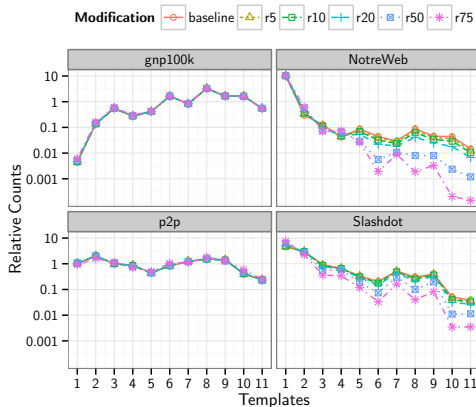
- 5, 10, 20, 50, and 75% edges deleted
- Max disagreement value of 1.2 with Gnutella p2p snapshot
- Demonstrated relative treelet counts are useful analytic even on networks with a relatively high proportion of known vertices to known edges (e.g. PPI networks)



# Results

## Edge rewiring

- 5, 10, 20, 50, and 75% edges rewired
- 6.6 and 10.4 disagreements with Slashdot and Notre Dam webcrawl
- Minimal change with random and Gnutella networks; less inherent structure?



# Conclusions

- Partitioned subgraph counting (i.e., partitioned dynamic programming table) makes it feasible to analyze large-scale networks on clusters.
- Distributed subgraph counting accelerates subgraph counting for small networks.
- Treelets counts seem to be as powerful as graphlets for network analysis.
- Relative treelet counts more sensitive to vertex sampling than edge sampling.

# Bibliography

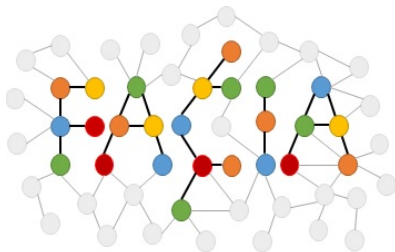
- N. Alon, R. Yuster, and U. Zwick. Color-coding. *J. ACM*, 42(4):844–856, 1995.
- N. Alon, P. Dao, I. Hajirasouliha, F. Hormozdiari, and S.C. Sahinalp. Biomolecular network motif counting and discovery by color coding. *Bioinformatics*, 24(13):i241–i249, 2008.
- T. Milenković and N. Pržulj. Uncovering biological network function via graphlet degree signatures. *Cancer Informatics*, 6:257–273, 2008.
- R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. Network motifs: Simple building blocks of complex networks. *Science*, 298(5594):824–827, 2002.
- N. Pržulj. Modeling interactome, scale-free or geometric? *Bioinformatics*, 20(18):3508–3515, 2004.
- N. Pržulj. Biological network comparison using graphlet degree distribution. *Bioinformatics*, 23(2):e177–83, 2007.
- G. M. Slota and K. Madduri. Fast approximate subgraph counting and enumeration. In *Proc. 42nd Int'l. Conf. on Parallel Processing (ICPP)*, pages 210–219, 2013.

# Acknowledgments

- NSF grant ACI-1253881
- Penn State systems, NSF grant OCI-0821527
- NSF XSEDE, supported by NSF grant OCI-1053575
- Sandia National Laboratories, for access to the Compton system
  - Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.



# Questions?



- fascia-psu.sf.net
- George M. Slota (gms5016@psu.edu)
- Kamesh Madduri (madduri@cse.psu.edu)