Fast Graphlet Decomposition

Theory, Algorithms, and Applications

Nesreen K. Ahmed

Research Scientist
Intel Labs

MMDS Workshop, Berkeley CA
June 24th, 2016
Joint work with ...

Ted Willke
Intel Research Labs

Jennifer Neville
Purdue University

Nick Duffield
Texas A&M University

Ryan A. Rossi
PARC
Graphs – Rich Data Representation

- Social Network
- Political Blogs
- Internet (AS)
- Biological

Graph Mining
# Graphlets

<table>
<thead>
<tr>
<th>Connected</th>
<th>2-node Graphlets</th>
<th>3-node Graphlets</th>
<th>4-node Graphlets</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disconnected</td>
<td>2-node Graphlets</td>
<td>3-node Graphlets</td>
<td>4-node Graphlets</td>
</tr>
</tbody>
</table>

*Network Motifs: Simple Building Blocks of Complex Networks* – [Milo et. al – Science 2002]

Graphlets

- Small k-vertex induced subgraphs

Network Motifs: Simple Building Blocks of Complex Networks – [Milo et. al – Science 2002]
Graphlets

- Small k-vertex induced subgraphs

- **Motifs**: Occur in real-world networks with frequencies **significantly higher** than randomly generated networks

---

**Network Motifs: Simple Building Blocks of Complex Networks** – [Milo et. al – Science 2002]

**The Structure and Function of Complex Networks** – [Newman – Siam Review 2003]
**Graphlets**

<table>
<thead>
<tr>
<th>Connected</th>
<th>2-node Graphlets</th>
<th>3-node Graphlets</th>
<th>4-node Graphlets</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Connected" /></td>
<td><img src="image" alt="2-node" /></td>
<td><img src="image" alt="3-node" /></td>
<td><img src="image" alt="4-node" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Disconnected</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Disconnected" /></td>
<td><img src="image" alt="2-node" /></td>
<td><img src="image" alt="3-node" /></td>
<td><img src="image" alt="4-node" /></td>
</tr>
</tbody>
</table>

- Small k-vertex induced subgraphs
- **Motifs**: Occur in real-world networks with frequencies **significantly higher** than randomly generated networks
- Applied to food web, genetic, neural, web, and other networks
  - Found distinct graphlets in each case

**Network Motifs: Simple Building Blocks of Complex Networks** – [Milo et. al – Science 2002]
**The Structure and Function of Complex Networks** – [Newman – Siam Review 2003]
Efficient graphlet kernels for large graph comparison

Abstract

State-of-the-art graph kernels do not scale to large graphs with hundreds of nodes and thousands of edges. In this article we propose to compare graphs by counting graphlets, i.e., subgraphs with $k$ nodes where $k \in \{3, 4, 5\}$. Exhaustive enumeration of all graphlets being prohibitively expensive, we introduce two theoretically grounded speedup schemes, one based on sampling and the second one specifically designed for bounded degree graphs. In our experimental evaluation, our novel kernels allow us to efficiently compare large graphs that cannot be tackled by existing graph kernels.

Frequent subgraph mining algorithms, on the other hand, aim to detect subgraphs that are frequent in a given dataset of graphs. Afterwards, feature selection is applied to select the most discriminative subgraphs. Efficient methods such as gSpan (Yan & Han, 2003) have been developed for this task, which use elegant data structures and efficient branch-and-bound search strategies. Unfortunately, their computational complexity scales exponentially with graph size in the worst case.

Graph kernels represent an attractive middle ground. They respect and exploit graph topology, but restrict themselves to comparing substructures of graphs that are computable in polynomial time. Many different graph kernels have been defined, which focus on different types of substructures in graphs, such as random walks (Gärtner et al., 2003; Kashima et al.,
Efficient graphlet kernels for large graph comparison

Nino Shervashidze  
MPI for Biological Cybernetics  
MPI for Developmental Biology  
Tübingen, Germany

S.V.N. Vishwanathan  
Department of Statistics  
Purdue University  
West Lafayette, IN, USA

Tobias H. Petri  
Institute for Computer Science  
LMU München  
München, Germany

Kurt Mehlhorn  
MPI for Informatics  
Saarbrücken, Germany

Karsten M. Borgwardt  
MPI for Biological Cybernetics  
MPI for Developmental Biology  
Tübingen, Germany

Abstract

State-of-the-art graph kernels do not scale to large graphs with hundreds of nodes and thousands of edges. In this article we propose to compare graphs by counting graphlets, i.e., subgraphs with \(k\) nodes where \(k = \{3, 4, 5\}\). Exhaustive enumeration of all graphlets being prohibitively expensive, we introduce two theoretically grounded speedup schemes, one based on sampling and the second one specifically designed for bounded degree graphs. In our experimental evaluation, our novel kernels allow us to efficiently compare large graphs that cannot be tackled by existing graph kernels.

Biological network comparison using graphlet degree distribution

Nataša Pržulj  
Computer Science Department, University of California, Irvine, CA 92697-3425, USA

ABSTRACT

Motivation: Analogous to biological sequence comparison, comparing cellular networks is an important problem that could provide insight into biological understanding and therapeutics. For technical reasons, comparing large networks is computationally infeasible, and thus heuristics, such as the degree distribution, clustering coefficient, diameter, and relative graphlet frequency distribution have been sought. It is easy to demonstrate that two networks are different by simply showing a short list of properties in which they differ. It is much harder to show that two networks are similar, as it requires demonstrating their similarity in all of their exponentially many properties. Clearly, it is computationally prohibitive to analyze all network properties, but the larger the number of constraints we impose in determining network similarity, the more likely it is that the networks will truly be similar.

Therefore, analogous to the BLAST heuristic (Altschul et al., 1990) for biological sequence comparison, we need to design a heuristic tool for the full-scale comparison of large cellular networks (Berg and Lassig, 2004). The current network comparisons consist of heuristics falling into two major classes: (1) global heuristics, such as counting the number of connections between various parts of the network (the ‘degree distribution’), computing the average density of node neighborhoods (the ‘clustering coefficients’), or the average length of shortest paths between all pairs of nodes (the ‘diameter’); and (2) local heuristics that measure relative distance between concentrations of small subgraphs (called graphlets) in two networks (Pržulj et al., 2004).

Since cellular networks are incompletely explored, global statistics on such incomplete data may be substantially biased, or even
Applications of Graphlets

- Biological Networks
  - network alignment, protein function prediction
- Social Networks
  - triad analysis, community detection, Exp. Random Models
- Computer Networks
- Internet AS
- Cyber Security
  - spam detection
- Ecology
  ...

![Diagram of biological networks and social networks](image1)

![Diagram of computer networks and cyber security](image2)
Problem: Counting Graphlets of Size $k$

Given a family of graphlets of size $k$ nodes and an input graph

Family of Graphlets of size $k$ nodes: $G_k = \{g_{k_1}, g_{k_2}, \ldots, g_{k_m}\}$

Input graph: $G = (V, E)$ where $\Delta$ is the maximum node degree

Count the number of embeddings (appearances) of each graphlet $g_{k_i} \in G_k$ in the input graph $G$. 
Problem: Counting Graphlets of Size k

Given a family of graphlets of size $k$ nodes and an input graph

*Family of Graphlets of size $k$ nodes: $G_k = \{g_{k1}, g_{k2}, ..., g_{km}\}$*

*Input graph: $G = (V, E)$ where $\Delta$ is the maximum node degree*

Count the number of embeddings (appearances) of each graphlet $g_{ki} \in G_k$ in the input graph $G$.

**Ex: Given an input graph $G$**

- How many triangles in $G$?
- How many cliques of size 4-nodes in $G$?
- How many cycles of size 4-nodes in $G$?
Problem: Counting Graphlets of Size $k$

Given a family of graphlets of size $k$ nodes and an input graph

*Family of Graphlets of size $k$ nodes: $G_k = \{g_{k_1}, g_{k_2}, \ldots, g_{k_m}\}$*

*Input graph: $G = (V, E)$ where $\Delta$ is the maximum node degree*

Count the number of embeddings (appearances) of each graphlet $g_{k_i} \in G_k$ in the input graph $G$.

**Ex: Given an input graph $G$**

- How many triangles in $G$?
- How many cliques of size 4-nodes in $G$?
- How many cycles of size 4-nodes in $G$?

→ In practice, we would like to count all $k$-vertex graphlets
**Problem: Counting Graphlets of Size $k$**

Given a family of graphlets of size $k$ nodes and an input graph

*Family of Graphlets of size $k$ nodes: $G_k = \{g_{k_1}, g_{k_2}, \ldots, g_{k_m}\}$*

*Input graph: $G = (V, E)$ where $\Delta$ is the maximum node degree*

Count the number of embeddings (appearances) of each graphlet $g_{k_i} \in G_k$ in the input graph $G$.

- Enumerate all possible graphlets
**Problem: Counting Graphlets of Size k**

Given a family of graphlets of size $k$ nodes and an input graph

*Family of Graphlets of size $k$ nodes: $G_k = \{g_{k_1}, g_{k_2}, \ldots, g_{k_m}\}$*

*Input graph: $G = (V, E)$ where $\Delta$ is the maximum node degree*

Count the number of embeddings (appearances) of each graphlet $g_{k_i} \in G_k$ in the input graph $G$.

- Enumerate all possible graphlets
  - *Exhaustive enumeration is too expensive* $O(|V|^k)$
Problem: Counting Graphlets of Size $k$

Given a family of graphlets of size $k$ nodes and an input graph

- Family of Graphlets of size $k$ nodes: $G_k = \{g_{k_1}, g_{k_2}, ..., g_{k_m}\}$
- Input graph: $G = (V, E)$ where $\Delta$ is the maximum node degree

Count the number of embeddings (appearances) of each graphlet $g_{k_i} \in G_k$ in the input graph $G$.

- Enumerate all possible graphlets
  - Exhaustive enumeration is too expensive $O(|V|^k)$
- Count graphlets for each node – and combine all node counts
  [Shervashidze et. al – AISTAT 2009] $O(|V| \cdot \Delta^{k-1})$
Problem: Counting Graphlets of Size k

Given a family of graphlets of size $k$ nodes and an input graph:

*Family of Graphlets of size $k$ nodes: $G_k = \{g_{k_1}, g_{k_2}, \ldots, g_{k_m}\}$*

*Input graph: $G = (V, E)$ where $\Delta$ is the maximum node degree*

Count the number of embeddings (appearances) of each graphlet $g_{k_i} \in G_k$ in the input graph $G$.

- Enumerate all possible graphlets
  - Exhaustive enumeration is too expensive
    $\mathcal{O}(|V|^k)$
- Count graphlets for each node – and combine all node counts
  - Still expensive for relatively large $k$ [Shervashidze et. al – AISTAT 2009]
    $\mathcal{O}(|V| \cdot \Delta^{k-1})$
Problem: Counting Graphlets of Size k

Given a family of graphlets of size $k$ nodes and an input graph $G$. Given any node $x \in G_k$ in the input graph $G$.

- **Not practical** – scales only for small graphs with few hundred/thousand nodes/edges
  - taking 2400 secs for a graph with 26K nodes

- Enumerate all possible graphlets
  - *Exhaustive enumeration is too expensive* $\mathcal{O}(|V|^k)$

- Count graphlets for each node – and combine all node counts
  - *Still expensive for relatively large $k$* [Shervashidze et. al – AISTAT 2009] $\mathcal{O}(|V| \cdot \Delta^{k-1})$

- Other recent work counts only connected graphlets of size $k=4$
  - [Marcus & Shavitt – Computer Networks 2012] $\mathcal{O}(|E| \cdot \Delta + |E|^2)$
Most work focused on graphlets of $k=3$ nodes

In this work, we focus on graphlets of $k=3,4$ nodes

*Efficient Graphlet Counting for Large Networks* [Ahmed et al., ICDM 2015]

Our Approach – (Edge-centric, parallel)

① For each edge do

 Searching Edge Neighborhoods
Our Approach – (Edge-centric, parallel)

① For each edge do
• Count All 3-node graphlets

Searching Edge Neighborhoods

② Merge counts from all edges
Our Approach – (Edge-centric, parallel)

1. For each edge do
   - Count All 3-node graphlets

   Triangle 2-star 1-edge Independent

   - We only need to find/count triangles
   - Use equations to get counts of others in $o(1)$

   $O(|E| \cdot \Delta)$

2. Merge counts from all edges
Edge-centric, Parallel, Memory-efficient Framework
How to count all 4-node graphlets?

- 4-Clique
- 4-Chrodal-Cycle
- Tailed-triangle
- 4-Cycle
- 4-Path
- 3-Star

- 4-node-triangle
- 4-node-2star
- 4-node-2edge
- 4-node-1edge
- Independent
Our Approach – (Edge-centric, parallel)

Step 1
Searching Edge Neighborhoods
For each edge
Find the triangles

Step 2
Count 4-node graphlets
For each edge
Count 4-node cliques and 4-node cycles only

Step 3
Count 4-node graphlets
For each edge
Use combinatorial relationships to compute counts of other graphlets in constant time

Step 4
Merge counts from all edges
4-Node Graphlet Transition Diagram

± 1 edge
4-Node Graphlet Transition Diagram

Maximum no. triangles Incident to an edge
\[ T_{max} \ll \Delta \]

\[ O(|E| \cdot \Delta \cdot T_{max}) \]

4-Cliques

Count Cliques & Cycles ONLY

Use relationships & transitions to count all other graphlets in constant time

Maximum no. stars Incident to an edge
\[ S_{max} \leq \Delta \]

\[ O(|E| \cdot \Delta \cdot S_{max}) \]
Relationship between 4-cliques & 4-ChordalCycles

4-Cliques

4-ChordalCycle

\[ \text{No. 4-ChordalCycles} = \sum_{e=(u,v)\in E} \left( \binom{\text{No. Tri}(e)}{2} \right) - 6\cdot\text{No. 4-Cliques} \]

Proof in Lemma 1 - Ahmed et al., ICDM 2015
Relationship between 4-cliques & 4-ChordalCycles

No. ChordalCycles = \[ \sum_{e=(u,v) \in E} \left( \binom{\text{No. Tri}(e)}{2} \right) - 6 \cdot \text{No. Cliques} \]

Proof in Lemma 1 - Ahmed et al., ICDM 2015
Experiments & Results
Experiments

- Shared Memory Implementation

- Tested on graphs with over a billion edges

- Largest systematic investigation on 300+ networks
  - Social, web, technological, biological, co-authorship, infrastructure...
  - Facebook 100 networks from a variety of US schools
  - Dense graphs from the DIMACS challenge
  - Large collections of biological and chemical graphs

Details in the paper
Data/code online
Comparison to RAGE [Marcus & Shavitt – J. Computer Networks 2011]

Facebook100 Networks from US Schools

| graph              | $|V|$ | $|E|$ | $|T|$ | $|S|_3|$ | $|K|_4|$ | $C_{4,ch}$ | $|C|_4|$ | $|P|_4|$ | $|S|_4|$ | Time in Seconds |
|--------------------|-----|-----|-----|------|------|--------|--------|--------|--------|------|-------------|
| socfb-American75   | 6.4k| 218k| 1.5M| 23M  | 6.5M | 36M    | 23M    | 2.2B   | 1.2B   | 0.46 | 202.85      |
| socfb-Amherst41    | 2.2k| 91k | 916k| 9.0M | 4.5M | 33M    | 18M    | 754M   | 361M   | 0.16 | 63.61       |
| socfb-BC17         | 12k | 487k| 3.5M| 63M  | 16M  | 117M   | 77M    | 7.5B   | 4.2B   | 1.71 | 1311.34     |
| socfb-Berkeley13   | 23k | 852k| 5.4M| 125M | 27M  | 153M   | 87M    | 17B    | 25B    | 4.94 | 2514.59     |
| socfb-Brown11      | 8.6k| 385k| 3.0M| 52M  | 12M  | 96M    | 72M    | 6.7B   | 3.5B   | 1.4  | 781.09      |
| socfb-Caltech36    | 769 | 17k | 120k| 873k | 460k | 965k   | 34M    | 19M    |        | 0.01 | 4.09        |
| socfb-Carnegie49   | 6.6k| 250k| 2.3M| 30M  | 14M  | 84M    | 46M    | 3.5B   | 1.9B   | 0.79 | 412.72      |
| socfb-Columbia2    | 12k | 44k | 3.3M| 66M  | 14M  | 119M   | 76M    | 8.2B   | 12B    | 1.81 | 1897.8      |
| socfb-Cornell5     | 19k | 791k| 6.1M| 117M | 36M  | 206M   | 112M   | 16B    | 18B    | 4.33 | 2271        |
| socfb-Dartmouth6   | 7.7k| 304k| 2.3M| 38M  | 8.8M | 72M    | 53M    | 4.4B   | 2.3B   | 0.93 | 467.8       |
| socfb-Duke14       | 9.9k| 506k| 5.1M| 78M  | 31M  | 206M   | 126M   | 11B    | 5.9B   | 2.52 | 1509.58     |
| socfb-FSU53        | 28k | 1.0M| 7.9M| 130M | 63M  | 242M   | 95M    | 16B    | 10B    | 5.55 | 2192.94     |
| socfb-GWU54        | 12k | 470k| 3.2M| 60M  | 15M  | 90M    | 56M    | 7.1B   | 5.6B   | 1.55 | 749.1       |
| socfb-Hamilton46   | 2.3k| 96k | 913k| 9.8M | 4.1M | 31M    | 18M    | 826M   | 421M   | 0.18 | 60.91       |
| socfb-Haverford76  | 1.4k| 60k | 628k| 5.6M | 3.1M | 24M    | 13M    | 427M   | 195M   | 0.09 | 45.6        |
| socfb-Indiana69    | 30k | 1.3M| 9.4M| 181M | 60M  | 269M   | 141M   | 25B    | 13B    | 8.44 | 3212.1      |
| socfb-JMU79        | 14k | 486k| 2.7M| 54M  | 13M  | 63M    | 36M    | 5.4B   | 8.8B   | 1.24 | 511.06      |
| socfb-JohnsHopkins55 | 5.2k | 187k | 1.6M | 21M | 9.1M | 54M   | 30M | 2.1B | 1.1B | 0.46 | 202.66 |
| socfb-Maine59      | 9.1k | 243k | 1.1M | 20M | 3.5M | 21M | 13M | 1.6B | 920M | 0.36 | 101.38 |
| socfb-Maryland58   | 21k | 745k | 4.7M | 94M | 25M | 117M | 64M | 11B | 16B | 2.99 | 1229.2 |
| socfb-Michigan23   | 30k | 1.2M | 8.3M | 162M | 49M | 277M | 146M | 23B | 13B | 7.57 | 2995.83 |
| socfb-MU78         | 15k | 649k | 4.6M | 78M | 27M | 116M | 59M | 9.3B | 4.0B | 2.27 | 1018.51 |
| socfb-Northeastern19 | 14k | 382k | 1.7M | 34M | 5.4M | 32M | 19M | 3.0B | 1.7B | 0.69 | 209.78 |
| socfb-Northwestern25 | 11k | 488k | 4.4M | 69M | 25M | 146M | 85M | 9.1B | 6.2B | 2   | 1113.17 |
| socfb-NotreDame57  | 12k | 541k | 3.5M | 73M | 12M | 98M | 75M | 9.4B | 5.5B | 1.99 | 1022.4 |
| socfb-NYU9         | 22k | 716k | 3.6M | 90M | 13M | 90M | 59M | 11B | 11B | 3.04 | 1151.58 |
The table below shows the comparison of RAGE and our implementation in terms of time taken to compute graphlet counts for various graphs.

### Time in Seconds

<table>
<thead>
<tr>
<th></th>
<th>Ours</th>
<th>RAGE</th>
<th></th>
<th>Ours</th>
<th>RAGE</th>
<th></th>
<th>Ours</th>
<th>RAGE</th>
<th></th>
<th>Ours</th>
<th>RAGE</th>
<th></th>
<th>Ours</th>
<th>RAGE</th>
<th></th>
<th>Ours</th>
<th>RAGE</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Time-in-Seconds</td>
<td>Baseline%</td>
<td>RAGE%</td>
<td>did not finish for most graphs</td>
<td>Time-in-Seconds</td>
<td>Baseline%</td>
<td>RAGE%</td>
<td>did not finish for most graphs</td>
<td>Time-in-Seconds</td>
<td>Baseline%</td>
<td>RAGE%</td>
<td>did not finish for most graphs</td>
<td>Time-in-Seconds</td>
<td>Baseline%</td>
<td>RAGE%</td>
<td>did not finish for most graphs</td>
<td>Time-in-Seconds</td>
<td>Baseline%</td>
</tr>
<tr>
<td>soc-orkut</td>
<td>3.1M</td>
<td>117M</td>
<td>628M</td>
<td>44B</td>
<td>3.2B</td>
<td>48B</td>
<td>70B</td>
<td>19T</td>
<td>98T</td>
<td>1.5T</td>
<td>2694.55</td>
<td>N/A</td>
<td>0.38</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>soc-sinaweibo</td>
<td>58M</td>
<td>261M</td>
<td>212M</td>
<td>804B</td>
<td>662M</td>
<td>27B</td>
<td>259B</td>
<td>157T</td>
<td>8.48T</td>
<td>3.80T</td>
<td>33359.7</td>
<td>N/A</td>
<td>0.38</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>soc-friendster</td>
<td>65.6M</td>
<td>1.8B</td>
<td>4.17B</td>
<td>708.1B</td>
<td>8.96B</td>
<td>131.4B</td>
<td>307.5B</td>
<td>364.7T</td>
<td>247.3T</td>
<td>5.79T</td>
<td>N/A</td>
<td>N/A</td>
<td>0.38</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bio-celegans</td>
<td>453</td>
<td>2.0k</td>
<td>3.3k</td>
<td>69k</td>
<td>3.0k</td>
<td>37k</td>
<td>4.5k</td>
<td>495k</td>
<td>2.9M</td>
<td>363k</td>
<td>&lt;0.001</td>
<td>1.7</td>
<td>0.44</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bio-disease</td>
<td>516</td>
<td>1.2k</td>
<td>1.4k</td>
<td>5.4k</td>
<td>1.4k</td>
<td>923</td>
<td>42</td>
<td>18k</td>
<td>27k</td>
<td>19k</td>
<td>&lt;0.001</td>
<td>0.01</td>
<td>2.47</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bio-dmela</td>
<td>7.4k</td>
<td>26k</td>
<td>2.9k</td>
<td>572k</td>
<td>393</td>
<td>13k</td>
<td>107k</td>
<td>11M</td>
<td>9.2M</td>
<td>312k</td>
<td>0.01</td>
<td>5.3</td>
<td>0.53</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bio-yeast-protein-inter</td>
<td>1.8k</td>
<td>2.2k</td>
<td>222</td>
<td>11k</td>
<td>41</td>
<td>198</td>
<td>140</td>
<td>31k</td>
<td>72k</td>
<td>2.6k</td>
<td>&lt;0.001</td>
<td>0.43</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bio-yeast</td>
<td>1.5k</td>
<td>1.9k</td>
<td>206</td>
<td>11k</td>
<td>39</td>
<td>195</td>
<td>139</td>
<td>31k</td>
<td>72k</td>
<td>2.6k</td>
<td>&lt;0.001</td>
<td>0.43</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bio-human-gene2</td>
<td>14k</td>
<td>9.0M</td>
<td>4.9B</td>
<td>10B</td>
<td>2.3T</td>
<td>3.7T</td>
<td>90B</td>
<td>4.4T</td>
<td>5.3T</td>
<td>8.4T</td>
<td>8023.84</td>
<td>N/A</td>
<td>5515.6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bio-mouse-gene</td>
<td>44k</td>
<td>13M</td>
<td>4.6B</td>
<td>15B</td>
<td>670B</td>
<td>2.1T</td>
<td>223B</td>
<td>9.0T</td>
<td>6.7T</td>
<td>7.7T</td>
<td>N/A</td>
<td>N/A</td>
<td>0.38</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ca-CSphd</td>
<td>1.9k</td>
<td>1.7k</td>
<td>8</td>
<td>6.6k</td>
<td>0</td>
<td>5</td>
<td>8</td>
<td>9.4k</td>
<td>32k</td>
<td>93</td>
<td>&lt;0.001</td>
<td>1.25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ca-GrQc</td>
<td>4.2k</td>
<td>13k</td>
<td>48k</td>
<td>85k</td>
<td>329k</td>
<td>66k</td>
<td>1.1k</td>
<td>553k</td>
<td>406k</td>
<td>628k</td>
<td>&lt;0.001</td>
<td>0.01</td>
<td>5.99</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ca-dblp-2012</td>
<td>317k</td>
<td>1.0M</td>
<td>2.2M</td>
<td>15M</td>
<td>17M</td>
<td>4.8M</td>
<td>203k</td>
<td>252M</td>
<td>259M</td>
<td>97M</td>
<td>0.48</td>
<td>227.79</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ca-cit-HepTh</td>
<td>23k</td>
<td>2.4M</td>
<td>191M</td>
<td>1.6B</td>
<td>13B</td>
<td>47B</td>
<td>7.3B</td>
<td>538B</td>
<td>976B</td>
<td>385B</td>
<td>132.66</td>
<td>N/A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ca-cit-HepPh</td>
<td>28k</td>
<td>3.1M</td>
<td>196M</td>
<td>1.5B</td>
<td>9.8B</td>
<td>34B</td>
<td>6.1B</td>
<td>536B</td>
<td>479B</td>
<td>276B</td>
<td>125.49</td>
<td>N/A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ca-coauthors-dblp</td>
<td>540k</td>
<td>15M</td>
<td>444M</td>
<td>698M</td>
<td>15B</td>
<td>3.4B</td>
<td>31M</td>
<td>42B</td>
<td>27B</td>
<td>67B</td>
<td>40.26</td>
<td>N/A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

We take ~45 mins for soc-orkut (117M edges)

We take ~40 secs for ca-dblp (15M edges)
Most graphlet counts in orders of $10^6 - 10^{15}$

| |V| |E| | \begin{tabular}{lllllllllllll} 
 tech-as-caida2007 & 26k & 53k & 36k & 15M & 54k & 1.7M & 407k & 285M & 7.8B & 47M & 0.19 & 36.83 \\
 tech-p2p-gnutella & 63k & 148k & 2.0k & 1.6M & 16 & 826 & 42k & 15M & 8.1M & 71k & 0.02 & 7.44 \\
 tech-RL-caida & 191k & 608k & 455k & 21M & 423k & 7.4M & 40M & 583M & 1.7B & 77M & 0.39 & 71.74 \\
 tech-WHOIS & 7.5k & 57k & 782k & 5.3M & 12M & 31M & 2.9M & 229M & 566M & 194M & 0.14 & 44.52 \\
 tech-as-skitter & 1.7M & 11M & 29M & 16B & 149M & 20B & 43B & 819B & 96T & 162B & 476.06 & N/A \\
 web-edu & 3.0k & 6.5k & 10k & 81k & 40k & 4.6k & 18 & 435k & 1.3M & 186k & <0.001 & 0.52 \\
 web-google-dir & 876k & 4.3M & 13M & 687M & 40M & 382M & 38M & 4.1B & 650B & 6.7B & 4.45 & N/A \\
 web-indochina-2004 & 11k & 48k & 210k & 481k & 1.2M & 88k & 9.2k & 5.5M & 12M & 4.9M & 0.01 & 24.36 \\
 web-wikipedia-growth & 1.9M & 37M & 127M & 123B & 288M & 38B & 68B & 29T & 3.1P & 3.2T & 22389.2 & N/A \\
 web-ClueWeb09-50m & 148M & 447M & 1.2B & 494B & 5.6B & 243B & 774B & 34T & 24P & 3.4T & 15655.9 & N/A \\
 inf-italy-osm & 6.7M & 7.0M & 7.4k & 8.2M & 0 & 244 & 47k & 9.9M & 992k & 27k & 0.85 & N/A \\
 inf-openflights & 2.9k & 16k & 73k & 639k & 286k & 1.5M & 319k & 17M & 17M & 9.0M & 0.01 & 2.46 \\
 inf-power & 4.9k & 6.6k & 651 & 17k & 90 & 385 & 324 & 38k & 20k & 5.1k & <0.001 & 0.58 \\
 inf-roadNet-CA & 2.0M & 2.8M & 120k & 5.6M & 40 & 13k & 249k & 11M & 2.4M & 521k & 0.35 & N/A \\
 inf-roadNet-PA & 1.1M & 1.5M & 67k & 3.2M & 16 & 5.7k & 152k & 6.2M & 1.4M & 295k & 0.19 & N/A \\
 inf-road-usa & 24M & 29M & 439k & 50M & 90 & 21k & 1.6M & 81M & 18M & 1.5M & 4.05 & N/A \\

We take ~4.5 secs for web-google (4.3M edges)

We take ~4 secs for inf-road-usa (29M edges)
Parallel Scaling

Strong scaling results

Intel Xeon 3.10 Ghz E5-2687W server, 16 cores
Applications
Each Protein is represented by a graph

Binary label represents the function of the protein
Assume we know the labels of a few graphs

How to predict the labels of the unlabeled graphs?
Graphs

Protein Graphs

Graphlet Feature Extraction

Graph

$\mathbf{f}(G_k, G_i)$

Features

Predict Labels of Unlabeled Graphs

Model Learning
**Application:**

**Classification of Unlabeled Biological Networks**

- D&D – 1178 protein graphs. Binary labeled as Enzymes vs. Non. Enzymes
- MUTAG – 188 mutagenic compounds. Binary labeled (whether or not they have a mutagenic effect on the Gram-negative bacterium)
- 10-fold validation, Support Vector Machine
- Used 2,3,4 node graphlets as features

<table>
<thead>
<tr>
<th>graph</th>
<th>Type</th>
<th>No. Graphs</th>
<th>Accuracy(%)</th>
<th>Total Time(sec)</th>
<th>Avg Time per G (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>D&amp;D</td>
<td>Protein</td>
<td>1178</td>
<td>76.13 ± 0.03</td>
<td>1.05</td>
<td>8.95x10^{-4}</td>
</tr>
<tr>
<td>MUTAG</td>
<td>Chemicals</td>
<td>188</td>
<td>86.4 ± 0.21</td>
<td>0.14</td>
<td>7.47x10^{-4}</td>
</tr>
</tbody>
</table>
Application: Classification of Unlabeled Biological Networks

Previous work: in machine learning & biological networks
Shervashidze et. al [AISTATS 2009]

Feature Extraction Time:
D&D 2 hours, 45 mins
MUTAG 4.73 secs

<table>
<thead>
<tr>
<th>graph</th>
<th>Type</th>
<th>No. Graphs</th>
<th>Accuracy(%)</th>
<th>Total Time(sec)</th>
<th>Avg Time per G (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>D&amp;D</td>
<td>Protein</td>
<td>1178</td>
<td>76.13 ± 0.03</td>
<td>1.05</td>
<td>8.95x10^{-4}</td>
</tr>
<tr>
<td>MUTAG</td>
<td>Chemicals</td>
<td>188</td>
<td>86.4 ± 0.21</td>
<td>0.14</td>
<td>7.47x10^{-4}</td>
</tr>
</tbody>
</table>
Visualization of Human Diseasome Network
[Barabasi – 2007]

Network of disorders and disease genes linked by their known associations

Ranking by graphlet counts

Links are colored/weighted by stars of size 4 nodes

Nodes are colored/weighted by triangle counts
Work in Progress

- Local Graphlet Decomposition

Role discovery, relational learning, multi-label classification
Work in Progress

- Unbiased Estimation of Graphlet Counts

Estimation of counts of 4-vertex clique
Summary

- **Framework & Algorithms**
  - One of the first parallel approaches for graphlet counting
  - On average 460x faster than current methods
  - Edge-centric computations (only requires access to edge neighborhood)
  - Time and space-efficient
  - Sampling/estimation methods
  - Local/global counting

- **Applications**
  - **Large-scale** graph comparison, classification, and anomaly detection
  - Visual analytics and **real-time graphlet mining**
Code/Data are available online

Parallel Parameterized Graphlet Decomposition (PGD) Library

A fast parallel parameterized graphlet decomposition library for large graphs.
Please refer to the paper [1] for detailed discussion on the algorithm.

Download PGD
Graph Classification Data
Journal Paper (Under Review)

Synopsis

In short, a parameterized high performance library for counting motifs in large sparse graphs.

Setup

First, you’ll need to compile PGD.

```
$ cd path/to/pgd/
$ make
```

Afterwards, the following should work:

```
# compute the motif counts
./pgd -f sample_graph.gov
```

Code

http://nesreennahmed.com/graphlets
https://github.com/nkahmed/PGD

Data

http://networkrepository.com
Thank you!

Questions?

nesreen.k.ahmed@intel.com
http://nesreenahmed.com

MMDS Workshop, Berkeley CA
June 24th, 2016