2-3 Cuckoo Filters for Faster Triangle Listing and Set Intersection

David Eppstein¹, Michael T. Goodrich¹, Michael Mitzenmacher², and Manuel Torres¹

¹University of California, Irvine
²Harvard University
Triangle Listing Problem

• List all triangles in a network.
Set Intersection Problem

• Preprocess a collection of sets so as to quickly answer set-intersection queries.
Listing Triangles Using Set Intersection Queries

• For each edge, \( e=(v,w) \):
  – Intersect the adjacency lists for \( v \) and \( w \).
Improved Algorithm

- Order G’s vertices by a **k-degeneracy order**:
  - Each vertex has out degree at most k.
  - Can be done by a greedy algorithm.
  - k is proportional to the arboricity, $A(G)$, of the graph.

- Improved Algorithm: for each edge $e=(v,w)$:
  - Intersect the out-going adjacencies for v and w.
  - Runs in $O(m A(G))$ time
    - [e.g., see Chiba-Nishizeki ’85, Ortmann-Brandes ‘14]
Further Improvements for Real-World Computational Models

• Take advantage of bit-level operations
• This model of computation is known as the **word-RAM** or **practical-RAM** model.
  – E.g., use built-in operations of C, C++, Java, Python, T-SQL.
• Related to external-memory model
Previous Results / Our Results

• Kopelowitz et al. ‘15 introduce a set intersection data structure and use it to list the triangles in a graph $G$ in expected time $O(m(A(G) \log^2 w)/w + \log w + k)$.
  – $w$ is the word size (in bits), $k$ is output size.

• We introduce a new set intersection data structure for listing the triangles in a graph in $O(m(A(G) \log w)/w + k)$ expected time.

• We also give an external-memory version.
Review: Cuckoo Hash Tables

- Each element is mapped to 1-out-of-2 possible locations by a pair of random hash functions.

  T: cat  [ ]  [ ]  [ ]  [ ]  pig  dog

- Insertions “bounce” elements to their alternative location as needed. [Pagh, Radler ’04]

- We can add a small stash cache of size $s$ to reduce the probability of failure to be $1/n^s$.
  - Analysis involves characterizing the “cuckoo graph” defined by pairs of locations defined by each element’s 2 locations [Kirsch et. al ‘10]
Review: Cuckoo Filters

• A cuckoo table and parallel **cuckoo filter**.
  • See Fan et al. ‘14, Eppstein ‘16.
• Provides improved functionality over Bloom filters.

<table>
<thead>
<tr>
<th>T:</th>
<th>cat</th>
<th></th>
<th></th>
<th></th>
<th>pig</th>
<th>dog</th>
</tr>
</thead>
<tbody>
<tr>
<td>M:</td>
<td>111</td>
<td>000</td>
<td>000</td>
<td>000</td>
<td>000</td>
<td>000</td>
</tr>
<tr>
<td>F:</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td>4</td>
</tr>
</tbody>
</table>

1 word 1 word 1 word
New: 2-3 Cuckoo Filters

- A cuckoo table and parallel cuckoo filter, where each element is stored in \textit{2-out-of-3} possible locations.

\begin{itemize}
  \item T:
    \begin{tabular}{ccc}
      cat & dog & pig \\
      \hline
      cat & dog & pig \\
    \end{tabular}
  \item M:
    \begin{tabular}{cccccccccccc}
      111 & 000 & 111 & 111 & 000 & 111 & 111 & 111 & 000 \\
    \end{tabular}
  \item F:
    \begin{tabular}{cccccccccccc}
      2 & 4 & 5 & 2 & 5 & 4 \\
    \end{tabular}
\end{itemize}
Cuckoo Hypergraph

• Instead of analysis w/ a cuckoo graph, we use a **cuckoo hypergraph** (which is 3-uniform).
• Our 2-out-of-3 paradigm corresponds to a two-assignment.
Correctness

- **Correctness Lemma**: Any 3-uniform hypergraph has a 2-assignment if and only if each of its connected components is acyclic or unicyclic.
Small Components

• Let $C_v$ be the component containing $v$ in the randomly chosen hypergraph, and let $E_v$ represent the set of edges in $C_v$.

**Lemma 7.** There exists a constant $\beta \in (0, 1)$ such that for any fixed vertex $v$ and integer $k > 0$,

$$\Pr(|E_v| \geq k) \leq \beta^k.$$  

– Implies that components are small with high probability.
Cyclomatic Numbers

Let the **cyclomatic number** $\alpha(H)$ be the smallest number of triangles which should be removed from a 3-uniform hypergraph $H$ in order to make $H$ become acyclic.

**Lemma 8.** For every vertex $v$ and $t, k \geq 1, k \leq m^{1/3}$,

$$\Pr(\alpha(C_v) \geq a \mid |E_v| \leq k) \leq 2 \left( \frac{126e^5 k^3}{m} \right)^a,$$
2-3 Cuckoo Hashing with a Stash

• Skipping two pages of equations...

Theorem 2. For any constant integer $s \geq 1$, for a sufficiently large constant $C$, the size $S$ of the stash in a 2-3 cuckoo hash table after all items have been inserted satisfies $\Pr(S \geq s) = \tilde{O}(n^{-s})$.

The $\tilde{O}$ notation allows for extra polylogarithmic factors.
Intersecting Two 2-3 Cuckoo Filters

- At least one location must overlap:

\[ A = (M_i \text{ AND NOT } (F_i \text{ XOR } F_j)) \]

- We may have false-positives, though.

\[
\begin{array}{ccccccc}
T_1: & \text{cat} & \text{dog} & \text{pig} & & \text{cat} & \text{pig} & \text{dog} \\
M_1: & 111 & 000 & 111 & 111 & 000 & 111 & 111 & 111 & 000 \\
F_1: & 2 & 4 & 5 & & 2 & 5 & 4 \\
\end{array}
\]

\[
\begin{array}{ccccccc}
T_2: & \text{cat} & & & \text{fox} & \text{pig} & \text{pig} & \text{fox} \\
M_2: & 111 & 111 & 000 & 000 & 111 & 111 & 111 & 111 & 000 \\
F_2: & 2 & 2 & & 4 & 5 & 5 & 4 \\
\end{array}
\]
Set Intersection Analysis

- By above analysis, we can construct 2-3 cuckoo filters of size at least 2 with constant-size stashes with probability at least $1 - 1/w^c$.

- Choosing a fingerprint size of at least $\log w$ bits implies false positives occur with probability less than $1/w$.
  - Expected number of false positives is $O(n/w)$.

- Thus, expected time for a set intersection query is $O(n(\log w)/w + k)$, where $k$ is the output size.
Triangle Listing Algorithm

• Order vertices by a k-degenerate ordering.
• Build a 2-3 cuckoo filter for each out-going adjacency list. (Each is size \( \frac{A(G) \log w}{w} \).)
• For each edge \( e = (v, w) \):
  – Intersect the out-going adjacency lists for \( v \) and \( w \) by the above set-intersection algorithm.
  – For any adjacency lists where 2-3 cuckoo construction failed, do intersection by merging.
    • Probability of failure is at most \( \frac{1}{w^c} \).
• Expected running time: \( O(m(A(G) \log w)/w + k) \).
Preliminary Experiments

- This is admittedly a theory paper, but we nevertheless did some preliminary experiments.

![Comparison of Algorithms on Erdos-Renyi Graphs, $p = 1/\sqrt{n}$]

- Word size = 64
- Fingerprint size = 8
We also have an external-memory algorithm.

We can list all the triangles in $G$ using an expected number of I/Os that is:

$$O(\text{sort}(n \cdot A(G)) + \text{sort}(m \cdot A(G) \log w)/w) + \text{sort}(k)).$$
Conclusion

• 2-3 cuckoo hash-filters are simple and lead to improved set-intersection queries.

• Open problem:
  – Are there other applications for 2-3 cuckoo hash-filters?