Doulion: counting triangles in massive graphs with a coin

(CSCI6220 Course Project)

Presentation: Sola, Colin, Manqing, Rufeng

Nov. 22nd 2019

Doulion

Tsourakakis, Charalampos E., et al. "Doulion: counting triangles in massive graphs with a coin." Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2009.

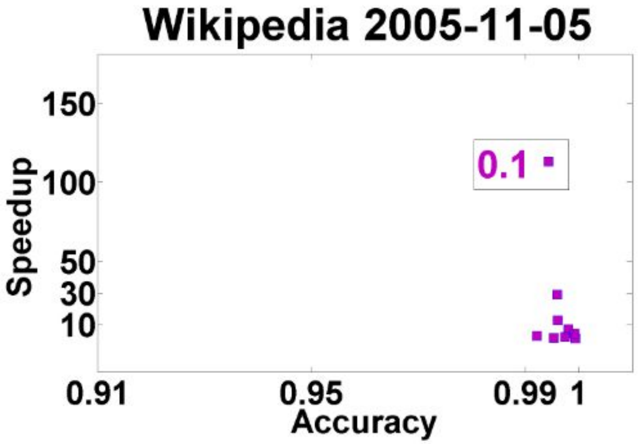
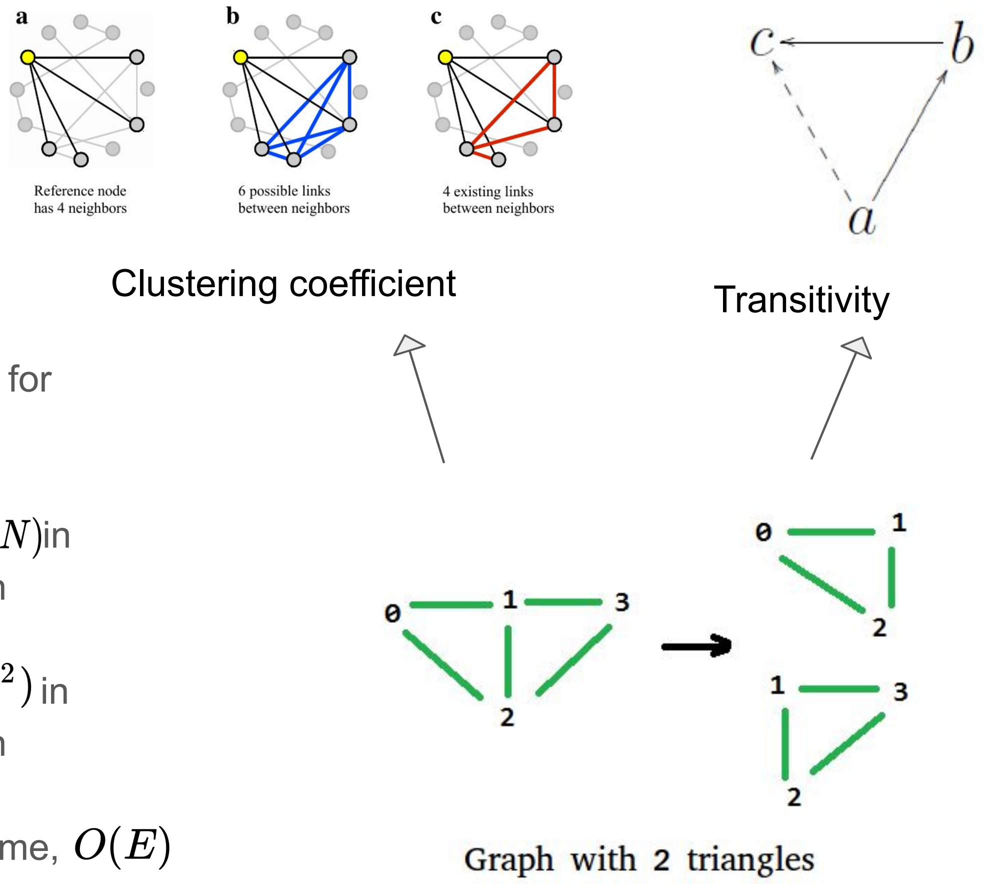


Figure 1: Speedup vs. Accuracy for the Wikipedia Graph snapshot on 2005 Nov. The graph has ≈ 1,7M nodes and 20M edges. As we see, even when keeping 10% of the edges of the initial graph accuracy is 99.5%. For p's ranging from 10% to 90% the mean accuracy is 99.7%, the accuracy standard deviation 0.0023 and the mean speedup 19.4.

- **A practical and effective meta-framework for generic triangle counting algorithms**
- **Main idea**: Chaining the random sampling with a straightforward triangle counting algorithm as a black box
- **Result:**
  - 166 experiments on real-world networks and on synthetic datasets
  - High accuracy more than 99%
  - **Significant speedups 130 times faster performance. (Using “NodeIterator”)**

Model Description

- Motivation
  - Triangle counting is essential for graph mining applications (i.e. clustering coefficient, transitivity, etc.)
  - To develop a practical, effective method for extreme large dataset
    - Deterministic methods are costly
      - Node-iterator: $O(Nd^2) = O(N)$in time, $O(d_{max}) = O(N)$ in space
      - Edge-iterator: $O(E) = O(N^2)$ in time, $O(2d_{max}) = O(N)$ in space
      - Trace Exact: $> O(N^2)$ in time, $O(E)$ in space



Model Description

- Design (pseudo code)

```
Require: Unweighted Graph G(V, E)
Require: Sparsification parameter p
Output: Δ'(G) global triangle estimation
  for each edge e_j do
    Toss a biased coin with success probability p
    if success then
      w(e_j) ← 1/p
    else
      w(e_j) ← 0
    end if
  end for
  Δ'(G) ← TRIANGLECOUNTINGALGORITHM(G)
  return Δ'(G)
```

Algorithm 1: The DOULION counting framework

```
Require: Unweighted Graph G(V, E)
Require: Sparsification parameter p
Output: Δ'(G) global triangle estimation
  Δ'(G) ← 0
  for each edge e_j do
    Toss a biased coin with success probability p
    if success then
      w(e_j) ← 1/p
    else
      w(e_j) ← 0
    end if
  end for
  for v ∈ V(G) do
    for all pairs of neighbors (u, w) of v do
      if (u, w) ∈ E(G) then
        if u < v < w then
          Δ'(G) ← Δ'(G) + 1
        end if
      end if
    end for
  end for
  Δ'(G) ← Δ'(G) * 1/p^3
  return Δ'(G)
```

Algorithm 2: The DOULION-NODEITERATOR algorithm

Node iterator

Performance Guarantees

"...we performed 166 experiments on real-world networks and on synthetic datasets as well, where we show that our method works with high accuracy, typically more than 99% and gives significant speedups, resulting in even ≈ 130 times faster performance."

- Theorem 1: DOULION expected value
- Theorem 2: DOULION variance
- Theorem 3: Accuracy
- Speedup, Eq (3) (for NodeIterator)

Theorem 1: Expected Value of DUOLION

The expected value of number of triangles in G' is equal to the actual number in G.

Proof: Let number of original triangles in graph G be $\triangle$, and let δ_i be an indicator variable representing whether triangle i is in the new graph G'. Let X be the number of triangles returned by the DUOLION algorithm. From the algorithm,

$X = \sum_{i=1}^{\triangle} \frac{1}{p^3}\delta_i$.

$$\mathbb{E}[X] = \mathbb{E}[\sum_{i=1}^{\triangle} \frac{1}{p^3}\delta_i] = \frac{1}{p^3}\sum_{i=1}^{\triangle}\mathbb{E}[\delta_i]$$

$\mathbb{E}[\delta_i]$ is 1 if all 3 edges in triangle i are kept, each with probability p. So $\mathbb{E}[\delta_i] = p^3$

$$\frac{1}{p^3}\sum_{i=1}^{\triangle}\mathbb{E}[\delta_i] = \frac{1}{p^3}\triangle p^3 = \triangle$$

.

Theorem 2: DOULION Variance

Let $\triangle$ be the total number of triangles in G. The variance is equal to:

$$Var(X) = \frac{\Delta(p^3 - p^6) + 2k(p^5 - p^6)}{p^6}$$

Where k is the number of pairs of triangles that are not edge disjoint.

Proof:

The random indicator variables δ_i are not independent (Fig. 3.2.1). Therefore,

$$Var(X) = Var(\frac{1}{p^3}\sum_{i=1}^{\Delta}\delta_i) = \frac{1}{p^6}\sum_{i=1}^{\Delta}\sum_{j=1}^{\Delta}Cov(\delta_i, \delta_j)$$

Theorem 2: DOULION Variance

We have

$$Var(X) = Var(\frac{1}{p^3}\sum_{i=1}^{\Delta}\delta_i) = \frac{1}{p^6}\sum_{i=1}^{\Delta}\sum_{j=1}^{\Delta}Cov(\delta_i, \delta_j)$$

And

$$Cov(\delta_i, \delta_j) = E(\delta_i\delta_j) - E[\delta_i]E[\delta_j]$$

There are Δ^2 terms in this sum. Δ of them are the variance of the indicator variables.

$$\sum_{i=1}^{\Delta}Cov(\delta_i, \delta_i) = \sum_{i=1}^{\Delta}E[\delta_i^2] - E[\delta_i]^2$$

Theorem 2: DOULION Variance

Since $E[\delta_i^2] = E[\delta_i] = p^3$ we have

$$\sum_{i=1}^{\Delta} Cov(\delta_i, \delta_i) = \Delta(p^3 - p^6)$$

The rest 2 * (Δ choose 2) terms corresponding to the pairs of indicator variables. Let $k$out of 2 * (Δ choose 2) pairs of indicator variables corresponding to triangles that share one edge. In that case

$$Cov(\delta_i, \delta_j) = E[\delta_i \delta_j] - E[\delta_i]E[\delta_j] = p^5 - p^6.$$

For the rest 2 * (Δ choose 2) - k terms, where triangles don't share an edge and thus independent with each other. We have

$$Cov(\delta_i, \delta_j) = E[\delta_i \delta_j] - E[\delta_i]E[\delta_j] = p^6 - p^6 = 0.$$

Overall we have

$$Var(X) = \frac{1}{p^6}(\Delta(p^3 - p^6) + 2k(p^5 - p^6))$$

Theorem 3: Accuracy Bounds of DOULION

$$\mathbb{P}(|X - \triangle| > \epsilon\triangle) \leq \frac{(p^3 - p^6)}{p^6 \epsilon^2 \triangle} + \frac{2k(p^5 - p^6)}{p^6 \epsilon^2 \triangle^2}$$

Proof: By applying Chebyshev's inequality with the calculated expected value and variance, we get

$$\mathbb{P}(|X - \triangle| > \epsilon\triangle) \leq \frac{\mathrm{Var}(X)}{\epsilon^2 \triangle^2} = \frac{1}{p^6 \epsilon^2 \triangle^2}(\triangle(p^3 - p^6) + 2k(p^5 - p^6))$$

$$= \frac{(p^3 - p^6)}{p^6 \epsilon^2 \triangle} + \frac{2k(p^5 - p^6)}{p^6 \epsilon^2 \triangle^2}$$

Speedup on Deterministic Algorithms

Deterministic algorithm NodeIterator runs in $O\left(\sum_v \deg(v)^2\right)$ time. Each vertex in the DOULLION graph G' has expected degree

$$\mathbb{E}[D(v)] = p\deg(v)$$

Therefore, running NodeIterator on new graph G' has expected runtime

$$O\left(\sum_v D(v)^2\right) = O\left(\sum_v (p\deg(v))^2\right) = O\left(p^2 \sum_v (\deg(v))^2\right)$$

This results in $\frac{1}{p^2}$ speedup.

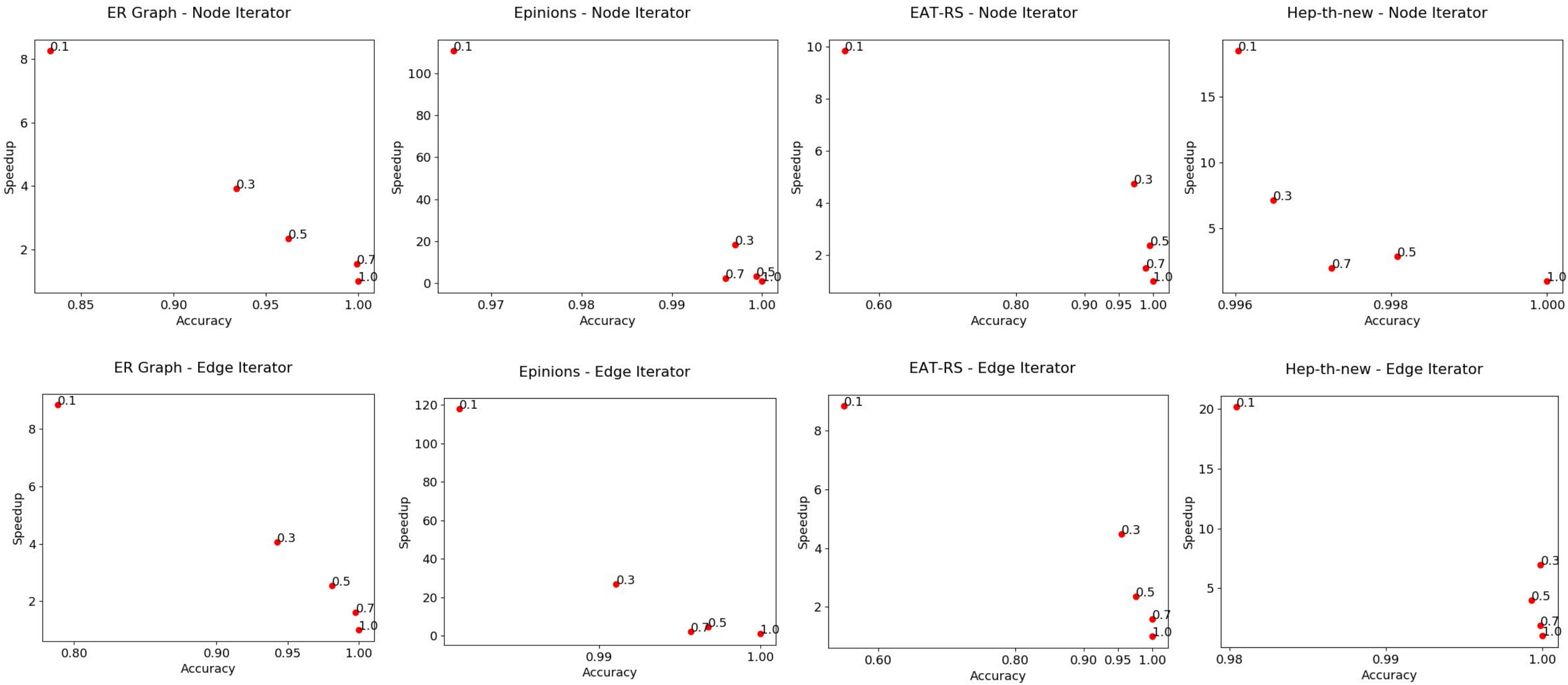
Empirical Results

- Datasets

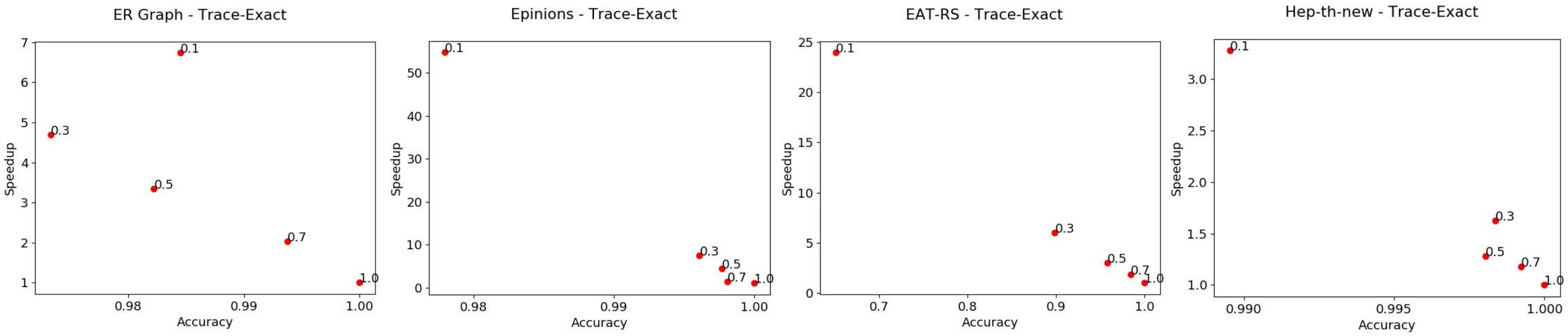
Dataset	N	E	Edge Density	Description
ER	25,000	-	-	A synthetic network
Epinions	75,877	405,740	0.0001	A product review social network
EAT-RS	23,219	304,937	0.001	A language network
HEP-th-new	27,770	352,285	0.001	An academic collaboration network

- Algorithms
  - Node Iterator
  - Edge Iterator
  - Trace Exact

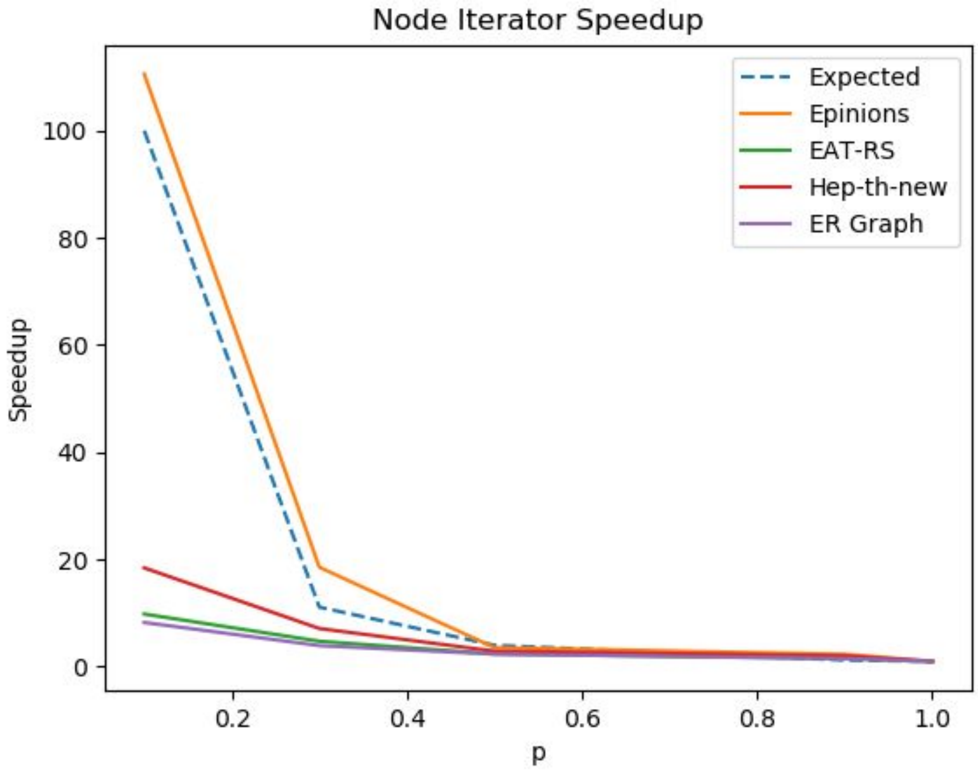
Results: Accuracy vs Speedup



Results: Accuracy vs Speedup



Results: Runtime



Discussion

For performance on specific graphs,

- EAT-RS tended to show greater drop-off in accuracy at low *p* values
- Of the two dataset having similar N and E with the synthetic ER-graph, Hep-th-new shows similar behavior with the ER-graph, with a similar speedup yet much better accuracy.
- Using the *NodeIterator* algorithm, Epinions has the speedup closest to the theoretical expected value (when *p=0.1,* the speedup is around 100= $\frac{1}{p^2}$). This leads to an proposition that the neighborhood of the Epinions' nodes might be highly independent.

Discussion

In general,

- At lower *p* values, accuracy can vary a lot for certain graphs. However, when $p > 0.3$, the accuracy spikes up and goes beyond *0.9* with high probability.
- Overall , the "speedup-accuracy" scatter plots show a universal across all graphs and deterministic algorithms: with the *p* increases for Doulion (less sparsity for sampling), the accuracy increases while the speedup drops, with a relationship almost linearly.

Questions & Comments