

Counting and Sampling Triangles from a Graph Stream

A. Pavan, Kanat Tangwongsan, Srikanta Tirthapura, Kun-Lung Wu

2013, VLDB Conference

Marten Heidemeyer

Presentation for CMPT 843 Class

Outline

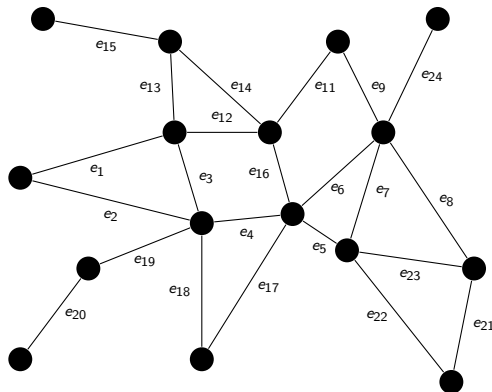
- 1 Motivation
- 2 Neighborhood Sampling
- 3 Counting Triangles
- 4 Bulk Processing
- 5 Sampling Triangles
- 6 Experiments and Results
- 7 Discussion

Motivation

- Triangle Counting is important in:
 - social networks
 - spam and fraud detection
 - link classification and recommendation
- Why use a streaming algorithm?
 - real-time processing of live data
 - analysis of **large** disk-resident graph data

The Adjacency Stream Model

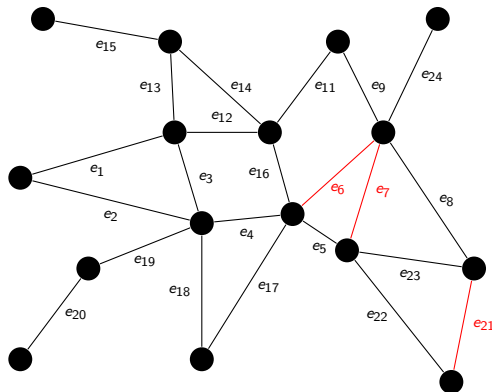
1



... e_{23} e_{17} e_4 e_{11} e_{12} e_1 e_2 e_5 e_{16} e_{18} e_{19} e_6 e_{21} e_7

The Adjacency Stream Model

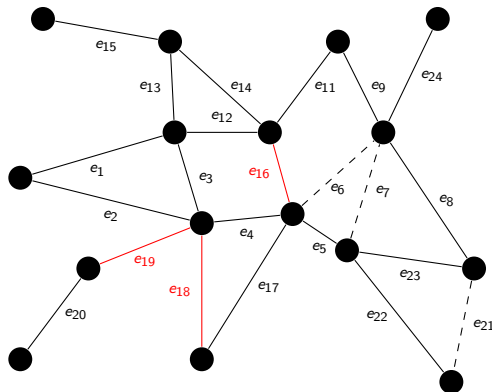
2



... e23 e17 e4 e11 e12 e1 e2 e5 e16 e18 e19 e6 e21 e7

The Adjacency Stream Model

3



... e_{24} e_{14} e_{22} e_{23} e_{17} e_4 e_{11} e_{12} e_1 e_2 e_5 e_{16} e_{18} e_{19}

Neighborhood Sampling

- first sample a random edge r_1 from the edge stream
- then sample a random edge r_2 from those edges that appear after r_1 and are adjacent to r_1
- try to close r_1 and r_2 with a subsequent edge, to form a triangle t
- simple data structure: (r_1, r_2, t, c)
- sample r_1 and r_2 using *reservoir sampling*

The Algorithm

Upon receiving edge e at time step m

if $\text{coin}(1/m) = \text{"head"}$ **then**

$(r_1, r_2, t, c) = (e, \emptyset, \emptyset, 0)$

else

if e is adjacent to r_1 **then**

$c \leftarrow c + 1$

if $\text{coin}(1/c) = \text{"head"}$ **then**

$(r_2, t) \leftarrow (e, \emptyset)$

else

if e forms a triangle with r_1 and r_2 **then**

$t \leftarrow \{r_1, r_2, e\}$

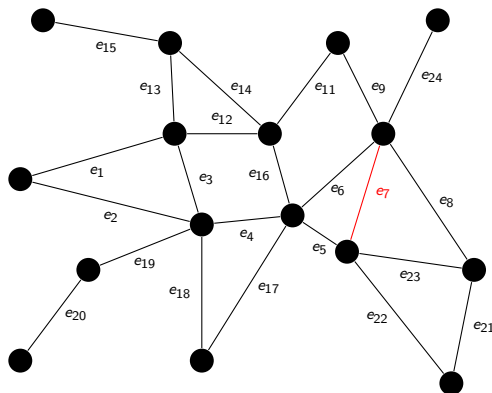
end if

end if

end if

end if

Example 1

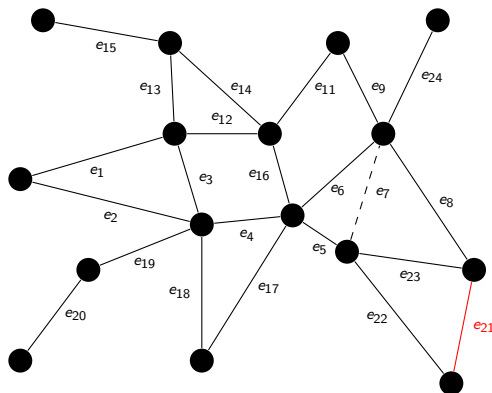


$\dots e_{23} e_{17} e_4 e_{11} e_{12} e_1 e_2 e_3 e_8 e_5 e_{14} e_{23} e_{21} \boxed{e_7} \dots$

$m = 1042$

estimator : $r_1 = e_{15}$ $r_2 = e_{39}$ $t = \emptyset$ $c = 15$

Example 2

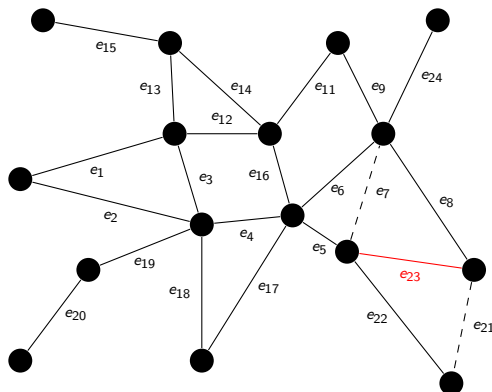


$\dots e_{13} e_{23} e_{17} e_4 e_{11} e_{12} e_1 e_2 e_3 e_8 e_5 e_{14} e_{23} \boxed{e_{21}} \dots$

$m = 1043$

estimator : $r_1 = e_7$ $r_2 = \emptyset$ $t = \emptyset$ $c = 0$

Example 3

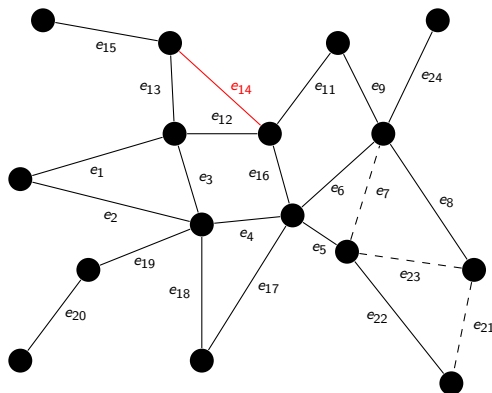


$\dots e_9 e_{13} e_{23} e_{17} e_4 e_{11} e_{12} e_1 e_2 e_3 e_8 e_5 e_{14} \boxed{e_{23}} \dots$

$m = 1044$

estimator : $r_1 = e_7$ $r_2 = e_{23}$ $t = \emptyset$ $c = 1$

Example 4

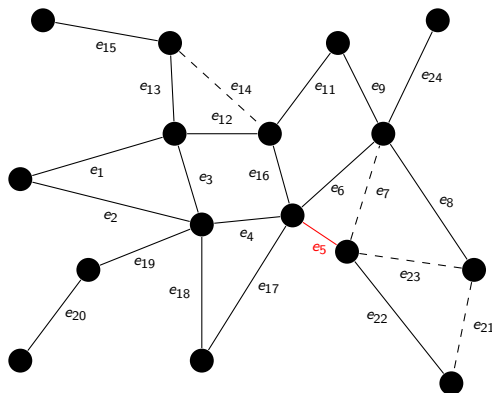


$\dots e_{16} e_9 e_{13} e_{23} e_{17} e_4 e_{11} e_{12} e_1 e_2 e_3 e_8 e_5$ e_{14} \dots

$m = 1045$

estimator : $r_1 = e_7$ $r_2 = e_{23}$ $t = \emptyset$ $c = 1$

Example 5

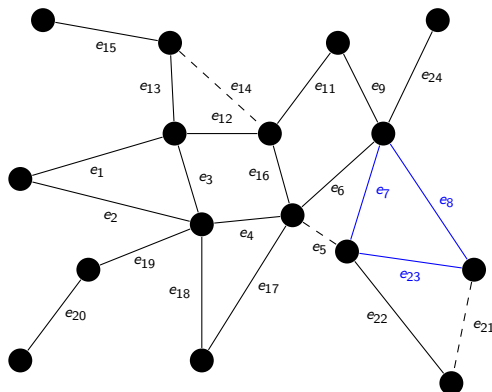


$\dots e_{18} e_{16} e_9 e_{13} e_{23} e_{17} e_4 e_{11} e_{12} e_1 e_2 e_3 e_8$ e_5 \dots

$m = 1046$

estimator : $r_1 = e_7$ $r_2 = e_{23}$ $t = \emptyset$ $c = 2$

Example 6



$\dots e_{20} e_{18} e_{16} e_9 e_{13} e_{23} e_{17} e_4 e_{11} e_{12} e_1 e_2 e_3$ e_8 \dots

$m = 1047$

estimator : $r_1 = e_7$ $r_2 = e_{23}$ $t = \{e_7, e_{23}, e_8\}$ $c = 3$

Expectation value of τ

- We want to find the number of triangles $\tau(G)$
- Let t and c be the values the neighborhood sampling algorithm maintains and m be the number of edges observed so far. Define:

$$\tilde{\tau} = \begin{cases} c \times m & \text{if } t \neq \emptyset \\ 0 & \text{otherwise} \end{cases}$$

Then $\mathbf{E}[\tilde{\tau}] = \tau(G)$

An (ϵ, δ) -approximation to the triangle count

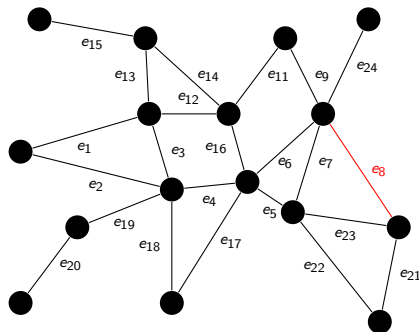
- Take the average of r such estimators with
- With probability $1 - \delta$ the average of the estimators is in
$$[(1 - \epsilon)\tau(G), (1 + \epsilon)\tau(G)]$$

An (ϵ, δ) -approximation to the triangle count

- Take the average of r such estimators with
- With probability $1 - \delta$ the average of the estimators is in $[(1 - \epsilon)\tau(G), (1 + \epsilon)\tau(G)]$
- Relation between number of estimators and error of the approximation:

$$r \geq \frac{6}{\epsilon^2} \frac{m\Delta}{\tau(G)} \log\left(\frac{2}{\delta}\right)$$

- LiveJournal:
 - given $m = 34.7M$, $\Delta = 14815$, $\tau(G) = 177,820,130$
 - chose $\epsilon = 0.1$, $\delta = 0.05$
 - then $r \geq 6.3M$



estimator1 : $r_1 = e_{101}$ $r_2 = \emptyset$ $t = \emptyset$ $c = 0$
estimator2 : $r_1 = e_9$ $r_2 = e_{23}$ $t = \emptyset$ $c = 13$
estimator3 : $r_1 = e_8$ $r_2 = \emptyset$ $t = \emptyset$ $c = 33$
estimator4 : $r_1 = e_{99}$ $r_2 = e_{42}$ $t = \{e_{99}, e_{42}, e_{101}\}$ $c = 33$
estimator5 : $r_1 = e_7$ $r_2 = \emptyset$ $t = \emptyset$ $c = 0$
estimator6 : $r_1 = e_8$ $r_2 = e_{23}$ $t = \{e_7, e_{23}, e_8\}$ $c = 31$
estimator7 : $r_1 = e_{42}$ $r_2 = \emptyset$ $t = \emptyset$ $c = 0$
estimator8 : $r_1 = e_{17}$ $r_2 = e_{42}$ $t = \{e_{17}, e_{42}, e_{23}\}$ $c = 92$

...

$m = 1042$

... e_{20} e_{18} e_{16} e_9 e_{13} e_{23} e_{17} e_4 e_{11} e_{12} e_1 e_2 e_3 e_8 ...

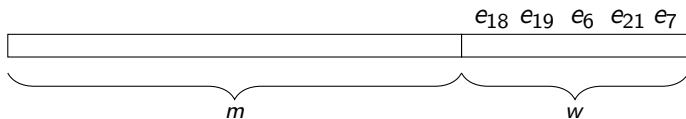
Nearly-Linear Time Triangle Counting

- so far we have a $O(mr)$ -time implementation
- with a small constant factor increase in space, we are able to achieve $O(m + r)$ -time bound
- **Bulk Processing** Process edges in bulk. Read w edges at a time and update all estimators simultaneously.

... e_{23} e_{17} e_4 e_{11} e_{12} e_1 e_2 e_5 e_{16} e_{18} e_{19} e_6 e_{21} e_7

Bulk Processing Step 1: Resample Level 1 edges

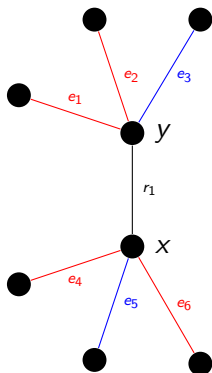
- sample new r_1 edges from the batch of size w . Keep current edge with probability $\frac{m}{w+m}$ and with remaining probability, replace it with edge uniformly chosen from B .
- for each estimator: draw random number between 1 and $m + w$.



Bulk Processing Step 2: Identify Level-2 candidates and sample from them

- The sample space we want to sample r_2 edges from, is those edges that are adjacent to r_1 and arrive after r_1 .
- Let $N(r_1)$ be those edges that arrive after r_1 in the stream.
- Suppose for each estimator we have the following values ($r_1 = (x, y)$):
 - $|N(r_1)| = c^- + c^+$
 - c^- : # edges in $N(r_1)$ before the badge.
 - $c^+ = a + b$: # of edges in $N(r_1)$ inside the badge.
 - a : # of edges in $N(r_1)$, sharing endpoint x with r_1 , inside badge.
 - b : # of edges in $N(r_1)$, sharing endpoint y with r_1 , inside badge.

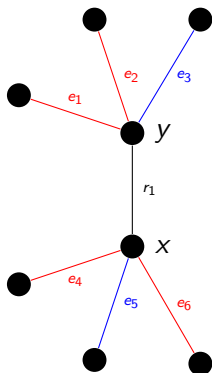
Example: c^-, c^+, a, b



- $|N(r_1)| = c^- + c^+ = 6$
- $c^- = 2$
- $c^+ = a + b = 4$
- $a = 2$
- $b = 2$

e_6 e_4 e_{23} e_1 e_{13} e_2 e_{42} e_{17} e_5 e_3 r_1

Example: c^-, c^+, a, b

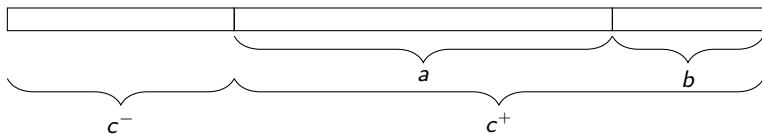


- $|N(r_1)| = c^- + c^+ = 6$
- $c^- = 2$
- $c^+ = a + b = 4$
- $a = 2$
- $b = 2$
- with probability $\frac{c^+}{c^+ + c^-}$ sample a r_1 adjacent edge from the badge as new r_2

e₆ e₄ e₂₃ e₁ e₁₃ e₂ e₄₂ e₁₇ e₅ e₃ r₁

Sampling new r_2 edges from the badge

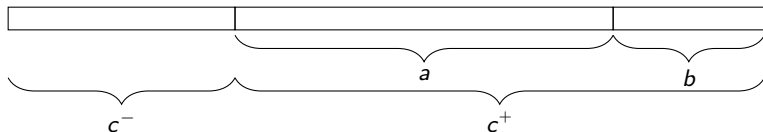
- For each estimator draw random number φ between 1 and $c^- + c^+$



- if $\varphi \leq c^-$ don't sample new r_2 edge
- if $c^- \leq \varphi \leq c^- + a$ sample one of the edges that share endpoint x
- else sample one of the edges that share endpoint y

Sampling r_2 edges summary

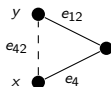
- paper presents simple algorithm to get c^-, c^+, a, b which goes over the badge once
- then we go over all estimators to sample the r_2 edges
- then go over badge again to find edge that corresponds to offset in range $c^-, c^- + a$ or $c^- + a, c^- + a + b$



Bulk Processing Step 3: Detect edges that close the wedges

- For each estimator with r_1 and r_2 edges, check if badge contains an edge that comes after r_2 and closes r_1, r_2 to form a triangle
- Create hashtable that maps the needed edges to the estimators.
- Check if badge contains this edge and whether it comes after r_2

est.1: $r_1 = e_{12}, r_2 = e_4, t = \emptyset \dots$



x, y	est.1	e_2 e_{42} e_{17} e_5 e_3 e_4
c, d	est.3	
u, v	est.12	

