Counting Triangles under Updates in Worst-Case Optimal Time

Ahmet Kara, Hung Q. Ngo, Milos Nikolic
Dan Olteanu, and Haozhe Zhang

fdbresearch.github.io

ICDT 2019, Lisbon
Problem Setting

The triangle count $Q$ returns the number of tuples in the join of $R$, $S$, and $T$.

Maintain the triangle count $Q$ under single-tuple updates to $R$, $S$, and $T$!
Data Model

- Relations are functions mapping tuples to multiplicities.

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$R$</td>
<td>$S$</td>
<td>$T$</td>
<td></td>
</tr>
<tr>
<td>$A B$</td>
<td>$B C$</td>
<td>$C A$</td>
<td></td>
</tr>
<tr>
<td>$a_1 b_1$</td>
<td>$b_1 c_1$</td>
<td>$c_1 a_1$</td>
<td>2</td>
</tr>
<tr>
<td>$a_2 b_1$</td>
<td>$b_1 c_2$</td>
<td>$c_2 a_1$</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$c_2 a_2$</td>
<td>3</td>
</tr>
</tbody>
</table>

(positive for insertions, negative for deletions)
Data Model

- Relations are functions mapping tuples to multiplicities.

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>#</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a₁</td>
<td>b₁</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>a₂</td>
<td>b₁</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>#</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>b₁</td>
<td>c₁</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>b₁</td>
<td>c₂</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>#</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c₁</td>
<td>a₁</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>c₂</td>
<td>a₁</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>c₂</td>
<td>a₂</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>#</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>B</td>
<td>C</td>
<td></td>
</tr>
<tr>
<td>a₁</td>
<td>b₁</td>
<td>c₁</td>
<td>2 ⋅ 2 ⋅ 1 = 4</td>
</tr>
</tbody>
</table>
Data Model

- Relations are functions mapping tuples to multiplicities.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>#</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>a₁</td>
<td>b₁</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>a₂</td>
<td>b₁</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>B</th>
<th>C</th>
<th>#</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>b₁</td>
<td>c₁</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>b₁</td>
<td>c₂</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>C</th>
<th>A</th>
<th>#</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>c₁</td>
<td>a₁</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>c₂</td>
<td>a₁</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>c₂</td>
<td>a₂</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>#</th>
</tr>
</thead>
<tbody>
<tr>
<td>R · S · T</td>
<td>a₁</td>
<td>b₁</td>
<td>c₁</td>
<td>2 · 2 · 1 = 4</td>
</tr>
<tr>
<td></td>
<td>a₁</td>
<td>b₁</td>
<td>c₂</td>
<td>2 · 1 · 3 = 6</td>
</tr>
<tr>
<td></td>
<td>a₂</td>
<td>b₁</td>
<td>c₂</td>
<td>3 · 1 · 3 = 9</td>
</tr>
</tbody>
</table>
Data Model

- Relations are functions mapping tuples to multiplicities.
- $Q() = \sum_{a,b,c} R(a, b) \cdot S(b, c) \cdot T(c, a)$

<table>
<thead>
<tr>
<th>$R$</th>
<th>$S$</th>
<th>$T$</th>
<th>$R \cdot S \cdot T$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A B$</td>
<td>$B C$</td>
<td>$C A$</td>
<td>$A B C$</td>
</tr>
<tr>
<td>$#$</td>
<td>$#$</td>
<td>$#$</td>
<td>$#$</td>
</tr>
<tr>
<td>$a_1$</td>
<td>$b_1$</td>
<td>$c_1$</td>
<td>$a_1 b_1 c_1$</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>$2 \cdot 2 \cdot 1 = 4$</td>
</tr>
<tr>
<td>$a_2$</td>
<td>$b_1$</td>
<td>$c_2$</td>
<td>$a_1 b_1 c_2$</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>3</td>
<td>$2 \cdot 1 \cdot 3 = 6$</td>
</tr>
<tr>
<td></td>
<td>$b_1$</td>
<td>$c_2$</td>
<td>$a_2 b_1 c_2$</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>3</td>
<td>$3 \cdot 1 \cdot 3 = 9$</td>
</tr>
</tbody>
</table>

\[ Q() = 4 + 6 + 9 = 19 \]
Data Model

- Relations are functions mapping tuples to multiplicities.
- \( Q() = \sum_{a,b,c} R(a, b) \cdot S(b, c) \cdot T(c, a) \)
- A single-tuple update is a relation mapping a tuple to a non-zero value (positive for insertions, negative for deletions)

\[
\begin{array}{ccc|c}
R & & \# \\
A & B & \\
\hline
a_1 & b_1 & 2 \\
a_2 & b_1 & 3 \\
\end{array}
\quad
\begin{array}{ccc|c}
S & & \# \\
B & C & \\
\hline
b_1 & c_1 & 2 \\
b_1 & c_2 & 1 \\
\end{array}
\quad
\begin{array}{ccc|c}
T & & \# \\
C & A & \\
\hline
c_1 & a_1 & 1 \\
c_2 & a_1 & 3 \\
c_2 & a_2 & 3 \\
\end{array}
\quad
\begin{array}{ccc|c}
R \cdot S \cdot T & & \# \\
A & B & C & \\
\hline
a_1 & b_1 & c_1 & 2 \cdot 2 \cdot 1 = 4 \\
a_1 & b_1 & c_2 & 2 \cdot 1 \cdot 3 = 6 \\
a_2 & b_1 & c_2 & 3 \cdot 1 \cdot 3 = 9 \\
\end{array}
\]

\( \delta R = \{(a_2, b_1) \mapsto -2\} \)

\[
\begin{array}{ccc|c}
A & B & \# \\
\hline
a_2 & b_1 & -2 \\
\end{array}
\]

\[
\begin{array}{c|c}
Q() & \# \\
\hline
\emptyset & \\
( ) & 4 + 6 + 9 = 19 \\
\end{array}
\]
Data Model

- Relations are functions mapping tuples to multiplicities.
- \( Q() = \sum_{a,b,c} R(a, b) \cdot S(b, c) \cdot T(c, a) \)
- A single-tuple update is a relation mapping a tuple to a non-zero value (positive for insertions, negative for deletions)

<table>
<thead>
<tr>
<th>( R )</th>
<th>( S )</th>
<th>( T )</th>
<th>( R \cdot S \cdot T )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( A )</td>
<td>( B )</td>
<td>#</td>
<td>( A )</td>
</tr>
<tr>
<td>( a_1 )</td>
<td>( b_1 )</td>
<td>2</td>
<td>( b_1 )</td>
</tr>
<tr>
<td>( a_2 )</td>
<td>( b_1 )</td>
<td>3</td>
<td>( b_1 )</td>
</tr>
<tr>
<td>( a_2 )</td>
<td>( b_1 )</td>
<td>3</td>
<td>( c_2 )</td>
</tr>
<tr>
<td>( a_2 )</td>
<td>( b_1 )</td>
<td>3</td>
<td>( a_1 )</td>
</tr>
<tr>
<td>( a_2 )</td>
<td>( b_1 )</td>
<td>3</td>
<td>( a_1 )</td>
</tr>
<tr>
<td>( a_2 )</td>
<td>( b_1 )</td>
<td>3</td>
<td>( a_2 )</td>
</tr>
</tbody>
</table>

\[ \delta R = \{(a_2, b_1) \mapsto -2\} \]

<table>
<thead>
<tr>
<th>( A )</th>
<th>( B )</th>
<th>#</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a_2 )</td>
<td>( b_1 )</td>
<td>-2</td>
</tr>
</tbody>
</table>

\[ Q() \]

<table>
<thead>
<tr>
<th>( \emptyset )</th>
<th>#</th>
</tr>
</thead>
<tbody>
<tr>
<td>( ( ) )</td>
<td>4 \cdot 6 \cdot 9 = 19</td>
</tr>
</tbody>
</table>
Data Model

- Relations are functions mapping tuples to multiplicities.
- \( Q() = \sum_{a,b,c} R(a, b) \cdot S(b, c) \cdot T(c, a) \)
- A single-tuple update is a relation mapping a tuple to a non-zero value (positive for insertions, negative for deletions)

<table>
<thead>
<tr>
<th>( R )</th>
<th>( S )</th>
<th>( T )</th>
<th>( R \cdot S \cdot T )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( A ) ( B )</td>
<td>#</td>
<td>( B ) ( C )</td>
<td>#</td>
</tr>
<tr>
<td>( a_1 ) ( b_1 )</td>
<td>2</td>
<td>( b_1 ) ( c_1 )</td>
<td>2</td>
</tr>
<tr>
<td>( a_2 ) ( b_1 )</td>
<td>3</td>
<td>( b_1 ) ( c_2 )</td>
<td>1</td>
</tr>
<tr>
<td>( a_2 ) ( b_1 )</td>
<td>1</td>
<td>( c_2 ) ( a_2 )</td>
<td>3</td>
</tr>
</tbody>
</table>

\[ \delta R = \{(a_2, b_1) \mapsto -2\} \]

<table>
<thead>
<tr>
<th>( A ) ( B )</th>
<th>#</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a_2 ) ( b_1 )</td>
<td>-2</td>
</tr>
</tbody>
</table>

\[ Q() \]

<table>
<thead>
<tr>
<th>( \emptyset )</th>
<th>#</th>
</tr>
</thead>
<tbody>
<tr>
<td>( )</td>
<td>4 + 6 + 9 = 19</td>
</tr>
</tbody>
</table>
**Data Model**

- Relations are functions mapping tuples to multiplicities.
- $Q() = \sum_{a,b,c} R(a, b) \cdot S(b, c) \cdot T(c, a)$
- A single-tuple update is a relation mapping a tuple to a non-zero value (positive for insertions, negative for deletions)

<table>
<thead>
<tr>
<th>$R$</th>
<th>$S$</th>
<th>$T$</th>
<th>$R \cdot S \cdot T$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A$</td>
<td>$B$</td>
<td>#</td>
<td>$B$</td>
</tr>
<tr>
<td>$a_1$</td>
<td>$b_1$</td>
<td>2</td>
<td>$b_1$</td>
</tr>
<tr>
<td>$a_2$</td>
<td>$b_1$</td>
<td>3</td>
<td>$b_1$</td>
</tr>
<tr>
<td>$a_2$</td>
<td>$b_1$</td>
<td>1</td>
<td>$c_2$</td>
</tr>
</tbody>
</table>

$\delta R = \{(a_2, b_1) \mapsto -2\}$

<table>
<thead>
<tr>
<th>$A$</th>
<th>$B$</th>
<th>#</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_2$</td>
<td>$b_1$</td>
<td>$-2$</td>
</tr>
</tbody>
</table>

$Q()$

| # |
| $\emptyset$ |
| $( )$ | $4 + 6 + 9 = 19$ |
Data Model

- Relations are functions mapping tuples to multiplicities.
- \( Q() = \sum_{a,b,c} R(a, b) \cdot S(b, c) \cdot T(c, a) \)
- A single-tuple update is a relation mapping a tuple to a non-zero value (positive for insertions, negative for deletions)

<table>
<thead>
<tr>
<th>( R )</th>
<th>( S )</th>
<th>( T )</th>
<th>( R \cdot S \cdot T )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( A )</td>
<td>( B )</td>
<td>( # )</td>
<td>( B )</td>
</tr>
<tr>
<td>( a_1 )</td>
<td>( b_1 )</td>
<td>2</td>
<td>( b_1 )</td>
</tr>
<tr>
<td>( a_2 )</td>
<td>( b_1 )</td>
<td>3</td>
<td>( b_1 )</td>
</tr>
<tr>
<td>( a_2 )</td>
<td>( b_1 )</td>
<td>1</td>
<td>( b_1 )</td>
</tr>
<tr>
<td>( a_2 )</td>
<td>( b_1 )</td>
<td>1</td>
<td>( b_1 )</td>
</tr>
</tbody>
</table>

\[ \delta R = \{(a_2, b_1) \mapsto -2\} \]

<table>
<thead>
<tr>
<th>( A )</th>
<th>( B )</th>
<th>( # )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a_2 )</td>
<td>( b_1 )</td>
<td>( -2 )</td>
</tr>
</tbody>
</table>

\[ Q() \]

<table>
<thead>
<tr>
<th>( \emptyset )</th>
<th>( # )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( )</td>
<td>4 + 6 + 9 = 19</td>
</tr>
</tbody>
</table>
## Data Model

- Relations are functions mapping tuples to multiplicities.
- \( Q() = \sum_{a,b,c} R(a, b) \cdot S(b, c) \cdot T(c, a) \)
- A single-tuple update is a relation mapping a tuple to a non-zero value (positive for insertions, negative for deletions)

<table>
<thead>
<tr>
<th>( R )</th>
<th>( S )</th>
<th>( T )</th>
<th>( R \cdot S \cdot T )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( A )</td>
<td>( B )</td>
<td>#</td>
<td>( B )</td>
</tr>
<tr>
<td>( a_1 )</td>
<td>( b_1 )</td>
<td>2</td>
<td>( b_1 )</td>
</tr>
<tr>
<td>( a_2 )</td>
<td>( b_1 )</td>
<td>3</td>
<td>( b_1 )</td>
</tr>
<tr>
<td>( a_2 )</td>
<td>( b_1 )</td>
<td>1</td>
<td>( c_2 )</td>
</tr>
</tbody>
</table>

\[ \delta R = \{(a_2, b_1) \mapsto -2\} \]

<table>
<thead>
<tr>
<th>( A )</th>
<th>( B )</th>
<th>#</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a_2 )</td>
<td>( b_1 )</td>
<td>-2</td>
</tr>
</tbody>
</table>

\[ Q() \]

<table>
<thead>
<tr>
<th>( \emptyset )</th>
<th>#</th>
</tr>
</thead>
<tbody>
<tr>
<td>()</td>
<td>4 + 6 + 9 = 19</td>
</tr>
<tr>
<td>()</td>
<td>4 + 6 + 3 = 13</td>
</tr>
</tbody>
</table>
Previous Results
Much Ado about Triangles

The Triangle Query Served as Milestone in Many Fields

- Worst-case optimal join algorithms [Algorithmica 1997, SIGMOD R. 2013]
- Parallel query evaluation [Found. & Trends DB 2018]
- Randomized approximation in static settings [FOCS 2015]
- Randomized approximation in data streams

Intensive Investigation of Answering Queries under Updates

- Theoretical developments [PODS 2017, ICDT 2018]
- Lower bounds [STOC 2015, ICM 2018]

So far:

No dynamic algorithm maintaining the exact triangle count in worst-case optimal time!
Naïve Maintenance

“Compute from scratch!”

\[
Q() = \sum_{a,b,c} R(a,b) \cdot S(b,c) \cdot T(c,a)
\]

N is the database size
Update time: \(O(N^{1.5})\) using worst-case optimal join algorithms

[Algorithmica 1997, SIGMOD 2013, ICDT 2014]
Naïve Maintenance

“Compute from scratch!”

\[ \delta R = \{ (\alpha, \beta) \mapsto m \} \]

\[ Q() = \sum_{a,b,c} R(a, b) + \delta R(a, b) \cdot S(b, c) \cdot T(c, a) \]

\( N \) is the database size

Update time: \( O(N^{1.5}) \) using worst-case optimal join algorithms

[Algorithmica 1997, SIGMOD R. 2013, ICDT 2014]

Space: \( O(N) \) to store input relations
Naïve Maintenance

“Compute from scratch!”

$$\delta R = \{(\alpha, \beta) \mapsto m\}$$

$$Q() = \sum_{a,b,c} R(a, b) + \delta R(a, b) \cdot S(b, c) \cdot T(c, a)$$

$N$ is the database size

Update time: $O(N^{1.5})$ using worst-case optimal join algorithms

[Algorithmica 1997, SIGMOD R. 2013, ICDT 2014]

Space: $O(N)$ to store input relations
Naïve Maintenance

“Compute from scratch!”

\[ \delta R = \{ (\alpha, \beta) \mapsto m \} \]

\[ Q() = \sum_{a,b,c} R(a,b) + \delta R(a,b) \]

\[ S(b,c) \]

\[ T(c,a) \]

- \( N \) is the database size
- Update time: \( \mathcal{O}(N^{1.5}) \) using worst-case optimal join algorithms
  
  [Algorithmica 1997, SIGMOD R. 2013, ICDT 2014]
- Space: \( \mathcal{O}(N) \) to store input relations
Classical Incremental View Maintenance [Found. & Trends DB 2018]

“Compute the difference!”

\[
Q() = \sum_{a,b,c} R(a, b) \cdot S(b, c) \cdot T(c, a)
\]
"Compute the difference!"

\[ \delta R = \{ (\alpha, \beta) \mapsto m \} \]

\[
Q() = \sum_{a,b,c} \delta R(a, b) \cdot S(b, c) \cdot T(c, a)
\]

\[
\delta Q() = \sum_{a,b,c} \alpha \beta \cdot S(b, c) \cdot T(c, a)
\]
“Compute the difference!”

\[ \delta R = \{ (\alpha, \beta) \mapsto m \} \]

\[ Q() = \sum_{a,b,c} R(a,b) \cdot S(b,c) \cdot T(c,a) \]

\[ \delta Q() = \sum_{a,b,c} \delta R(a,b) \cdot S(b,c) \cdot T(c,a) \]
Classical Incremental View Maintenance  [Found. & Trends DB 2018]

“Compute the difference!”

\[ \delta R = \{ (\alpha, \beta) \mapsto m \} \]

\[
Q() = \sum_{a,b,c} R(a, b) \cdot S(b, c) \cdot T(c, a)
\]

\[ \delta Q() = \alpha \beta \cdot \sum_{c} \beta \cdot c \cdot \delta R(\alpha, \beta) \]

Update time: \( O(N) \) to intersect \( C \)-values from \( S \) and \( T \)

Space: \( O(N) \) to store input relations
Classical Incremental View Maintenance

“Compute the difference!”

\[
\delta R = \{(\alpha, \beta) \mapsto m\}
\]

\[
Q() = \sum_{a,b,c} R(a,b) \bullet S(b,c) \bullet T(c,a)
\]

\[
\delta Q() = \alpha \beta \cdot \sum_c \beta \cdot c \cdot \mathcal{O}(N)
\]

Update time: \(\mathcal{O}(N)\) to intersect \(C\)-values from \(S\) and \(T\)

Space: \(\mathcal{O}(N)\) to store input relations
"Compute the difference!"

\[ \delta R = \{ (\alpha, \beta) \mapsto m \} \]

\[
Q() = \sum_{a,b,c} R(a,b) \cdot S(b,c) \cdot T(c,a)
\]

\[
\delta Q() = \alpha \beta \cdot \sum_c \begin{cases} \beta & c \\ c & \cdots \end{cases} \cdot \begin{cases} c & \cdots \\ \alpha & c \end{cases}
\]

\[
Q() = Q() + \delta Q()
\]
“Compute the difference!”

\[ \delta R = \{ (\alpha, \beta) \mapsto m \} \]

\[
Q() = \sum_{a,b,c} R(a,b) \cdot S(b,c) \cdot T(c,a)
\]

\[
\delta Q() = \alpha \beta \cdot \sum_c S(\beta, c) \cdot T(c, \alpha)
\]

\[
Q() = Q() + \delta Q()
\]

- Update time: \( \mathcal{O}(N) \) to intersect \( C \)-values from \( S \) and \( T \)
- Space: \( \mathcal{O}(N) \) to store input relations
“Compute the difference by using pre-materialized views!”

\[
\delta R(\alpha, \beta) = \{ (\alpha, \beta) \mapsto m \} \\
\delta Q() = \sum_{a,b,c} R(a,b) \cdot S(b,c) \cdot T(c,a)
\]
"Compute the difference by using pre-materialized views!"

\[ \delta R = \{ (\alpha, \beta) \mapsto m \} \]

\[
Q() = \sum_{a,b,c} R(a, b) \cdot S(b, c) \cdot T(c, a)
\]

\[
V_{ST}(b, a) = \sum_{c} S(b, c) \cdot T(c, a)
\]
“Compute the difference by using pre-materialized views!”

$$\delta R = \{ (\alpha, \beta) \mapsto m \}$$

$$Q() = \sum_{a,b,c} R(a, b) \cdot V_{ST}(b, a)$$
"Compute the difference by using pre-materialized views!"

\[ \delta R = \{(\alpha, \beta) \mapsto m\} \]

\[ Q() = \sum_{a,b,c} R(a,b) \cdot V_{ST}(b,a) \]

\[ \delta Q() = \sum_{a,b} \delta R(a,b) \cdot V_{ST}(b,a) \]
“Compute the difference by using pre-materialized views!”

\[ \delta R = \{((\alpha, \beta) \mapsto m) \} \]

\[ Q() = \sum_{a,b,c} \]

\[ R(a, b) \quad \cdot \quad V_{ST}(b, a) \]

\[ \delta Q() = \sum_{a,b,c} \]

\[ \delta R(\alpha, \beta) \quad \cdot \quad V_{ST}(\beta, \alpha) \]
"Compute the difference by using pre-materialized views!"

\[
\delta R = \{ (\alpha, \beta) \mapsto m \}
\]

\[
Q() = \sum_{a,b,c} R(a,b) \cdot V_{ST}(b,a)
\]

\[
\delta Q() = \sum_{\alpha,\beta} \delta R(\alpha,\beta) \cdot V_{ST}(\beta,\alpha)
\]

\[
Q() = Q() + \delta Q()
\]
“Compute the difference by using pre-materialized views!”

\[ \delta R = \{ (\alpha, \beta) \mapsto m \} \]

\[
R(a, b) \quad \bullet \quad V_{ST}(b, a)
\]

\[
Q() = \sum_{a, b, c} \delta R(\alpha, \beta) \cdot V_{ST}(\beta, \alpha)
\]

\[
\delta Q() = \alpha \beta \cdot \beta \alpha
\]

\[
Q() = Q() + \delta Q()
\]

- Time for updates to \( R \): \( O(1) \) to look up in \( V_{ST} \)
Factorized Incremental View Maintenance

Maintain $V_{ST}$ under updates

$$V_{ST}(b, a) = \sum_c S(b, c) \cdot T(c, a)$$

Time for updates to $S$ and $T$: $O(N)$ to maintain $V_{ST}$

Space: $O(N^2)$ to store input relations and $V_{ST}$
Maintain $V_{ST}$ under updates

\[ \delta S = \{(\beta, \gamma) \mapsto m\} \]

\[
V_{ST}(b, a) = \sum_c S(b, c) \cdot \delta S(\beta, c) \cdot T(c, a)
\]

\[
\delta V_{ST}(\beta, a) = \sum_c \beta \gamma \cdot \delta S(\beta, c) \cdot T(c, a)
\]

Time for updates to $S$ and $T$: $O(N)$ to maintain $V_{ST}$

Space: $O(N^2)$ to store input relations and $V_{ST}$
Maintain $V_{ST}$ under updates

$\delta S = \{ (\beta, \gamma) \mapsto m \}$

$V_{ST}(b, a) = \sum_c S(b, c) \cdot \delta S(\beta, \gamma) \cdot T(c, a)$

$\delta V_{ST}(\beta, a) = \beta \gamma \cdot \delta S(\beta, \gamma) \cdot \gamma \ldots a$
Maintain $V_{ST}$ under updates

\[ \delta S = \{ (\beta, \gamma) \mapsto m \} \]

\[ V_{ST}(b, a) = \sum_c S(b, c) \cdot \delta S(\beta, \gamma) \cdot T(c, a) \]

\[ \delta V_{ST}(\beta, a) = \sum_c \beta \gamma \cdot T(\gamma, a) \]

Time for updates to $S$ and $T$: $O(N)$ to maintain $V_{ST}$

Space: $O(N^2)$ to store input relations and $V_{ST}$
Maintain $V_{ST}$ under updates

\[ \delta S = \{ (\beta, \gamma) \mapsto m \} \]

\[ S(b, c) \]

\[ T(c, a) \]

\[ V_{ST}(b, a) = \sum_{c} S(b, c) \cdot T(c, a) \]

\[ \delta V_{ST}(\beta, a) = \beta \cdot \gamma \cdot T(\gamma, a) \]

\[ O(N) \]

\[ V_{ST}(\beta, a) = V_{ST}(\beta, a) + \delta V_{ST}(\beta, a) \]
Maintain $V_{ST}$ under updates

$$\delta S = \{ (\beta, \gamma) \mapsto m \}$$

$$S(b, c)$$

$$T(c, a)$$

$$V_{ST}(b, a) = \sum_c S(b, c) \cdot T(c, a)$$

$$\delta V_{ST}(\beta, a) = \delta S(\beta, \gamma) \cdot T(\gamma, a)$$

$$V_{ST}(\beta, a) = V_{ST}(\beta, a) + \delta V_{ST}(\beta, a)$$

- Time for updates to $S$ and $T$: $O(N)$ to maintain $V_{ST}$
- Space: $O(N^2)$ to store input relations and $V_{ST}$
Our Contribution
**Closing the Complexity Gap**

Complexity bounds for the maintenance of the triangle count

<table>
<thead>
<tr>
<th>Known Upper Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Update Time:</strong> $O(N)$</td>
</tr>
<tr>
<td><strong>Space:</strong> $O(N)$</td>
</tr>
</tbody>
</table>

**Known Lower Bound**

Amortized update time: not $O(N^{0.5-\gamma})$ for any $\gamma > 0$

under the Online Matrix-Vector Multiplication Conjecture (follows from [PODS 2017])
Closing the Complexity Gap

Complexity bounds for the maintenance of the triangle count

<table>
<thead>
<tr>
<th>Known Upper Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Update Time: ( \mathcal{O}(N) )</td>
</tr>
<tr>
<td>Space: ( \mathcal{O}(N) )</td>
</tr>
</tbody>
</table>

Can the triangle count be maintained with sublinear update time?

<table>
<thead>
<tr>
<th>Known Lower Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amortized update time: not ( \mathcal{O}(N^{0.5-\gamma}) ) for any ( \gamma &gt; 0 )</td>
</tr>
<tr>
<td>under the Online Matrix-Vector Multiplication Conjecture (follows from [PODS 2017])</td>
</tr>
</tbody>
</table>
Closing the Complexity Gap

Complexity bounds for the maintenance of the triangle count

### Known Upper Bound

<table>
<thead>
<tr>
<th>Update Time</th>
<th>Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O(N)$</td>
<td>$O(N)$</td>
</tr>
</tbody>
</table>

Can the triangle count be maintained with sublinear update time?

**Yes!**

We propose: IVM$^\varepsilon$

Amortized update time: $O(N^{0.5})$

This is worst-case optimal!

### Known Lower Bound

Amortized update time: not $O(N^{0.5-\gamma})$ for any $\gamma > 0$

under the Online Matrix-Vector Multiplication Conjecture

(follows from [PODS 2017])
Given $\varepsilon \in [0, 1]$, $\text{IVM}^\varepsilon$ maintains the triangle count with

- $O(N^{\max\{\varepsilon, 1-\varepsilon\}})$ amortized update time
- $O(N^{1+\min\{\varepsilon, 1-\varepsilon\}})$ space
- $O(N^{3/2})$ preprocessing time
- $O(1)$ answer time.

Known maintenance approaches are recovered by $\text{IVM}^\varepsilon$. 
Main Ideas in IVM$^\varepsilon$

- Compute the difference like in classical IVM!
- Materialize views like in Factorized IVM!
- **New ingredient**: Use adaptive processing based on data skew!
  - Treat *heavy* values differently from *light* values!
Quick Look inside IVM$^\varepsilon$
Quick Look inside IVM

Partition $R$ based on $A$ into

- a light part $R_L = \{ t \in R \mid |\sigma_{A=t.A}| < N^{\epsilon} \}$,
- a heavy part $R_H = R \setminus R_L$!

\[ R_L \]
\[ R_H \]
Partition $R$ based on $A$ into

- a light part $R_L = \{ t \in R \mid |\sigma_{A=t}.A| < N^\varepsilon \}$,
- a heavy part $R_H = R \setminus R_L$.

**Derived Bounds**

**from light part:**
for all $A$-values $a$, $|\sigma_{A=a} R_L| < N^\varepsilon$

**from heavy part:**
$|\pi_A R_H| \leq N^{1-\varepsilon}$, since
for all $A$-values $a$, $|\sigma_{A=a} R_H| \geq N^\varepsilon$
and $|\pi_A R_H| \cdot N^\varepsilon \leq N$
Likewise, partition

- $S = S_L \cup S_H$ based on $B$, and
- $T = T_L \cup T_H$ based on $C$!

$Q$ is the sum of skew-aware views

$$
\sum_{a,b,c} R_U(a, b) \cdot S_V(b, c) \cdot T_W(c, a) \text{ with } U, V, W \in \{L, H\}.
$$
Adaptive Maintenance Strategy

Given an update $\delta R_* = \{(\alpha, \beta) \mapsto m\}$, compute the difference for each of the following skew-aware views using a different strategy:

$$Q_{*LL}() = \sum_{a,b,c} R_* (a, b) \cdot S_L (b, c) \cdot T_L (c, a)$$

$$Q_{*HH}() = \sum_{a,b,c} R_* (a, b) \cdot S_H (b, c) \cdot T_H (c, a)$$

$$Q_{*LH}() = \sum_{a,b,c} R_* (a, b) \cdot S_L (b, c) \cdot T_H (c, a)$$

$$Q_{*HL}() = \sum_{a,b,c} R_* (a, b) \cdot S_H (b, c) \cdot T_L (c, a)$$
Adaptive Maintenance Strategy

Given an update $\delta R_* = \{(\alpha, \beta) \mapsto m\}$, compute the difference for each of the following skew-aware views using a different strategy:

\[
\begin{align*}
\delta Q_{*LL}() &= \delta R_*(\alpha, \beta) \cdot \sum_c S_L(\beta, c) \cdot T_L(c, \alpha) \\
\delta Q_{*HH}() &= \delta R_*(\alpha, \beta) \cdot \sum_c S_H(\beta, c) \cdot T_H(c, \alpha) \\
\delta Q_{*LH}() &= \delta R_*(\alpha, \beta) \cdot \sum_c S_L(\beta, c) \cdot T_H(c, \alpha) \\
\delta Q_{*HL}() &= \delta R_*(\alpha, \beta) \cdot \sum_c S_H(\beta, c) \cdot T_L(c, \alpha)
\end{align*}
\]
Adaptive Maintenance Strategy

\[ \delta Q_{*LL}(\cdot) = \delta R_*(\alpha, \beta) \cdot \sum_c S_L(\beta, c) \cdot T_L(c, \alpha) \]
Adaptive Maintenance Strategy

\[ \delta Q_{\text{LL}}(\alpha, \beta) = \delta R_*(\alpha, \beta) \cdot \sum_c S_L(\beta, c) \cdot T_L(c, \alpha) \]

\[ \delta Q_{\text{LL}}(\alpha, \beta) = \alpha \beta \cdot \sum_c S_L(\beta, c) < N^\varepsilon \cdot T_L(c, \alpha) \]

\[ S_L(\beta, c) \]

\[ T_L(c, \alpha) \]
Adaptive Maintenance Strategy

\[ \delta Q_{*LL}() = \delta R_*(\alpha, \beta) \cdot \sum_c S_L(\beta, c) \cdot T_L(c, \alpha) \]

Update time: \( \mathcal{O}(N^\varepsilon) \) to intersect \( C \)-values from \( S_L \) and \( T_L \)
Adaptive Maintenance Strategy

\[ \delta Q_{*HH}(\cdot) = \delta R_*(\alpha, \beta) \cdot \sum_c S_H(\beta, c) \cdot T_H(c, \alpha) \]

\[ \delta R_*(\alpha, \beta) \]

\[ \delta Q_{*HH}(\cdot) = \begin{bmatrix} \alpha & \beta \end{bmatrix} \cdot \sum_c \begin{bmatrix} \beta & c \\ c & \ldots \\ \ldots & c \end{bmatrix} \cdot \begin{bmatrix} a \\ \alpha \\ a \\ \ldots \\ a \end{bmatrix} \]
Adaptive Maintenance Strategy

\[ \delta Q_{*HH}(\cdot) = \delta R_*(\alpha, \beta) \cdot \sum_c S_H(\beta, c) \cdot T_H(c, \alpha) \]

\[ \delta R_*(\alpha, \beta) \]

\[ \delta Q_{*HH}(\cdot) = \alpha \beta \cdot \sum_c \beta \cdot c \cdot \ldots \cdot a \cdot \alpha \cdot c \cdot \ldots \cdot a \]

Update time: \( O(N^{1-\varepsilon}) \) to intersect \( C \)-values from \( S_H \) and \( T_H \).
Adaptive Maintenance Strategy

\[
\delta Q_{*HH}(\cdot) = \delta R_*(\alpha, \beta) \cdot \sum_c S_H(\beta, c) \cdot T_H(c, \alpha)
\]

\[
\delta R_*(\alpha, \beta)
\]

\[
\delta Q_{*HH}(\cdot) = \begin{pmatrix} \alpha & \beta \end{pmatrix} \cdot \sum_c \begin{pmatrix} \beta & \ldots & c \end{pmatrix} \cdot \begin{pmatrix} \ldots & \alpha & \ldots \\ c & \ldots & \ldots \\ \ldots & \ldots & \ldots \end{pmatrix}
\]

Update time: \( \mathcal{O}(N^{1-\varepsilon}) \) to intersect \( C \)-values from \( S_H \) and \( T_H \)
Adaptive Maintenance Strategy

\[ \delta Q_{*LH}(\cdot) = \delta R_*(\alpha, \beta) \cdot \sum_c S_L(\beta, c) \cdot T_H(c, \alpha) \]

\[ \delta R_*(\alpha, \beta) \]

\[ \delta Q_{*LH}(\cdot) = \alpha \beta \cdot \sum_c S_L(\beta, c) \cdot T_H(c, \alpha) \]
Adaptive Maintenance Strategy

\[ \delta Q_{LH}(\cdot) = \delta R_*(\alpha, \beta) \cdot \sum_c S_L(\beta, c) \cdot T_H(c, \alpha) \]

\[ \delta R_*(\alpha, \beta) \]

\[ \delta Q_{LH}(\cdot) = \begin{array}{cc} \alpha & \beta \\ \end{array} \cdot \sum_c \begin{array}{c} \beta \\ \cdots \\ c \\ \end{array} \cdot \begin{array}{c} c \\ \cdots \\ a \\ \cdots \\ a \\ \cdots \\ a \\ \cdots \\ c \\ \end{array} < N^\varepsilon \]

**Update time:** \( O(\min\{\varepsilon, 1 - \varepsilon\}) \) to intersect \( C \)-values from \( S_L \) and \( T_H \)
Adaptive Maintenance Strategy

\[ \delta Q_{*LH}(\cdot) = \delta R_*(\alpha, \beta) \cdot \sum_c S_L(\beta, c) \cdot T_H(c, \alpha) \]

\[ \delta R_*(\alpha, \beta) \]

\[ \delta Q_{*LH}(\cdot) = \alpha \beta \cdot \sum_c S_L(\beta, c) \cdot T_H(c, \alpha) \]
Adaptive Maintenance Strategy

\[ \delta Q_{* LH}(\cdot) = \delta R_*(\alpha, \beta) \cdot \sum_c S_L(\beta, c) \cdot T_H(c, \alpha) \]

Update time: \( \mathcal{O}(N^{\min\{\epsilon, 1-\epsilon\}}) \) to intersect \( C \)-values from \( S_L \) and \( T_H \)
Adaptive Maintenance Strategy

\[ \delta Q_{HL}(\cdot) = \delta R_*(\alpha, \beta) \cdot \sum_c S_H(\beta, c) \cdot T_L(c, \alpha) \]

\[ \delta R_*(a, b) \quad S_H(b, c) \quad T_L(c, a) \]

\[ \delta Q_{HL}(\cdot) = \sum_{a,b}^{\alpha \beta} \cdot \cdot \]
Adaptive Maintenance Strategy

\[ \delta Q_{\star HL}() = \delta R_{\star}(\alpha, \beta) \cdot \sum_c S_H(\beta, c) \cdot T_L(c, \alpha) \]

\[ \delta Q_{\star HL}() = \sum_{a,b} \delta R_{\star}(a, b) \cdot S_H(b, c) \cdot T_L(c, a) \]

\[ V_{ST}(b, a) = \sum_c S_H(b, c) \cdot T_L(c, a) \]
Adaptive Maintenance Strategy

\[ \delta Q_{*HL}() = \delta R_*(\alpha, \beta) \cdot \sum_c S_H(\beta, c) \cdot T_L(c, \alpha) \]

\[ \delta R_*(a, b) \quad V_{ST}(b, a) \]

\[ \delta Q_{*HL}() = \sum_{a,b} \alpha \quad \beta \cdot \cdot \]

Update time: \( O(1) \) to look up in \( V_{ST} \)
Adaptive Maintenance Strategy

\[ \delta Q_{\ast HL}() = \delta R_\ast(\alpha, \beta) \cdot \sum_c S_H(\beta, c) \cdot T_L(c, \alpha) \]

\[ \delta R_\ast(\alpha, \beta) \]

\[ V_{ST}(\beta, \alpha) \]

\[ \delta Q_{\ast HL}() = \alpha \quad \beta \quad \beta \quad \alpha \]
Adaptive Maintenance Strategy

\[ \delta Q_{*HL}(\cdot) = \delta R_*(\alpha, \beta) \cdot \sum_c S_H(\beta, c) \cdot T_L(c, \alpha) \]

\[ \delta R_*(\alpha, \beta) \]

\[ V_{ST}(\beta, \alpha) \]

Update time: \( \mathcal{O}(1) \) to look up in \( V_{ST} \)
Summary of Adaptive Maintenance Strategies

Given an update $\delta R_\ast = \{(\alpha, \beta) \mapsto m\}$, compute the difference for each skew-aware view using a different strategy:

<table>
<thead>
<tr>
<th>Skew-aware View</th>
<th>Evaluation from left to right</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sum_{a,b,c} R_\ast(a, b) \cdot S_L(b, c) \cdot T_L(c, a)$</td>
<td>$\delta R_\ast(\alpha, \beta) \cdot \sum_c S_L(\beta, c) \cdot T_L(c, \alpha)$</td>
<td>$\mathcal{O}(N^\varepsilon)$</td>
</tr>
<tr>
<td>$\sum_{a,b,c} R_\ast(a, b) \cdot S_H(b, c) \cdot T_H(c, a)$</td>
<td>$\delta R_\ast(\alpha, \beta) \cdot \sum_c T_H(c, \alpha) \cdot S_H(\beta, c)$</td>
<td>$\mathcal{O}(N^{1-\varepsilon})$</td>
</tr>
<tr>
<td>$\sum_{a,b,c} R_\ast(a, b) \cdot S_L(b, c) \cdot T_H(c, a)$</td>
<td>$\delta R_\ast(\alpha, \beta) \cdot \sum_c S_L(\beta, c) \cdot T_H(c, \alpha)$</td>
<td>$\mathcal{O}(N^\varepsilon)$</td>
</tr>
<tr>
<td>$\sum_{a,b,c} R_\ast(a, b) \cdot S_H(b, c) \cdot T_L(c, a)$</td>
<td>$\delta R_\ast(\alpha, \beta) \cdot \sum_c T_H(c, \alpha) \cdot S_L(\beta, c)$</td>
<td>$\mathcal{O}(N^{1-\varepsilon})$</td>
</tr>
</tbody>
</table>

Overall update time: $\mathcal{O}(N^{\max(\varepsilon, 1-\varepsilon)})$
Materialized Auxiliary Views

\[ V_{RS}(a, c) = \sum_b R_H(a, b) \cdot S_L(b, c) \]

\[ V_{ST}(b, a) = \sum_c S_H(b, c) \cdot T_L(c, a) \]

\[ V_{TR}(a, c) = \sum_a T_H(c, a) \cdot R_L(a, b) \]
Materialized Auxiliary Views

Maintain $V_{ST}(b, a) = \sum_c S_H(b, c) \cdot T_L(c, a)$ under an update

$\delta S_H = \{(\beta, \gamma) \mapsto m\}$

$\delta V_{ST}(\beta, a) = \sum_c S_H(\beta, c) \cdot T_L(c, a)$

Update time: $O(N^2 \varepsilon)$ to iterate over $A$-values paired with $\gamma$ from $T_L$. 

23 / 33
Materialized Auxiliary Views

Maintain \( V_{ST}(b, a) = \sum_c S_H(b, c) \cdot T_L(c, a) \) under an update

\[ \delta S_H = \{(\beta, \gamma) \mapsto m\} \]

\[ \delta V_{ST}(\beta, a) = \delta S_H(\beta, \gamma) \cdot T_L(\gamma, a) \]
Materialized Auxiliary Views

Maintain $V_{ST}(b, a) = \sum_c S_H(b, c) \cdot T_L(c, a)$ under an update

$\delta S_H = \{(\beta, \gamma) \mapsto m\}$

$\delta V_{ST}(\beta, a) = \delta S_H(\beta, \gamma) \cdot T_L(\gamma, a)$

Update time: $O(N^\varepsilon)$ to iterate over $A$-values paired with $\gamma$ from $T_L$. 

$< N^\varepsilon$
Materialized Auxiliary Views

Maintain $V_{ST}(b, a) = \sum_c S_H(b, c) \cdot T_L(c, a)$ under an update

$$\delta S_H = \{(\beta, \gamma) \mapsto m\}$$

$$\delta V_{ST}(\beta, a) = \delta S_H(\beta, \gamma) \cdot T_L(\gamma, a)$$

Update time: $O(N^\varepsilon)$ to iterate over $A$-values paired with $\gamma$ from $T_L$
Materialized Auxiliary Views

Maintain $V_{ST}(b, a) = \sum_c S_H(b, c) \cdot T_L(c, a)$ under an update

$\delta T_L = \{(\gamma, \alpha) \mapsto m\}$

$\delta V_{ST}(b, \alpha) = \sum_c \delta T_L(c, \alpha) \cdot S_H(b, c)$
Materialized Auxiliary Views

Maintain \( V_{ST}(b, a) = \sum_c S_H(b, c) \cdot T_L(c, a) \) under an update

\[ \delta T_L = \{(\gamma, \alpha) \mapsto m\} \]

\[ \delta V_{ST}(b, \alpha) = \sum_{\gamma} S_H(b, \gamma) \cdot T_L(\gamma, \alpha) \]

Update time: \( O(N^{1-\varepsilon}) \) to iterate over \( B \)-values paired with \( \gamma \) from \( S_H \).
Materialized Auxiliary Views

Maintain $V_{ST}(b, a) = \sum_c S_H(b, c) \cdot T_L(c, a)$ under an update

$$\delta T_L = \{(\gamma, \alpha) \mapsto m\}$$

$$\delta V_{ST}(b, \alpha) = \delta T_L(\gamma, \alpha) \cdot S_H(b, \gamma)$$

Update time: $O(N^{1-\varepsilon})$ to iterate over $B$-values paired with $\gamma$ from $S_H$. 
Materialized Auxiliary Views

Maintain $V_{ST}(b, a) = \sum_c S_H(b, c) \cdot T_L(c, a)$ under an update

$$\delta T_L = \{(\gamma, \alpha) \mapsto m\}$$

Update time: $\mathcal{O}(N^{1-\varepsilon})$ to iterate over $B$-values paired with $\gamma$ from $S_H$
Materialized Auxiliary Views

\[ V_{RS}(a, c) = \sum_b R_H(a, b) \cdot S_L(b, c) \]
\[ V_{ST}(b, a) = \sum_c S_H(b, c) \cdot T_L(c, a) \]
\[ V_{TR}(a, c) = \sum_a T_H(c, a) \cdot R_L(a, b) \]

Maintenance Complexity

- **Time:** \( O(N^{\max\{\varepsilon, 1-\varepsilon\}}) \)
- **Space:** \( O(N^{1+\min\{\varepsilon, 1-\varepsilon\}}) \)
Updates can change the frequencies of values in the relation parts!

\[ R_L \]

Insertions

\[ a \quad b \quad \ldots \quad b \\]

\[ a \text{ is light} \]

Threshold

\[ \begin{array}{c}
\text{a} \\
\text{b} \\
\text{b} \\
\end{array} \]

\[ a \text{ is heavy} \]

**Minor Rebalancing**

- Transfer \( O(N^\varepsilon) \) tuples from one to the other part of the same relation!
- Time complexity: \( O(N^\varepsilon + \max\{\varepsilon, 1-\varepsilon\}) \)
Rebalancing Partitions

Updates can change the heavy-light threshold!

\[ R_H \]

Database size increases

\[ \text{Threshold} \]

\[ \begin{array}{c}
\text{Threshold} \\
\hline
a \\
\hline
b \\
\hline
\end{array} \]

\[ \begin{array}{c}
\text{Threshold} \\
\hline
a \\
\hline
b \\
\hline
\end{array} \]

\[ a \text{ is heavy} \]

\[ a \text{ is light} \]

Major Rebalancing

- Recompute partitions and views from scratch!
- Time complexity: \( O(N^{1+\min\{\varepsilon,1-\varepsilon\}}) \)
Amortization of Rebalancing Times

- Both forms of rebalancing require superlinear time.
Amortization of Rebalancing Times

- Both forms of rebalancing require superlinear time.
- The rebalancing times amortize over sequences of updates.
  - Amortized minor rebalancing time: $O(N^{\max \{\epsilon, 1-\epsilon\}})$
  - Amortized major rebalancing time: $O(N^{\min \{\epsilon, 1-\epsilon\}})$
Amortization of Rebalancing Times

- Both forms of rebalancing require superlinear time.
- The rebalancing times amortize over sequences of updates.
  - Amortized minor rebalancing time: \( O(N^{\max\{\varepsilon,1-\varepsilon\}}) \)
  - Amortized major rebalancing time: \( O(N^{\min\{\varepsilon,1-\varepsilon\}}) \)
- Overall amortized rebalancing time: \( O(N^{\max\{\varepsilon,1-\varepsilon\}}) \)
Quo Vadis IVM$^\varepsilon$?
Quo Vadis $\text{IVM}^\varepsilon$?

### Generalization of $\text{IVM}^\varepsilon$

- $\text{IVM}^\varepsilon$ variants admit sublinear update time for counting versions of
  - Loomis-Whitney
  - Queries built from 3 relations
  - 4-cycle
  - 4-path

- $\text{IVM}^\varepsilon$ variants admit sublinear update time and constant-delay enumeration for the full triangle query.

### Ongoing Work

- Characterization of the class of conjunctive count queries that admit sublinear maintenance time
- Implementation of $\text{IVM}^\varepsilon$ on top of DBToaster


[PODS 2006] Luciana S Buriol, Gereon Frahling, Stefano Leonardi, Alberto Marchetti-Spaccamela, and Christian Sohler. *Counting Triangles in Data Streams.* In PODS 2006


[PODS 2017] Christoph Berkholz, Jens Keppeler, and Nicole Schweikardt. *Answering Conjunctive Queries Under Updates*. In PODS 2017


[ICDT 2018] Christoph Berkholz, Jens Keppeler, and Nicole Schweikardt. Answering UCQs Under Updates and in the Presence of Integrity Constraints. In ICDT 2018


[SIGMOD 2018] Nikolic, Milos, and Dan Olteanu. Incremental view maintenance with triple lock factorization benefits. In SIGMOD 2018