gSketch:
On Query Estimation in Graph Streams

Peixiang Zhao (Florida State University)

Charu C. Aggarwal (IBM Research, Yorktown Heights)

Min Wang (HP Labs, China)

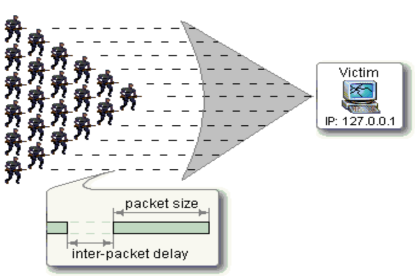
Istanbul, Turkey, August, 2012

Synopsis

1. Introduction
2. Problem Formulation
3. A Naïve Solution: Global Sketching
4. A Better Sketch Partitioning Solution: gSketch
5. Experimental Evaluation
6. Conclusions

Introduction

- **Graph stream = Graph + Data stream**
  - The edge set is massive
  - Edges are received and updated rapidly in a form of a stream
- **Most existing network applications can be naturally modeled as graph streams**
  - Representative applications
    - Intrusion detection on Internet
    - Social networks
    - Telecommunications



Challenges

- **Graph streams**
  - In a very large scale, the data cannot be stored explicitly in main memory, or even on disk
  - The arriving rate of graph streams is fast
    - "*You can never step in the same stream twice*" --- Heraclitus
- **Graph** streams
  - The universe we are keeping track of is extremely large
  - The dynamic nature hampers a direct application of many algorithms for static memory-resident graphs

Problem Formulation

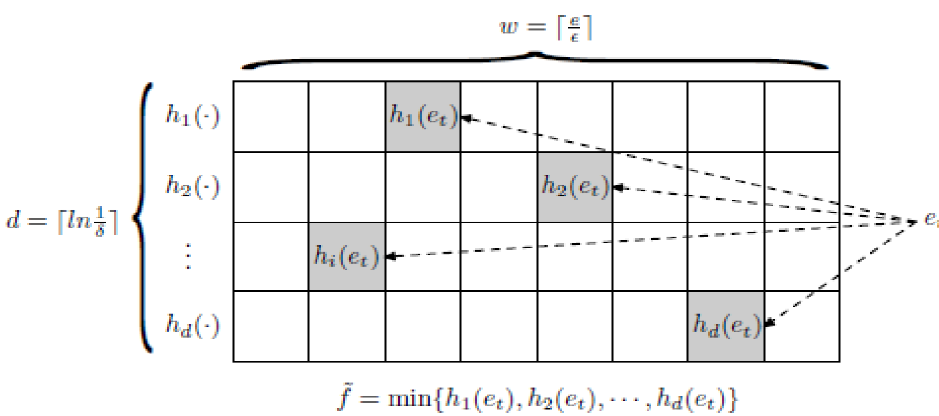
- **Graph streams**
  - $G = (V, E)$ a labeled, directed graph
    - $|V| = N$;
    - $E = \{<u_{t1}, v_{t1}>: f_{t1}; <u_{t2}, v_{t2}>: f_{t2}; \ldots\ldots; <u_{ti}, v_{ti}>: f_{ti}; \ldots\ldots\}$
- **Queries to be estimated**

1. **Edge query**
   - Determine the frequency of the edge $<X, Y>$: $f(X, Y) = \sum_{ti \in T} f(X, Y; t_i)$
2. **Aggregate subgraph query**
   - Determine the aggregate frequency behavior of the edges in a subgraph

A Naïve Solution: Global Sketching

- **Global sketching**
  - A direct application of any existing sketch method for data streams
    - AMS[STOC'96], Lossy-Count[VLDB'02], CountMin[J.Alg'05,SIGMOD'11], Bottom-k[VLDB'08], ……
  - **CountMin** sketch
    - Given a data stream with *N* arrivals till the time-stamp *t*, the estimated frequency $\tilde{f}$ is bounded up *w.h.p.* $(1 - e^{-d})$

$$f \leq \tilde{f} \leq f + e * N/w$$



A Naïve Solution: Global Sketching

- **The vulnerabilities of global sketching**
  - The relative error of query estimation on edge i is $\frac{e}{w}N/f_i$, which is proportional to N/f_i !
  - Such an estimation error incurred can be **extremely high**
    - Edge frequencies of a graph stream are distributed quite unevenly
    - “Low-frequency" edges are quite relevant for querying, and may show up repeatedly in the workload

A Better Solution: Sketch Partitioning

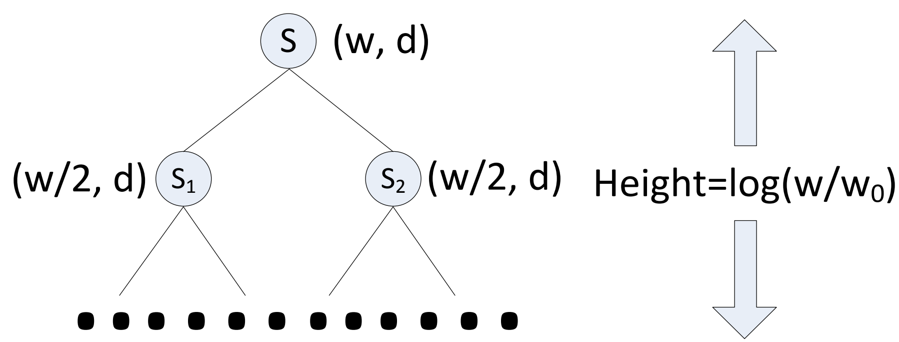
- **Broad intuition**
  - Common characteristics of real graph streams
    - Global Heterogeneity and Skews: the relative frequencies of different edges are very uneven
    - Local Similarity: within structurally localized regions of the graph, relative frequencies of edges are often correlated
  - **(Data/workload) samples** are always available
- **Key idea:**
  - Partitioning the global sketch, so that edges with similar frequencies are maintained and queried in **localized sketches** in order to achieve better estimation accuracy

gSketch: Overview

- **Objective**
  - Given a space limit S, to partition the global sketch over different regions of the graph
    - Partition based on vertices toward counting edges with sufficient frequency uniformity within a sketch
- **Sampling-based partitioning**
  - A sample of the original stream is available
  - Both a sample of the stream and a sample of the query workload are available

Sketching Partitioning with Data Sample

- **Recursive partitioning in a top-down fashion as in a decision tree**
  - Data Samples are used to estimate edge frequencies based on local similarity of **edges emanating from different vertices**
  - Optimize the partitioning of S into S_1 and S_2



$$min\ E = min\left(\sum_{m \in S_1} \frac{\tilde{d}(m) * \tilde{F}(S_1)}{\frac{\tilde{f}_v(m)}{\tilde{d}(m)}} + \sum_{m \in S_2} \frac{\tilde{d}(m) * \tilde{F}(S_2)}{\frac{\tilde{f}_v(m)}{\tilde{d}(m)}}\right)$$

Sketching Partitioning with Data/Workload Samples

- **Recursive partitioning in a top-down fashion as in a decision tree**
  - **Workload Samples** are used to estimate "**relative weights**" of different edges

$$min\ E = min\left(\sum_{m \in S_1} \frac{\widetilde{w}(m) * \tilde{F}(S_1)}{\frac{\widetilde{f_v}(m)}{\tilde{d}(m)}} + \sum_{m \in S_2} \frac{\widetilde{w}(m) * \tilde{F}(S_2)}{\frac{\widetilde{f_v}(m)}{\tilde{d}(m)}}\right)$$

Early Termination of the Recursive Partition

1. **The width of a partitioned sketch** at a given level is less than a particular threshold w_0: $Width(S_i) \leq w_0$
2. **The number of distinct edges** being counted in a sketch is less than a given factor of the sketch table width: $\sum_{m \in S} \tilde{d}(m) < C * Width(S_i)$
   - The probability of any collision in a particular cell in S can be bounded by C

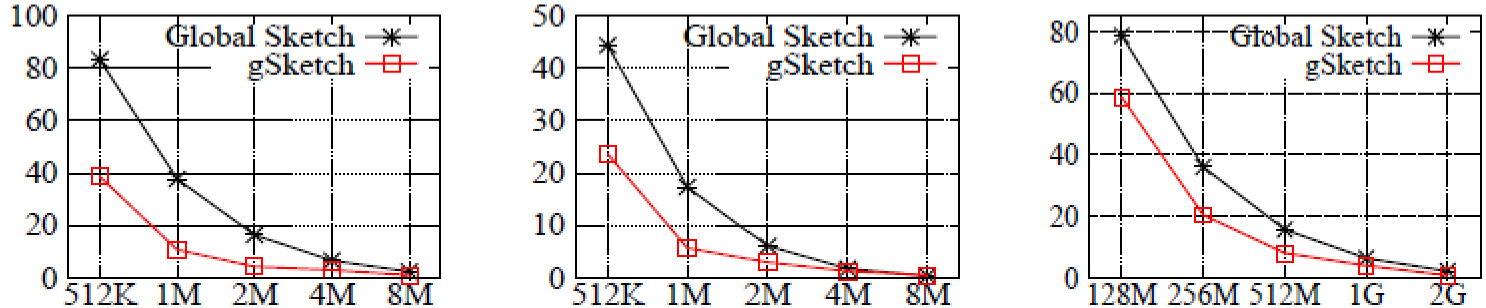
gSketch: Query Processing

- **Sketch partitioning is performed on the sample data as a preprocessing step**
  - Data samples only
  - Data and query workload samples
- **After sketch partitioning, graph streams are maintained and queried by a set of partitioned localized sketches**
  - Each edge is dispatched to its corresponding local sketch for frequency maintenance and query processing
  - Edges not in the data sample are uniformed dispatched to an **outlier sketch**

Experimental Evaluation

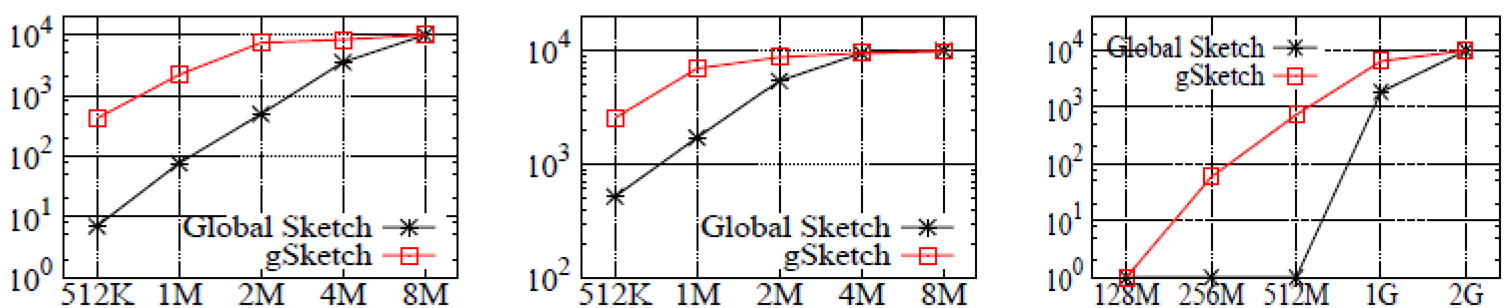
- **gSketch vs. Global sketching**
- **Evaluation methods**
  - Average relative error
  - Number of effective queries
- **Two real data sets and one synthetic data set**
  - DBLP (1,954,776 edges)
  - IBM-Attack Sensor Streaming Data (3,781,471 edges)
  - GTGraph (10^8 vertices and 10^9 edges)

Query Estimation Accuracy of Edge Queries (Data Sample Only)



(a) DBLP (b) IP Attack Network (c) GTGraph

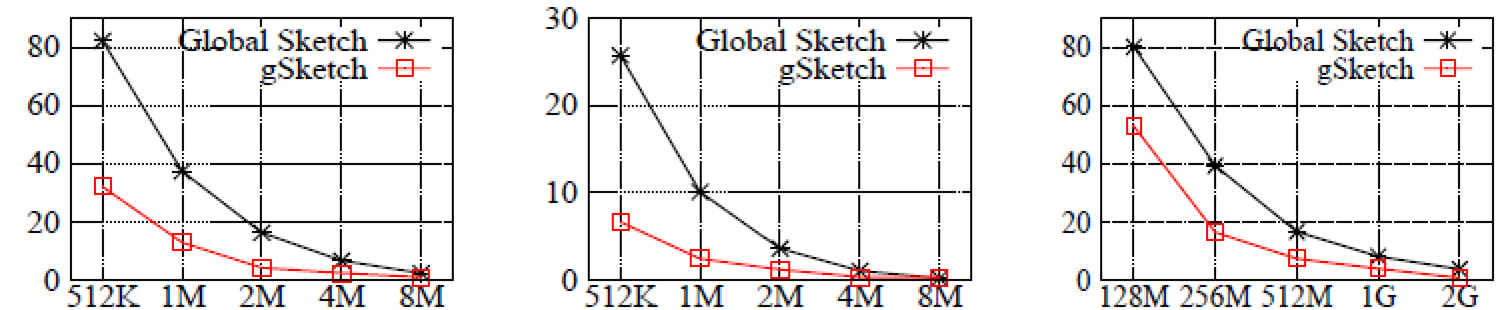
Figure: Average Relative Error



(a) DBLP (b) IP Attack Network (c) GTGraph

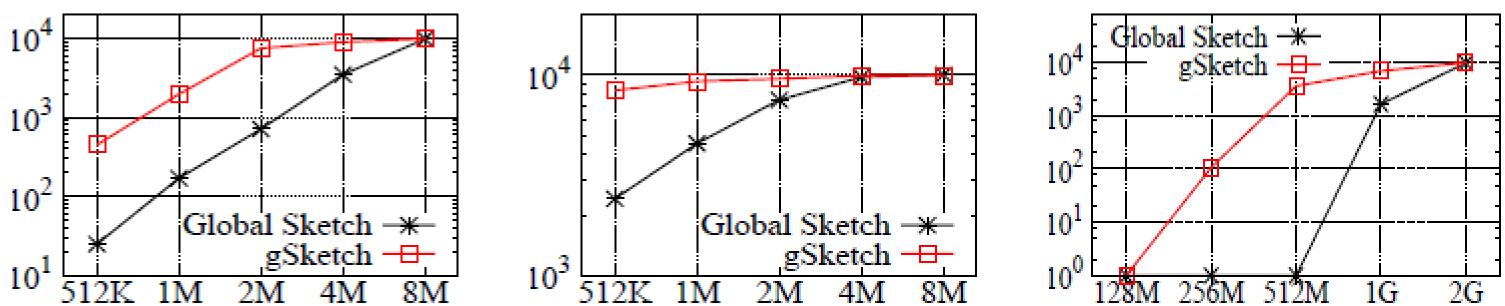
Figure: Number of Effective Queries

Query Estimation Accuracy of Edge Queries (Data and Query Workload Samples)



(a) DBLP (b) IP Attack Network (c) GTGraph

Figure: Average Relative Error (Zipf Skewness $\alpha = 1.5$)



(a) DBLP (b) IP Attack Network (c) GTGraph

Figure: Number of Effective Queries (Zipf Skewness $\alpha = 1.5$)

Query Estimation Accuracy of Edge Queries (Data and Query Workload Samples)

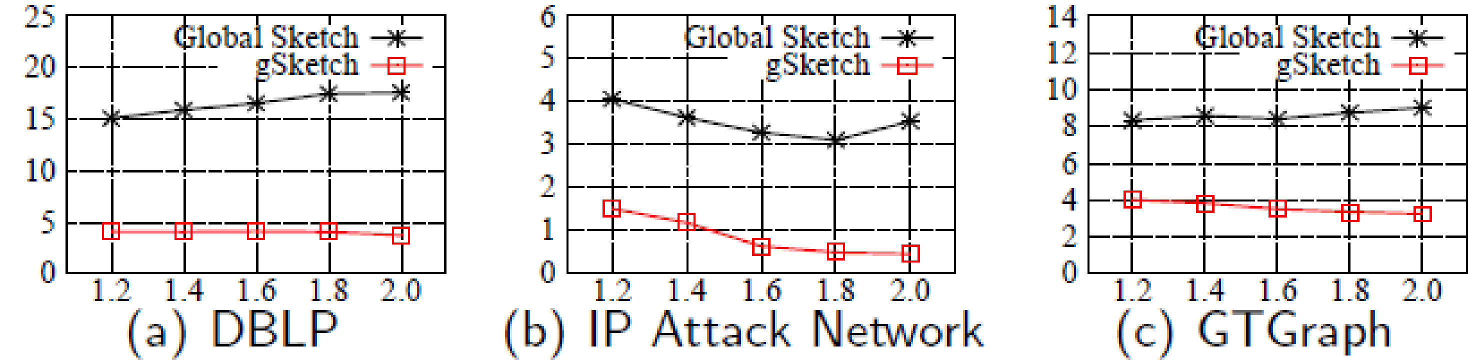


Figure: Average Relative Error (1G Memory)

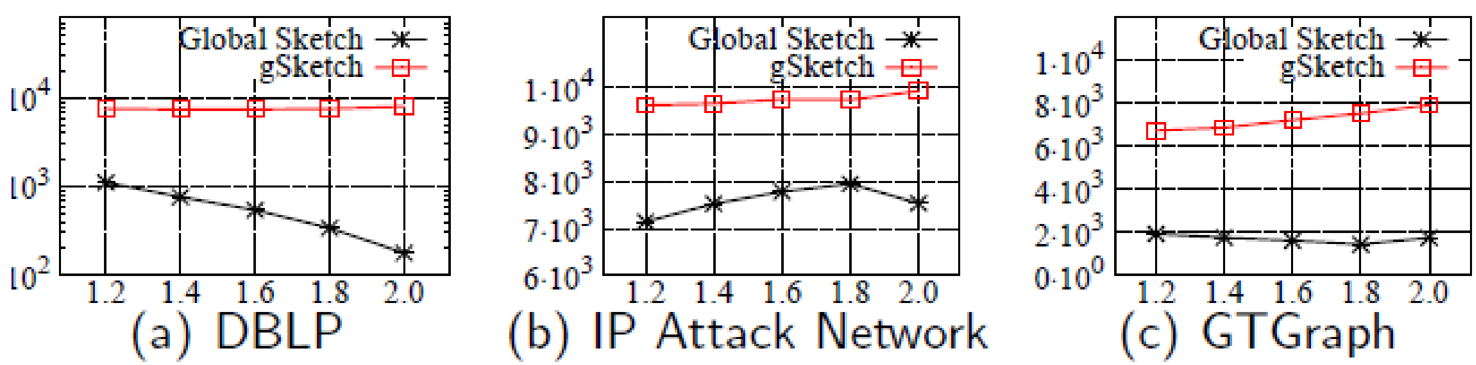
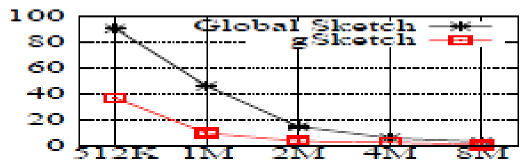
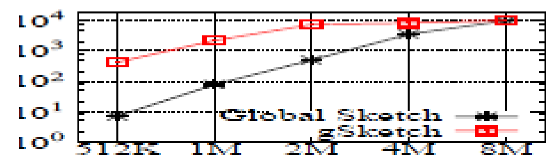


Figure: Number of Effective Queries (1G Memory)

Query Estimation Accuracy of Aggregate Subgraph Queries

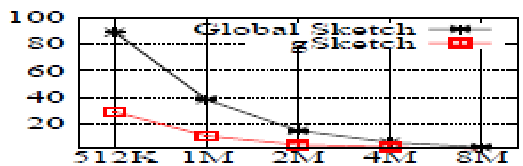


(a) Avg. Relative Error

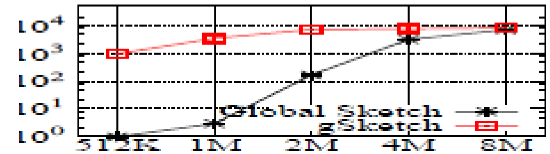


(b) # Effective Query

Figure: Data Sample Only

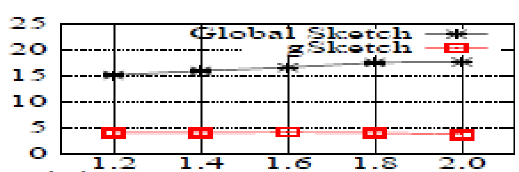


(a) Avg. Relative Error

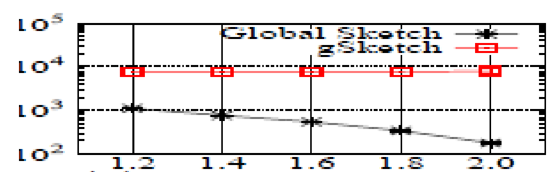


(b) # Effective Query

Figure: Data & Workload Samples ($\alpha = 1.5$)



(a) Avg. Relative Error



(b) # Effective Query

Figure: Data & Workload Samples (Memory = 1G)

Query Efficiency

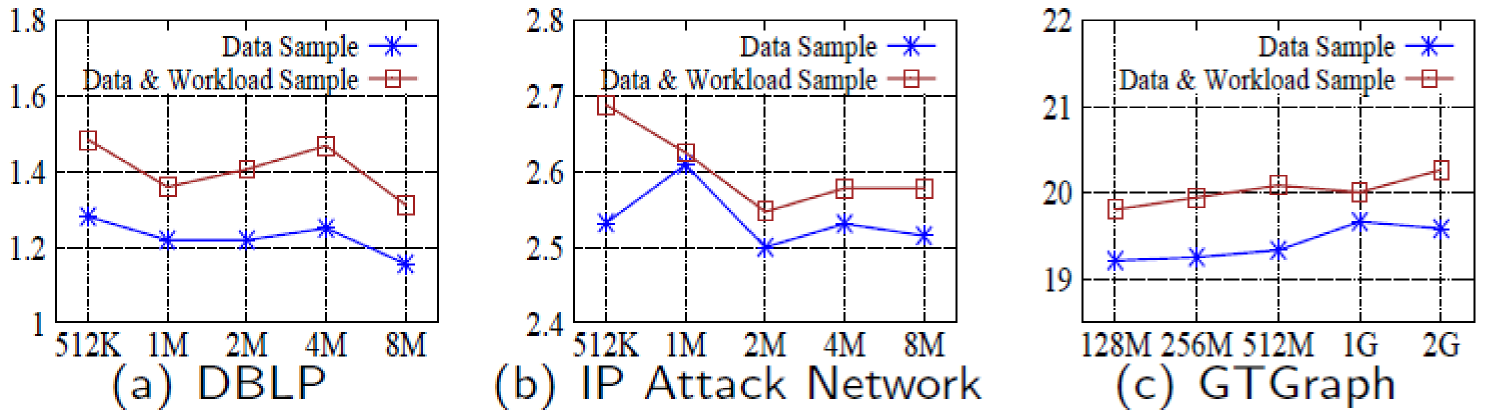


Figure: Sketch Construction Time (Seconds)

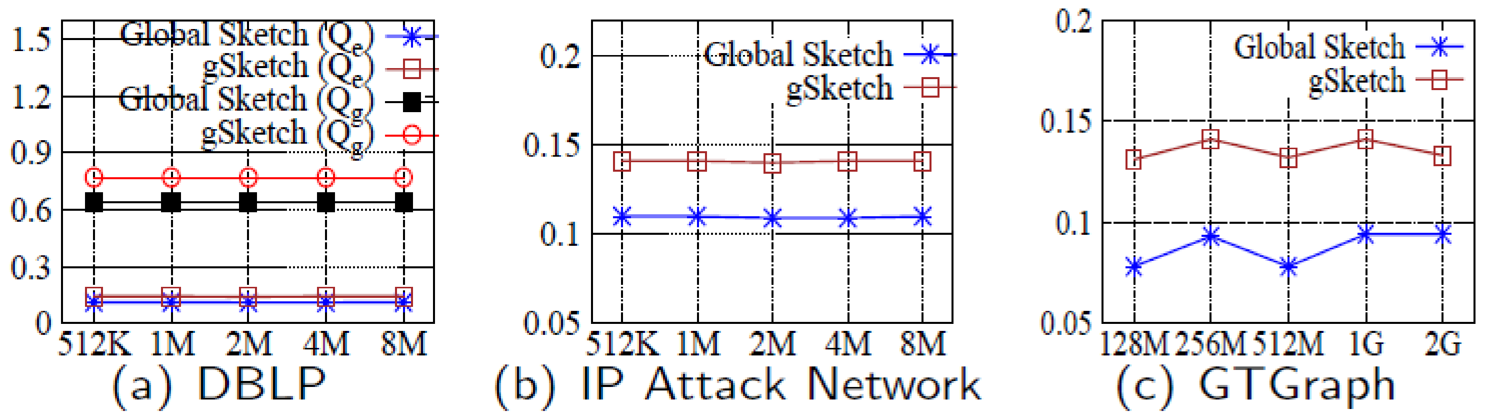


Figure: Query Processing Time (Seconds)

Effects of New Vertices/Edges

		Memory		Size		
		128M	256M	512M	1G	2G
Average relative error	gSketch	58.5968	20.381	8.0068	3.9345	0.7257
	Outlier sketch	58.5971	20.392	8.0081	3.9557	0.7837

Table: Average Relative Error of gSketch and Outlier Sketch in GTGraph

Conclusions

- **gSketch: a Partition-based sketch method for better query estimation in massive graph streams**
  - Adaptation of well-known sketching methods in conventional data streams
  - Leveraging common structural characteristics of massive graphs
  - Achieving up to an order of magnitude improvement in estimation accuracy
- **Future directions**
  - Computation of complex functions of edge frequencies in subgraph queries
  - Structural queries

Thank you!

Q & A

zhao@cs.fsu.edu
charu@us.ibm.com
min.wang6@hp.com