Charu C. Aggarwal
IBM T J Watson Research Center

Yuchen Zhao, Philip S. Yu
University of Illinois at Chicago

Outlier Detection in Graph Streams

International Conference on Data Engineering, 2011

Introduction

- Many real life applications result in edge stream objects which arrive continuously over time.

- Such edge streams may contain anomalies which provide use-ful insights about the underlying stream

- We design methods for outlier detection in graph streams.

Examples

- A bibliographic object from the DBLP network may be expressed as a graph with nodes corresponding to authors, conference, or topic area.

- A movie object from IMDB can be represented as an entity-relation graph, with edges corresponding to relationships between different elements.

- Events in social networks may lead to local patterns of activity, which may be modeled as streams of graph objects.

- The user browsing pattern at a web site is a stream of graph objects.
  - Edges $\Rightarrow$ Path taken by the user across the different objects.

Challenges

- Graphs are defined over a *massive domain of nodes*.
  - Node labels are drawn over universe of distinct identifers.
  - URL addresses in a web graph
  - IP-address in a network application
  - User identifier in a social networking application.
- The stream scenario makes the problem even more challenging:
  - The speed of the stream.
  - The entire graph is not available at a given time for analysis.

Outliers in Graphs: Goals

- Outliers represent significant deviations from the "normal" structural patterns in the underlying graphs.

- *Unusual relationships* in the graphs may be represented as *edges between regions of the graph that rarely occur together*.

- The goal of a stream-based outlier detection algorithm is to *identify graph objects which contain such unusual bridging edges*.

- Challenging because of the speed of the stream and large number of distinct edges.

Outlier Examples

- Objects correspond to entity-relation graphs for a movie. The nodes could correspond to actors that are drawn from very diverse genres or regions which do not normally occur together.

- In an academic network, two co-authors which are drawn from groups that usually do not work together may sometimes publish together (cross-disciplinary paper).

- Such outliers provide unique insights about the relationships in the underlying network.

Contributions

- Propose a real-time algorithm for discovering outliers in a graph stream.

- **Additional Results:** Dynamic construction of reservoir samples of the graph *with specific structural criteria* satisfying a general condition referred to as *set monotonicity*.
  - Many potential applications which require graph sampling with structural criteria.

Notations and Definitions

- We have an *incoming stream of graph objects*, denoted by $G_1 \ldots G_t \ldots$.
  - Graph objects could be information network objects.
  - Entity-relation graphs for a stream of incoming complex objects
  - Local patterns of activity in a social network
  - Patterns of intrusion in a computer network.
- Each graph G_i has a set of nodes, which are drawn from the node set N (base domain).

Outlier Detection with Partitioning

- The large size of the underlying graph precludes the storage of the entire network explicitly.
- The decision on whether an incoming graph object is an outlier needs to be performed in real time.
- Solution: We dynamically maintain node partitions which can expose abnormalities in the connectivity structure.
  - Leverage partitions for making real-time decisions about incoming graph objects.
- Partitions should ideally represent the dense regions in the graph
- Rare edges across dense regions are exposed as outliers.

Challenges in Maintaining Dynamic Partitions

- Dynamic maintenance of partitioning representing dense regions in a graph stream is very challenging.
  - **Further challenge:** A statistical model needs to be concurrently maintained for outlier modeling.

- Design a structural reservoir-sampling approach in order to dynamically maintain the partitioning.

- The structural sampling process is biased towards maintaining the dense regions $\Rightarrow$ Suffices for outlier detection as long as multiple partitions are maintained.

Algorithm Components

- Creation of node partitions (structural reservoir sampling).

- Using of node partitions for the purpose of outlier modeling.

- Use of outlier model to make prediction about incoming object.

- Describe outlier modeling first.

Outlier Modeling from the Use of Partitions

- In order to define the abnormality of edge behavior we define the *likelihood fit of an edge* with respect to a *partition* of the nodes $\mathcal{C} = C_1 \dots C_{k(\mathcal{C})}$.

- The number of node partitions in $\mathcal{C}$ is denoted by $k(\mathcal{C})$.

- We define a *structural generation model* of edges with respect to node partitioning $\mathcal{C}$.

- This structural generation model will be useful in the probabilistic modeling of outliers $\Rightarrow$ Lower probability edge is more indicative of outlier behavior.

Outlier Modeling

- **Edge Generation Model:** The structural generation model of a node partitioning $\mathcal{C} = \{C_1 \ldots C_{k(\mathcal{C})}\}$ is defined as a set of $k(\mathcal{C})^2$ probabilities $p_{ij}(\mathcal{C})$, such that $p_{ij}(\mathcal{C})$ is the probability of a randomly chosen edge in the incoming graph object to be incident on partitions i and j.

- For a given node i, we denote the partition identifier for i by $I(i, \mathcal{C})$ (between 1 and $k(\mathcal{C})$)

- **Edge Likelihood Fit:** Consider an edge (i, j), a node partition $\mathcal{C}$, and edge generation probabilities $p_{ij}(\mathcal{C})$ with respect to the partition. Then, the likelihood fit of the edge (i, j) with respect to the partition $\mathcal{C}$ is denoted by $\mathcal{F}(i, j, \mathcal{C})$ and is given by $p_{I(i,\mathcal{C}),I(j,\mathcal{C})}$.

Increasing Robustness with Multiple Models

- We maintain *multiple models* in order to increase the robustness of likelihood estimation.

- This smooths out the local variations which are specific to a given partitioning.

- **Edge Likelihood Fit:** The composite edge likelihood fit over the different partitionings $\mathcal{C}_1 \ldots \mathcal{C}_r$ for the edge (i,j) is the median of the values of $\mathcal{F}(i,j,\mathcal{C}_1)$ ... $\mathcal{F}(i,j,\mathcal{C}_r)$.

- This value is denoted by $\mathcal{MF}(i,j,\mathcal{C}_1 \ldots \mathcal{C}_r)$.

Likelihood Fit of Graph Object

- The likelihood fit for a graph object G is defined as a function of the likelihood fits of the constituent edges in G.

- **Graph Object Likelihood Fit:** The likelihood fit $\mathcal{GF}(G, \mathcal{C}_1 \ldots \mathcal{C}_r)$ for a graph object G with respect to the partitions $\mathcal{C}_1 \ldots \mathcal{C}_r$ is the geometric mean of the (composite) likelihood fits of the edges in G. Therefore, we have:

$$\mathcal{GF}(G, \mathcal{C}_1 \ldots \mathcal{C}_r) = \left[\prod_{(i,j) \in G} \mathcal{MF}(i, j, \mathcal{C}_1 \ldots \mathcal{C}_r) \right]^{1/|G|} \tag{1}$$

Partitions from Edge Samples

- The outlier discovery algorithm uses the maintenance of a group of r different *partitions which are implicitly defined by dynamically maintained reservoirs from the graph stream*.

- At a given time, the algorithm maintains r different reservoir samples (edges), which are denoted by $S_1 \dots S_r$ respectively.

- Each set S_m induces a different partitioning of the nodes in the underlying network, which is denoted by the notation $\mathcal{C}_m$.

- The partition $\mathcal{C}_m$ is induced by determining the connected components in the subgraph defined by the edges in S_m.

Modeling Edge Generation Probabilities

- The statistical behavior of the edges in the set corresponding to $Q_m = \cup_{j=1}^{r} S_j - S_m$ is used in order to model the edge generation probabilities.

- We model the *sample-estimated* probability of an edge between partitions i and j as the fraction of the number of edges in Q_m which exist between partitions i and j.

- We model the *non-sample-estimated* probability of an edge between partitions i and j by assuming that the probability of an edge between any pair of nodes is equal $\Rightarrow p_{ij}^{n}(\mathcal{C}_m)$ (used for smoothing).

- The *estimated* probability $p_{ij}(\mathcal{C}_m)$ of an edge between partitions i and j is a weighted combination of the above two factors.

Online Outlier Detection

- For each incoming graph stream object, we compute the likelihood fit with the use of the above-mentioned estimated probabilities.

- Those objects for which this fit is t standard deviations below the average of the likelihood probabilities of all objects received so far are reported as outliers.

- We use the reservoir sampling approach in order to update the edges in the different samples $S_1 \ldots S_r$.

- These are also used in order to update the partitions $\mathcal{C}_1 \ldots \mathcal{C}_r$.

Online Outlier Detection

- Need to determine whether the likelihood probability of an object is t standard deviations below the likelihood probability of all objects received so far.
  - We need to dynamically maintain the mean and standard deviation of the likelihood probabilities of all objects maintained so far, in a way which can be *additively achieved* for a data stream.
- Can be achieved by maintaining all moments upto order 2 (micro-clustering concept).

Structural Reservoir Sampling (Aims and Goals)

- The aim of the partitioning is to construct clusters of dense nodes, which would expose the outliers well.

- Since clustering is a challenging problem for the stream scenario, we induce node partitions from *edge samples*.

- It is well known that the use of edge sampling to create such partitions are biased towards creating partitions which are dense.

- We also need to *maintain specific structural properties* in the underlying partitions.

  - Eg. (a) The number of points in each partition is constrained. (b) Number of components are constrained

Structural Reservoir Sampling (Definition)

- We extend the approach to an unbiased sample of a structured graph with a *structural stopping criterion*.

- Many natural and desirable structural properties of the sample are maintained with the help of a *monotonic set function* of the underlying edges in the reservoir.

- **Monotonic Set Function** A monotonically non-decreasing (non-increasing) set function is a function $f(\cdot)$ whose argument is a set, and value is a real number which always satisfies the following property:

  – If S_1 is a superset (subset) of S_2, then $f(S_1) \geq f(S_2)$

Stochastic Stopping Criterion

- Some examples of a monotonic set function:
  - The function value is the number of connected components in the edge set S (monotonically non-increasing).
  - The function value is the number of nodes in the largest connected component in edge set S (monotonically non-decreasing).

- In some cases, we can use *thresholds* on the above properties, which are also referred to as *stochastic stopping criteria*.

Stopping Criterion with Random Sort Sample

- The edges are sorted in random order, and can be added to S only in **sort order priority**.

- **Sort Sample with Stopping Criterion:** Let $\mathcal{D}$ be a set of edges. A sort sample S from $\mathcal{D}$ with stopping threshold α is defined as follows:

  – We pick the *smallest* subset S from $\mathcal{D}$ **among all subsets which satisfy the sort-order priority**, such that $f(S)$ is at least (at most) α.

- Example: Smallest subset of edges which results in violation of the size constraint on number of nodes in connected component

  – Use penultimate set by removing the last added edge

Observations

- For the case of a *fixed data set*, it is fairly easy to create a sort sample with a structural stopping criterion.

- We achieve this by sorting the edges in random order and adding them sequentially, until the stopping criterion can no longer be satisfied.

- In the case of a data stream, a random sample or reservoir needs to be maintained *dynamically*.

- Once edges have been dropped from the sample, how does one compare their sort order to the incoming edges in the stream, and correspondingly update the sample?

Maintaining Sort order in Stream Case

- The key idea is use a *fixed random hash function*, which is computed as a function of the node labels on the edge.

- This hash function is used to create a sort order among the different edges.

- This hash function serves to provide *landmarks* for incoming edges when they are compared to the previously received edges from the data stream.

- The use of a hash function *fixes the sort order among the edges throughout the stream computation*.

- For edge (i, j) we compute the hash function $h(i \oplus j)$, where $i \oplus j$ is the concatenation of the node labels i and j.

Stopping Criterion with Hash Function

- Let $\mathcal{D}$ be a set of edges. let $f(\cdot)$ be a monotonically non-decreasing (non-increasing) set function defined on the edges. A sort sample S from $\mathcal{D}$ with stopping threshold α is equivalent to the following problem:
  - Apply a uniform random hash function $h(\cdot)$ to each edge (i, j) in $\mathcal{D}$.
  - Determine the smallest threshold q, such that the set S of edges which have hash function value at most q satisfy the condition that $f(S)$ is at least (at most) α.

Properties of Hash Function

- We denote the corresponding threshold value with respect to set function f, hash function h. data set $\mathcal{D}$ and stopping threshold criterion α by $H(f, h, \mathcal{D}, \alpha)$, and refer to it as the *stopping hash threshold* for data set $\mathcal{D}$.

- **Result:** *The stopping hash threshold exhibits a version of set monotonicity with respect to the underlying data set. Specifically, let us consider the monotonic set function $f(\cdot)$, hash function $h(\cdot)$, stopping threshold α. Let us consider two data sets $\mathcal{D}_1$ and $\mathcal{D}_2$, such that $\mathcal{D}_2 \supseteq \mathcal{D}_1$. Then, the stopping hash threshold $H(f, h, \mathcal{D}_2, \alpha)$ is at most equal to $H(f, h, \mathcal{D}_1, \alpha)$.*

Properties of Hash Function (Stream Context)

- *The stopping hash threshold is monotonically non-increasing over the life of the data stream.*

- This is a critical result, because it implies that edges which have not been included in the current sample will never be relevant for sampling over the future life of the data stream.

- The current sample is the only set we need for any future decisions about reservoir sample maintenance.

Structural Reservoir Sampling Algorithm

- Previous result implies simple algorithm in order to maintain the reservoir dynamically.

- We dynamically maintain the *current hash threshold* which is used to make decisions on whether or not incoming elements are to be included in the reservoir.

- For each incoming graph, we apply the hash function to each of its edges, and we add the edge to the reservoir, if the hash function value is less than the current threshold value.

Structural Reservoir Sampling Algorithm

- The addition of these edges will always result in the stopping criterion being met because of set function monotonicity.

- The set may no longer be the *smallest sort-ordered set* to do so.

- Edges may need to be removed in order to make it the smallest sort-ordered set to satisfy the stopping criterion.

- Process the edges in the reservoir *in decreasing order of the hash function value*, and continue to remove edges, until we are satisfied that the resulting reservoir is the smallest possible set which satisfies the stopping constraint.

Structural Reservoir Sampling Algorithm

- For each incoming graph, this process is repeated by first selectively adding the edges for which the hash function meets the threshold, and then removing edges if necessary.
- The corresponding hash threshold is then reduced to the largest hash function value of the *remaining edges in the reservoir* after removal.
- The removal of edges may result in a reduction of the hash threshold in each iteration.
- It will never result in an increase in the threshold.

Choice of Structural Function

- It is useful to set the function in such a way so as to prevent too small components from the reservoir.

- A good choice is to set $f(S)$ to the number of nodes in the largest connected component induced by the reservoir edge sample S.

- A corresponding stopping threshold of α is used on the function $f(S)$.

- The stopping criterion ensures that the sort sample defines the *smallest* set of edges, such that the largest connected component is *at least* α.

- This ensures that the largest partition has at most α nodes, *when the penultimate set derived from S is used*.

Experimental Results

- Tested on two real (IMDB and DBLP) and one synthetic data sets.
- For real data sets case studies are presented for effectiveness
- For synthetic data sets, precision and recall can be presented.
- Efficiency results are presented for both types of data sets.

Examples of Anomalous Bibliographic Objects (DBLP)

- Yihong Gong, Guido Proietti, Christos Faloutsos, *Image Indexing and Retrieval Based on Human Perceptual Color Clustering*, CVPR 1998: 578-585.
  - Edges across different kinds of partitions (a) Computer Vision and Multimedia Dominated : *Yihong Gong* (b) Database and Data Mining Dominated : *Christos Faloutsos*

- The two kinds of nodes were assigned to different partitions by the structural reservoir sampling algorithm

Examples of Anomalous Bibliographic Objects (DBLP)

- Natasha Alechina, Mehdi Dastani, Brian Logan, John-Jules Ch Meyer, *A Logic of Agent Programs*, AAAI 2007: 795-800.

- The co-authorship behavior of these cohorts was defined by geographical proximity.

- The first partition includes a group of researchers in the United Kingdom, while the second partition is composed of researchers in the Netherlands.

- The different groups were naturally assigned to different clusters.

Examples of Anomalous Movie Objects

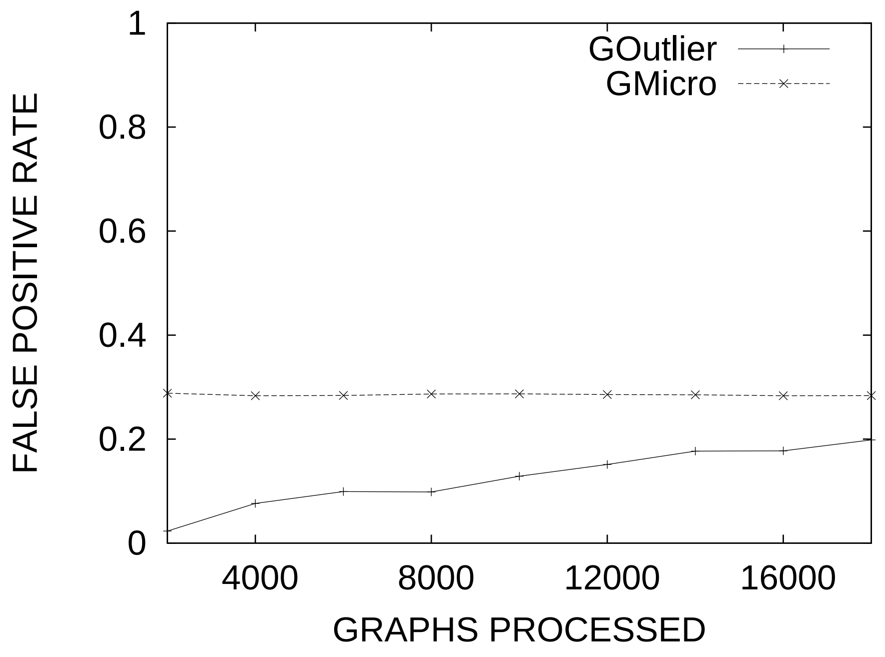
- *Movie Title:* Cradle 2 the Grave (2003)

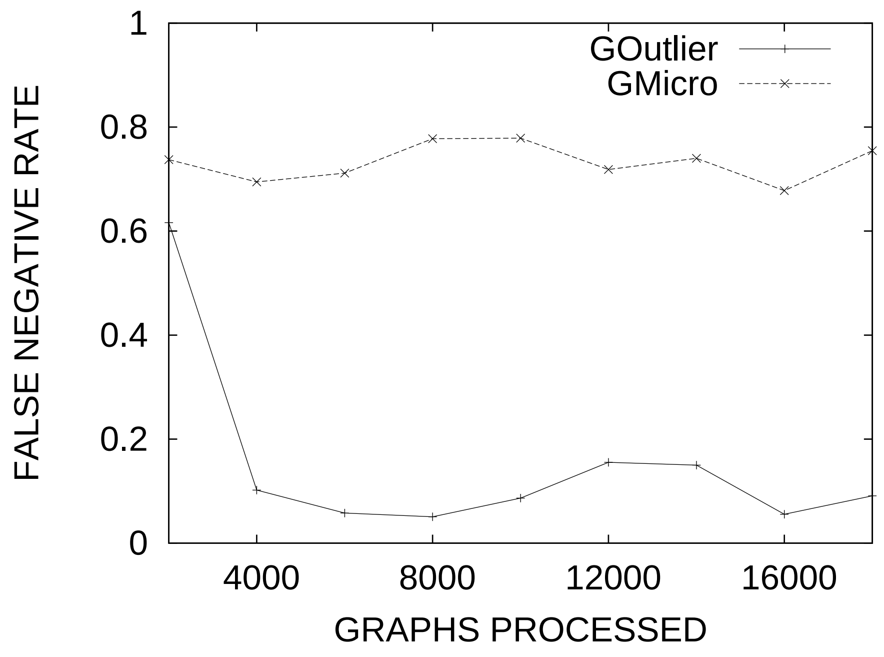
- This movie was directed by Andrzej Bartkowiak, and the actors include Jet Li, DMX (I), etc.

- Non-chinese director which contains an international cast along with many chinese actors.

- *Movie Title:* Memoirs of a Geisha, 2005: Contains participants from Chinese, Japanese and American backgrounds

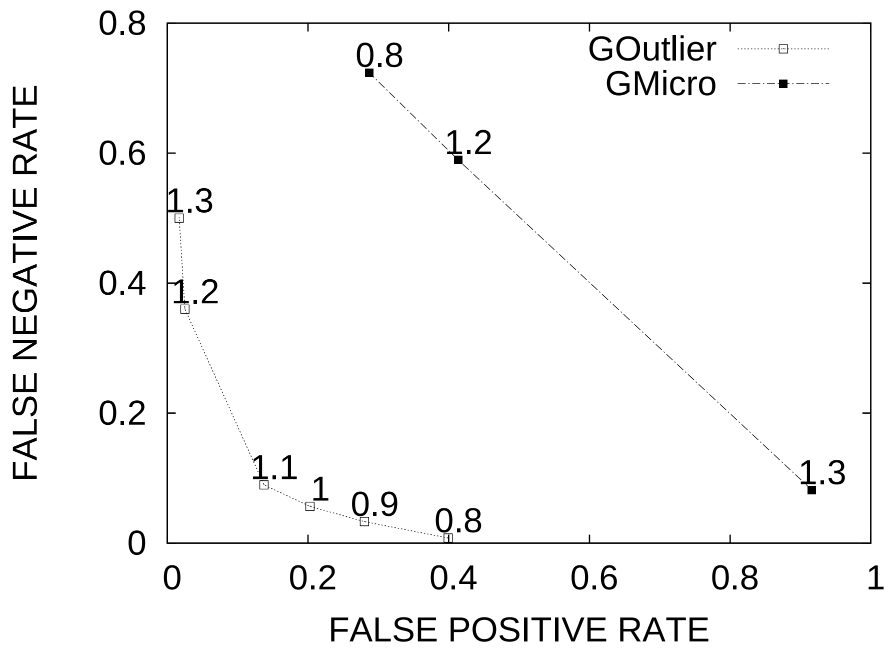
Effectiveness Results On Synthetic Data Set





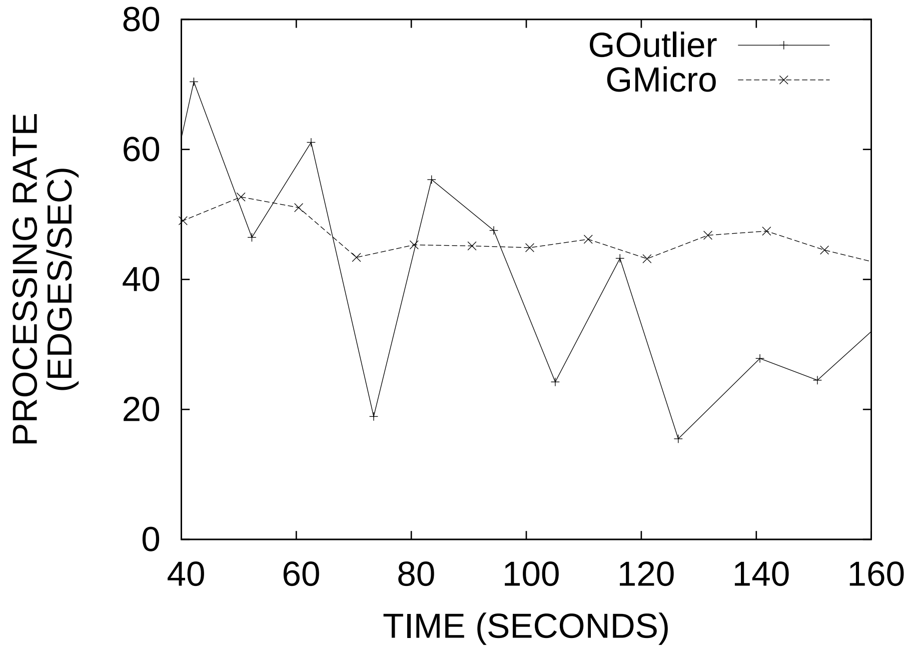
- Effectiveness Results on Synthetic Data Set

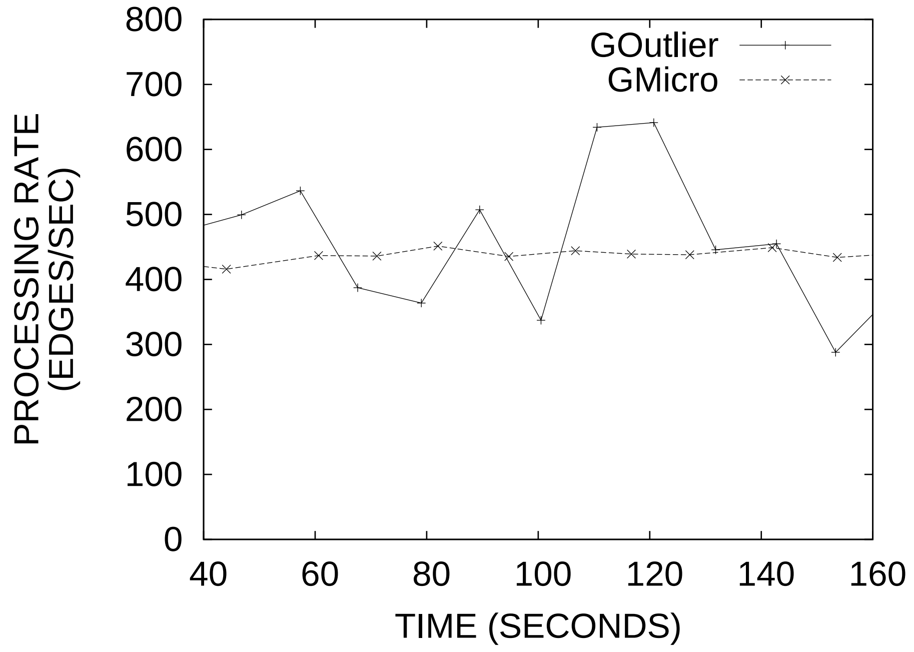
Effectiveness Results On Synthetic Data Set



- Effectiveness Results on Synthetic Data Set

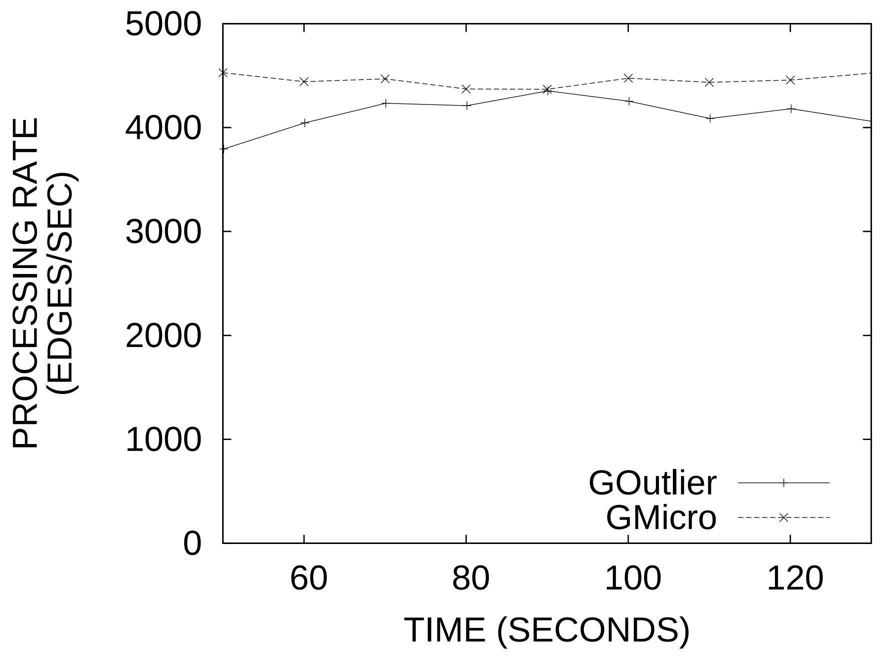
Efficiency Results (Real Data Sets)





- Efficiency Results on Real Data Sets (DBLP and IMDB)

Efficiency Results (Synthetic Data Sets)



- Efficiency Results on Synthetic Data Set

Conclusions and Summary

- New method for outlier detection in graphs streams
- Proposed methods for structural reservoir sampling, which may have other applications
- Presented results on real and synthetic data sets