Fast Memory-Efficient Anomaly Detection in Streaming Heterogenous Graphs


Emaad Manzoor*

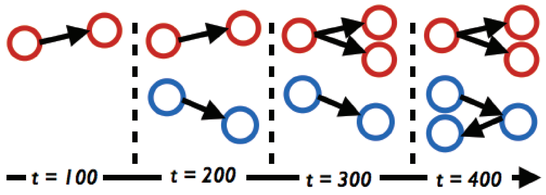

Sadegh M. Milajerdi


Leman Akoglu

emanzoor@cs.stonybrook.edu

Code and data at **bit.ly/streamspot**

StreamSpot tracks anomalous heterogenous graph objects as they evolve from a stream of typed edges.



Challenges

Bounded-space graph representation

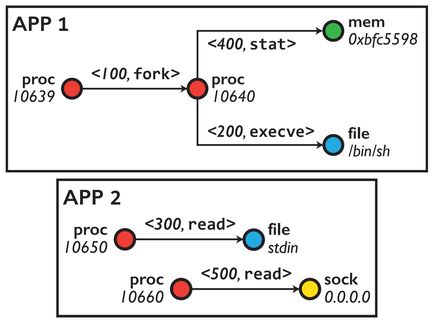
Streaming graph comparision

Fast and constant-time per-edge

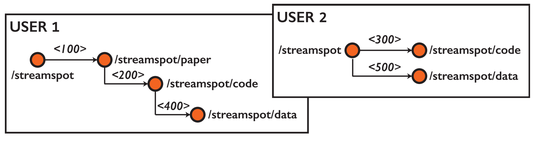
Accurate anomaly scoring

Applications

Detecting **malicious software** from a stream of **system call logs**



Detecting **malicious users** from a stream of **filesystem navigation** traces

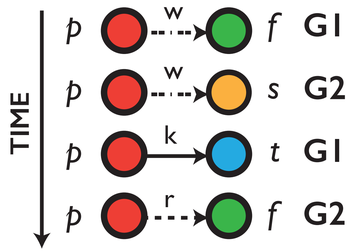


Thanks to support from DARPA NSF NORTHROP GRUMMAN

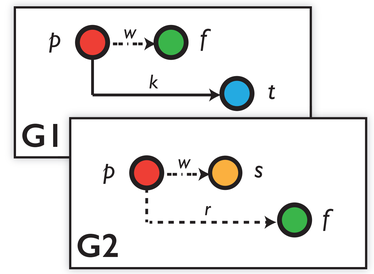
Typed event stream as dynamic heterogenous graphs

Node types
- (p)rocess
- (t)hread
- (f)ile
- (s)ocket

Edge types
- for(k)
- (r)ead
- (w)rite



Typed Edge Stream

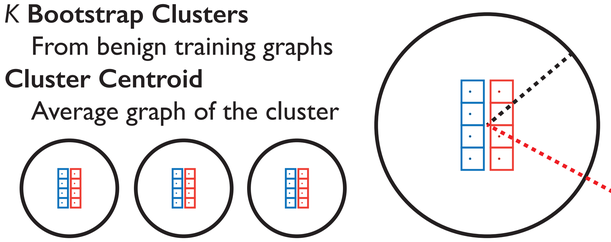


Dynamic Heterogenous Graphs

Clustering-based anomaly detection

K **Bootstrap Clusters**
From benign training graphs

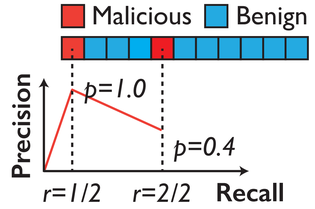
Cluster Centroid
Average graph of the cluster



Cluster Radius
3 standard-deviations from the mean graph to centroid distance

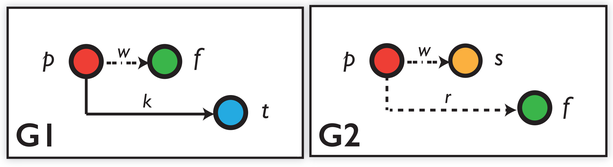
Anomaly Score
Distance to nearest cluster centroid

Anomaly Ranking



Dynamic heterogenous graphs as vectors by shingling

OkBFT Ordered k-hop Breadth-first Traversal



Shingle from *p* = pwfkt Shingle from *p* = pwsrf

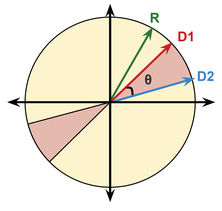
Shingle Vectors

	pwfkt	pwsrf
G1	1	0
G2	0	1

Shingle Universe *S*
Large!
Unknown!

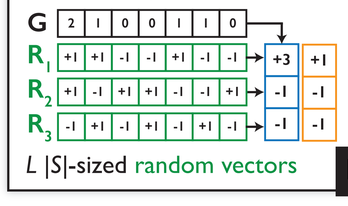
Streaming pairwise graph similarity with StreamHash

Why does SimHash fail?



Probability [D1 and D2 on the **same side** of R]

~ *cosine*(D1,D2)



L dot-products

L bit sketch and signed random projection

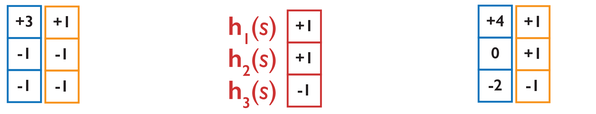
L |S|-sized random vectors

But *S* is unknown!

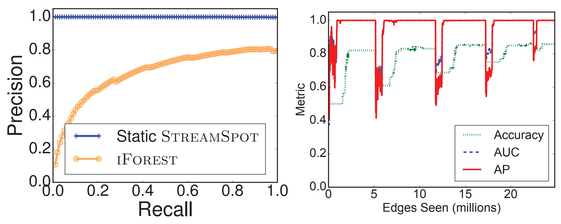
Why cache when you can hash?

$h_i(s)$: s → {+1, -1}
Store h_1 ... h_L in **constant space**

Incremental sketch construction
Add/remove shingle *s* in **constant time**

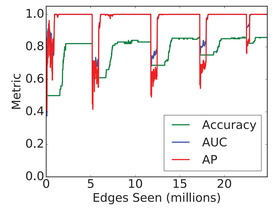


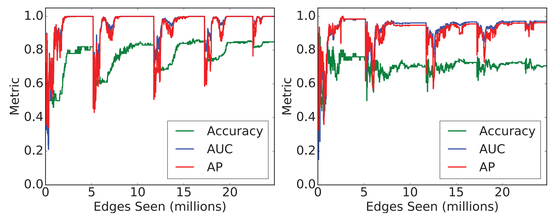
Accuracy



Offline Online

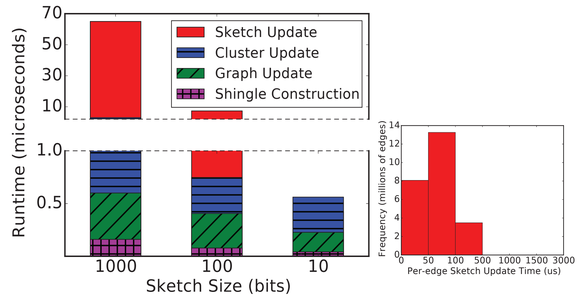
L = 1000





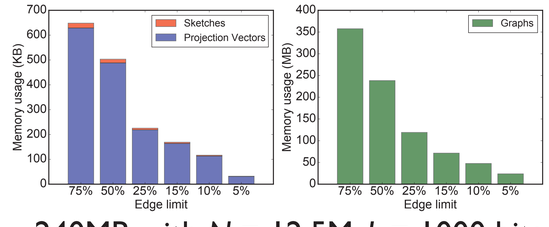
L = 100 *L* = 10

Time



L = 100 → 100,000 edges/s

Space



~240MB with *N* = 12.5M, *L* = 1000 bits