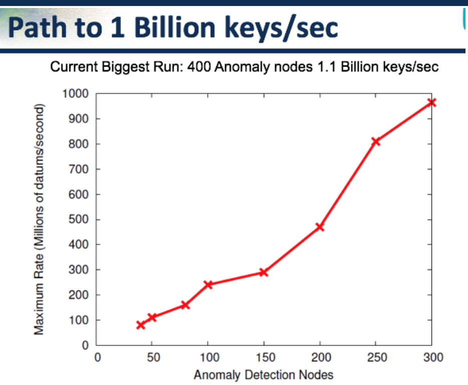
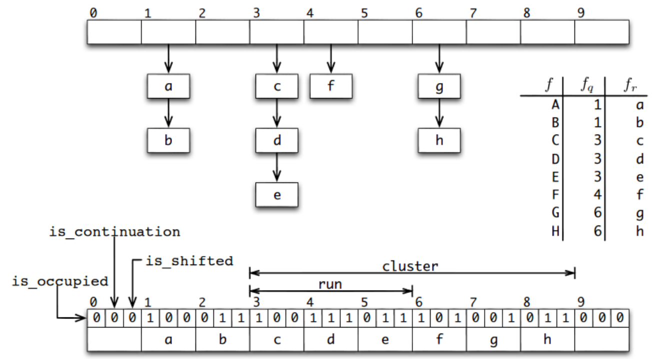
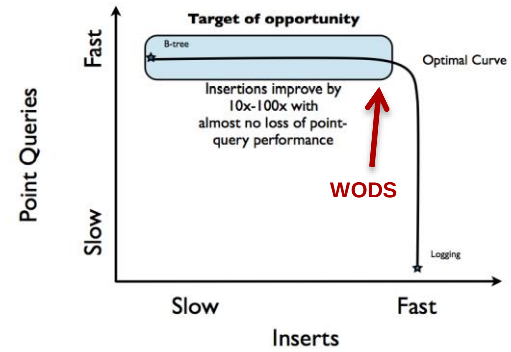
Exceptional service in the national interest

Sandia National Laboratories







Advanced Data Algorithms & Architectures for Security Monitoring

Thomas M Kroeger, Cindy A Phillips, Eric D Thomas, Brian J Wright
Rutgers, Stony Brook University, VMWare RG

U.S. DEPARTMENT OF ENERGY

NNSA National Nuclear Security Administration

Sandia National Laboratories is a multi-mission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525. SAND2020-11253 C

Too much data to use it effectively

Current systems don't support querying historical data in a timely manner.

Sensors are collecting data at incredible rates.

Typically linearly logs with little to no organization for example: cyber connections or power grid state.

Analytics are starting to understand this data

- Typically overwhelmed w/ data
- Stay in RAM and respond quickly
- Use disk and respond in days

Responding at Machine Speed

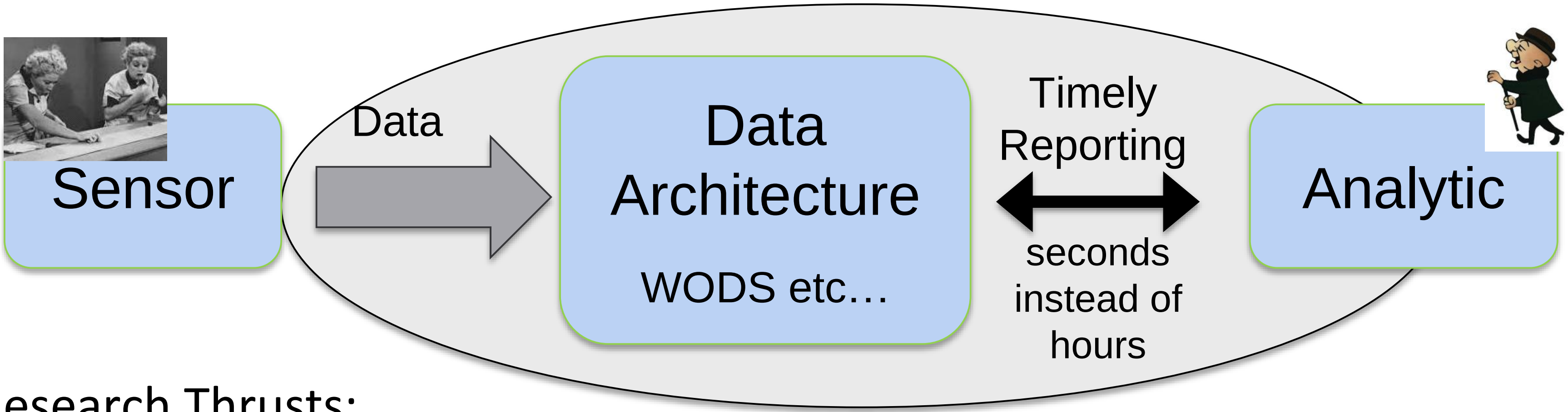
- Systems that respond and prevent attacks requires analytics that work at machine speed.
- Current disk/log based tools take hours.
- Ram based systems loose data quickly
- Low and slow attackers exploit this

Data Architectures to Bridge this Gap

Bottom line up front (BLUF)

Use Write Optimized Data Structures (WODS) to build new architectures to bridge this gap and enable machine speed analytics

- Track data sets far larger than core memory
- Enable sustained long-term low-maintenance operations



Research Thrusts:

1. **New data architectures** to support our cyber missions
2. **Algorithm research** to address known limits, and
3. **Rethink** how we do **analytics** using these new capabilities

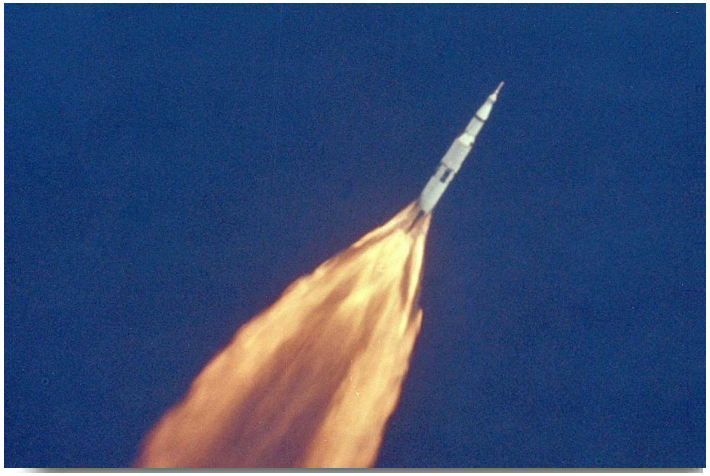
Memory and Disk access times

RAM: ~60 nanoseconds per access
Disks: ~6 milliseconds per access.
disk is ~100,000 times slower

Analogy:

- RAM = escape velocity from earth (25,000 mph)
- disk = walking speed of the giant tortoise (0.3mph)

~83,333x slower



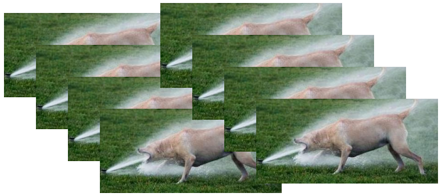
Current Approaches

No capability of timely reporting across data larger than RAM

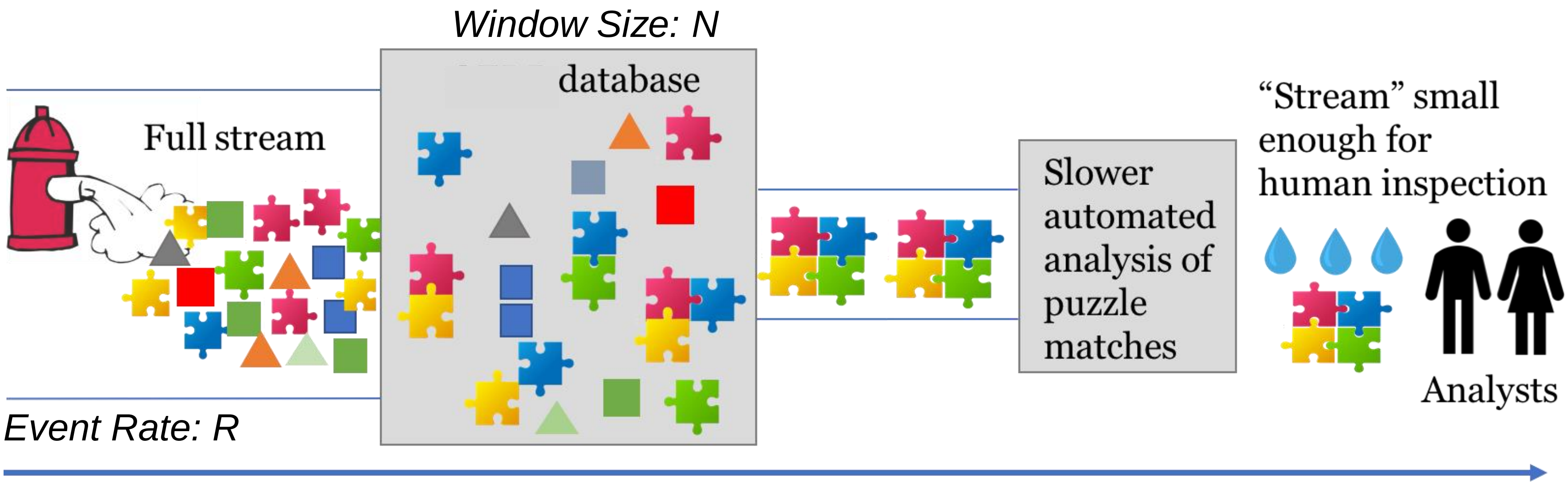
- One disk write per insert takes ~6ms
- Best rates of 200 – 2000 inserts per second
- We see rates of 100K to millions

Clustering?

- Log processing tools and large scale parallel data stores (hadoop, Splunk and postgres)
- Cyber responders have long been fighting issues of ingestion rate, query response and data size.
  - They have many parallel machines and lots of experts to tune the system at some cost.
  - In the end they still do grep in parallel across large logs.



Standing Queries & Firehose



Database requirements:

- No false negatives
- Limited false positives
- Immediate response preferred
- Window of size N limits insights
- Rate of R typically means RAM

Firehose benchmark

- Captures essence of monitoring
- Sandia + DoD partners
- Input: stream of (key, value) pairs
- Report a key when seen 24th time.

http://firehose.sandia.gov/

Limits of Current RAM Based Analytics

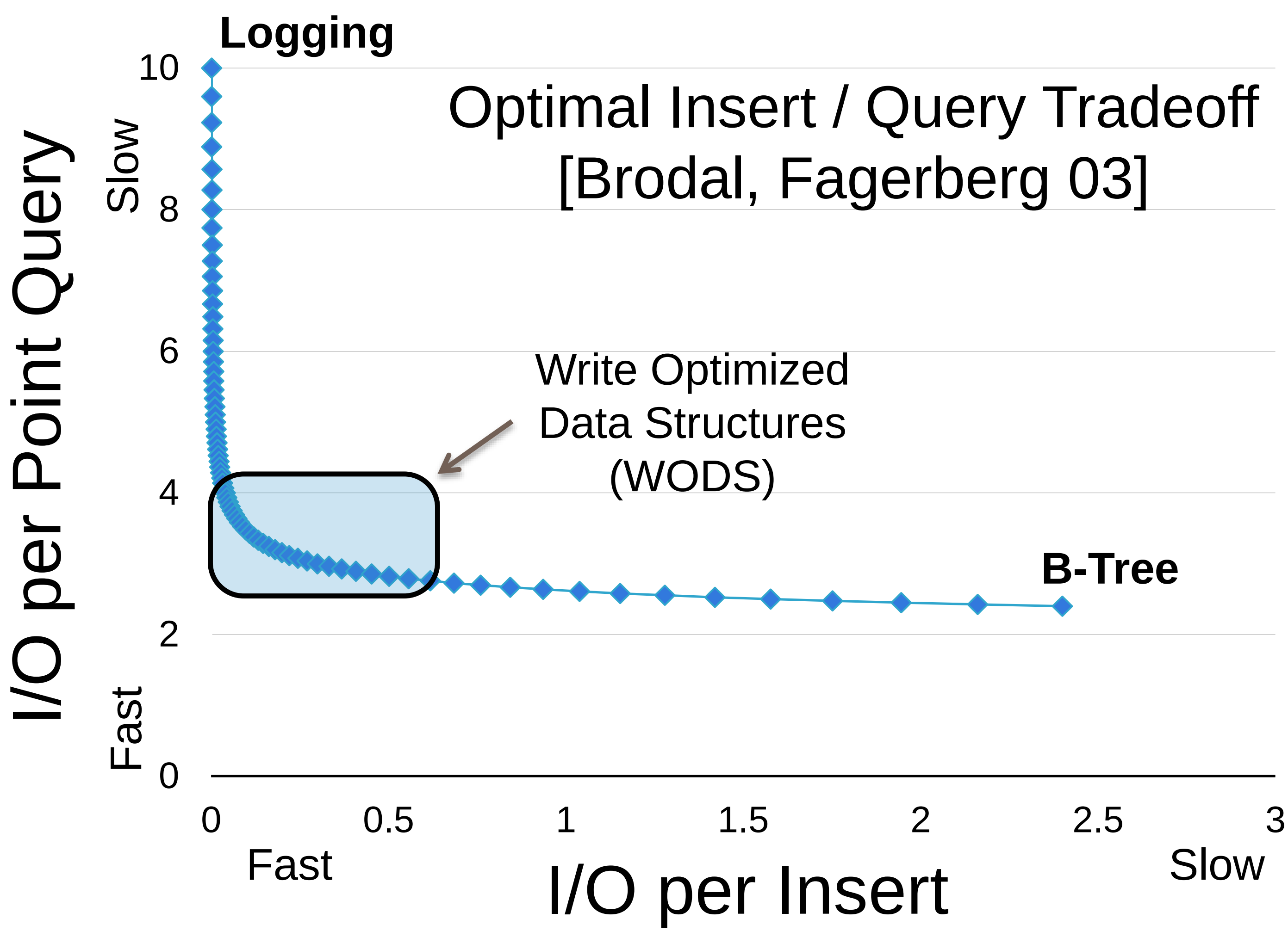
- Tested state of the art analytic, waterslide with firehose
  https://github.com/waterslideLTS/waterslide
- Accuracy of cyber-analytics depends on window size
- As the monitored set grew beyond RAM accuracy fell quickly

Analytic Size	Firehose Size	Ratio	Events Found
1048576	1048576	1x	66.04%
1048576	2097152	2x	23.82%
1048576	4194304	4x	**0.06%**

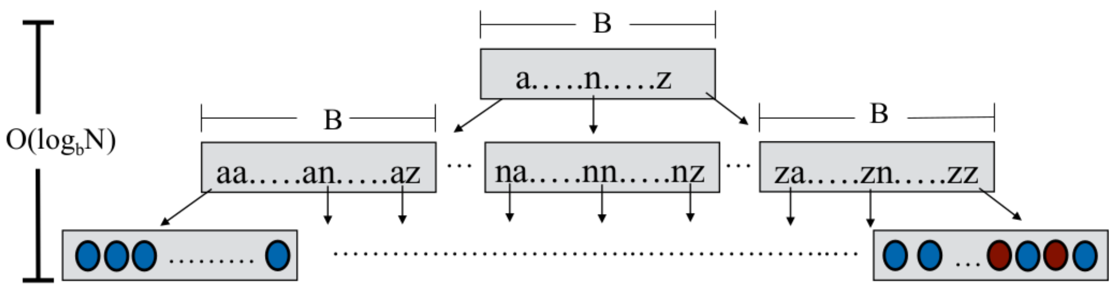
Its clear we need more space.

How do we integrate storage without loosing performance?

Write Optimized Data Structure

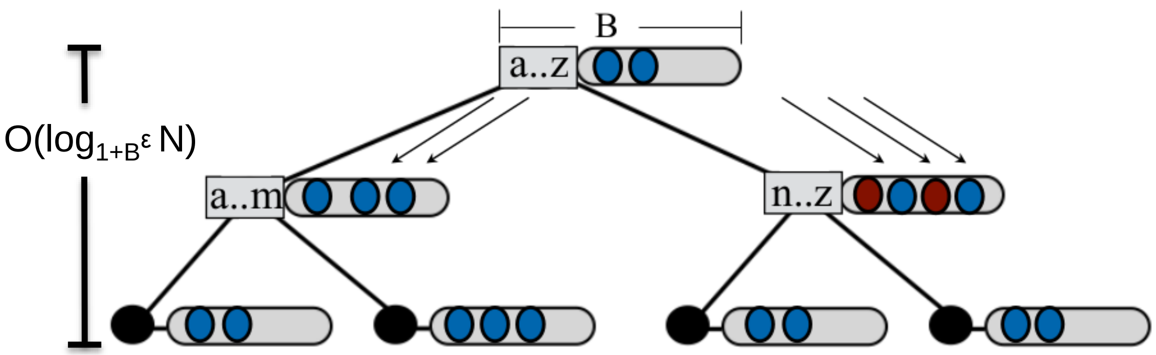


B-Tree & B^{ε}-Tree



B-Tree is used to index keys.

Insert & Lookup take $O(\log_B N)$



B^{ε}-Tree buffers inserts at each layer in the tree to aggregate writes.

Lookup takes $O(\log N)$

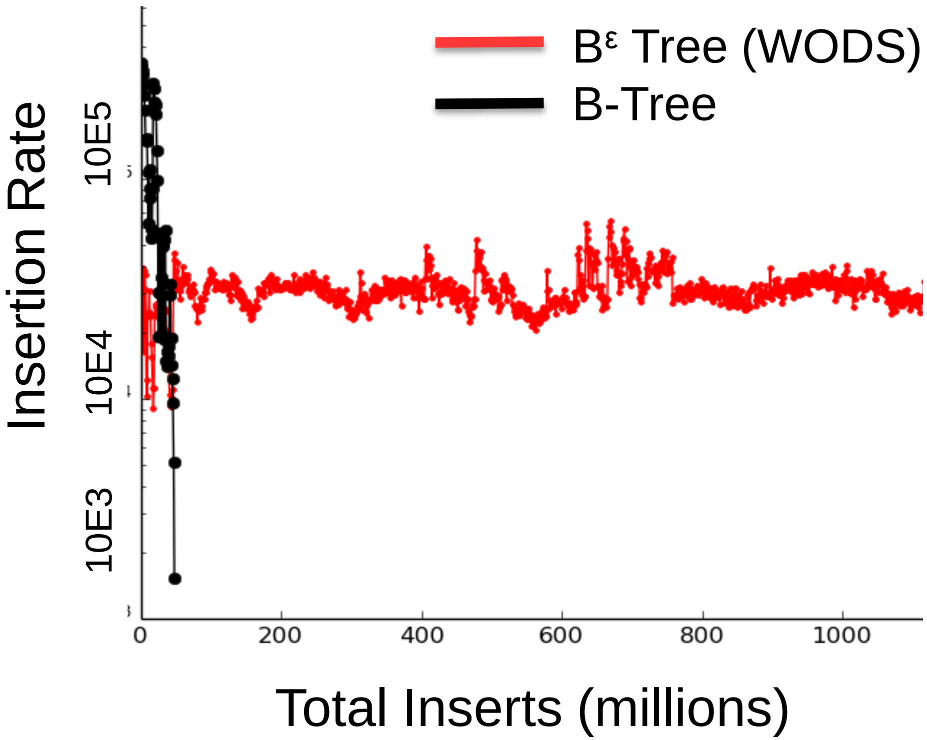
Insert takes $O\left(\frac{\log N}{B}\right)$

Inserts upto 100x faster

Take Away: WODS offers a balance between RAM and Disk for fast ingestion and organized data.

Comparing WODS to Traditional B-Trees

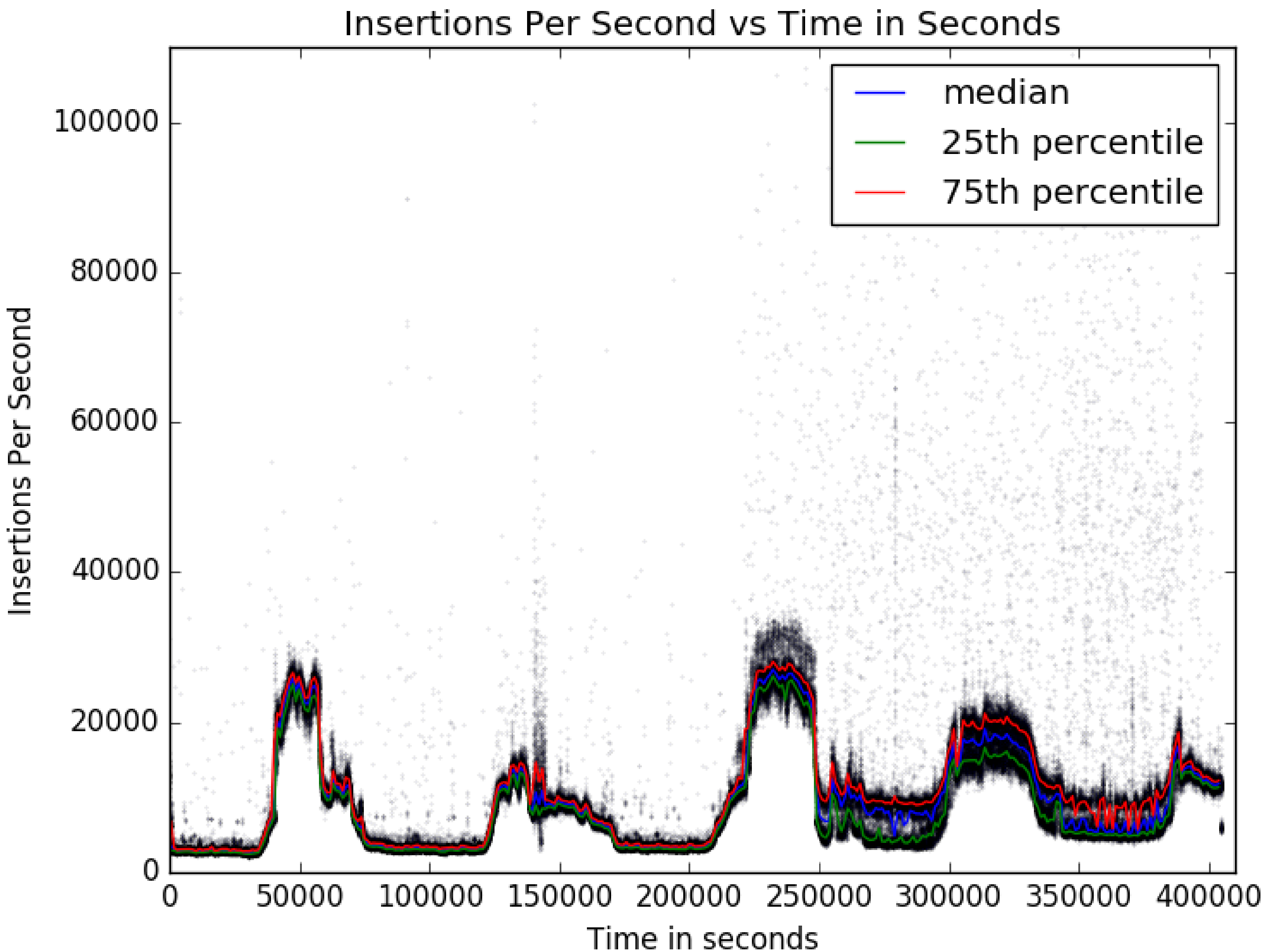
Insertion Rates B^ε Tree v B-Tree



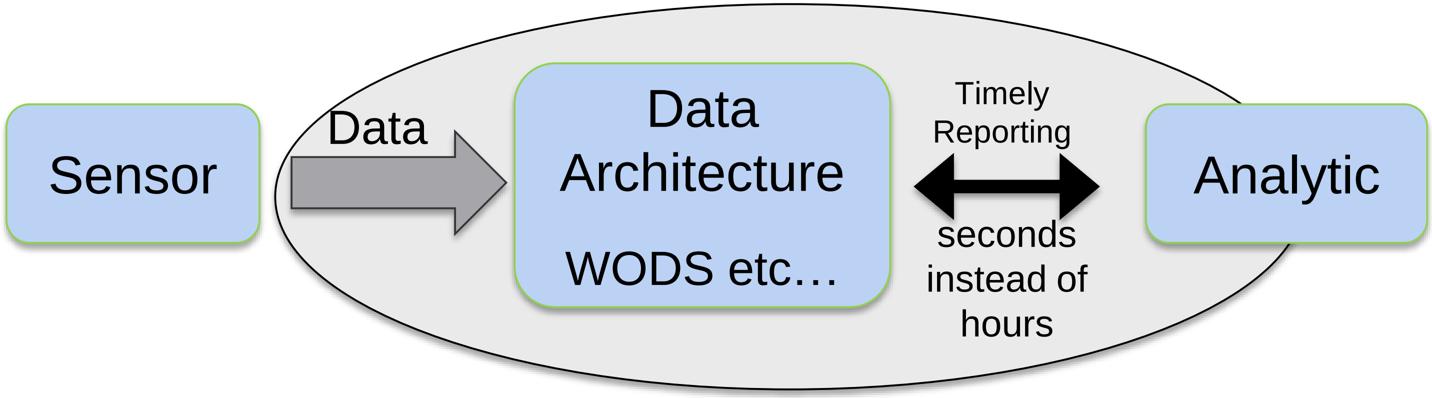
BADGERS 2015 Paper

- Compared indexing IP connections with B-Tree and WODS - B^ε Tree
- B-Tree initial better but
- Quickly reduced to unsustainable rates.
- B^ε Tree able to sustain reasonable indexing throughput

Tracking Network Connections at SCinet



Research Thrusts Going Forward



Research Thrusts:

1. **New data architectures** and prototype tools that use WODS to track real-world events to support our cyber missions
   - Our Demand query tool (DQT) & Standing query tool (SQT) serve as vehicles for researching advanced architectures and algorithms on real-world data.
2. **Algorithm research** to address infinite streams of data, including expiration, sustainability, and adaptability, and
3. **Rethink** how we do **analytics** using these new capabilities to support machine speed consequence mitigation

Didn’t <Big Tech.com> Already Solve This?

NO.

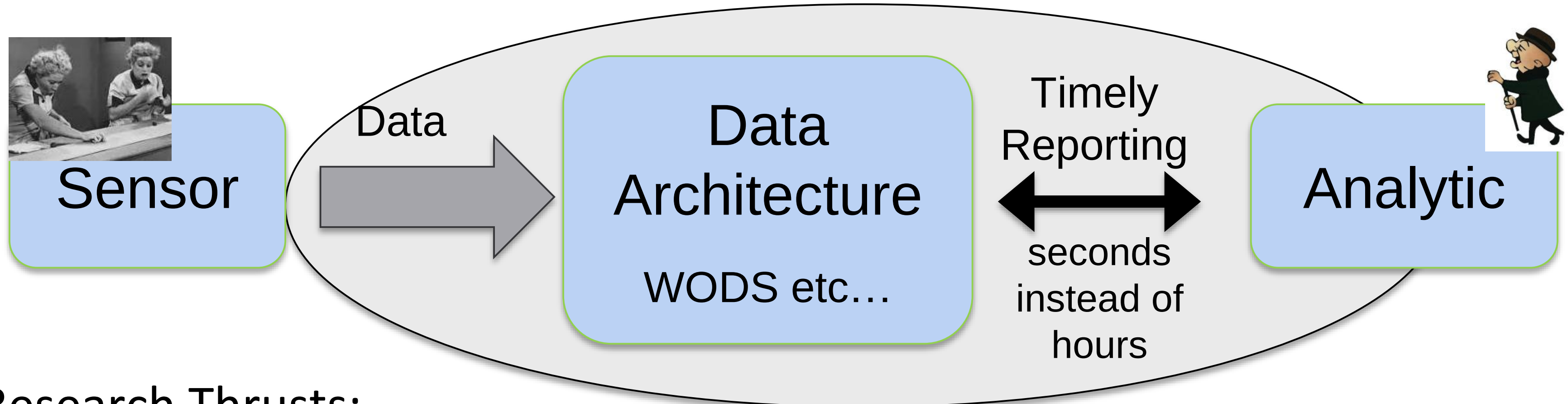
- Our problem space needs to ingest millions of events per second and answer questions in seconds while maintaining a state space on secondary storage.

- Some indexes the data over night and doesn’t have to provide answers up to the second
- They work in standing queries are at thousands per second we’re at 100k--millions.

Conclusion

Use Write Optimized Data Structures (WODS) to build new architectures to bridge this gap and enable machine speed analytics

- Track data sets far larger than core memory
- Enable sustained long-term low-maintenance operations



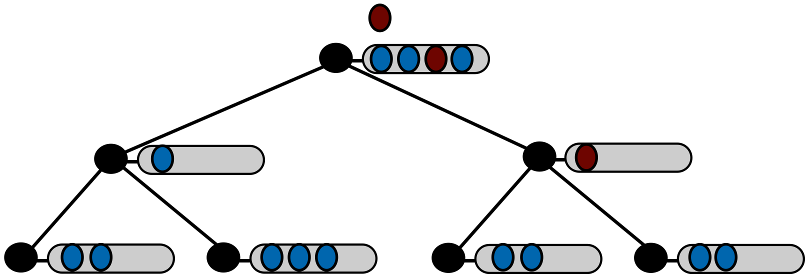
Research Thrusts:

1. **New data architectures** to support our cyber missions
2. **Algorithm research** to address known limits, and
3. **Rethink** how we do **analytics** using these new capabilities

Backup Slides

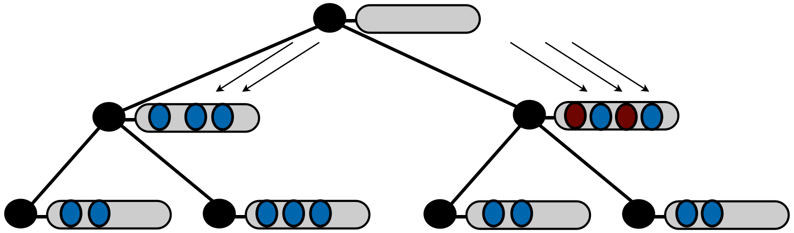
Write Optimized B-Tree

We used is a combination called B^eTree (pronounce B to the epsilon tree) that balances branching and buffering at each node.



Aggregates writes with a buffer of size B at each at each node. *e* slots are used as pivots and B-*e* are used as buffers.

Flush costs O(1) and happens O(1/B). The result is inserts are now O((logN)/B)



For a large B ~1024 this can be 100x faster in practice. [Bender 2007]

Memory and Disk access times

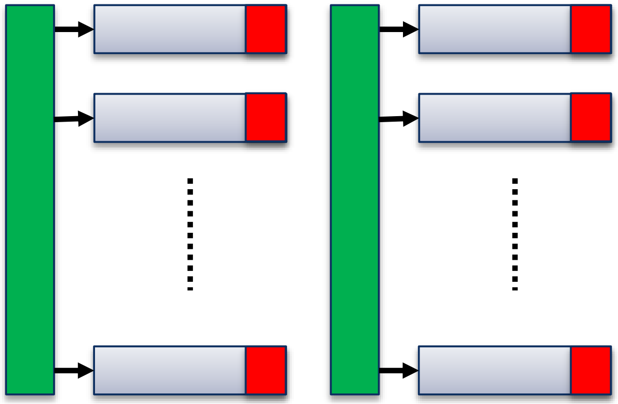
Disks: ~6 milliseconds per access.
RAM: ~60 nanoseconds per access

Analogy:

- disk = distance from home to first base (90 feet)
- RAM = distance from AT&T Park to Kauffman Stadium (1500 miles)

What is Happening?

- **Waterslide uses ‘d-left hashing’**
  - Two rows of buckets
  - Constant-size
  - Fast
  - Waterslide adds LRU expiration *per bucket*



Broder, Andrei, and Michael Mitzenmacher. "Using multiple hash functions to improve IP lookups." *INFOCOM 2001*

- **1/16 of all data is always subject to immediate expiration in steady state**

- **As active generator window grows, FIREHOSE accuracy quickly goes to zero**

Even when window size is only 4x data structure size, most reportable data are lost before It is reported.