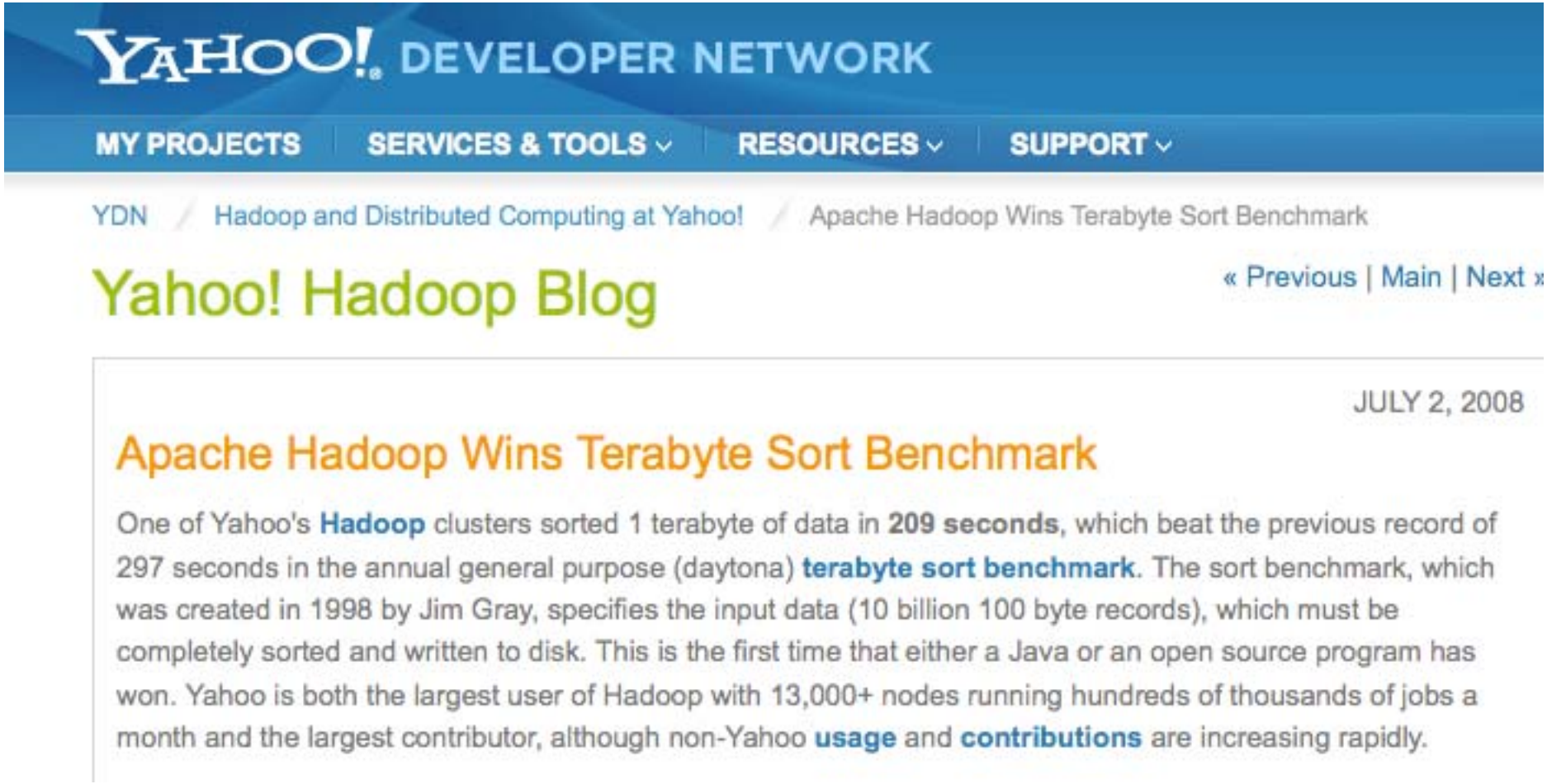
MalStone and MalGen

Robert Grossman
Open Data Group
Open Cloud Consortium

Joint work with Collin Bennett, David Locke, Jonathan Seidman and Steve Vejcik

Part 1. Other Communities are not Afraid of Benchmarks



Hadoop wins 2008 Terasort in 2008 in 209 seconds.

- Hadoop cluster with 910 nodes
- Sorted 1 TB of data consisting of 10 billion 100-byte records and writing results to disk
- Each node has 2 quad core 2.0 GHZ Xeons
- 8 GB RAM per node
- 40 nodes per rack
- 8 Gbps Ethernet uplinks from rack to switch

Why Is This Important?

- Helpful when designing out of memory algorithms.
- Helpful when porting applications to MapReduce and similar environments.
- Helpful when benchmarking different rack architectures.
- Helpful to those designing large data clouds to understand trade off space.

MapReduce Terasort

- The job used 1800 maps and 1800 reduces
- Hadoop pre-0.18 *with optimization patches so intermediate results not written to disk*
- Allocated enough memory buffers to hold intermediate data in memory
- Code checked in as Hadoop example by Hadoop team

A Measure of Transaction Processing Power[1]

Anon Et Al

February 1985

Proposed:

- sort
- scan
- DebitCredit

AlphaSort: A Cache-Sensitive Parallel External Sort

Chris Nyberg, Tom Barclay, Zarka Cvetanovic, Jim Gray, Dave Lomet

Proposed:

- minute sort
- penny sort

Sorting Benchmarks

- GraySort - TBs / min over large data (at least 100 TB)
- PennySort – Amount of data that can be sorted in penny's worth of system time
- MinuteSort – Amount of data that can be sorted in 60 seconds or less
- JouleSort – Amount of energy required to sort either 10 GB, 100 GB or 1 TB of data
- Terabyte Sort Time to sort 10 Billion records (depreciated, replaced by MinuteSort)
- Datamation Sort - time to sort 1 million records (depreciated, takes less than a second)

Source: sortbenchmark.org/

A new sort algorithm, called AlphaSort, demonstrates that commodity processors and disks can handle commercial batch workloads. Using commodity processors, memory, and arrays of SCSI disks, AlphaSort runs the industry-standard sort benchmark in seven seconds. This beats the best published record on a 32-CPU 32-disk Hypercube by 8:1. On another benchmark, AlphaSort sorted more than a gigabyte in a minute.

AlphaSort is a cache-sensitive memory-intensive sort algorithm. We argue that modern architectures require algorithm designers to re-examine their use of the memory hierarchy. AlphaSort uses clustered data structures to get good cache locality. It uses file striping to get high disk bandwidth. It uses QuickSort to generate runs and uses replacement-selection to merge the runs. It uses shared memory multiprocessors to break the sort into subsort chores.

Source: Abstract from AlphaSort: A Cache-Sensitive Parallel External Sort, Chris Nyberg, Tom Barclay, Zarka Cvetanovic, Jim Gray, Dave Lomet

Is Terasort relevant to the KDD community?

Not that much….

… So what benchmark is relevant for large scale analytics?

Part 2. Log Files are Everywhere

Log Files Are Everywhere

- Advertising systems
- Analyzing system logs
- Health and status monitoring
- Identifying online threats
  (“ghost in the browser”)

What are the Common Elements?

- Time stamps
- Sites
  - e.g. Web sites, computers, network devices
- Entities
  - e.g. visitors, users, flows
- Log files fill disks, many, many disks
- Behavior occurs at all scales
- Want to identify phenomena at all scales
- Need to group “similar behavior”
- Need to do statistics (not just sorting)

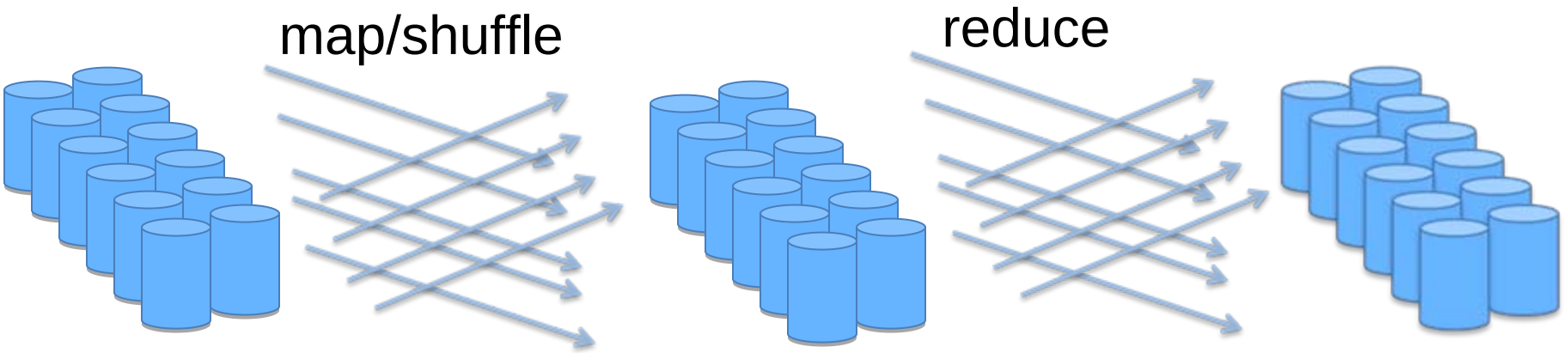
Abstract the Problem Using Site-Entity Logs

Example	Sites	Entities
Measuring online advertising	Web sites	Consumers
Drive-by exploits	Web sites	Computers (identified by cookies or IP)
Compromised systems	Compromised computers	User accounts

MalStone Schema

- Event ID
- Time stamp
- Site ID
- Entity ID
- Mark (categorical variable)
- Fit into 100 bytes

Toy Example



Events collected by device or processor in time order

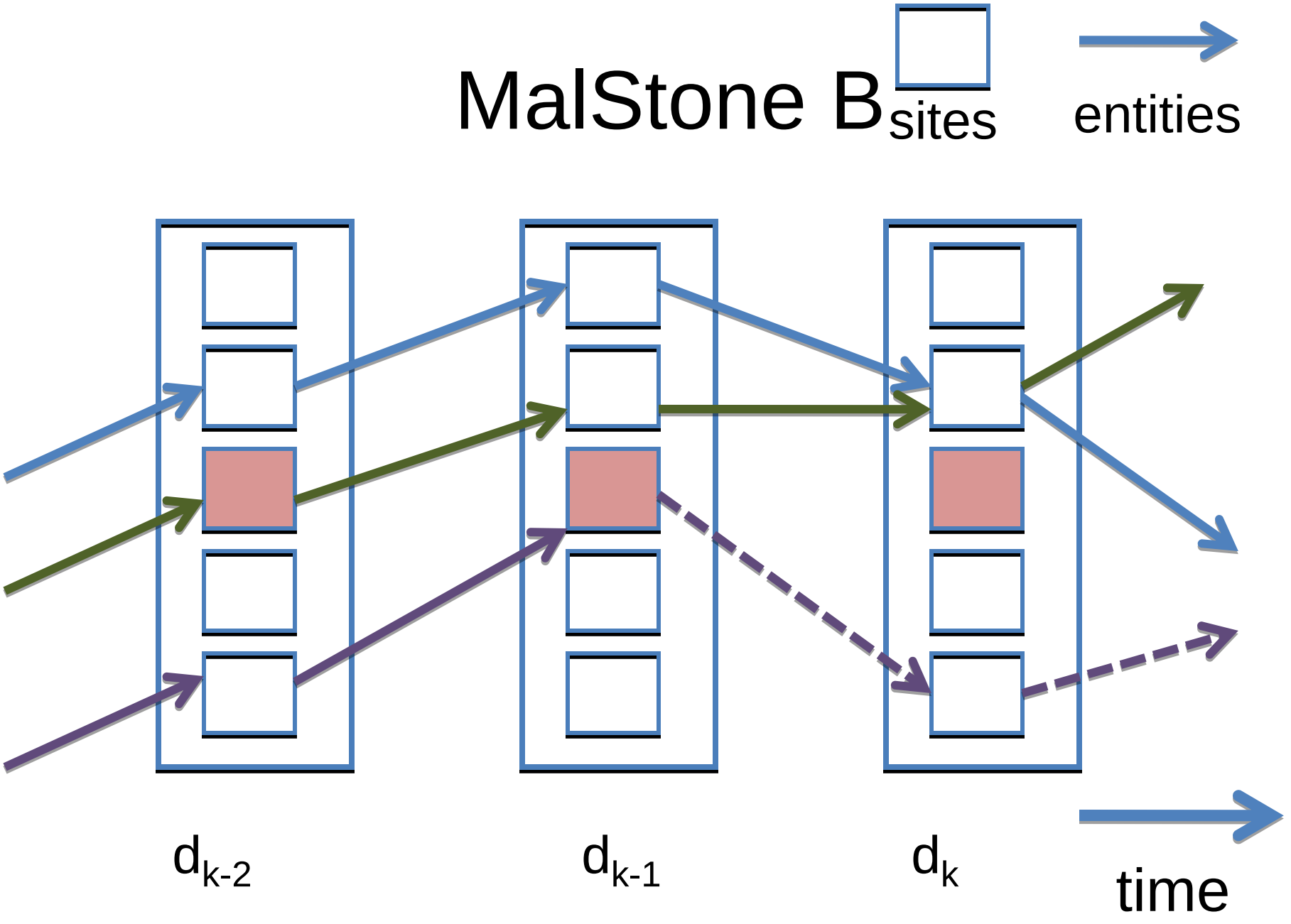
Map events by site

For each site, compute counts and ratios of events by type

Distributions

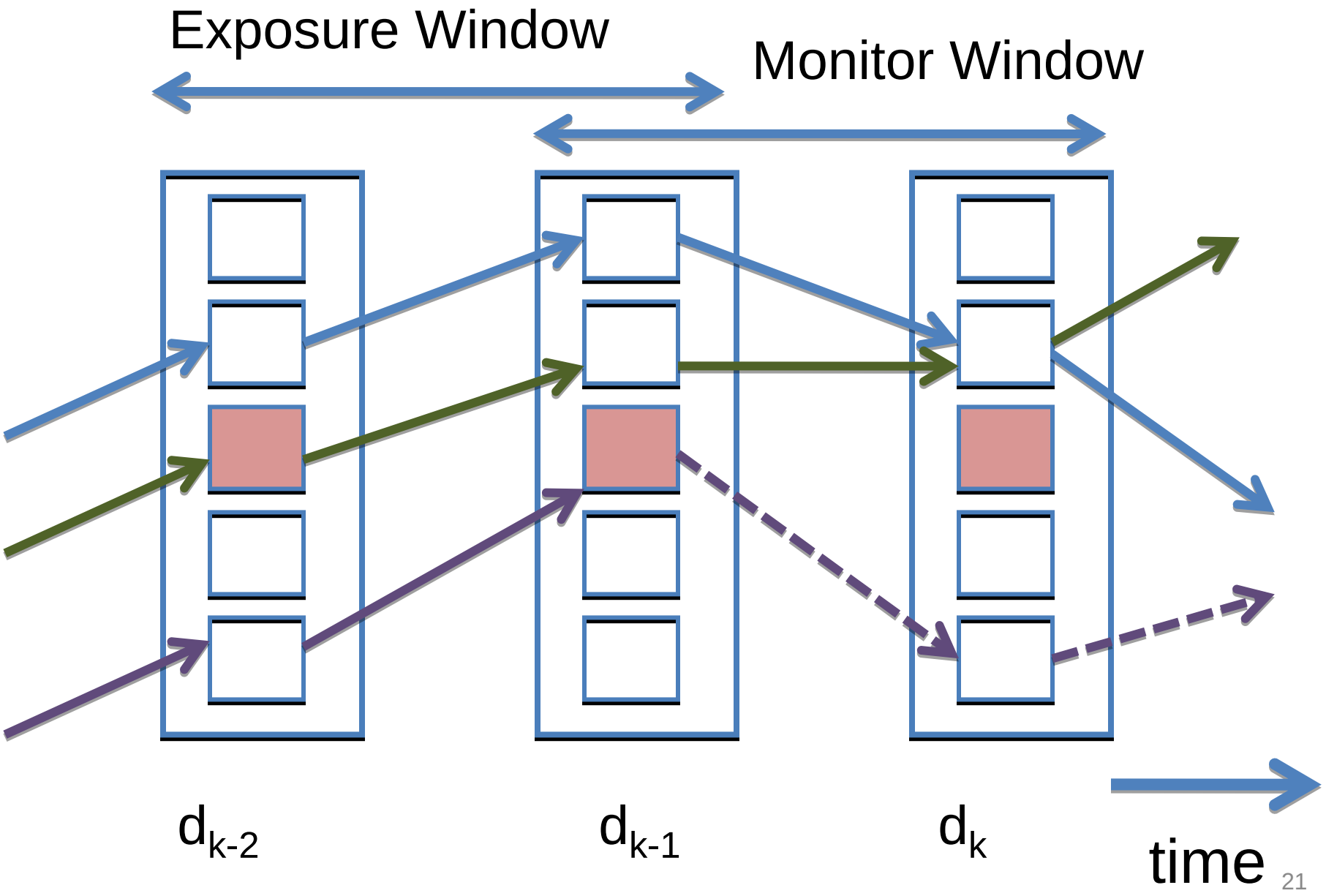
- Tens of millions of sites
- Hundreds of millions of entities
- Billions of events
- Most sites have a few number of events
- Some sites have many events
- Most entities visit a few sites
- Some visitors visit many sites

MalStone B



The Mark Model

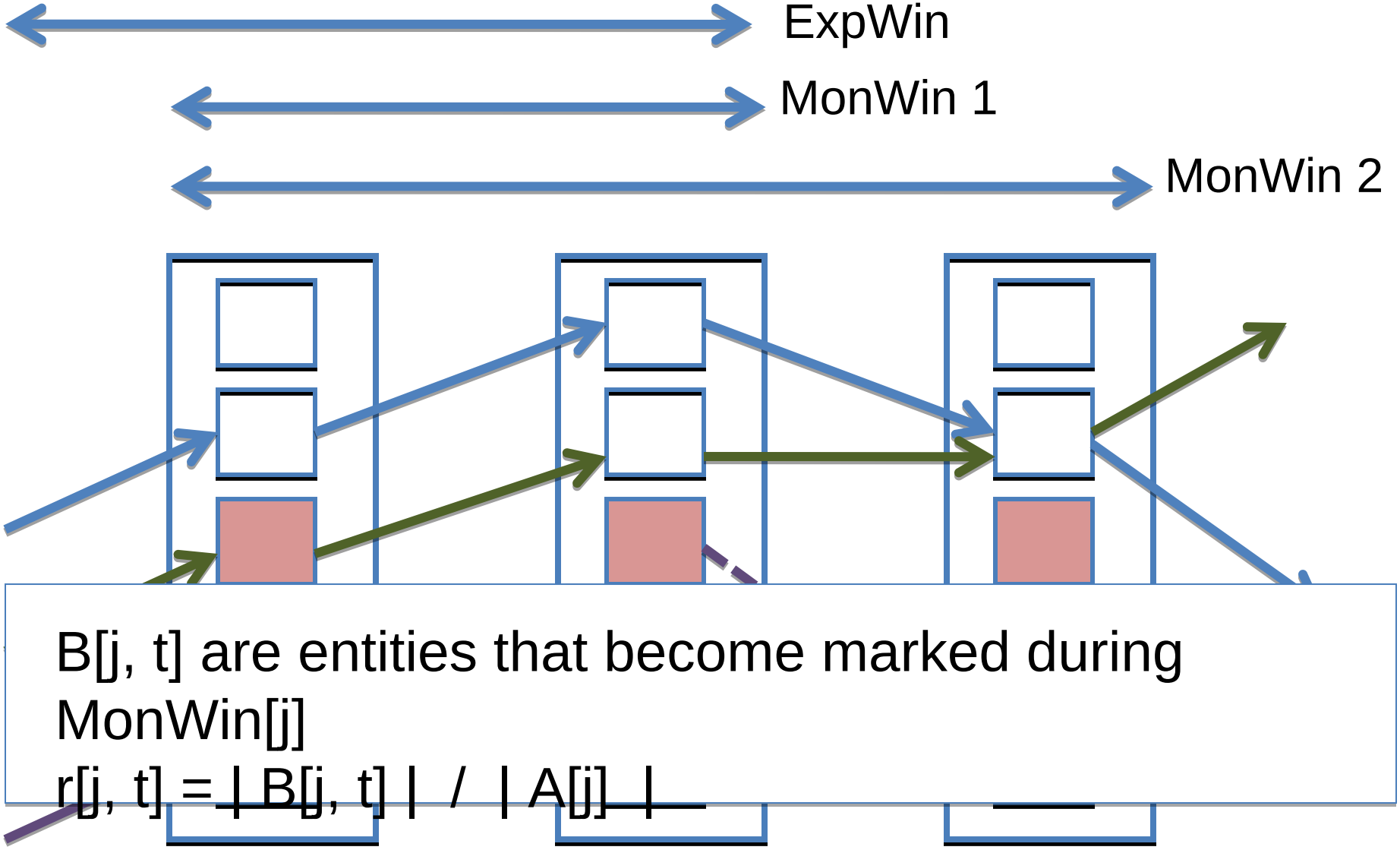
- Some sites are marked (percent of mark is a parameter and type of sites marked is a draw from a distribution)
- *Some* entities become marked after visiting a marked site (this is a draw from a distribution)
- There is a delay between the visit and the when the entity becomes marked (this is a draw from a distribution)
- There is a background process that marks some entities independent of visit (this adds noise to problem)


Exposure Window
Monitor Window
d_{k-2}
d_{k-1}
d_k
time

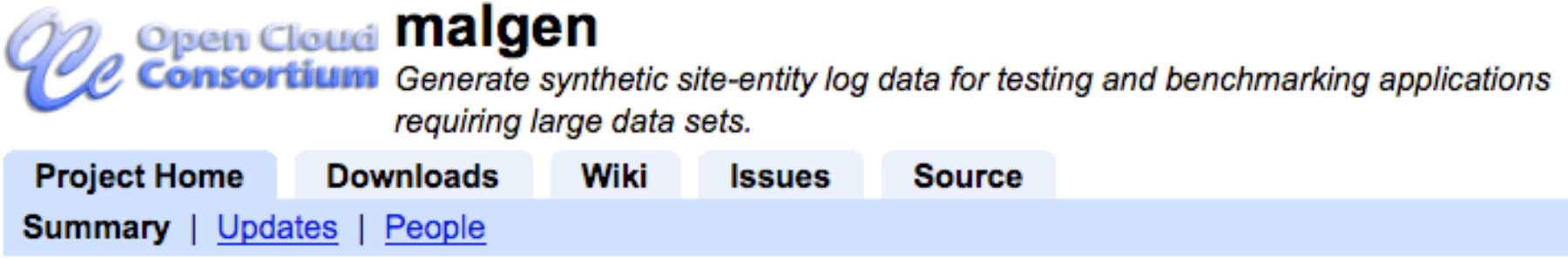
Notation

- Fix a site s[j]
- Let A[j] be entities that transact during ExpWin and if entity is marked, then visit occurs before mark
- Let B[j] be all entities in A[j] that become marked sometime during the MonWin
- Subsequent proportion of marks is

$$r[j] = |B[j]| \ / \ |A[j]|$$


ExpWin
MonWin 1
MonWin 2
B[j, t] are entities that become marked during MonWin[j]
r[j, t] = | B[j, t] | / | A[j] |
d_{k-2}
d_{k-1}
d_k
time

Part 3. MalStone Benchmarks



code.google.com/p/malgen/

MalGen and MalStone implementations are open source

MalStone Benchmark

- Benchmark developed by Open Cloud Consortium for clouds supporting data intensive computing.
- Code to generate synthetic data required is available from code.google.com/p/malgen
- Stylized analytic computation that is easy to implement in MapReduce and its generalizations.

MalStone A & B

Benchmark	Statistic	# records	Size
MalStone A-10	r[j]	10 billion	1 TB
MalStone A-100	r[j]	100 billion	10 TB
MalStone A-1000	r[j]	1 trillion	100 TB
MalStone B-10	r[j, t]	10 billion	1 TB
MalStone B-100	r[j, t]	100 billion	10 TB
MalStone B-1000	r[j, t]	1 trillion	100 TB

- MalStone B running on 10 Billion 100 byte records
- Hadoop version 0.18.3
- 20 nodes in the Open Cloud Testbed
- MapReduce required 799 minutes
- Hadoop streams required 142 minutes

68 minutes running MalStone B Benchmark

- 4 AMD 8435 processors with 24 cores running at 2.6 GHZ
- 64 Gigabytes of Memory
- RAID file system with of 5 SATA drives

Source: cs.pervasive.com/blogs/datarush/archive/2010/03/05/cluster-on-a-chip.asp
March 5, 2010

Design Trade Offs for Sector

	MalStone B
Sector/Sphere v1.20	44 min
# Nodes	20 nodes
# Records	10 Billion
Size of Dataset	1 TB

Tests done on Open Cloud Testbed.

Impact of Locality

Component	Impact
Without in-memory objects	117%
Without bucket localization	146%
With bucket combiner	106%
Without bucket fault tolerance	110%

Thank You!

For more information,

rgrossman.com