D4M 2.0 Schema: A General Purpose High Performance Schema for the Accumulo Database

Jeremy Kepner, Christian Anderson, William Arcand, David Bestor, Bill Bergeron, Chansup Byun, Matthew Hubbell, Peter Michaleas, Julie Mullen, David O'Gwynn, Andrew Prout, Albert Reuther, Antonio Rosa, Charles Yee

IEEE HPEC 2013

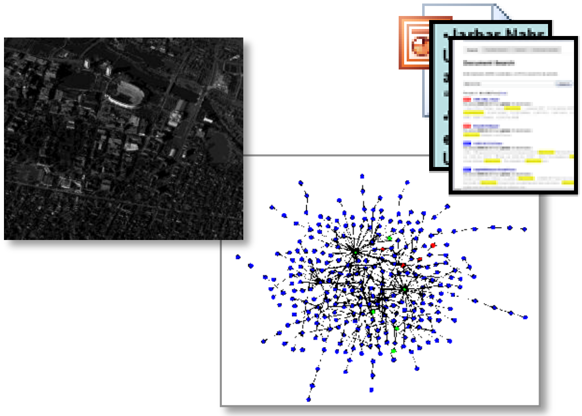
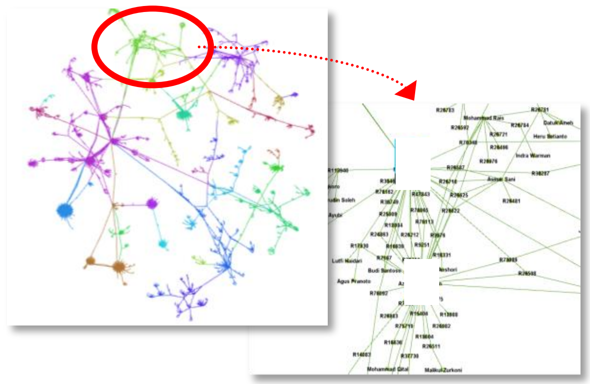
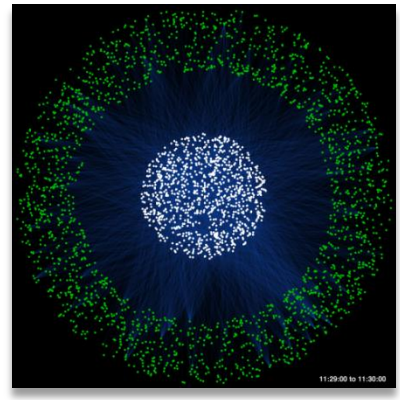
LINCOLN LABORATORY
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

This work is sponsored by the Department of the Air Force under Air Force Contract #FA8721-05-C-0002. Opinions, interpretations, recommendations and conclusions are those of the authors and are not necessarily endorsed by the United States Government.

Outline

- Introduction
- D4M
- Schema
- Twitter
- Summary

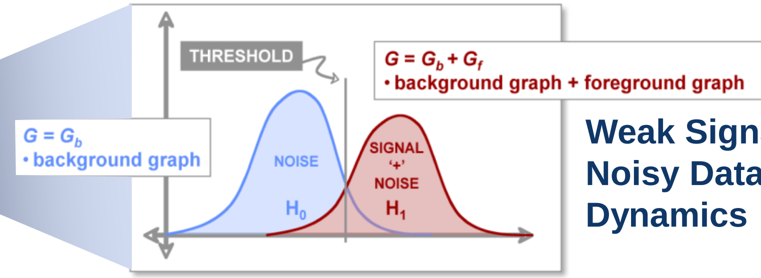
Example Big Data Applications

ISR	Social	Cyber
		
• Graphs represent entities and relationships detected through multi-INT sources • 1,000s – 1,000,000s tracks and locations • GOAL: Identify anomalous patterns of life	• Graphs represent relationships between individuals or documents • 10,000s – 10,000,000s individual and interactions • GOAL: Identify hidden social networks	• Graphs represent communication patterns of computers on a network • 1,000,000s – 1,000,000,000s network events • GOAL: Detect cyber attacks or malicious software

Cross-Mission Challenge: detection of subtle patterns in massive multi-source noisy datasets

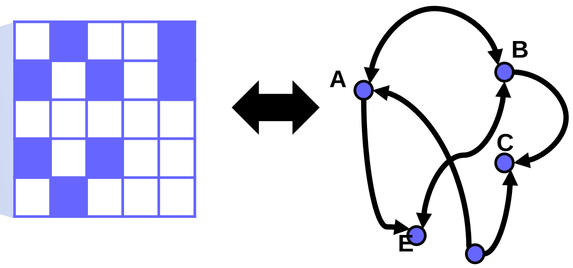
LLSuperCloud Software Stack: Big Data + Big Compute

Novel Analytics for: Text, Cyber, Bio



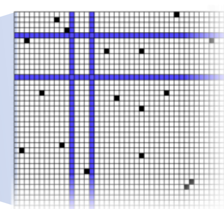
Weak Signatures, Noisy Data, Dynamics

High Level Composable API: D4M ("Databases for Matlab")



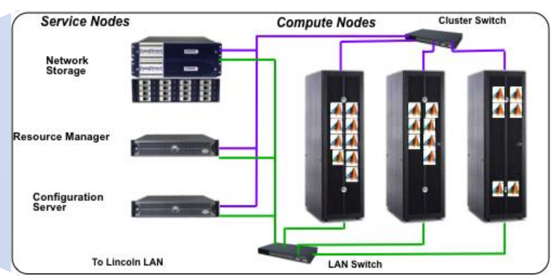
Array Algebra

Distributed Database: Accumulo (triple store)



Distributed Database/ Distributed File System

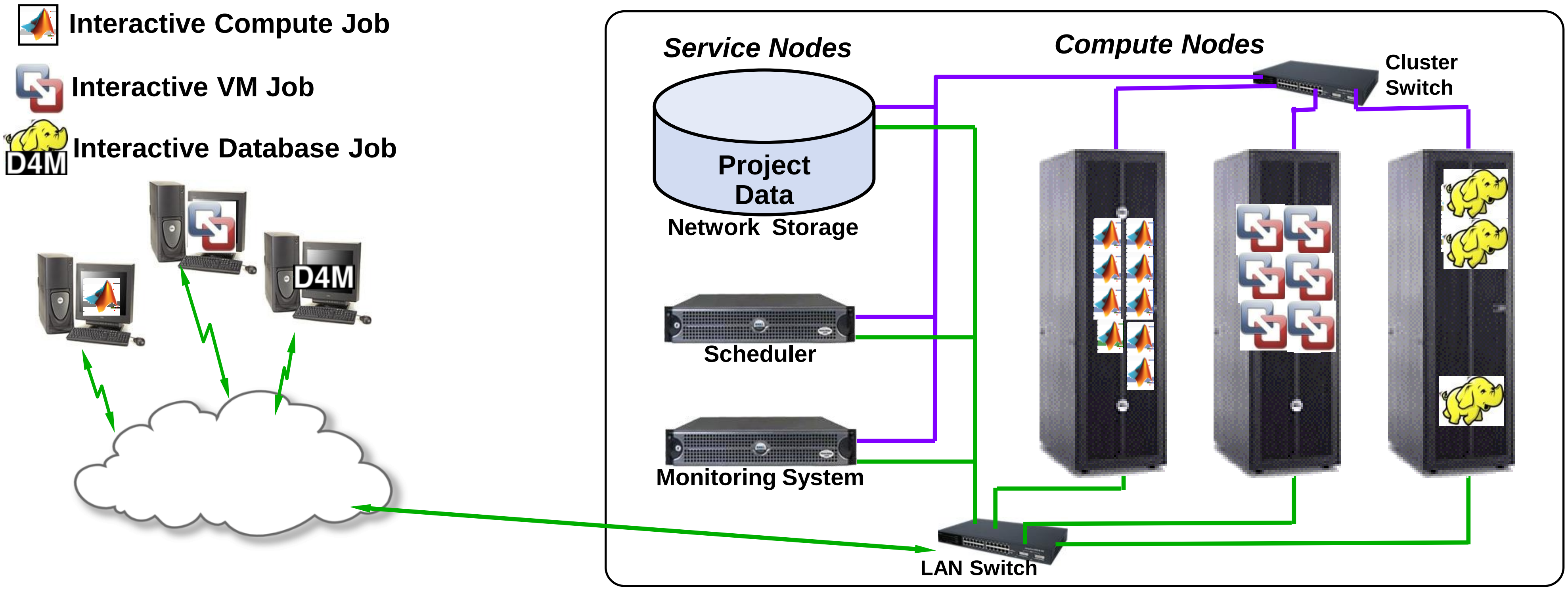
High Performance Computing: LLGrid + Hadoop



Interactive Super-computing

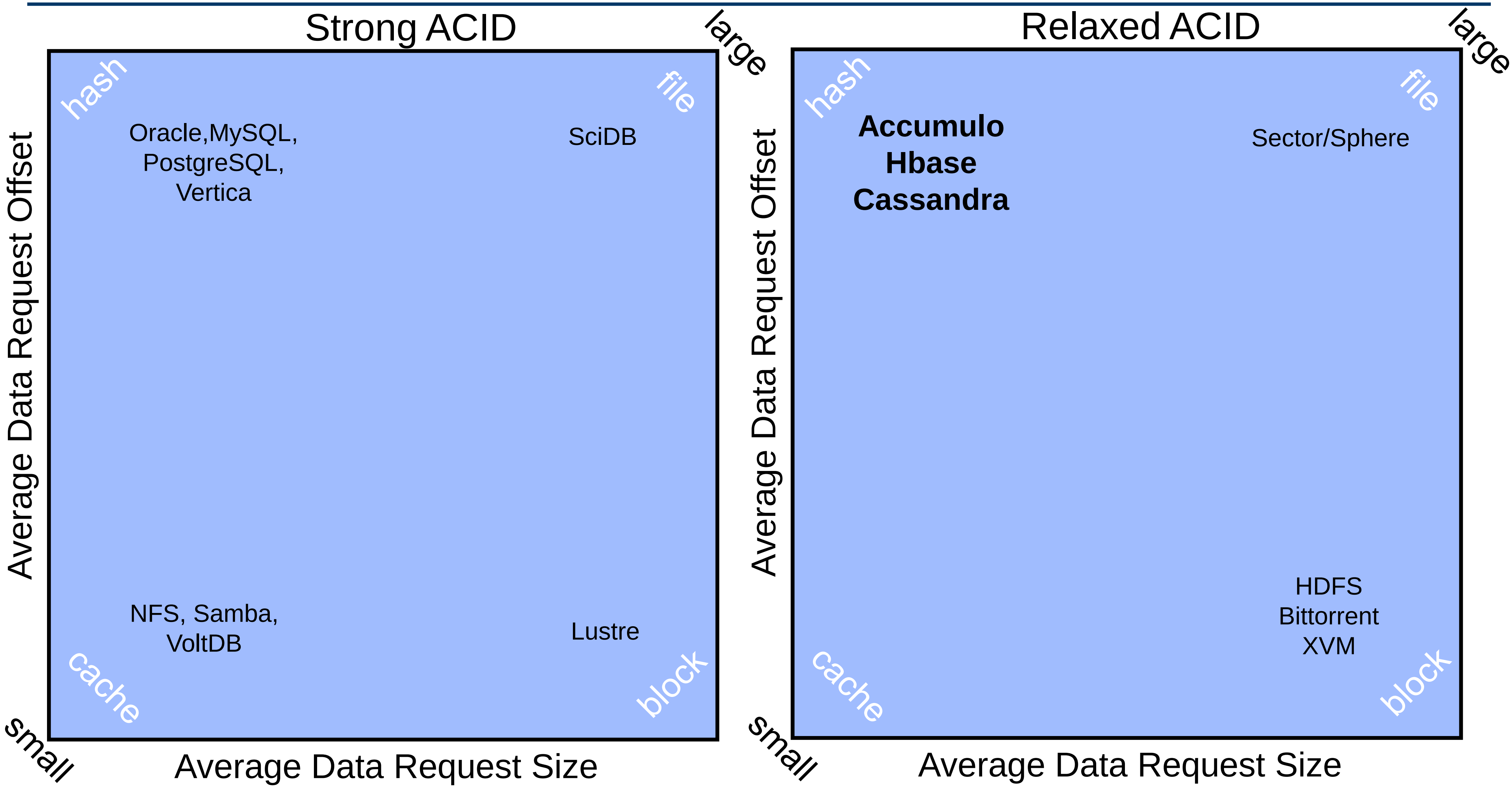
Combining Big Compute and Big Data enables entirely new domains

LLSuperCloud Test Bed



- LLSuperCloud allows traditional supercomputing, VMs and Hadoop/Accumulo to dynamically share the same hardware; allows users to:
- Dynamically stand up and test heterogeneous clouds
- Integrate different clouds for best mission solution
- Determine which clouds are best for which mission

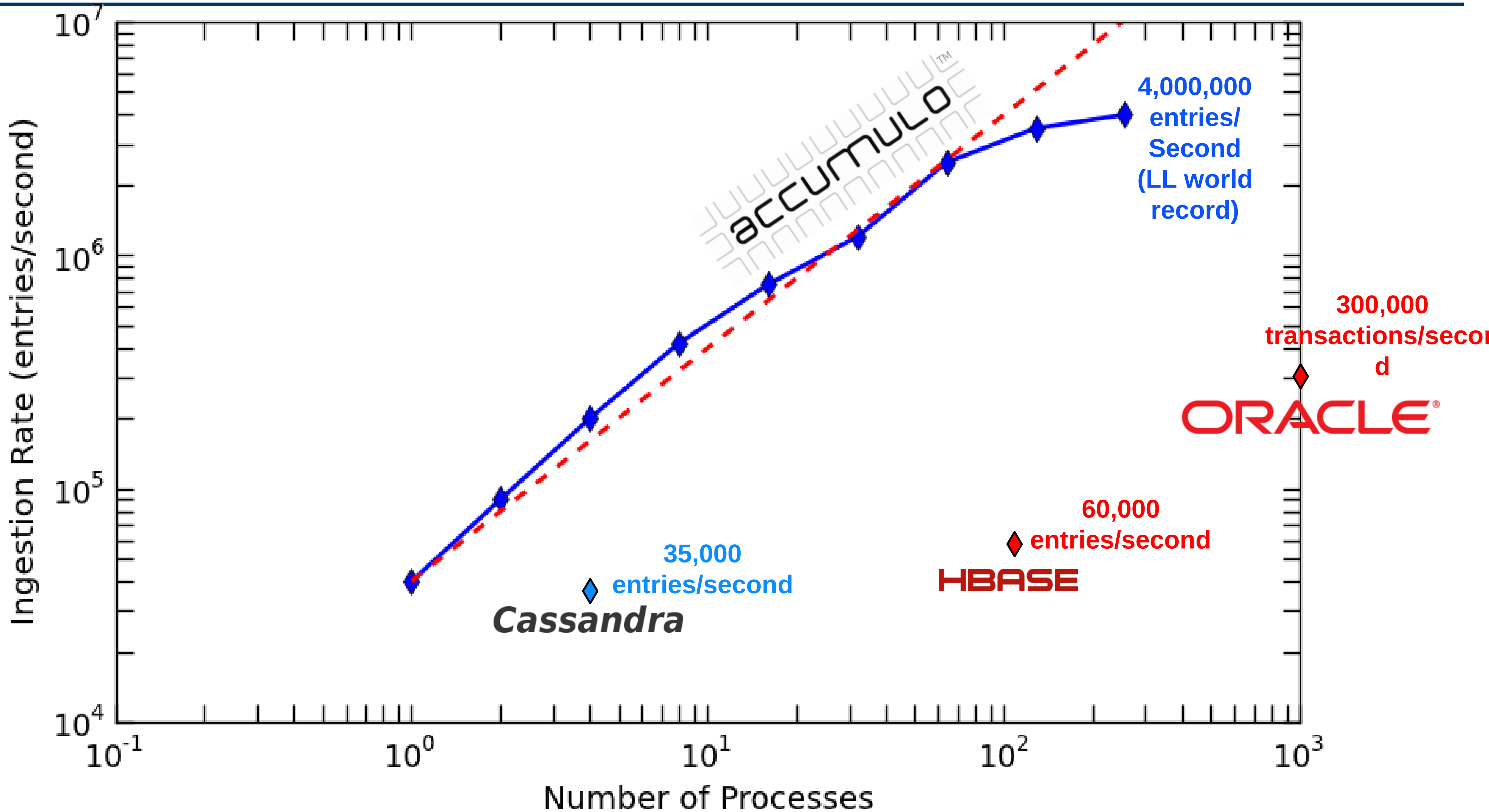
Data Storage Landscape



- Leading areas of innovation are in dense structured databases and sparse unstructured databases

ACID = Atomicity, Consistency, Isolation, Durability

Accumulo "Big Table" Database



- **Accumulo is the fastest open source database in the world**
- **Widely used for gov't applications**

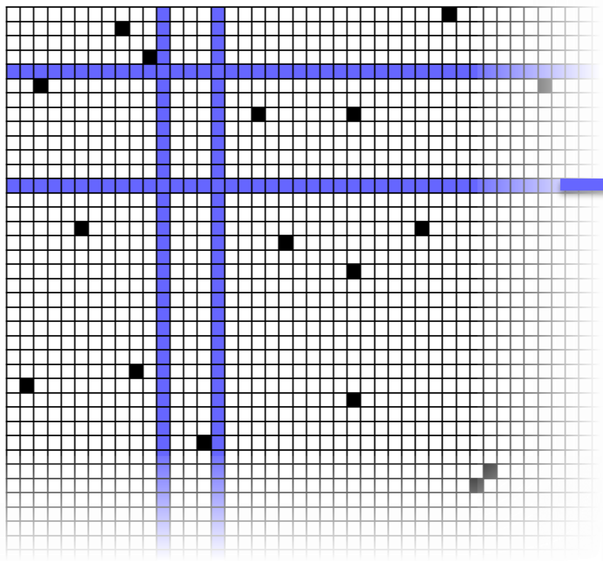
Outline

- Introduction
- D4M
- Schema
- Twitter
- Summary

High Level Language: D4M

http://www.mit.edu/~kepner/D4M

Accumulo Distributed Database

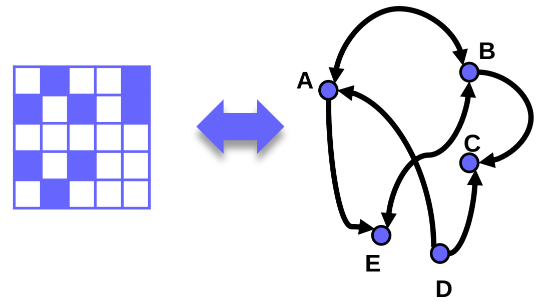


D4M
Dynamic
Distributed
Dimensional
Data Model

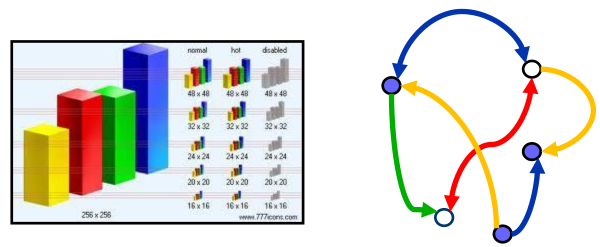
Query:
Alice
Bob
Cathy
David
Earl

Associative Arrays

Numerical Computing Environment



A D4M query returns a sparse matrix or a graph…



…for statistical signal processing or graph analysis in MATLAB

D4M binds associative arrays to databases, enabling rapid prototyping of data-intensive cloud analytics and visualization

D4M Key Concept: Associative Arrays Unify Four Abstractions

- **Extends associative arrays to 2D and mixed data types**

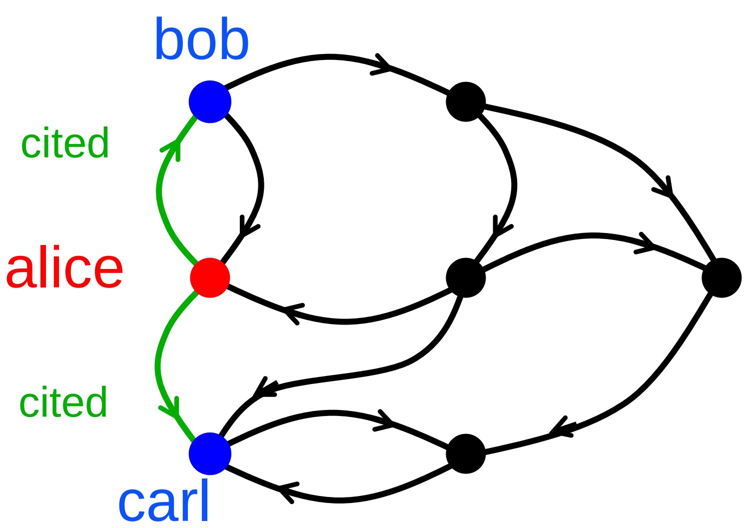
A('alice ','bob ') = 'cited '

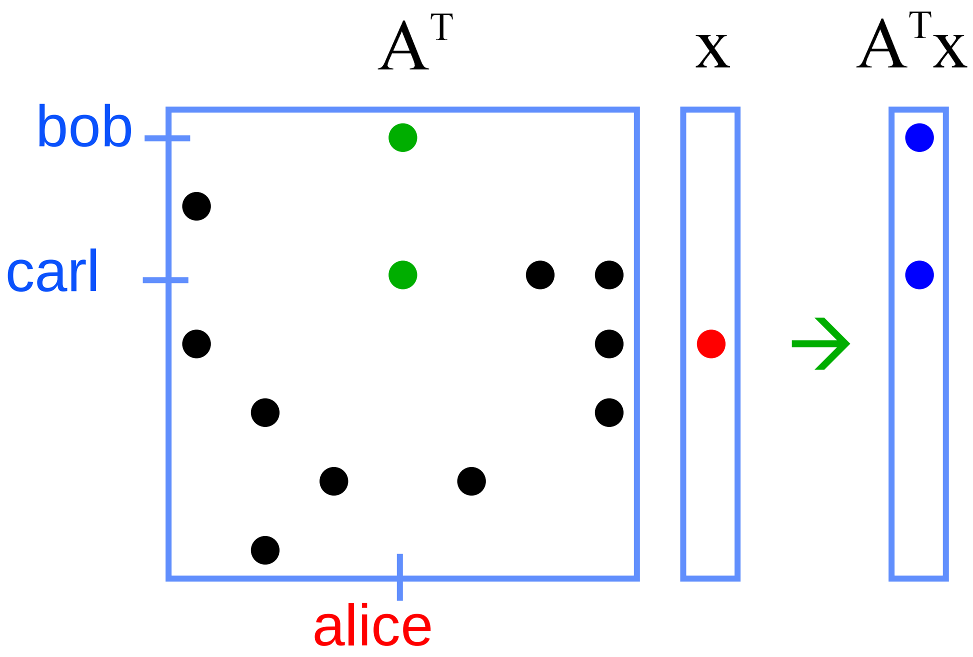
or A('alice ','bob ') = 47.0

- **Key innovation: 2D is 1-to-1 with triple store**

('alice ','bob ','cited ')

or ('alice ','bob ',47.0)





Composable Associative Arrays

- **Key innovation: mathematical closure**
  - **All associative array operations return associative arrays**
- **Enables composable mathematical operations**

  A + B   A - B   A & B   A|B   A*B

- **Enables composable query operations via array indexing**

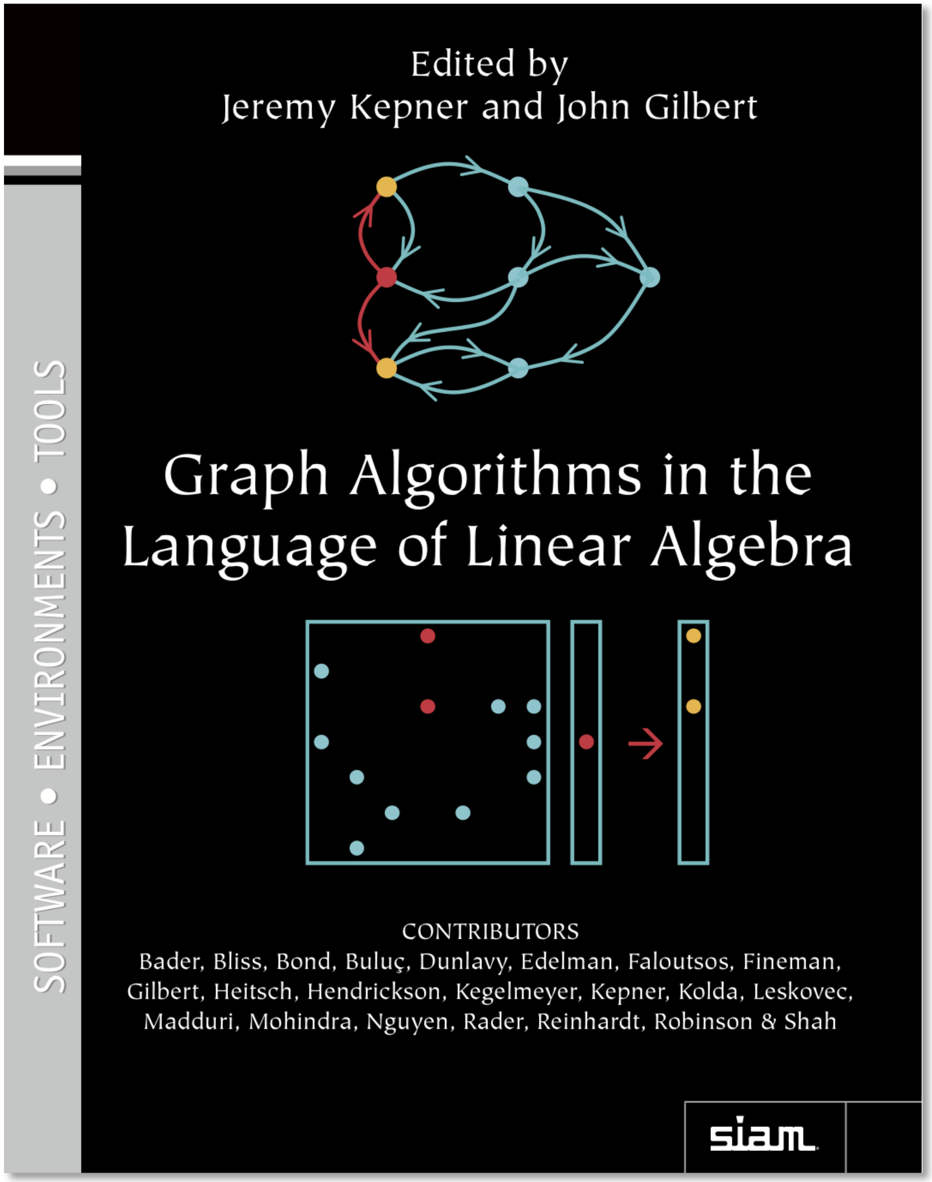
  A('alice bob ',:)   A('alice ',:)   A('al* ',:)

  A('alice : bob ',:)   A(1:2,:)   A == 47.0

- **Simple to implement in a library (~2000 lines) in programming environments with: 1st class support of 2D arrays, operator overloading, sparse linear algebra**

- **Complex queries with ~50x less effort than Java/SQL**
- **Naturally leads to high performance parallel implementation**

Reference & Database Workshop



Database Discovery Workshop

3 day hands-on workshop on:

Systems

- Parse, ingest, query, analysis & display

Usage

- Files vs. database, chunking & query planning

Detection theory

- Clutter, background, detection & tracking

Technology selection

- Knowing what to use is as important as knowing how to use it

Using state-of-the-art technologies:

D4M

Python

R

accumulo

Hadoop

SciDB

Outline

- **Introduction**
- **D4M**
- **Schema**
- **Twitter**
- **Summary**

Generic D4M Triple Store Exploded Schema

Accumulo Table: Ttranspose

	01-01-2001	02-01-2001	03-01-2001
Col1|a	1		
Col1|b		1	
Col2|b		1	
Col2|c			1
Col3|a	1		
Col3|c			1

Input Data

Time	Col1	Col2	Col3
2001-01-01	a		a
2001-01-02	b	b	
2001-01-03		c	c

	Col1|a	Col1|b	Col2|b	Col2|c	Col3|a	Col3|c
01-01-2001	1				1	
02-01-2001		1	1			
03-01-2001				1		1

Accumulo Table: T

- Tabular data expanded to create many type/value columns
- Transpose pairs allows quick look up of either row or column
- Flip time for parallel performance

Tables: SQL vs D4M+Accumulo

SQL Dense Table: T

log_id	src_ip	srv_ip
001	128.0.0.1	208.29.69.138
002	192.168.1.2	157.166.255.18
003	128.0.0.1	74.125.224.72 208.29.69.138

Use as row indices

Create columns for each unique type/value pair

	src_ip|128.0.0.1	src_ip|192.168.1.2	srv_ip|157.166.255.18	srv_ip|208.29.69.138	srv_ip|74.125.224.72
log_id|100	1			1	
log_id|200		1	1		
log_id|300	1			1	1

Accumulo D4M schema (aka NuWave) Tables: E and E^T

- **Both dense and sparse tables stored the same data**
- **Accumulo D4M schema uses table pairs to index every unique string for fast access to both rows and columns (ideal for graph analysis)**

Queries: SQL vs D4M

Query Operation	SQL	D4M
Select all	SELECT * FROM T	E(:,:)
Select column	SELECT src_ip FROM T	E(:,StartsWith('src_ip| '))
Select sub-column	SELECT src_ip FROM T WHERE src_ip=128.0.0.1	E(:,'src_ip|128.0.0.1 ')
Select sub-matrix	SELECT * FROM T WHERE src_ip=128.0.0.1	E(Row(E(:,'src_ip|128.0.0.1 ')),:)

- Queries are easy to represent in both SQL and D4M
- Pedigree (i.e., the source row ID) is always preserved since no information is lost

Analytics: SQL vs D4M

Query Operation	SQL	D4M
Histogram	SELECT COUNT(src_ip) FROM T GROUP BY src_ip	sum(E(:,StartsWith('src_ip| ')),2)
Graph traversal	SELECT * FROM T WHERE src_ip=128.0.0.1 ...	v0 = 'src_ip|128.0.0.1 ' v1 = Col(E(Row(E(:,v0)),:)) v2 = Col(E(Row(E(:,v1)),:))
Graph construction	… many lines …	A = E(:,StartsWith('src_ip| ')). ' * E(:,StartsWith('srv_ip| '))
Graph eigenvalues	… many lines …	eigs(Adj(A))

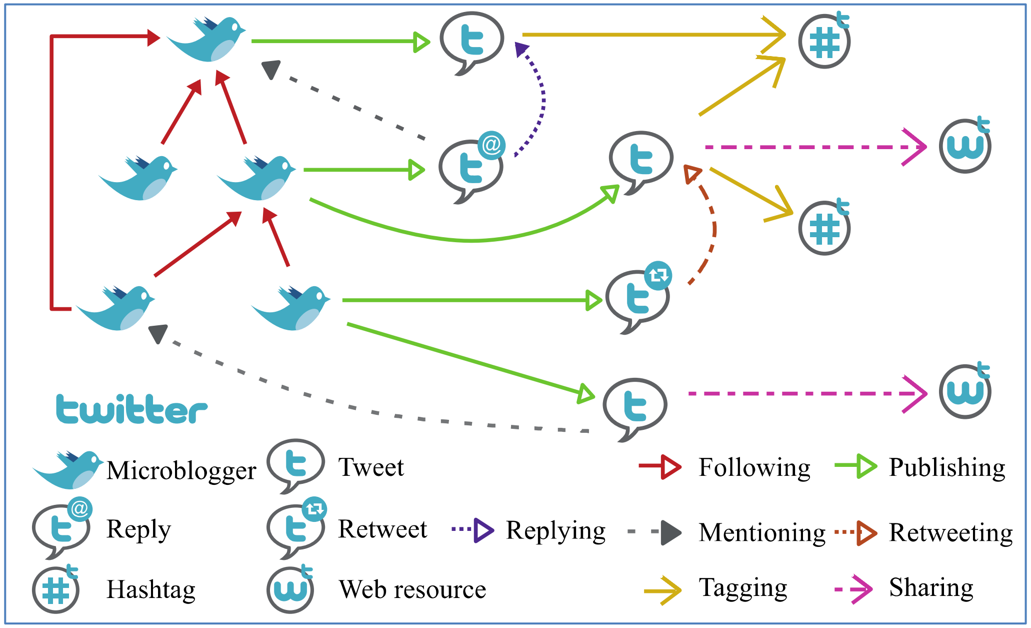
- **Analytics are easy to represent in D4M**
- **Pedigree (i.e., the source row ID) is usually lost since analytics are a projection of the data and some information is lost**

Outline

- **Introduction**
- **D4M**
- **Schema**
- **Twitter**
- **Summary**

Tweets2011 Corpus
http://trec.nist.gov/data/tweets/

- **Assembled for Text REtrieval Conference (TREC 2011)***
  - **Designed to be a reusable, representative sample of the twittersphere**
  - **Many languages**
- **16,141,812 million tweets sampled during 2011-01-23 to 2011-02-08 (16,951 from before)**
  - **11,595,844 undeleted tweets at time of scrape (2012-02-14)**
  - **161,735,518 distinct data entries**
  - **5,356,842 unique users**
  - **3,513,897 unique handles (@)**
  - **519,617 unique hashtags (#)**



Ben Jabur et al, ACM SAC 2012

*McCreadie et al, "On building a reusable Twitter corpus," ACM SIGIR 2012

Twitter Input Data

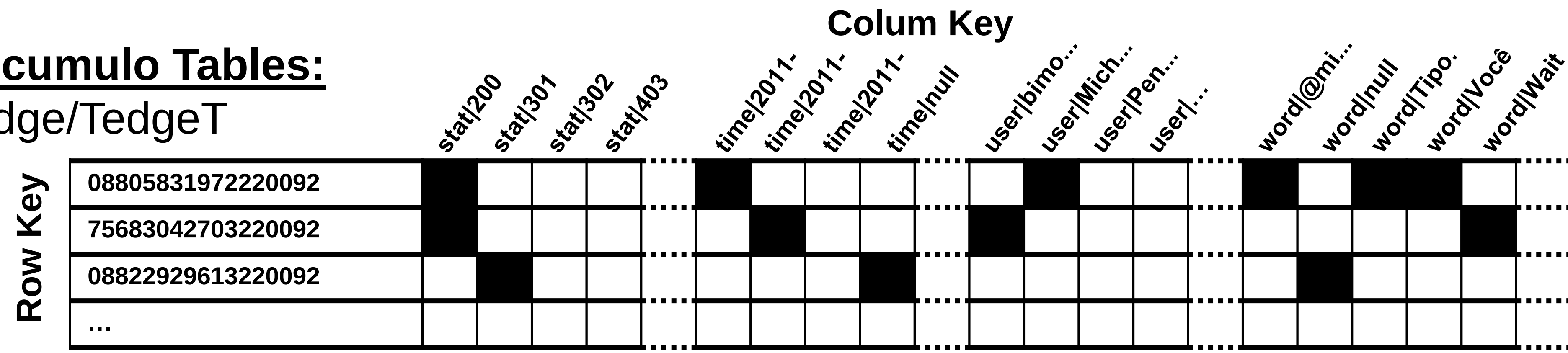
TweetID	User	Status	Time	Text
29002227913850880	Michislipstick	200	Sun Jan 23 02:27:24 +0000 2011	@mi_pegadejeito Tipo. Você ...
29002228131954688	__rosana__	200	Sun Jan 23 02:27:24 +0000 2011	para la semana q termino ...
29002228165509120	doasabo	200	Sun Jan 23 02:27:24 +0000 2011	お腹すいたずえ
29002228937265152	agusscastillo	200	Sun Jan 23 02:27:24 +0000 2011	A nadie le va a importar ...
29002229444771841	nob_sin	200	Sun Jan 23 02:27:24 +0000 2011	さて。札幌に帰るか。
29002230724038657	bimosephano	200	Sun Jan 23 02:27:25 +0000 2011	Wait :)
29002231177019392	_Word_Play	200	Sun Jan 23 02:27:25 +0000 2011	Shawty is 53% and he pick ...
29002231202193408	missogeeeeb	200	Sun Jan 23 02:27:25 +0000 2011	Lazy sunday ╰(◣﹏◢)╯ oooo !
29002231692922880	PennyCheco06	301	null	null
…	…	…	…	…

- **Mixture of structured (TweetID, User, Status, Time) and unstructured (Text)**
- **Fits well into standard D4M Exploded Schema**

Tweets2011 D4M Schema

Accumulo Tables:

Tedge/TedgeT



TedgeDeg^t

Row Key

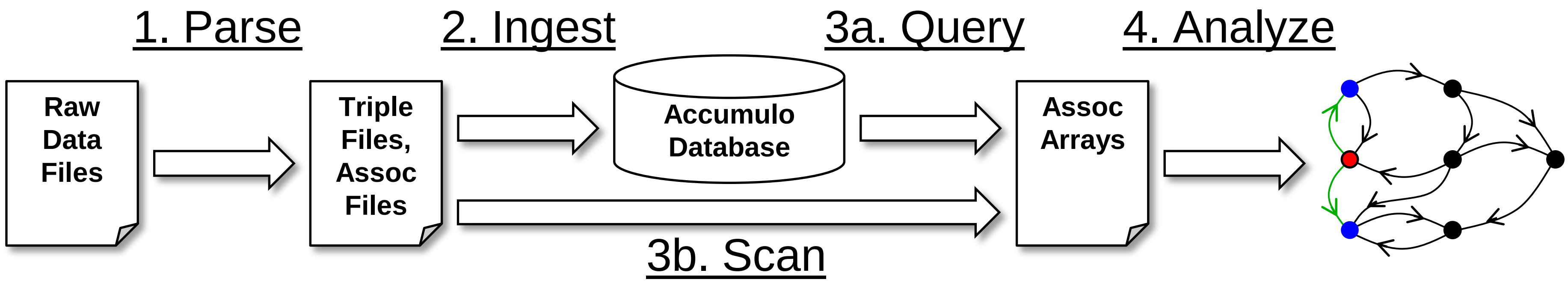
	stat|200	stat|301	stat|302	stat|403	...	time|2011-	time|2011-	time|2011-	time|null	...	user|bimo...	user|Mich...	user|Pen...	user|...	...	word|@mi...	word|null	word|Tipo.	word|Você	word|Wait	...
Degree	10864273	2861507	836327	825822		6	7	7	4545968		6	7	3			3	4546039	16	10223	1624	

TedgeTxt

Row Key	text
08805831972220092	@mi_pegadejeito Tipo. Você fazer uma plaquinha pra mim, com o nome do FC pra você tirar uma foto, pode fazer isso?
75683042703220092	Wait :)
08822929613220092	null
...	

- **Standard exploded schema indexes every unique string in data set**
- TedgeDeg **accumulate sums of every unique string**
- TedgeTxt **stores original text for viewing purposes**

D4M Pipeline



Tweets2011 (single node database)

- **Raw Data: 1.8GB (1621 Files; 100K tweets/file)**

1. Parse: 20 minutes (N_p = 16)

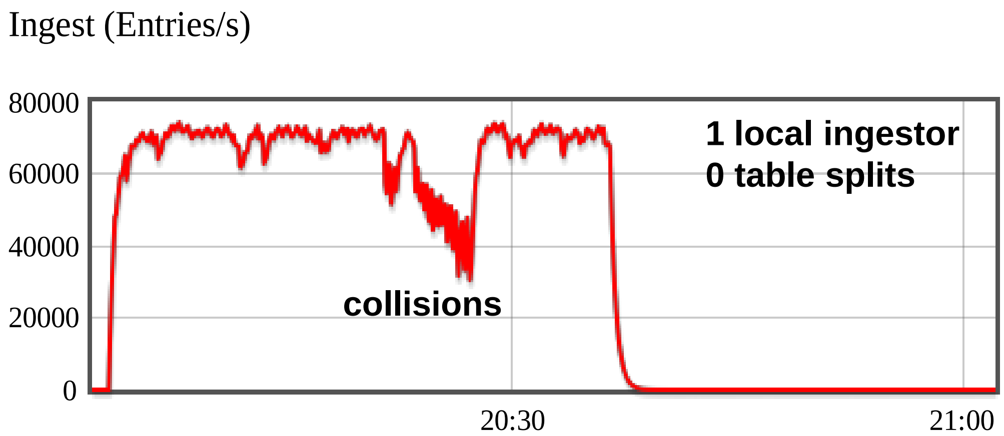
- **Triple Files & Assoc Files: 6.7GB (7x1621 files)**

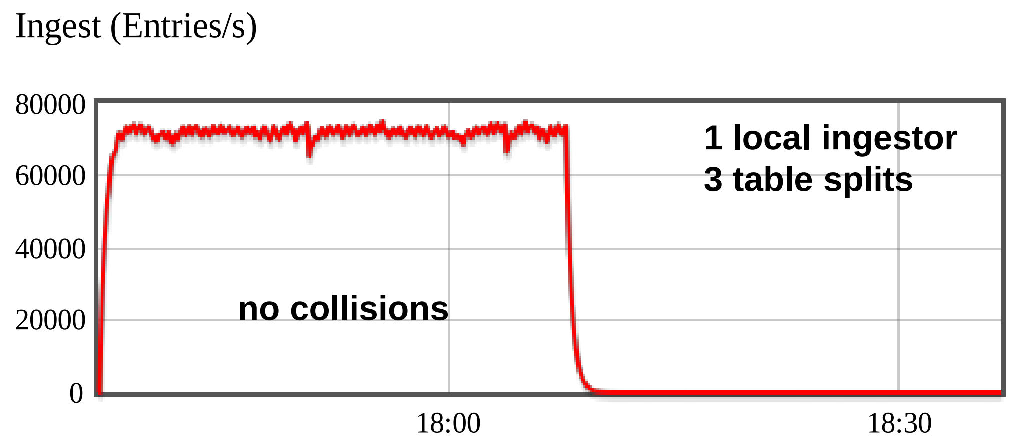
2. Ingest: 40 minutes (N_p = 3); 20 entries/tweet, 200K entries/second

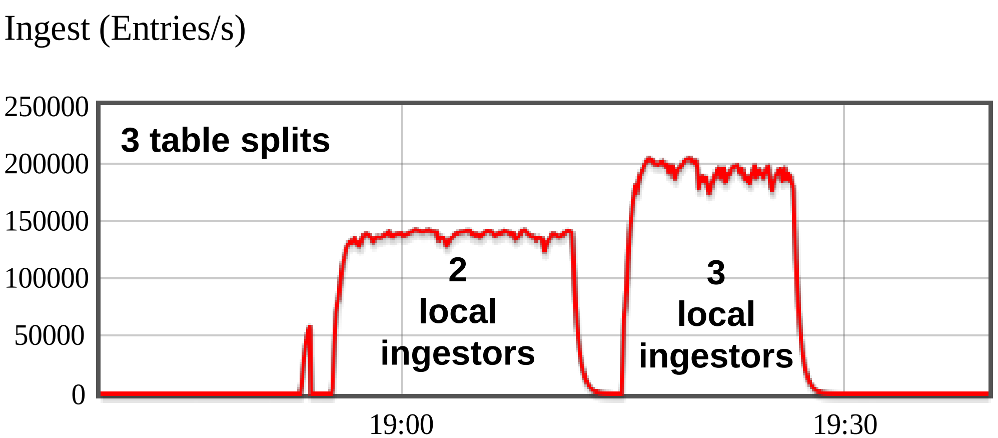
- **Accumulo Database: 14GB in 350M entries**

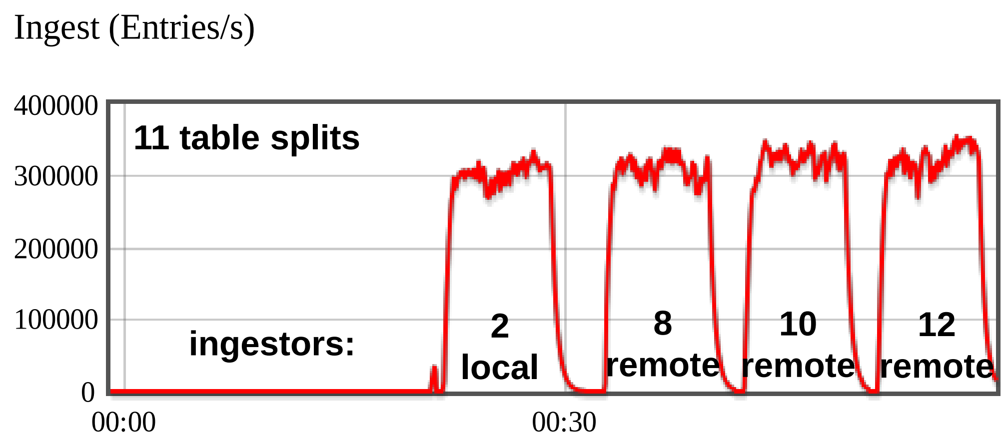
- **Raw data to fully indexed database < 1 hour**
- **Database good for queries, assoc files good for complete scans**

Accumulo Ingest Performance
Single Node Graph500 Benchmark Data









- Single node high performance ingest requires pre-splitting tables
- Can achieve high parallel ingest on single node with proper splitting

Outline

- Introduction
- Capabilities
- Data
- Analytics
- Summary

Word Tallies

- **D4M Code**

```
Tdeg = DB('TedgeDeg');
str2num(Tdeg(StartsWith('word|: '),:)) > 20000
```

- **D4M Result (1.2 seconds, N_p = 1)**

word|:	77946
word|:(	80637
word|:)	263222
word|:D	151191
word|:P	34340
word|:p	56696

- **Sum table** TedgeDeg **allows tallies to be seen instantly**

Users Who ReTweet the Most
Problem Size

- **D4M Code to check size of status codes**

```
Tdeg = DB('TedgeDeg');
Tdeg(StartsWith('stat| '),:))
```

- **D4M Results (0.02 seconds, N_p = 1)**

stat|200	10864273	**OK**
stat|301	2861507	**Moved permanently**
stat|302	836327	**ReTweet**
stat|403	825822	**Protected**
stat|404	753882	**Deleted tweet**
stat|408	1	**Request timeout**

- **Sum table** TedgeDeg **indicates 836K retweets (~5% of total)**
- **Small enough to hold all TweeetIDs in memory**
- **On boundary between database query and complete file scan**

Users Who ReTweet the Most
Parallel Database Query

- **D4M Parallel Database Code**

```
T = DB('Tedge','TedgeT');        Ar = T(:,'stat|302 ');
my = global_ind(zeros(size(Ar,2),1,map([Np 1],{},0:Np-1)));
An = Assoc('','','');            N = 10000;
for i=my(1):N:my(end)
  Ai = dblLogi(T(Row(Ar(i:min(i+N,my(end)),:)),:));
  An = An + sum(Ai(:,StartsWith('user|,')),1);
end
Asum = gagg(Asum > 2);
```

- **D4M Result (130 seconds, N_p = 8)**

user|Puque007 103, user|[redacted]Say 113,
user|carp_fans 115, user|habu_bot 111,
user|kakusan_RT 135, user|umaitofu 116

- **Each processor queries all the retweet TweetIDs and picks a subset**
- **Processors each sum all users in their tweets and then aggregate**

Users Who ReTweet the Most
Parallel File Scan

- **D4M Parallel File Scan Code**

```
Nfile = size(fileList);
my = global_ind(zeros(Nfile,1,map([Np 1],{},0:Np-1)));
An = Assoc('','','');
for i=my
  load(fileList{i});
  An = An + sum(A(Row(A(:,'stat|302,')),StartsWith('user|,')),1);
end
An = gagg(An > 2);
```

- **D4M Result (150 seconds, N_p = 16)**

user|Puque007 100, user|[redacted]Say 113,
user|carp_fans 114, user|habu_bot 109,
user|kakusan_RT 135, user|umaitofu 114

- **Each processor picks a subset of files and scans them for retweets**
- **Processors each sum all users in their tweets and then aggregate**

Summary

- **Big data is found across a wide range of areas**
  - **Document analysis**
  - **Computer network analysis**
  - **DNA Sequencing**
- **Non-traditional, relaxed consistency, triple store databases are the backbone of many web companies**
- **D4M Schema provides a general, high performance approach for harnessing the power of these databases**