

---

# **Dynamic Distributed Dimensional Data Model (D4M) Database and Computation System**

**Jeremy Kepner, William Arcand, William Bergeron,  
*Nadya Bliss*, Robert Bond, Chansup Byun, Gary Condon,  
Kenneth Gregson, Matthew Hubbell, Jonathan Kurz,  
Andrew McCabe, Peter Michaleas, Andrew Prout,  
Albert Reuther, Antonio Rosa, Charles Yee**

**ICASSP (March 2012)**





# Outline

---

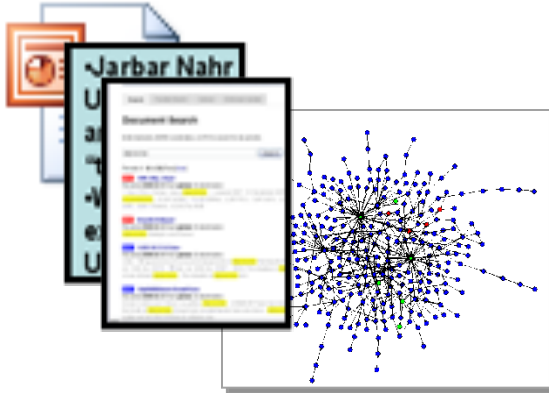


- **Introduction**
  - **Big Data**
  - **Challenge**
  - **D4M**
- **Technologies**
- **Results**
- **Summary**

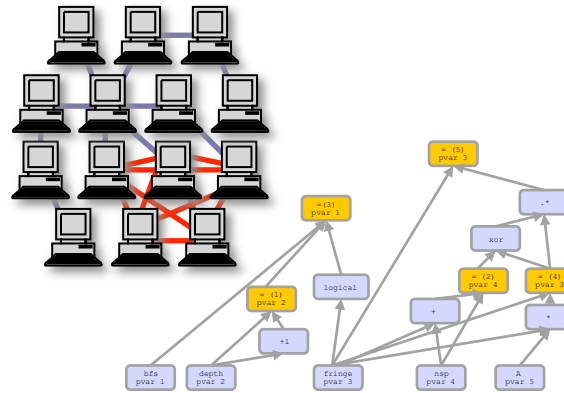


# Big Data Application Areas

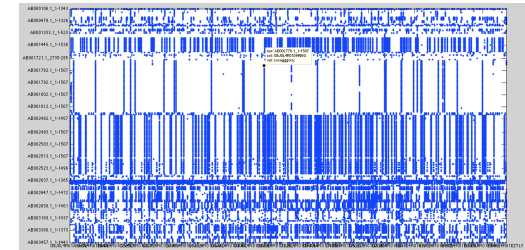
## DOCUMENTS



## COMPUTER NETWORKS



## DNA SEQUENCING



- Billions of documents
- Entities detected from multi-INT sources
- Analyze relationships between entities
- Millions of computers
- Analyze communication patterns
- Analyze program flow
- Find behaviors consistent with attack
- Thousands of species
- Consider interactions between species
- Identify and correlate

- Analysis significantly effected by data access time
- N large enough that  $O(N^2)$  algorithms are usually infeasible
- Cannot be performed on a single computer



# Algorithm Developer Tool Gap

## Scalable Databases (triple stores)



## Legacy Tools



## Distributed Mobile Display Devices



**Algorithm Developer Needs**  
Graphs + Strings + Numbers  
Composable Mathematics

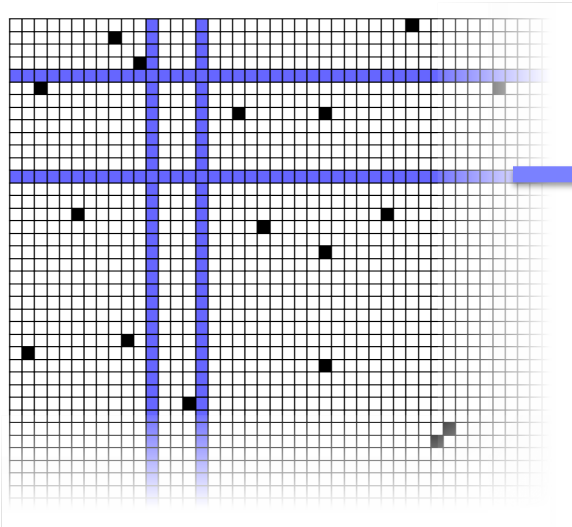
**Tool Requirements**  
Multi-dimensional arrays  
Operator overloading  
Sparse linear algebra

- Scalable databases provide low latency access to vast stores of data
- Mobile devices allows data to be viewed anywhere
- Legacy tools not intended for big data algorithm development



# D4M: “Databases For Matlab”

## Triple Store Distributed Database




Triple store are high performance distributed databases for heterogeneous data

## 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 graph from Accumulo ...

...for statistical signal processing or graph analysis in MATLAB

• D4M binds Associative Arrays to Triple Store, enabling rapid prototyping of data-intensive cloud analytics and visualization



# Outline

---

- Introduction
- • Technologies
  - Comparison
  - Associative arrays
  - Exploded schema
- Results
- Summary



# Technology Comparison

Feature	Perl	SQL	HBase	Linda	Comm BLAS	Tensor Toolbox	UPC	VSIP++	pMatlab	D4M
Associative Array										
1D	X	X	X	X						X
2D	X	X	X	X						X
String key/value	X	X	X	X						X
Numeric key/value		X	X	X						X
Composable query										X
Composable compute					X	X	X	X	X	X
Tuple Store			X							X
Parallel Client			X	X	X		X	X	X	X
Distributed array				X	X		X	X	X	X

- **D4M features can be found across a wide range of technologies**
- **D4M uniquely combines these features into a composable language for algorithm development**



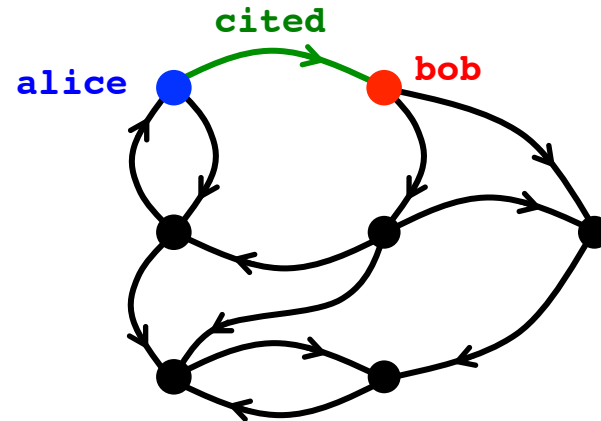
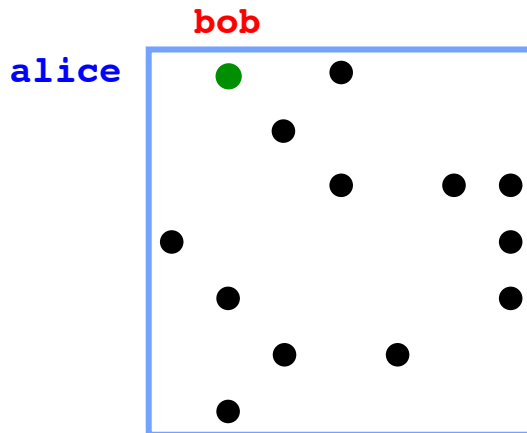
# Multi-Dimensional Associative Arrays

- 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)`



- Associative arrays unify four viewpoints into one concept





# Universal “Exploded” Schema

Time	src_ip	domain	dest_ip
2001-01-01	a		a
2001-01-02	b	b	
2001-01-03		c	c

Input Data

	src_ip/ a	src_ip/ b	domain /b	domain /c	dest_ip /a	dest_ip /c
2001-01-01	1				1	
2001-01-02		1	1			
2001-01-03				1		1

Triple Store Table: T

	2001-01-01	2001-01-02	2001-01-03
src_ip/a	1		
src_ip/b		1	
domain/b		1	
domain/c			1
dest_ip/a	1		
dest_ip/c			1

Triple Store Table: Ttranspose

## Key Innovations

- Handles all data into a single table representation
- Transpose pairs allows quick look up of either row or column



# 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**



# Associative Array Algebra

- Keys and values are from the infinite strict totally ordered set  $\mathbb{S}$
- Associative array  $A(k) : \mathbb{S}^d \rightarrow \mathbb{S}$ ,  $k=(k^1, \dots, k^d)$ , is a partial function from  $d$  keys (typically 2) to 1 value, where

$$A(k_i) = v_i \text{ and } \emptyset \text{ otherwise}$$

- Binary operations on associative arrays  $A_3 = A_1 \oplus A_2$ , where  $\oplus = \cup_{f()}$  or  $\cap_{f()}$ , have the properties

- If  $A_1(k_i) = v_1$  and  $A_2(k_i) = v_2$ , then  $A_3(k_i)$  is

$$v_1 \cup_{f()} v_2 = f(v_1, v_2) \text{ or } v_1 \cap_{f()} v_2 = f(v_1, v_2)$$

- If  $A_1(k_i) = v$  or  $\emptyset$  and  $A_2(k_i) = \emptyset$  or  $v$ , then  $A_3(k_i)$  is

$$v \cup_{f()} \emptyset = v \text{ or } v \cap_{f()} \emptyset = \emptyset$$

- High level usage dictated by these definitions
- Deeper algebraic properties set by the collision function  $f()$
- Frequent switching between “algebras”



# Outline

- Introduction
- Technologies
- • Results
  - Insert performance
  - Text query
  - Computer Networks
  - DNA Sequencing
- Summary



# Graph500 Benchmark Performance

Home Complete Results

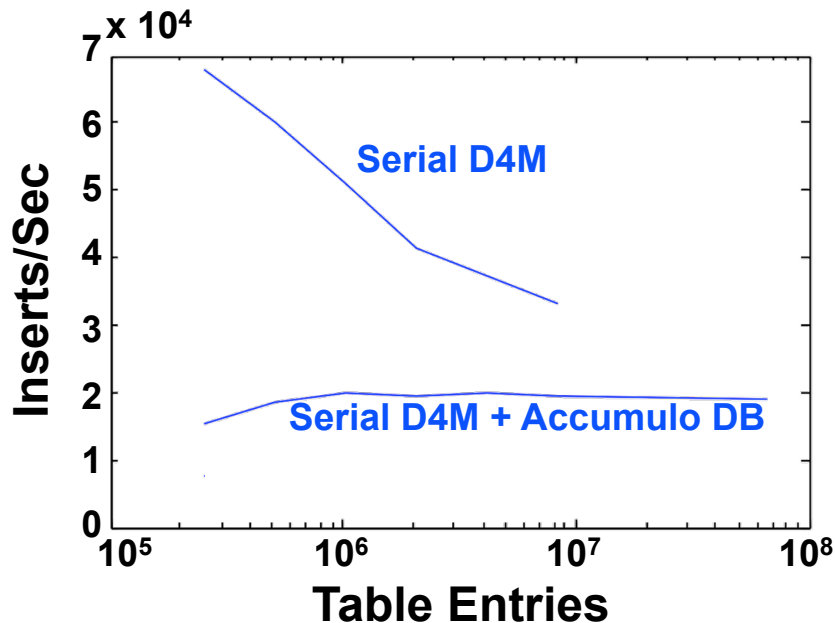
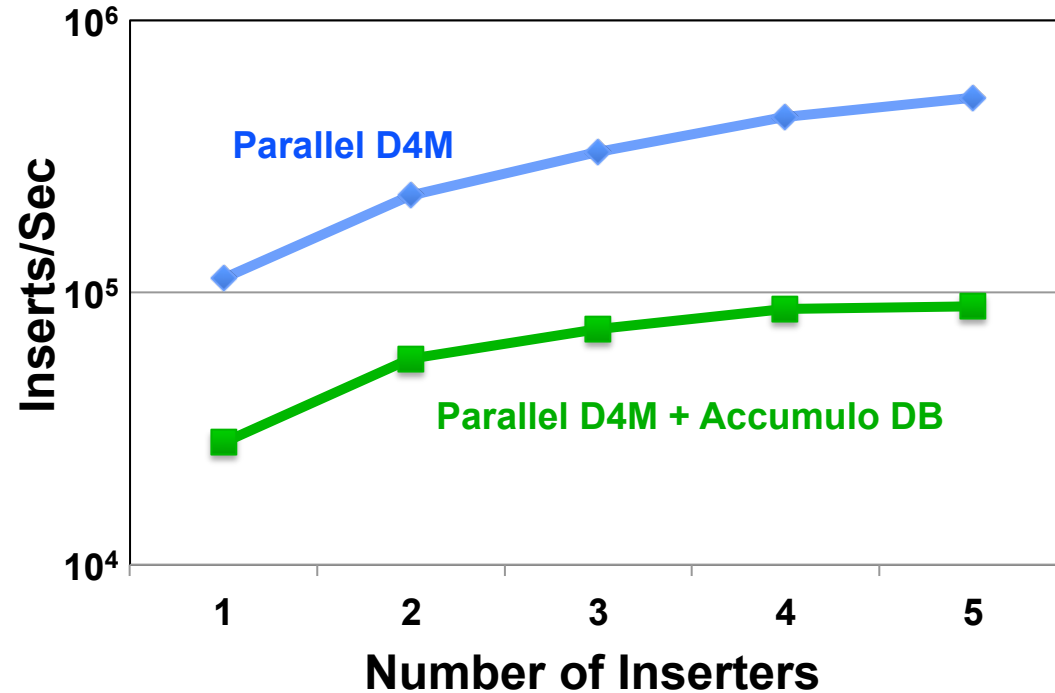
## GRAPH 500 The Graph 500 List

Top 10 (June 2011)

Rank	Machine
1	Intrepid (BG/P, 32768 nodes/ 131072 cores)
2	Jugene (IBM, 32k nodes)
3	Lomonosov (MPP, 4096 nodes/ 8192 cores)

**Brief Introduction**

Data intensive supercomputer applications are increasingly important for HPC workloads, but are ill-suited for platforms designed for 3D physics simulations. Current benchmarks and performance metrics do not provide useful information on the suitability of supercomputing systems for data intensive applications. A new set of benchmarks is needed in order to guide the design of hardware architectures and software systems intended to support such applications and to help procurements. Graph algorithms are a core part of many analytics workloads.



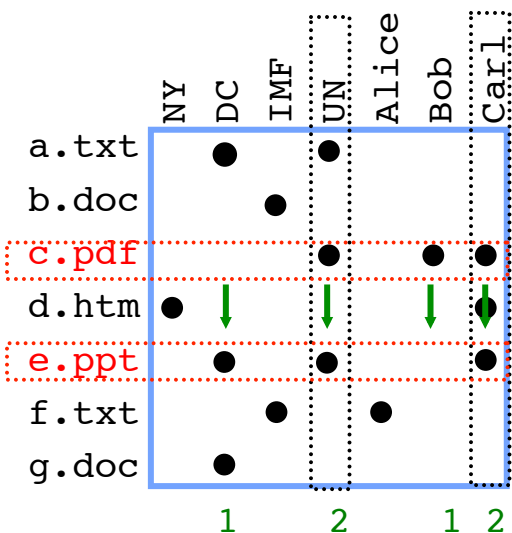
- Graph500 generates power law data
- D4M (in memory) + Accumulo (storage) provides scalable high performance



# Text Facet Search

## Algorithm

- Facets  $x=UN$ ,  $y=Carl$
- Documents that contain both  $A(:,x)$  &  $A(:,y)$
- Entity counts  $(A(:,x) \& A(:,y))^t A$



## Code

```
A=T(:, :); % Load Reuters docs.
x='LOCATION/new york,'
y='PERSON/john howard,'
(noCol(A(:,x)) & noCol(A(:,y))).' * A
```

## Results

```
LOCATION/asia, 1
LOCATION/australia, 3
LOCATION/london, 1
...
PERSON/bill clinton, 1
PERSON/david kemp, 1
...
```

## Reuters Corpus

```
797677 documents
1786 locations
141 organizations
37191 persons
8444 times
6132286 entries
```

• Dynamically computes histogram of entities within a subset of documents



# Computer Networks

## Network Events Table: T

Associative Array: A

Row	Key (time)	src_ip/a	src_ip/b	src_ip/c	src_ip/d	domain/a	domain/b	domain/c	domain/d	dest_ip/a	dest_ip/b	dest_ip/c	dest_ip/d	Recv/a	Recv/b	Recv/c	Recv/d	Recv/e	
1	2001-10-01 01 01 00																		
2	2001-10-01 01 02 00																		
3	2001-10-01 01 03 00																		
4	2001-10-01 01 04 00																		
5	2001-10-01 01 05 00																		
6	2001-10-01 01 06 00																		

- Define ranges of rows and columns

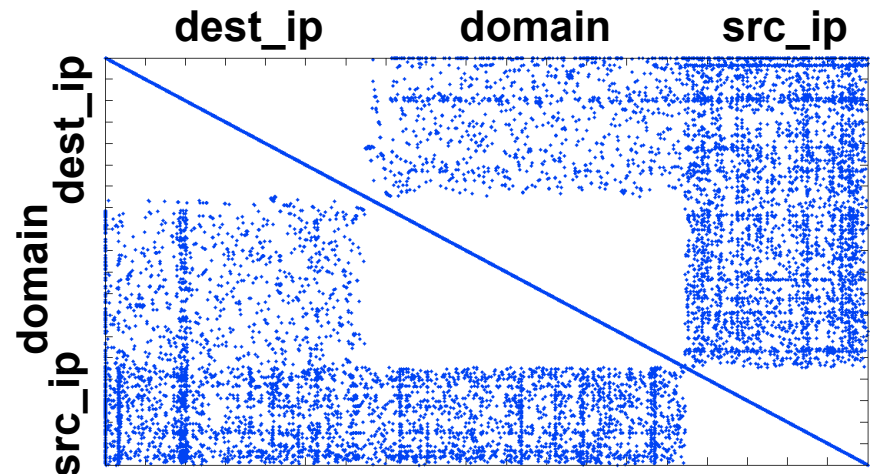
```
r = '2001-01-01 01 02 00, :, 2001-01-01 01 04 00, '
```

```
c = StartsWith('src_ip/, domain/, dest_ip/')
```

- Query table and find popular pairs

```
A = T(r, c)
```

```
A' * A > 200
```



- Good for identifying column types, gaps, clutter, and correlations



# Bio Sequencing

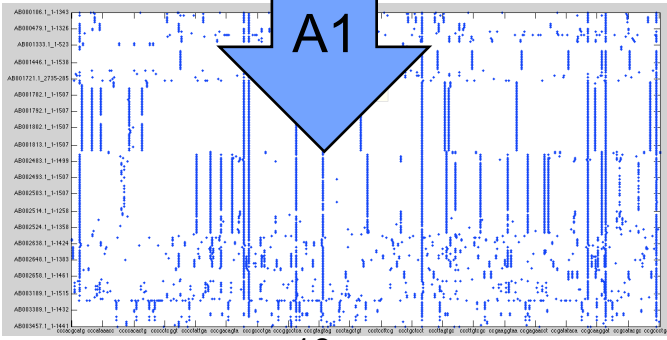
### Sequence Set 1

SeqID	sequence
G6J0L4R01AUYU3	TAGATACTGCTGCCTCCCG
G6J0L4R01DLKJM	TTTTTTTCGTGCTGCTGCC
G6J0L4R01D0SEN	TTATCGGCTGCTGCCTCCC
G6J0L4R01EOS3L	AGGTTGTCTGCTGCCTCTA

### Sequence Set 2

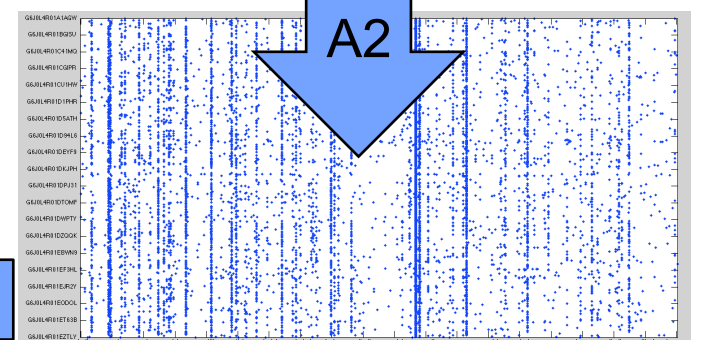
SeqID	sequence
AB000106.1_1-1343	ggaatctgcccttgggttcgg
AB000278.1_1-1410	caggcctaacacatgcaagt
AB000389.1_1-1508	ttgatcctggctcagattgaa
AB000390.2_1-1428	catgcaagtcgagcggaad

SeqID



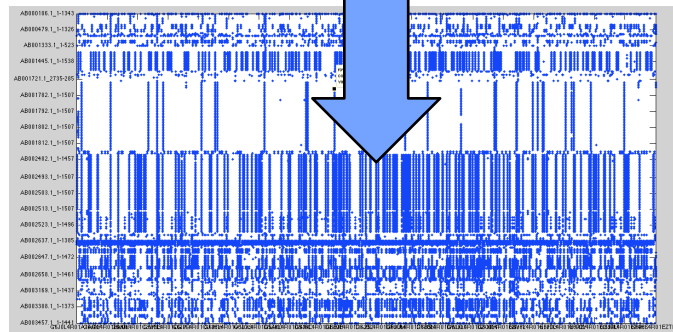
10mer

SeqID



10mer

$A1 * A2'$



**• Matrix multiply rapidly finds common 10 base sequences**





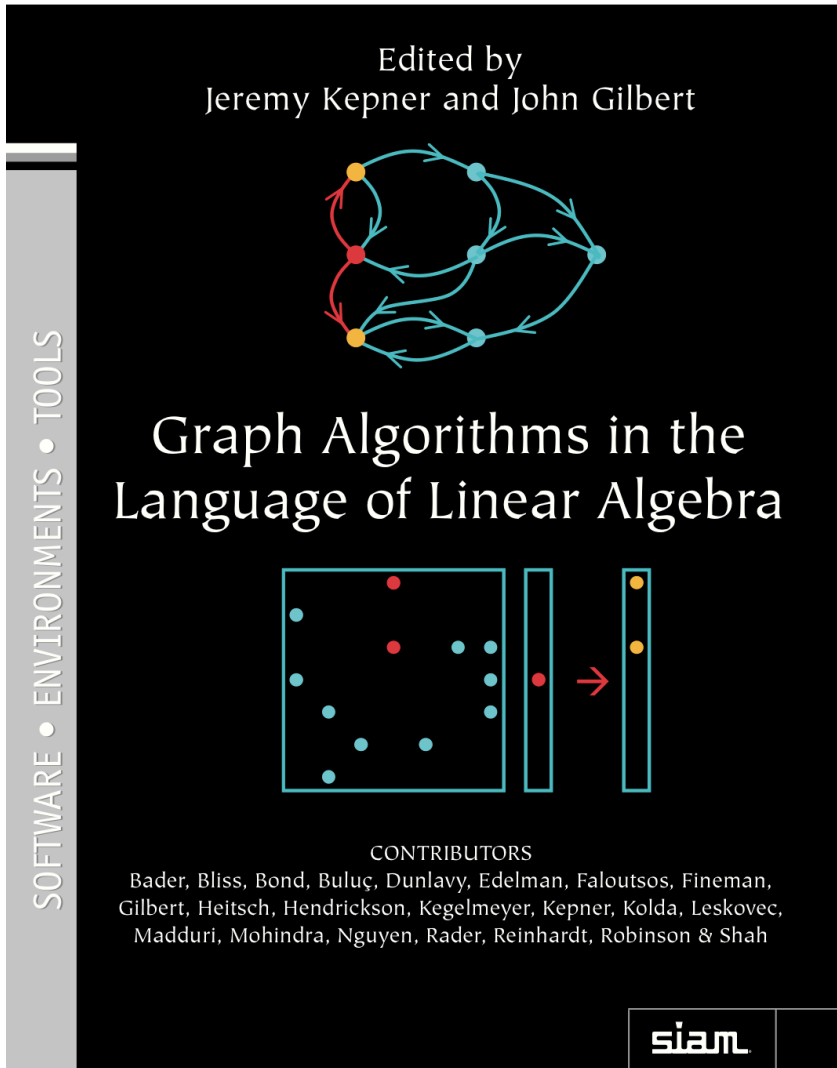
# Summary

---

- **Big data is found across a wide range of areas**
  - Document analysis
  - Computer network analysis
  - DNA Sequencing
- **Currently there is a gap in big data analysis tools for algorithm developers**
- **D4M fills this gap by providing algorithm developers composable associative arrays that admit linear algebraic manipulation**



# Reference



- **Editors: Kepner (MIT-LL) and Gilbert (UCSB)**
- **Contributors**
  - Bader (Ga Tech)
  - Bliss (MIT-LL)
  - Bond (MIT-LL)
  - Dunlavy (Sandia)
  - Faloutsos (CMU)
  - Fineman (CMU)
  - Gilbert (UCSB)
  - Heitsch (Ga Tech)
  - Hendrickson (Sandia)
  - Kegelmeyer (Sandia)
  - Kepner (MIT-LL)
  - Kolda (Sandia)
  - Leskovec (CMU)
  - Madduri (Ga Tech)
  - Mohindra (MIT-LL)
  - Nguyen (MIT)
  - Rader (MIT-LL)
  - Reinhardt (Microsoft)
  - Robinson (MIT-LL)
  - Shah (UCSB)