# Signal Processing on Databases

Jeremy Kepner

## Lecture 6: Bio Sequence Cross Correlation

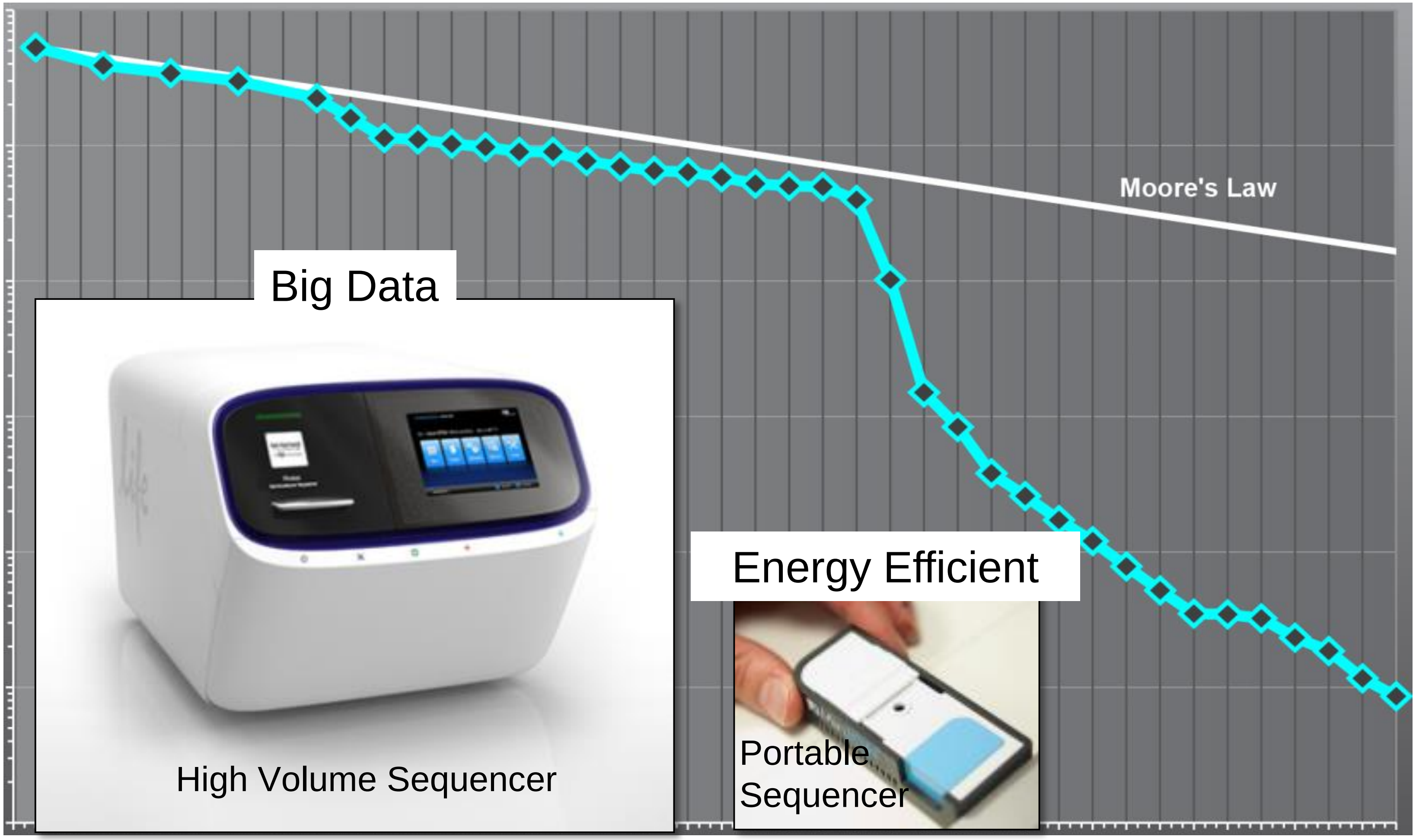
LINCOLN LABORATORY
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

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
- Algorithm
- Implementation
- Results
- Summary

# Relative Cost per DNA Sequence



Big Data

High Volume Sequencer

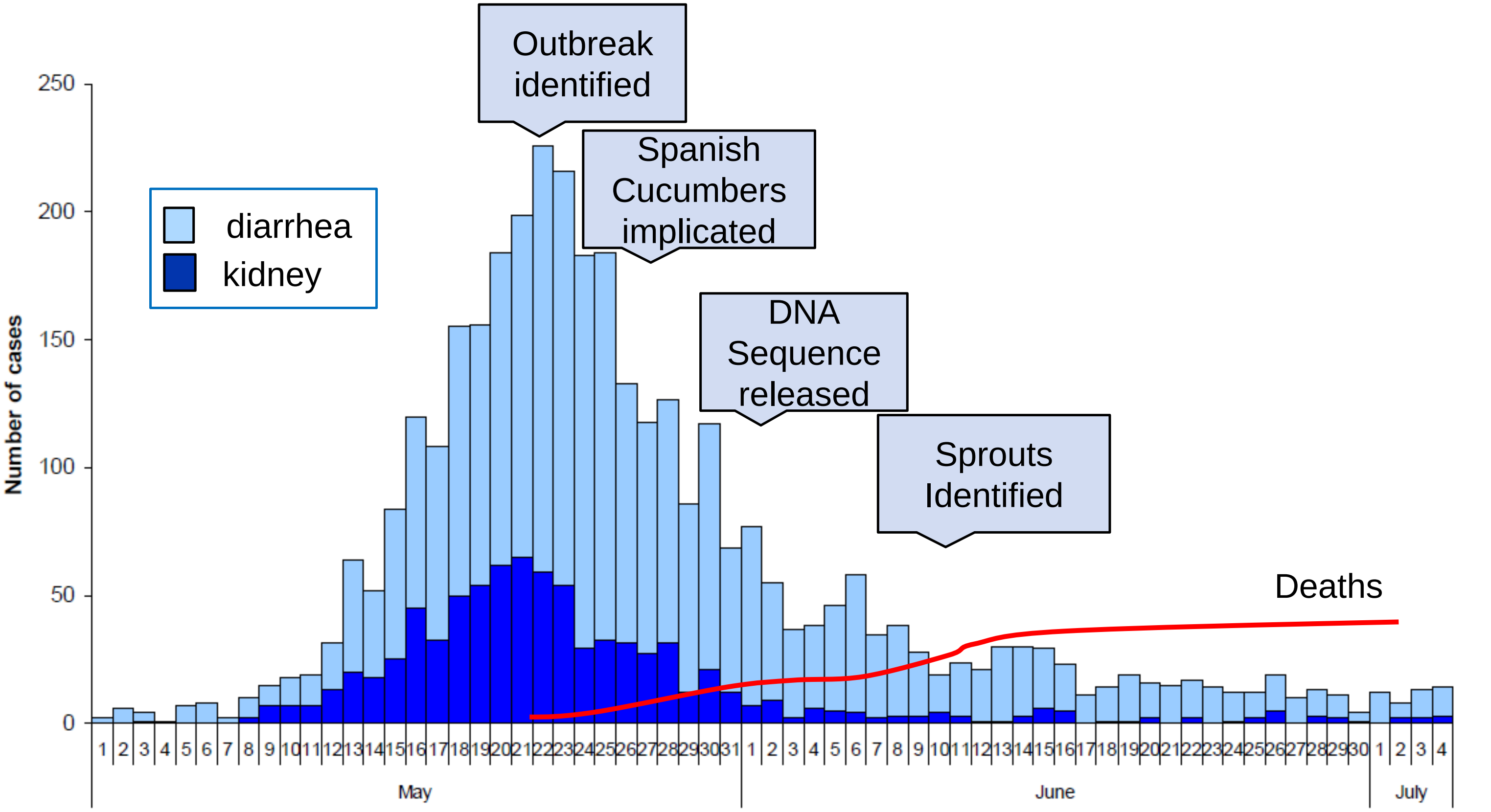
Energy Efficient

Portable Sequencer

Wetterstrand KA. DNA Sequencing Costs: Data from the NHGRI Large-Scale Genome Sequencing Program Available at: www.genome.gov/sequencingcosts. Accessed 03/08/2012

# Example Disease Outbreak

## May-July 2011 - Virulent *E. Coli* Outbreak Germany



Outbreak identified

Spanish Cucumbers implicated

DNA Sequence released

Sprouts Identified

Deaths

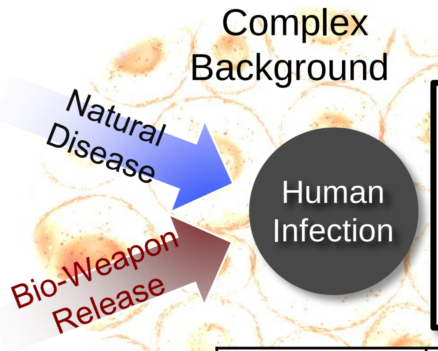
www.rki.de EHEC final report

Conclusions: Identification of E. Coli source too late to have substantial impact on illnesses
Publishing sequence data allowed for broad community to fully characterize pathogen

Courtesy of Robert Koch Institute. See Figure 2 in Report: Final presentation and evaluation of epidemiological findings in the EHEC O104:H4 outbreak, Germany 2011. Berlin 2011. Used with permission.

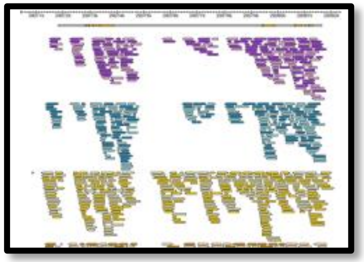
**Sequencing and crowd source analysis showed promising potential -> Still too slow**

# Example Processing Timeline

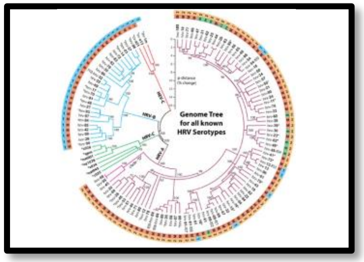


Collection & Shipment of Sample

Sample Preparation & Sequencing



Analysis of Samples



Time to Actionable Data

| | | | | |
|---|---|---|---|---|
| Current | 2-30 days | 1-3 days | 7-14 days | 10-45 days |
| Goal | on site | 3 hours | <12 hours | **< 1 day** |

- **Processing plays a key part in accelerating the overall time to solution**

Slide inspired by Nicole Rosenzweig, ECBC

# DNA Sequence Matching

## Goal

- Quickly compare two sets of DNA

## Applications

- Identification
- Mixture Analysis
- Kinship Analysis
- Ancestry Analysis

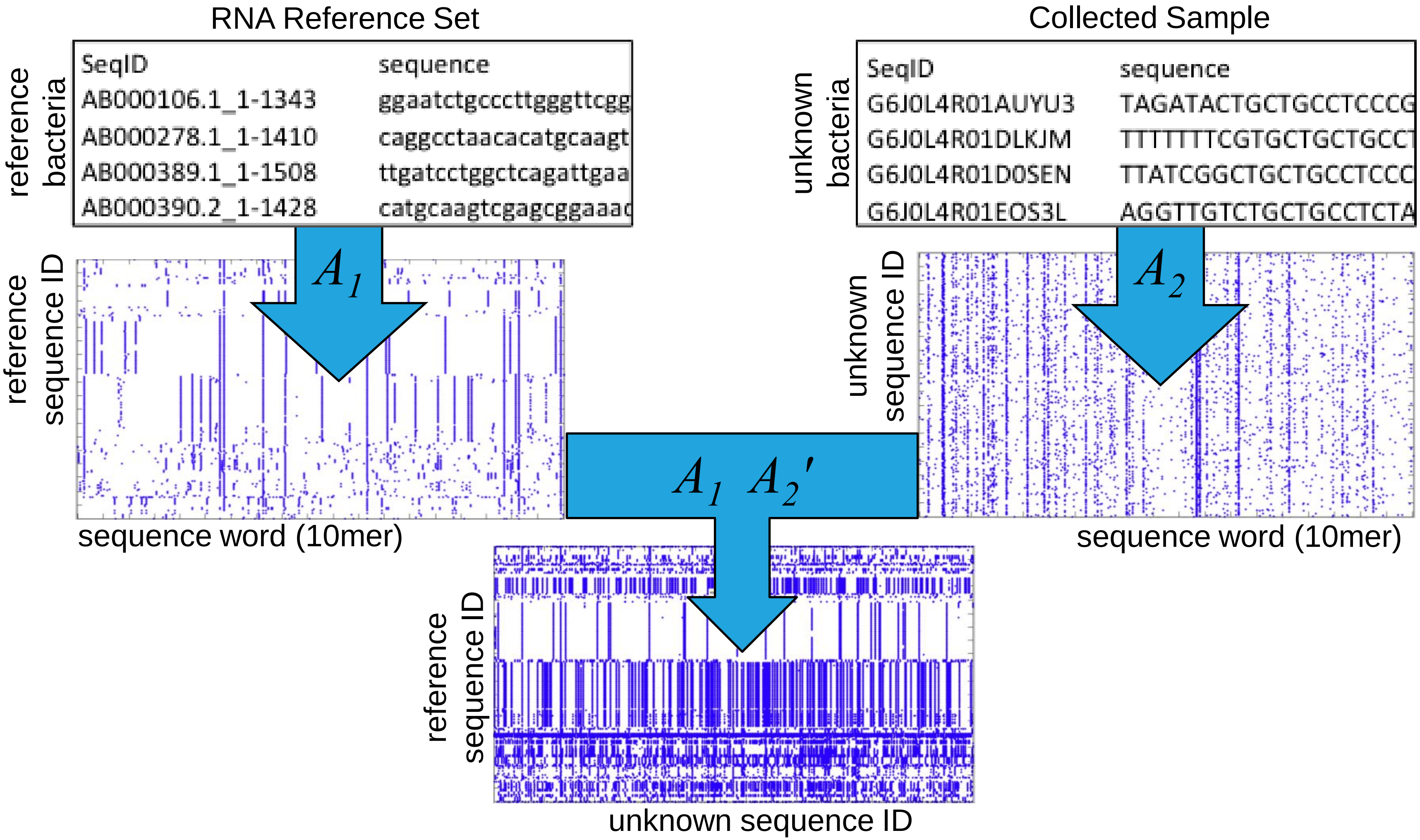
**Uses: disease outbreaks, criminal investigations, personal medicine, …**

- **Challenge: sequencing matching takes a long time, can we make it faster?**

# Outline

- Introduction
- ➡ Algorithm
- Implementation
- Results
- Summary

# Sequence Matching ⇔ Sparse Matrix Multiply in D4M

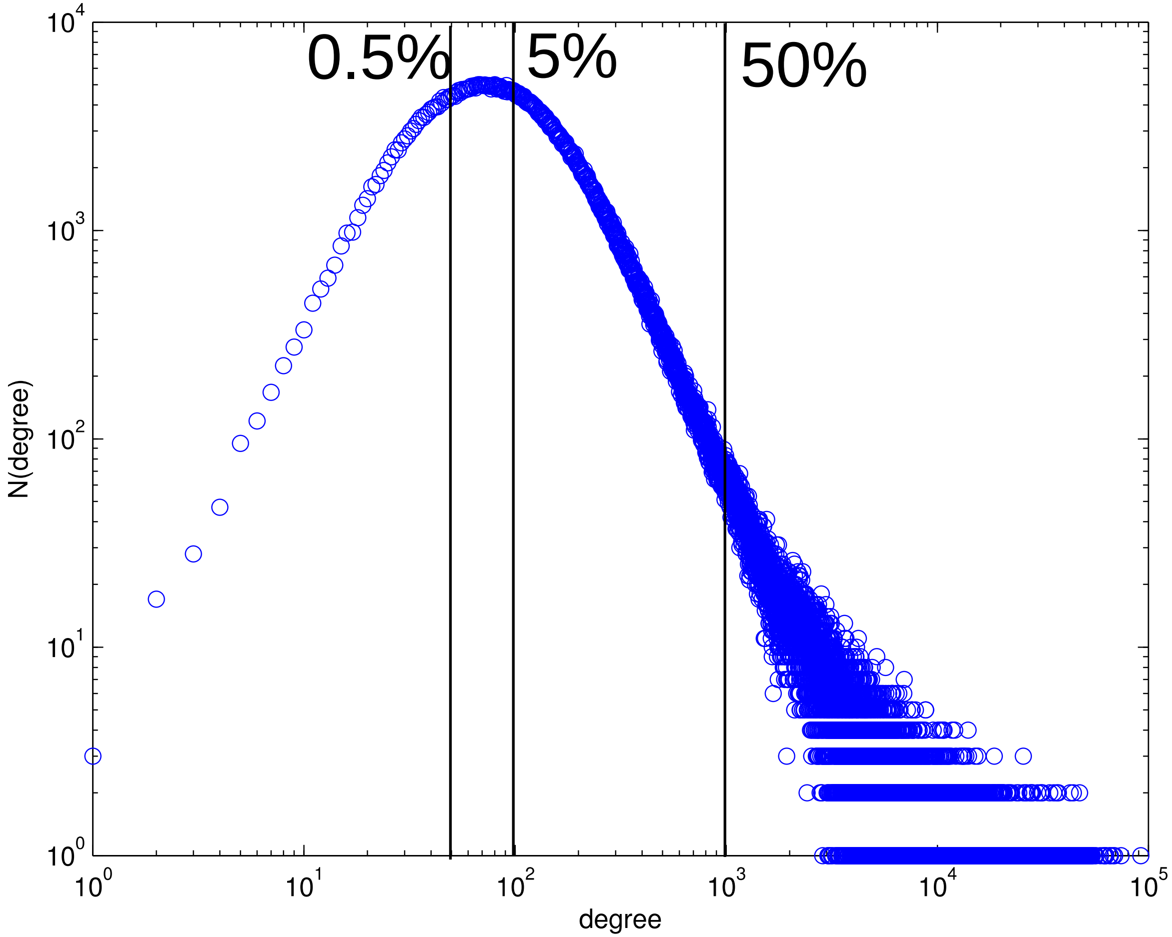


| SeqID | sequence |
|---|---|
| AB000106.1_1-1343 | ggaatctgcccttgggttcgg |
| AB000278.1_1-1410 | caggcctaacacatgcaagt |
| AB000389.1_1-1508 | ttgatcctggctcagattgaa |
| AB000390.2_1-1428 | catgcaagtcgagcggaaac |

| SeqID | sequence |
|---|---|
| G6J0L4R01AUYU3 | TAGATACTGCTGCCTCCCG |
| G6J0L4R01DLKJM | TTTTTTTCGTGCTGCTGCCT |
| G6J0L4R01D0SEN | TTATCGGCTGCTGCCTCCC |
| G6J0L4R01EOS3L | AGGTTGTCTGCTGCCTCTA |

- **Associative arrays provide a natural framework for sequence matching**

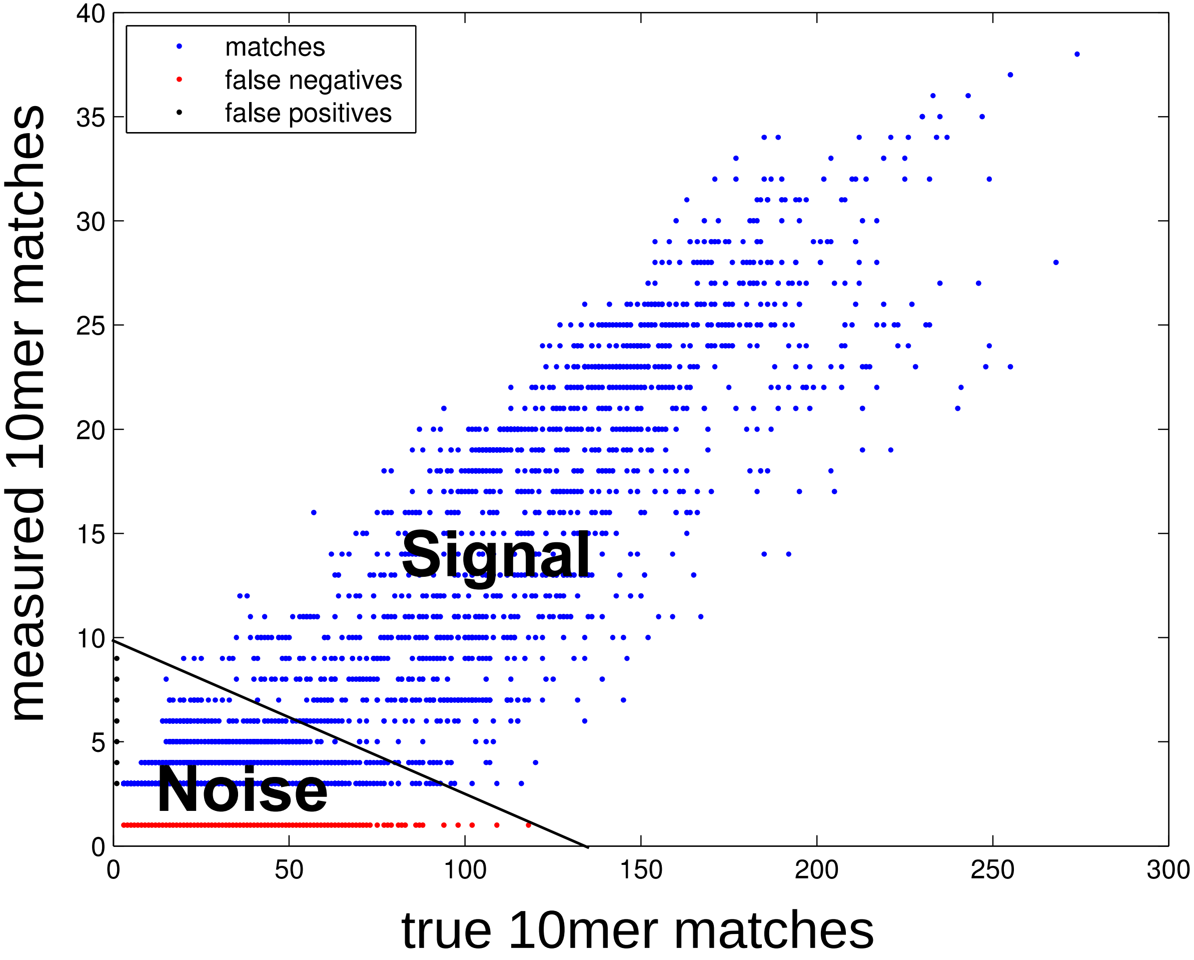
# Database Automatically Computes Reference 10mer Distribution



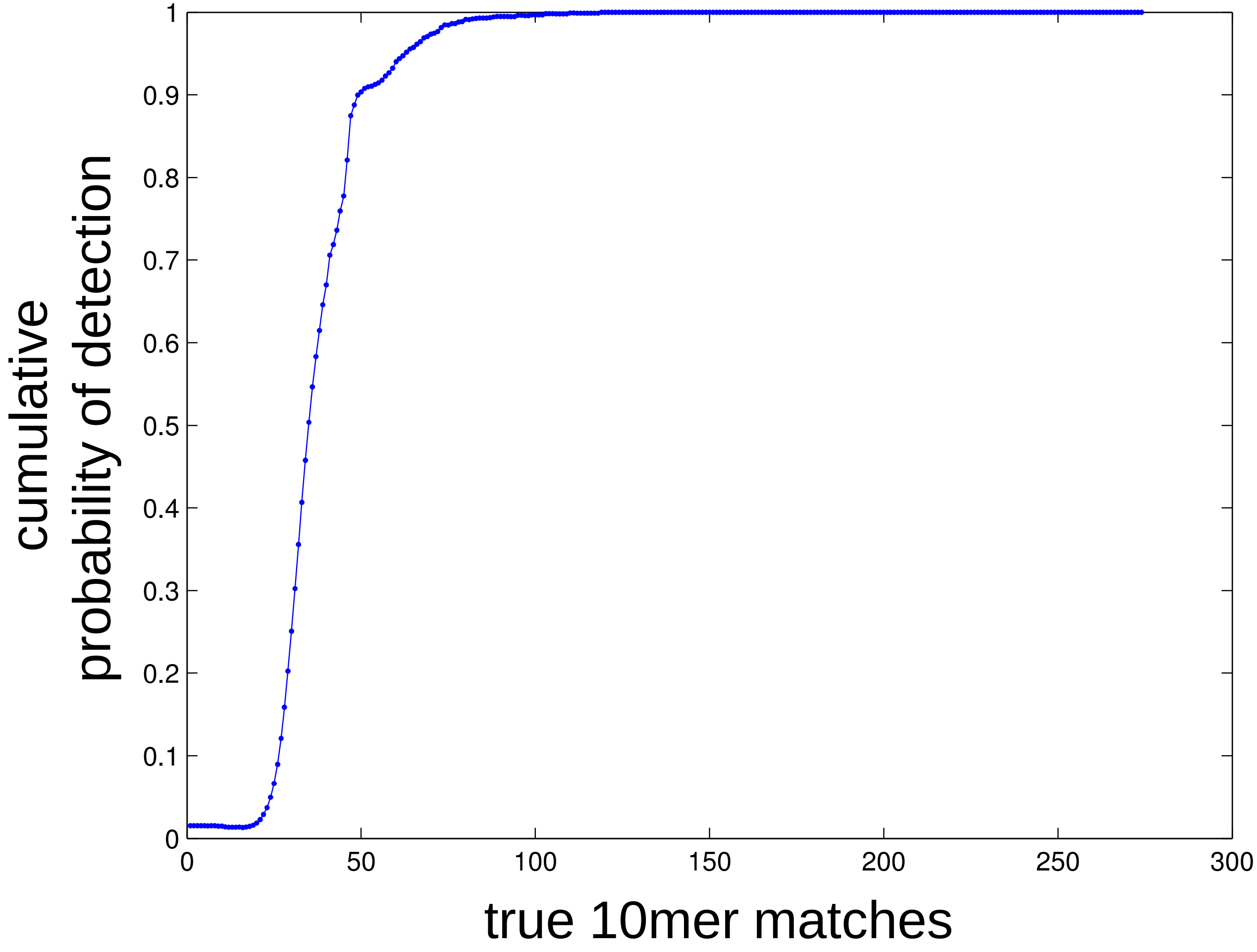
- **Using 10mer distribution can quickly select reference 10mers that maximally differentiate sample sequences and eliminate most 10mers**

# 0.5% Selection Results

- **Sample (20MB):**
  - **NGS from Roche 454**
- **Reference (500MB):**
  - **Virus DNA from GenBank**
- **All strong matches detected using 0.5% of data**

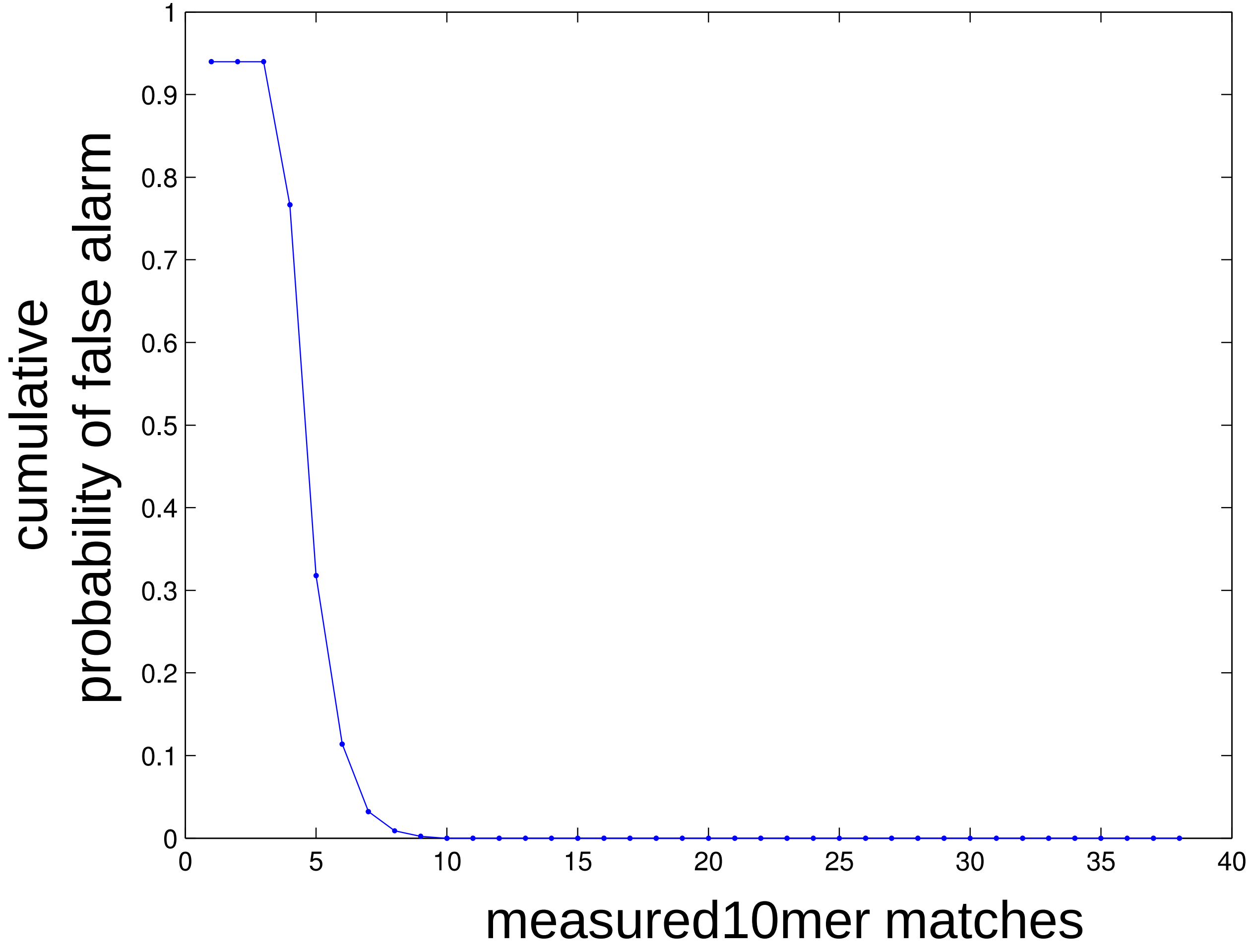


# Cumulative Probability of Detection



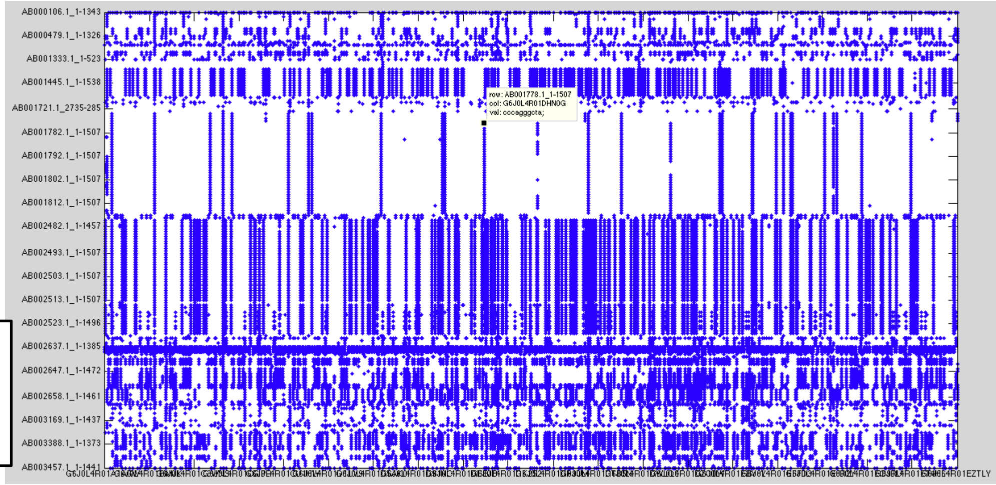
- **100% detection of all true matches > 100**

# Cumulative Probability of False Alarm



- **Measured matches > 10 are always matches**

# Finding Matches



A = A1 * A2'

Ak = CatKeyMul(A1,A2')

- **Find sequences with >6 word matches**

Am = Ak(Row(A > 6),Col(A > 6))

(AB001520.1_1-1428,G6J0L4R01B4UPM)

aaatctttaa;aatctttaaa;ctttaaataa;ggggaccagc;taaatcttta;ttaaataaaa;tttaaataaa;,

(AB002634.1_1-1419,G6J0L4R01EDJVA)

aaatgtcgtt;aatgtcgttt;atgtcgtttc;gtcgtttccc;gtctcagttc;tcgtttccct;tgtcgtttcc;,

- **Associative array cat multiply preserves pedigree of matches**

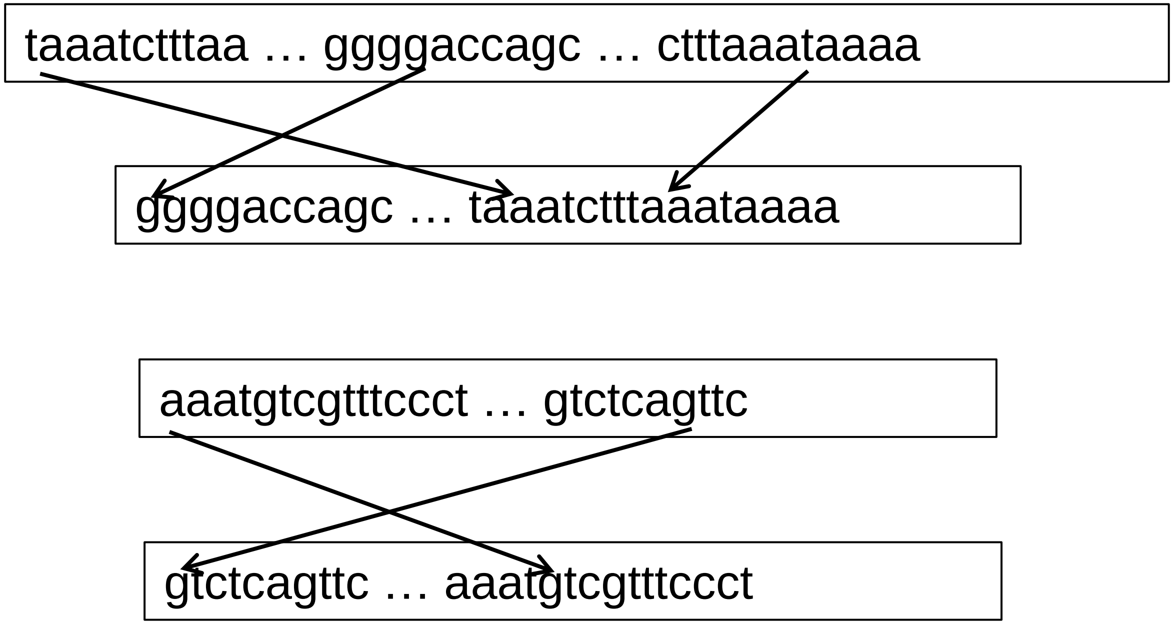
# Sequence Alignment

- **Show relative alignments of sequences**

A1(Row(Am),Val(Am)) + A2(Row(Am),Val(Am))

<u>reference</u> <u>sample</u>

| | | |
|---|---|---|
| | AB001520.1_1-1428 | G6J0L4R01B4UPM |
| aaatctttaa | 564 | 155 |
| aatctttaaa | 1227 | 156 |
| ctttaaataa | 1376 | 159 |
| ggggaccagc | 877 | 58 |
| taaatcttta | 563 | 154 |
| ttaaataaaa | 1378 | 161 |
| tttaaataaa | 1377 | 160 |
| | AB002634.1_1-1419 | G6J0L4R01EDJVA |
| aaatgtcgtt | 933 | 300 |
| aatgtcgttt | 934 | 301 |
| atgtcgtttc | 935 | 302 |
| gtcgtttccc | 937 | 304 |
| gtctcagttc | 1211 | 37 |
| tcgtttccct | 938 | 305 |
| tgtcgtttcc | 936 | 303 |



taaatctttaa … ggggaccagc … ctttaaataaaa

ggggaccagc … taaatctttaaataaaa

aaatgtcgtttccct … gtctcagttc

gtctcagttc … aaatgtcgtttccct

- **Sequence alignment found by indexing into associative array**

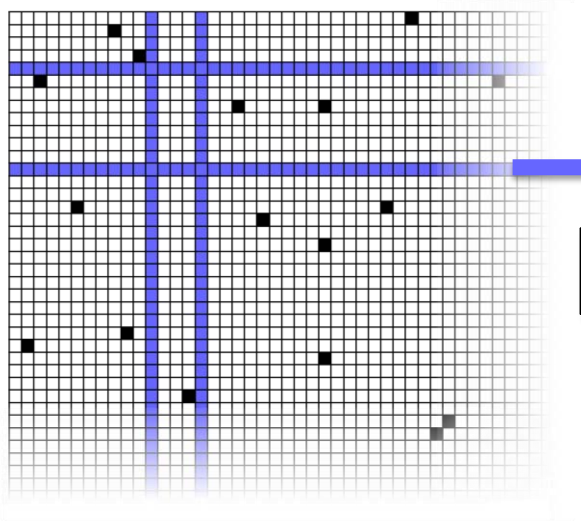
# Outline

- Introduction
- Algorithm
- Implementation
- Results
- Summary

# D4M Stores Giant Sparse Matrices in Accumulo Triple Store Database

Triple Store

Distributed Database



D4M

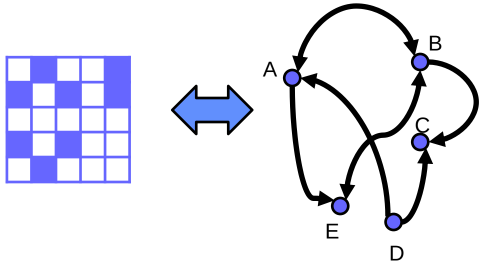
Dynamic
Distributed
Dimensional
Data
Model

Query:
T(:,ggaatctgcc)

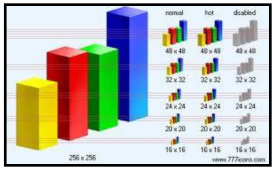
Triple store are high performance distributed databases for heterogeneous data

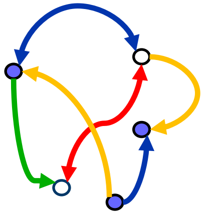
Associative Arrays

Numerical Computing Environment



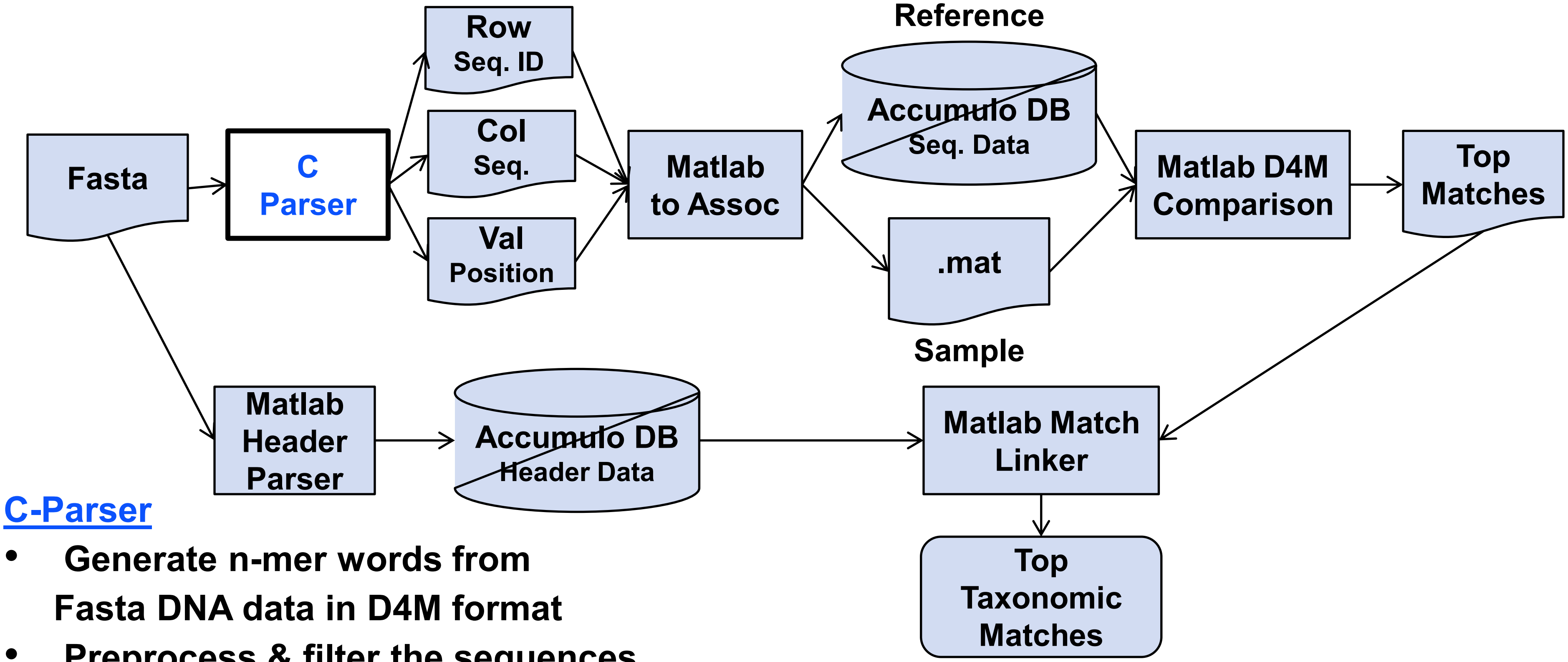
A D4M query returns a sparse matrix or graph from a triple store…





…for statistical signal processing or graph analysis in Matlab

# Sequence Processing Pipeline

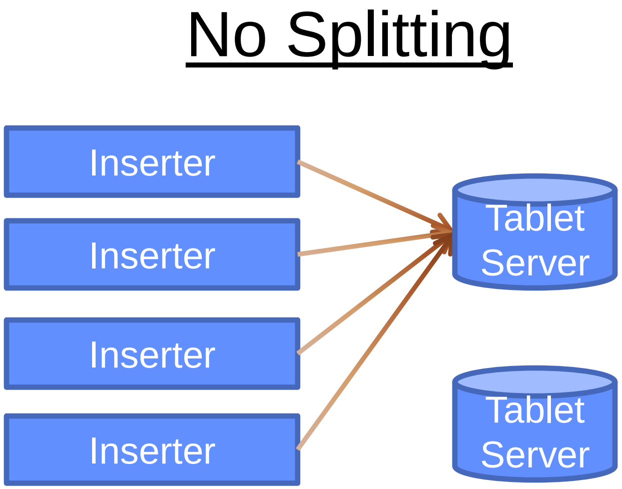


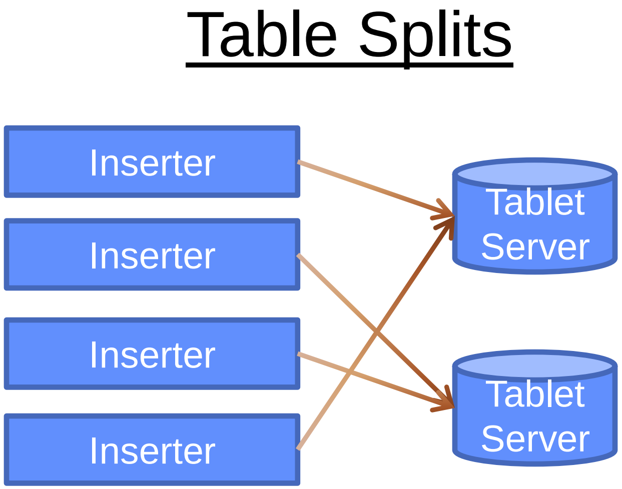
**C-Parser**

- Generate n-mer words from Fasta DNA data in D4M format
- Preprocess & filter the sequences
  – Ignore bad, common sequences
  – Break output files into manageable chunks, say 5MB
  – Generate reverse sequences
  – Break up big sequences into subsequences to preserve locality

# Database Table Splits

- **Initial inserts bottleneck on one tablet server until it fills up and splits**
- **Performance booster: pre-split table among several tablet servers for instant parallel insertion**
  - **Use advanced knowledge of row data patterns to choose splits**
- **Created functions to set and query table splits**

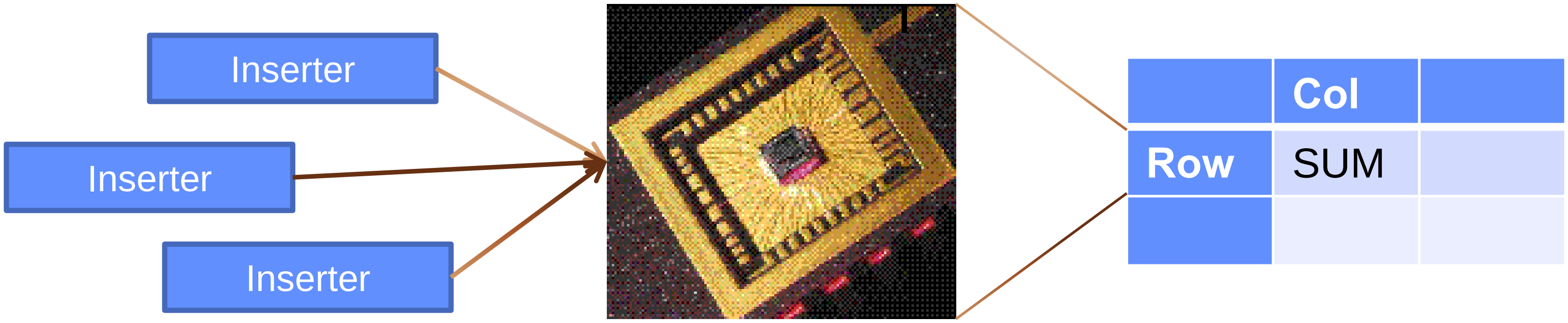




# Accumulator Columns

- Accumulator columns allow counting to be done on insert
  - Example: sequence counting
    - Row ID = 10 mer
    - Column = Count
    - Value = Count
  - Insert (aaatctttaa,Count,2) → DB has (Doc1, ‘bird’, 2)
  - Insert (aaatctttaa,Count,3) → DB has (aaatctttaa,Count,5)
- Works with any commutative operation
  - Addition, maximum, minimum, etc.

Database Accumu ator



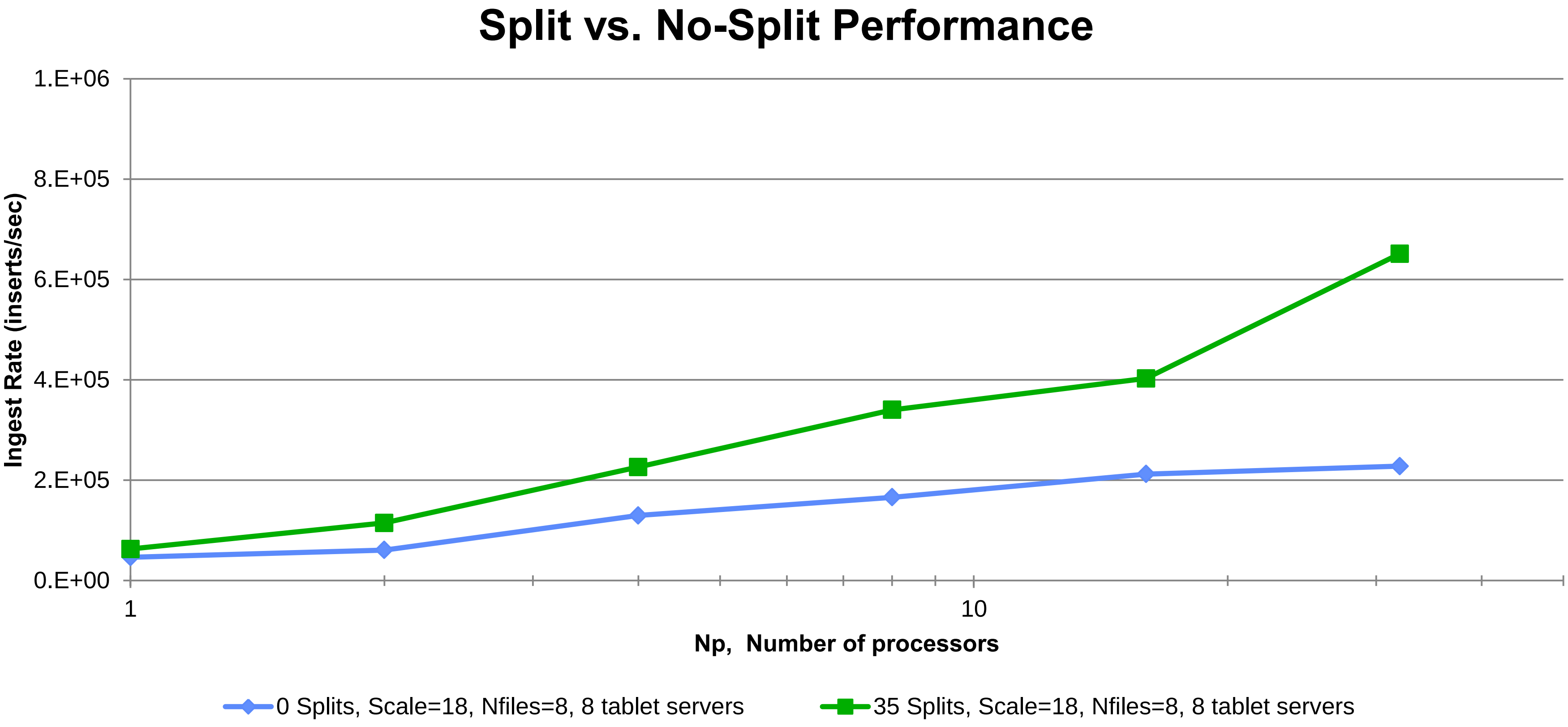
|  | Col |  |
|---|---|---|
| Row | SUM |  |
|  |  |  |

Courtesy of Jan Van der Spiegel.
Used with permission.

# Outline

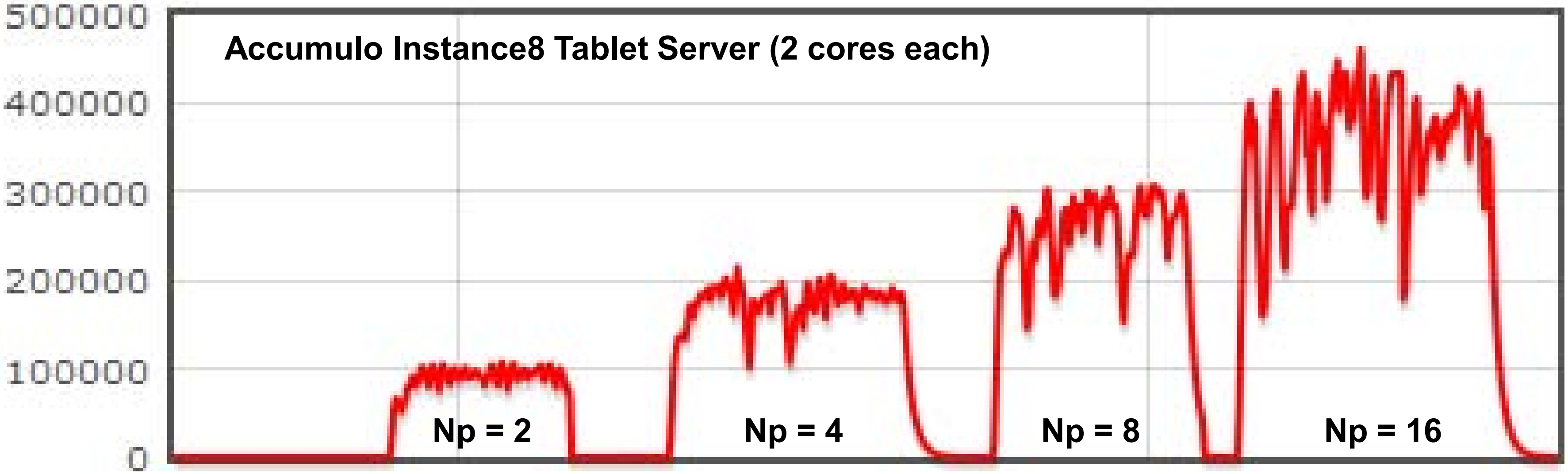
- Introduction
- Algorithm
- Implementation
- ➡ Results
- Summary

# Table Split Performance

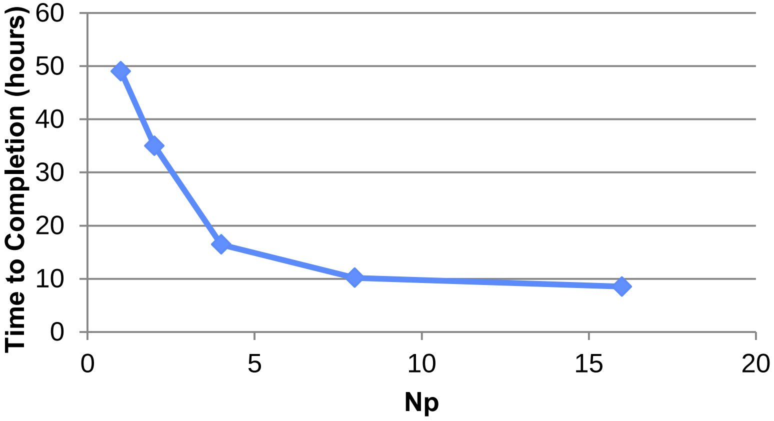


- **Pre-Splitting tables appropriately can double ingest rates at higher Np in multinode database environments**

# Human DNA DB Ingest Performance

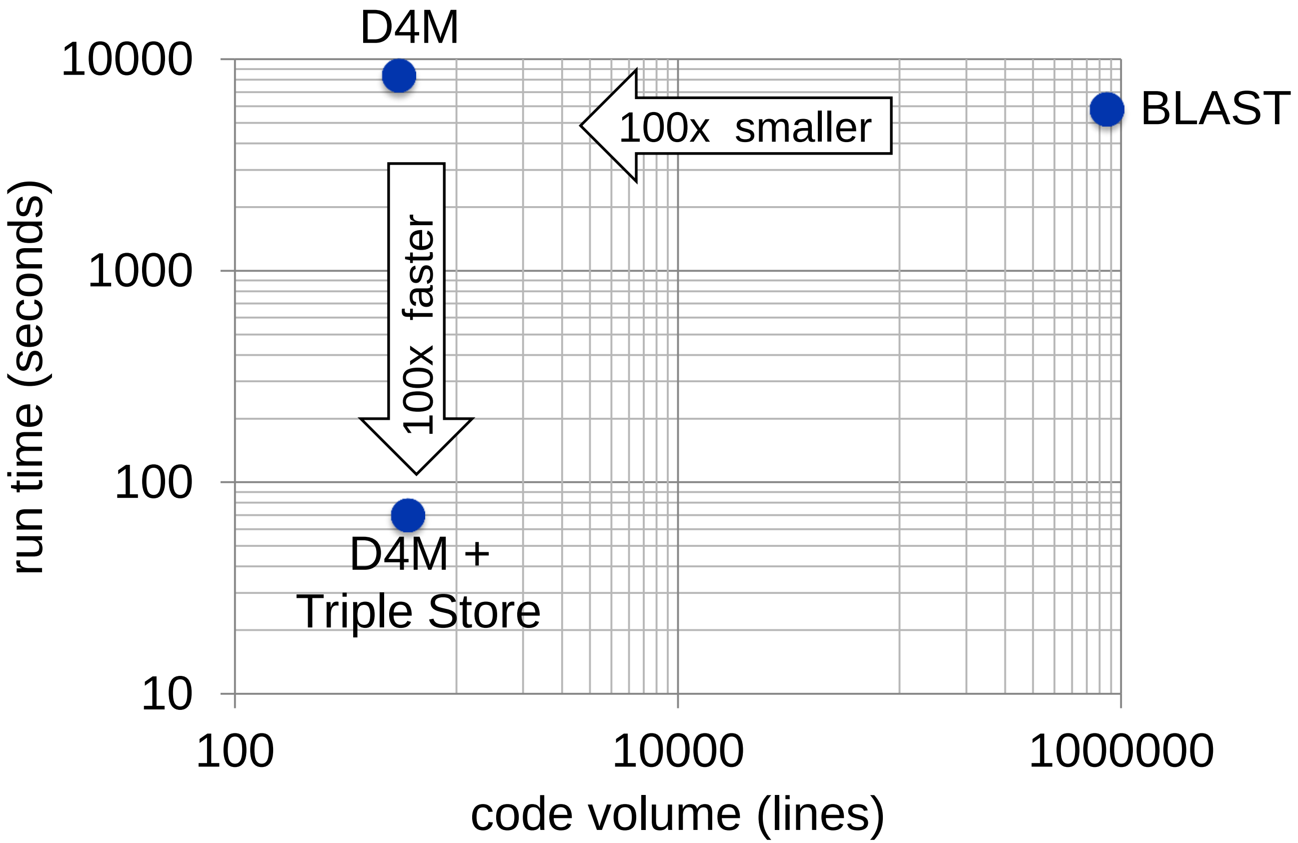


## Extrapolated Run Times



- 4.5 GB human Fasta file
- C Parser took 25 minutes
- 101 GB of row, col files
- Database ingest time ~10 hours

# Leveraging “Big Data” Technologies for High Speed Sequence Matching



- High performance triple store database trades computations for lookups
- Used Apache Accumulo database to accelerate comparison by 100x
- Used Lincoln D4M software to reduce code size by 100x

# 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

# Example Code & Assignment

- Example Code
  - d4m_api/examples/2Apps/4BioBlast
- Assignment
  - None

RES-LL.005 D4M: Signal Processing on Databases
Fall 2012