Adventures in NOSQL

Brian O'Connor
Postdoc – Nelson Lab
University of California, Los Angeles

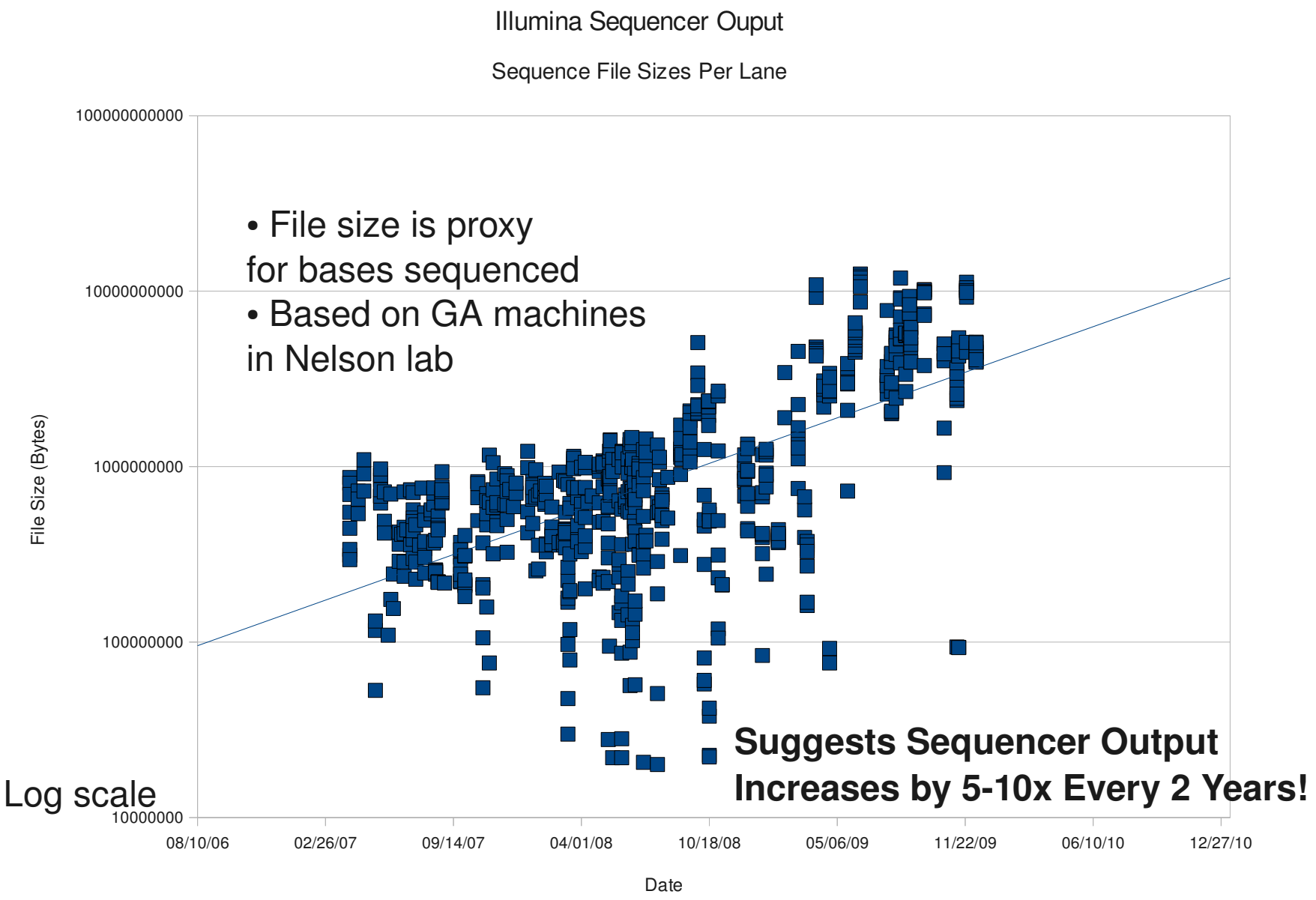
Overview

- Why I'm giving this talk
- My motivation
  - Stupendous data growth
  - Desire for scalability
- What NOSQL means
  - What you get
  - What you give up
- Progress in SeqWare project

My Background

- User of relational databases for many years
  - GMODWeb & Chado
  - Celsius & Chado
  - Other schema
- Seen relational DBs not scale well when pushed to hundreds of millions of rows, looking for alternatives
- I am just a user, *not an expert* and in the early stages of exploring NOSQL (**take everything I say with a grain of salt!**)

My Motivation



Pressures of Sequencing

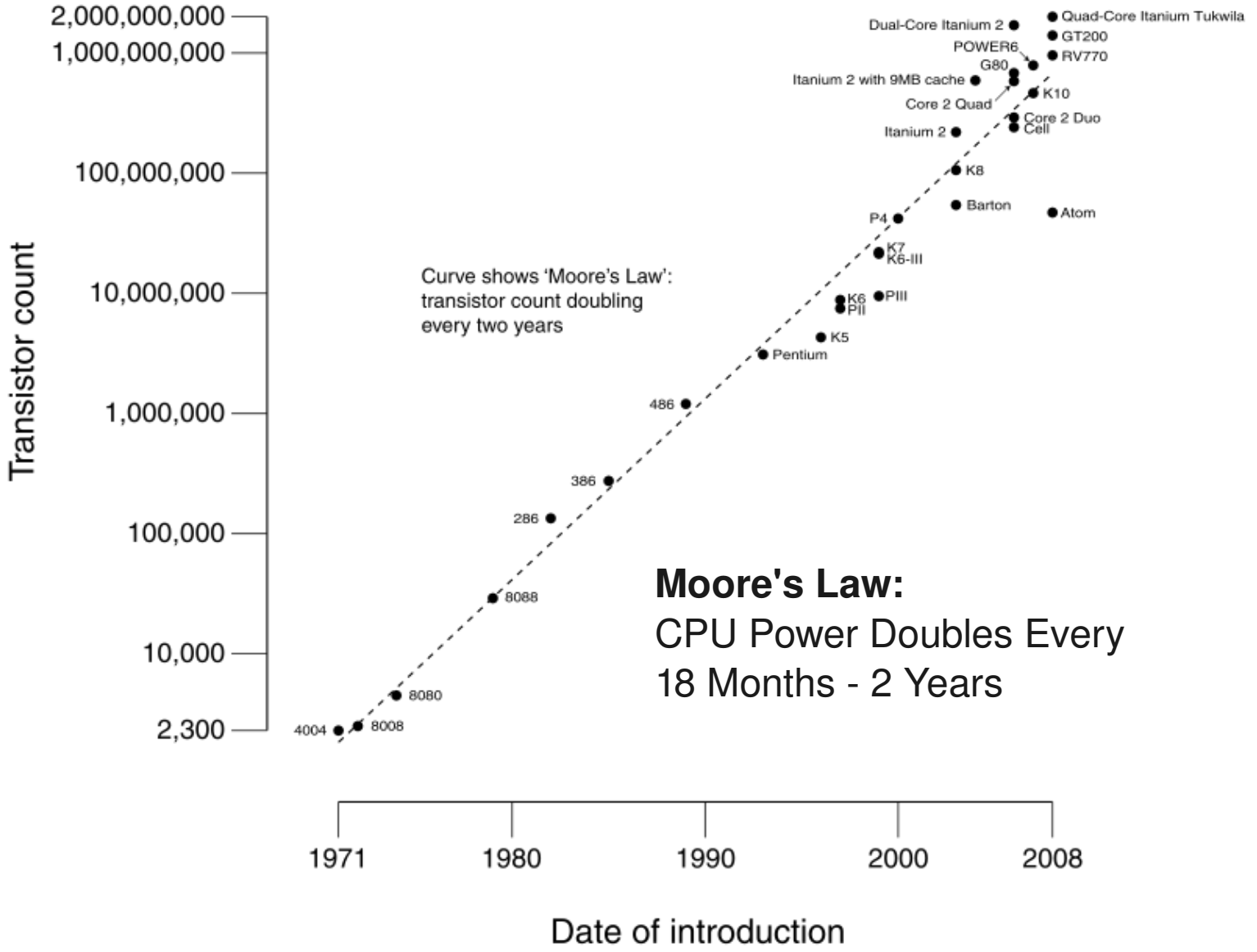
- A lot of data (50GB SRF file, 150GB alignment files, 60GB variants for a 20x human genome)
- Needs to be processed, annotated, and **queryable** (variants, coverage, annotations)!
- Want to ask simple questions:
  - “What SNVs are in 5'UTR of phosphatases?”
  - “What frameshift indels affect PTEN?”
  - “What genes include homozygous, non-synonymous SNVs?”

Requirements for System

- A **sequence database** must:
  - Provide a common “place” for lots of data and annotations
  - Support a rich level of annotation
  - Support very large variant databases (>300GB)
  - Be distributed across a cluster (>100 nodes)
  - Be queriable programmatically and/or using a simple RESTful web service
  - **Support a crazy growth of data scalably**

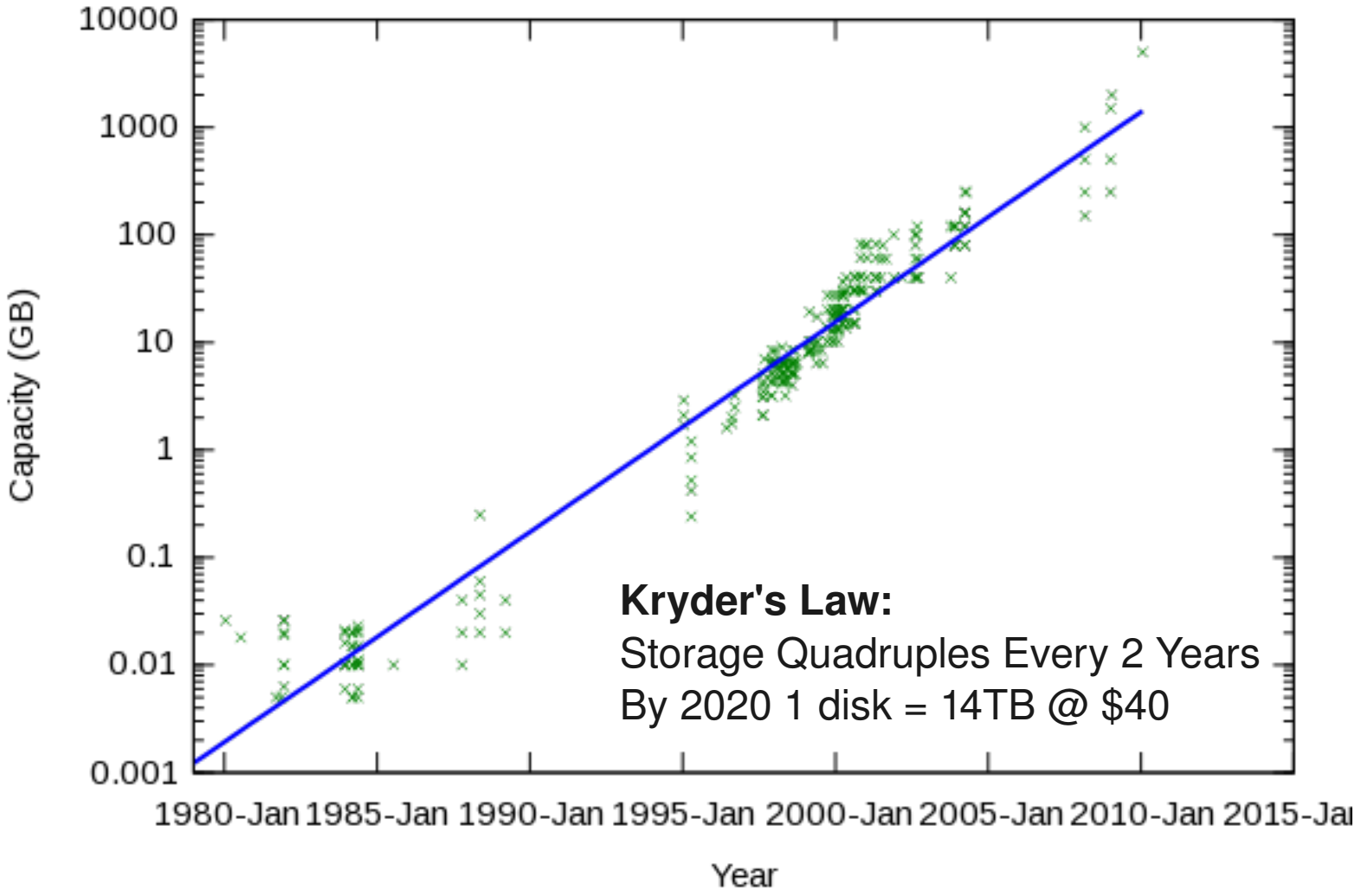
Reality of CPU Growth

CPU Transistor Counts 1971-2008 & Moore's Law



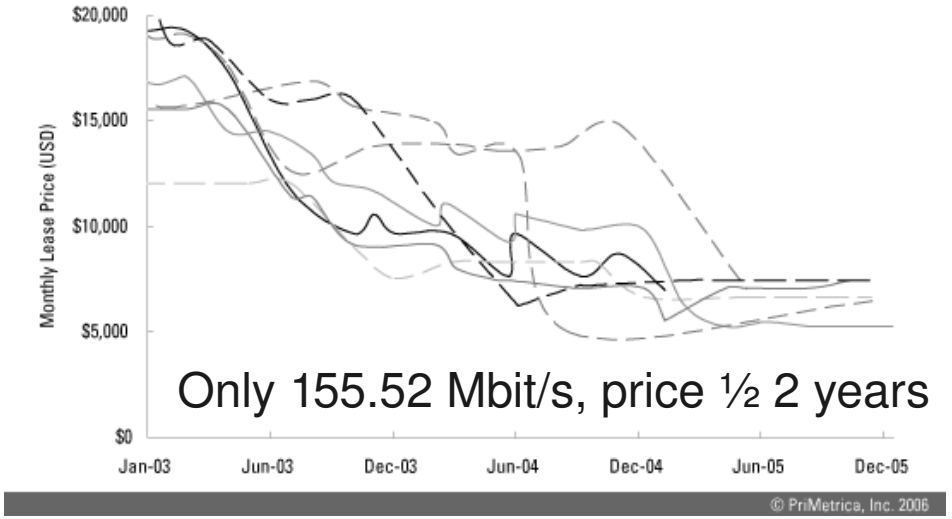
Moore's Law:
CPU Power Doubles Every
18 Months - 2 Years

Reality of Storage Growth



Reality of Bandwidth Growth

Los Angeles-New York OC-3 Lease



Only 155.52 Mbit/s, price ½ 2 years

Nielsen's Law:

Internet Bandwidth Doubles Every 2 Years

- Models home bandwidth (~10Mbps now)
- Institutions with GigE connections clearly much more expensive and constrained!

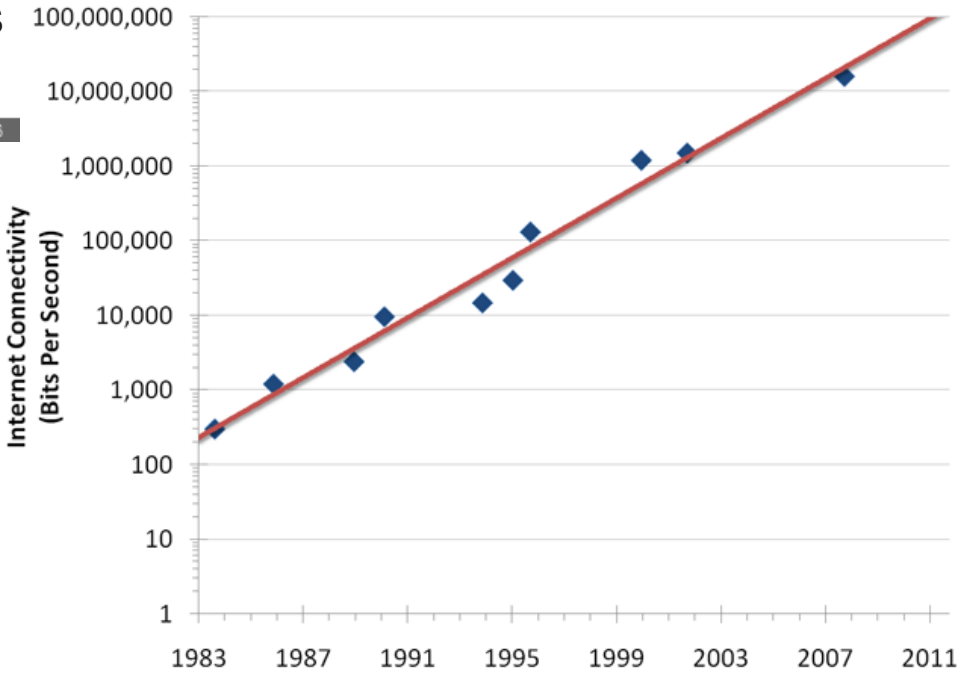
Internet2's backbone is 100Gbit/s

GigE IP Transit Price by City, Q2 09

http://www.telegeography.com

~$10K/month For GigE

Home Bandwidth Growth



http://www.useit.com/alertbox/980405.html

Sequencers vs. Information Technology

- Sequencers are increasing output by a factor of 10 every two years!
- Hard drives: 4x every 2 years
- CPUs: 2x every 2 years
- Bandwidth: 2x every 2 years (really?!)
- So there's a huge disconnect, **can't just throw more hardware at a single database server**!
- Must look for better ways to scale

Google to the Rescue?

- Companies like Google, Amazon, Facebook, etc have had to deal with massive scalability issues over the last 10+ years
- Solutions include:
  - Frameworks like *MapReduce*
  - Distributed file systems like *HDFS*
  - Distributed databases like *HBase*
- Focus here on **HBase**

Magically Scalable Databases

- Talking about *distributed* databases, forces a huge shift in what you can and can't do
- “NoSQL is an umbrella term for a loosely defined class of non-relational data stores that break with a long history of relational databases and ACID guarantees. Data stores that fall under this term may not require fixed table schemas, and usually avoid join operations. The term was first popularized in early 2009.”

http://en.wikipedia.org/wiki/Nosql

When Would You Use NOSQL?

- Horizontal scalability: not something common in database servers which tend to scale vertically
- Don't mind having weaker consistency guarantees
- Schema is less defined
- Data is sparse
- Query needs are not quite realtime (maybe)

http://blog.rapleaf.com/dev/?p=26

What Do You Give Up?

- SQL queries
- Well defined schema, normalized data structure
- Relationships manged by DB
- Flexible and easy indexing of table columns
- Existing tools that query a SQL database must be re-written
- Certain ACID aspects
- Software maturity, most distributed NOSQL projects are very new

What Do You Gain?

- Scalability is the clear win, you can have many processes on many nodes hit the collection of database servers
- Ability to look at very large datasets and do complex computations across a cluster
- More flexibility in representing information now and in the future
- HBase includes data timestamps/versions
- Integration with Hadoop

When Would You Not Use It?

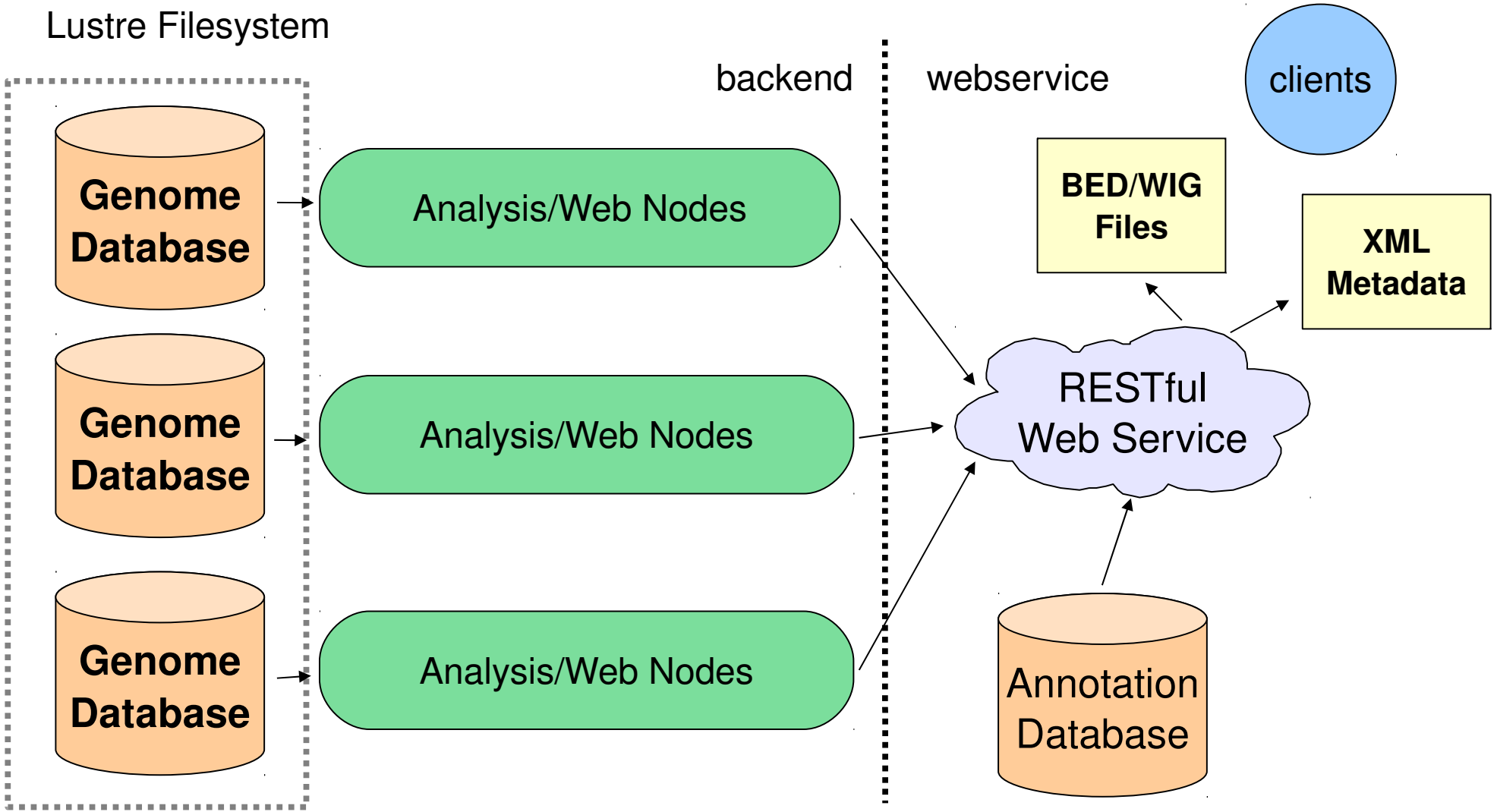
- When your data fits in a SQL database
- If your data is simple, fits well into a relational schema
- You data is not sparse
- You want to store large amounts of binary data

http://blog.rapleaf.com/dev/?p=26

SeqWare Query Engine

- **My first NOSQL project**
- Ask simple questions:
  - "What SNVs are in 5'UTR of phosphatases?"
  - "What frameshift indels affect PTEN?"
  - "What genes include homozygous, non-synonymous SNVs?"
- SeqWare Query Engine was designed to:
  - Support very large variant databases (>300GB)
  - Be distributed across a cluster (>100 nodes)
  - Be queriable using a simple RESTful web service (e.g. common programmatic interface)
  - Support a rich level of annotation (is_dbSNP, frameshift)

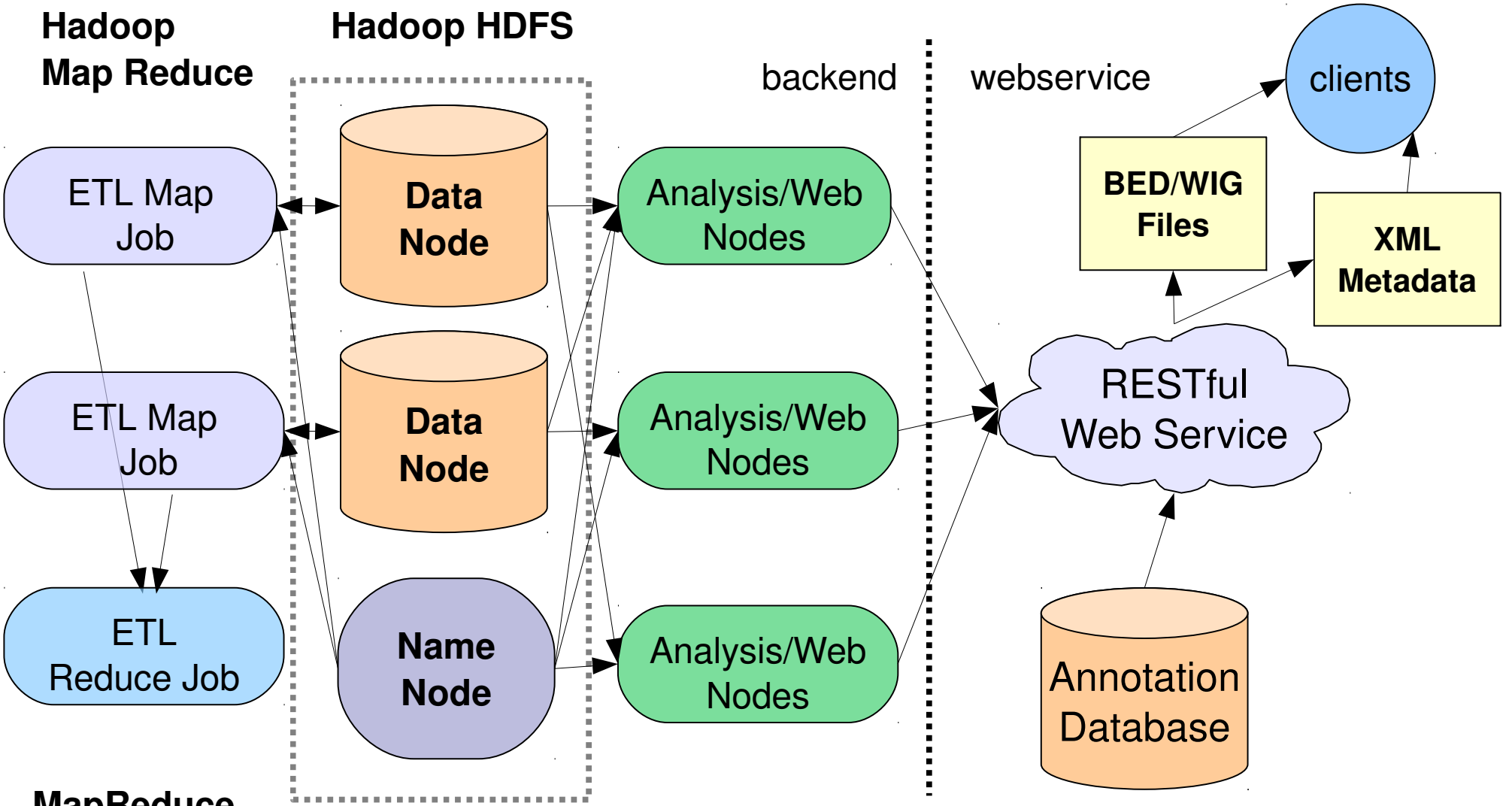
BerkeleyDB SeqWare Query Engine



Challenges with BerkeleyDB

- BerkeleyDB let me:
  - Create a database per genome, independent from a single database daemon
  - Provision database to cluster for distributed analysis
  - Adapt to key-value database semantics with nice API
- Limitations:
  - Creation on single node only
  - Want to query easily across genomes
  - Database are not distributed
  - I saw performance issues, high I/O wait

HBase SeqWare Query Engine



What HBase DB Looks Like

A Record in my HBase

Database on filesystem (HDFS)

Current Prototyping Work

- Validate creation of U87 (genome resequencing at 20x) genome database
  - SNVs
  - Coverage
  - Annotations
- Test fast querying of record subsets
- Test fast processing of whole DB using MapReduce
- Test stability, fault-tolerance, auto-balancing, and deployment issues along the way

What About Fast Queries?

- I'm fairly convinced I can create a distributed HBase database on a Hadoop cluster
- I have a prototype HBase database running on two nodes
- But HBase shines when bulk processing DB
- Big question is how to make individual lookups fast
- Possible solution is Hbase+Katta for indexes (distributed Lucene)

Resources

- Hbase & Hadoop: http://hadoop.apache.org
- When to use HBase: http://blog.rapleaf.com/dev/?p=26
- NOSQL presentations: http://blog.oskarsson.nu/2009/06/nosql-debrief.html
- Other DBs: CouchDB, Hypertable, Cassandra, Project Voldemort, and more...
- Data mining tools: Pig and Hive
- SeqWare: http://seqware.sourceforge.net

Acknowledgments

- Zugen Chen
- Michael Clark
- Ascia Eskin
- Bret Harry
- Nils Homer
- Stanley Nelson
- GMOD!
- Hane Lee
- Jordan Mendler
- Barry Merriman
- Dmitriy Skvortsov
- Colin Flinders
- John Klejnot
- Felipe Cervantes
- Kevin McKernan (ABI)

SeqWare: http://seqware.sf.net

boconnor@ucla.edu

Extra Slides

File Sizes for 20x Genome

- U87
  - 150GB for SRF file (sequence and quality)
  - 150GB for BAM (alignment)
  - 50GB for pileup (variant calls)
  - 30GB for BekeleyDB (variant calls, coverage, annotation, but no alignments)
  - Total = ~400GB per genome
- 2000 genomes x 400GB/genome = 800TB
- 2000 genomes x 30GB/genome = 60TB

File Sizes for ?x Genome

- HuRef
  - ?GB for SRF file (sequence and quality)
  - ?GB for BAM (alignment)
  - ?GB for pileup (variant calls)
  - 140GB for BekeleyDB (variant calls, coverage, annotation, but no alignments)
  - Total = ~?GB per genome
- 2000 genomes x ?GB/genome = ?TB
- 2000 genomes x 140GB/genome = 280TB

Would 2,000 Genomes Kill SQL?

- Say each genome has 5M variants (not counting coverage!)
- 5M variant rows x 2,000 genomes = 10 billion rows
- Our DB server running PostgreSQL (2xquad core, 64GB RAM) died with the Celsius (Chado) schema after loading ~1 billion rows
- So maybe conservatively we would have issues with 150+ genomes
- That threshold is probably 1 year away with public datasets available via SRA, 1000 genomes, TCGA

HBase Under the Hood

- Slide from architecture