# Scientific Computing (PHYS 2109/Ast 3100 H)
# I. Scientific Software Development

SciNet HPC Consortium
University of Toronto

Winter 2014

# Lecture 8

# Data Management

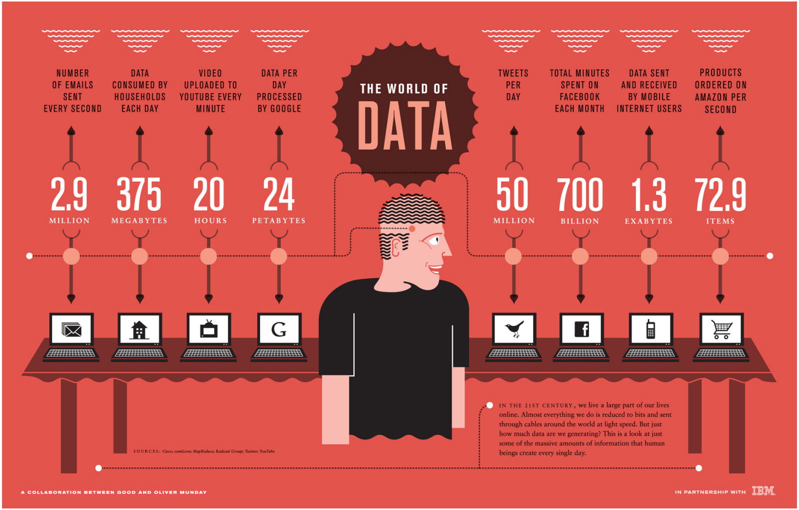
## To much of a good thing?

- Increase in computing power makes simulations larger/more frequent
- Increase in sensor technology makes experiments/observations larger
  - Large Hadron: $\sim$ 50-100 PB to date (4 years)
  - Square Kilometer Array: $\sim$ 1 EB /day !
- Data sizes that used to be measured in MB/GB now measured in TB/PB.
- Easier to make big data than to do something useful with it!
- Data access is the now the bottleneck.

# Big Data

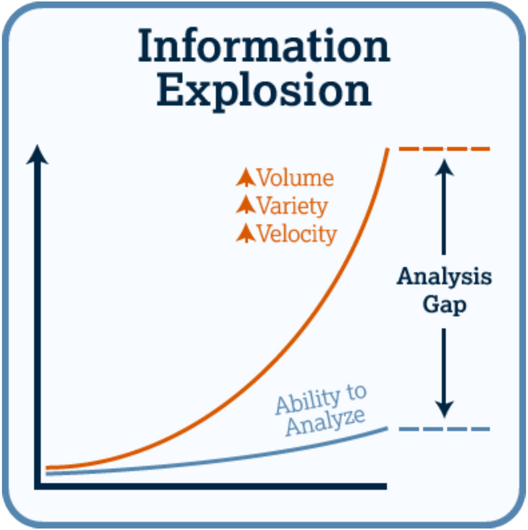

Ben Chams - Fotolia

# Big Data



THE WORLD OF
DATA
NUMBER OF EMAILS SENT EVERY SECOND
2.9 MILLION
DATA CONSUMED BY HOUSEHOLDS EACH DAY
375 MEGABYTES
VIDEO UPLOADED TO YOUTUBE EVERY MINUTE
20 HOURS
DATA PER DAY PROCESSED BY GOOGLE
24 PETABYTES
TWEETS PER DAY
50 MILLION
TOTAL MINUTES SPENT ON FACEBOOK EACH MONTH
700 BILLION
DATA SENT AND RECEIVED BY MOBILE INTERNET USERS
1.3 EXABYTES
PRODUCTS ORDERED ON AMAZON PER SECOND
72.9 ITEMS
IN THE 21ST CENTURY, we live a large part of our lives online. Almost everything we do is reduced to bits and sent through cables around the world at light speed. But just how much data are we generating? This is a look at just some of the massive amounts of information that human beings create every single day.
SOURCES: Cisco; comScore; MapReduce; Radicati Group; Twitter; YouTube
A COLLABORATION BETWEEN GOOD AND OLIVER MUNDAY
IN PARTNERSHIP WITH IBM

# Big Data

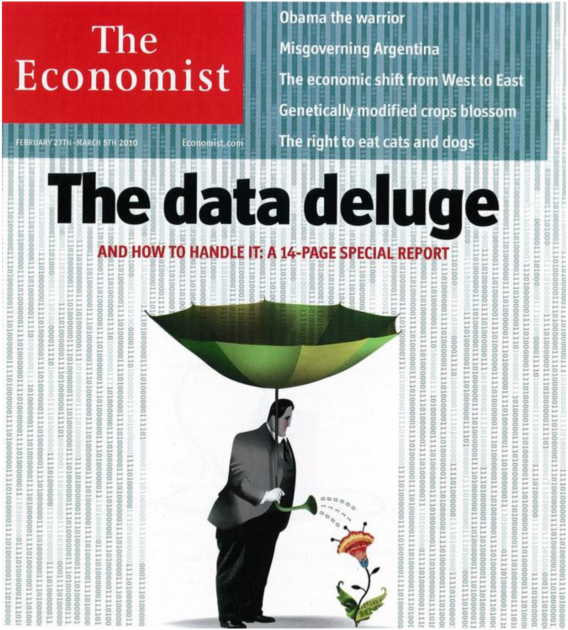


Information Explosion
Volume
Variety
Velocity
Analysis Gap
Ability to Analyze

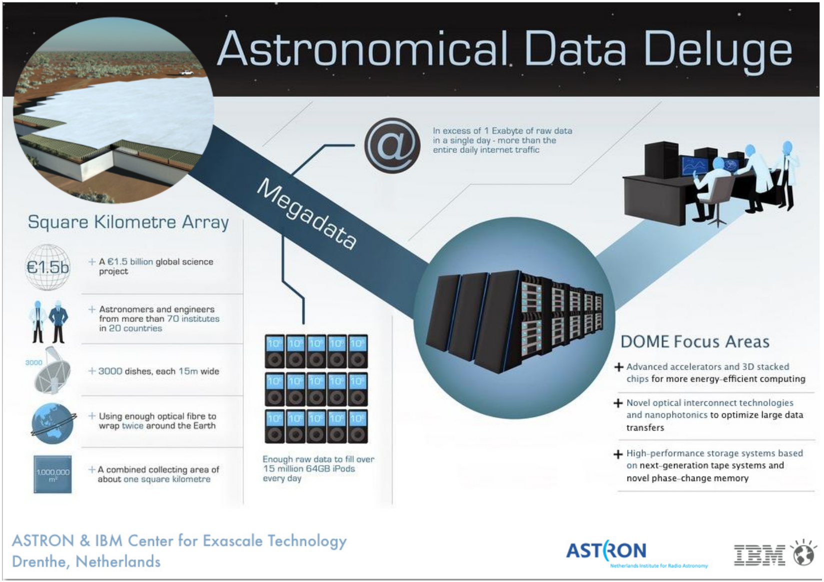
# Big Data

COMMUNICATIONS
OF THE ACM
CACM.ACM.ORG
12/08 VOL.51 NO.12
Surviving the Data Deluge
Open Information Extraction from the Web
CTOs on Virtualization
Living Machines
High-Performance Web Sites

# Big Data


The Economist
Obama the warrior
Misgoverning Argentina
The economic shift from West to East
Genetically modified crops blossom
The right to eat cats and dogs
FEBRUARY 27TH–MARCH 5TH 2010
Economist.com
The data deluge
AND HOW TO HANDLE IT: A 14-PAGE SPECIAL REPORT

# Big Data



Astronomical Data Deluge
Megadata
In excess of 1 Exabyte of raw data in a single day - more than the entire daily internet traffic
Square Kilometre Array
€1.5b
A €1.5 billion global science project
Astronomers and engineers from more than 70 institutes in 20 countries
3000
3000 dishes, each 15m wide
Using enough optical fibre to wrap twice around the Earth
1,000,000 m²
A combined collecting area of about one square kilometre
Enough raw data to fill over 15 million 64GB iPods every day
DOME Focus Areas
Advanced accelerators and 3D stacked chips for more energy-efficient computing
Novel optical interconnect technologies and nanophotonics to optimize large data transfers
High-performance storage systems based on next-generation tape systems and novel phase-change memory
ASTRON & IBM Center for Exascale Technology
Drenthe, Netherlands
ASTRON
Netherlands Institute for Radio Astronomy
IBM

# Data

## Things to think about

- Big is Relative
  - Too Big to Fit in Memory (16-256 GB today)
  - Too Big to Fit on Disk (1-100 TB today)
- Plan for Data Analysis
  - Don't just save everything.
  - On the fly analysis, post-processing automation.
  - Is it worth storing or just recomputing?

# Disk I/O

## Common Uses

- Checkpoint/Restart Files
- Data Analysis
- Data Organization
- Time accurate and/or Optimization Runs
- Batch and Data processing
- Database

# Disk I/O

## Common Bottlenecks

- Mechanical disks are slow!
- System call overhead (open, close, read, write)
- Shared file system (nfs, lustre, gpfs, etc)
- HPC systems typically designed for high bandwidth (GB/s) not IOPs
- Uncoordinated independent accesses

# Disk Access Rates over Time

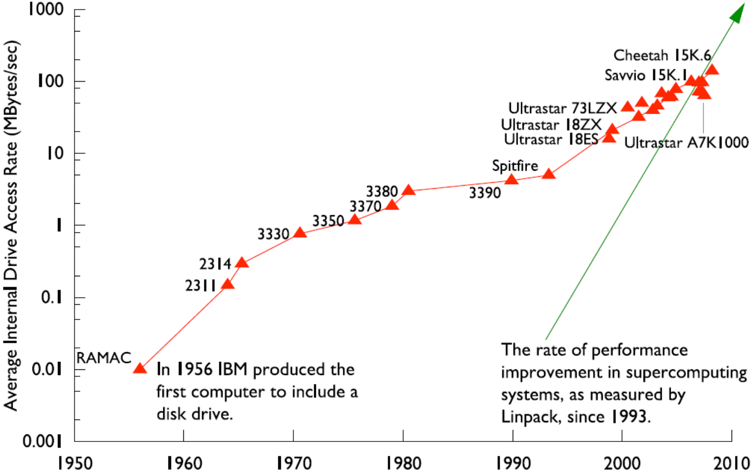



Figure by R. Ross, Argonne National Laboratory, CScADS09

# Memory/Storage Latency

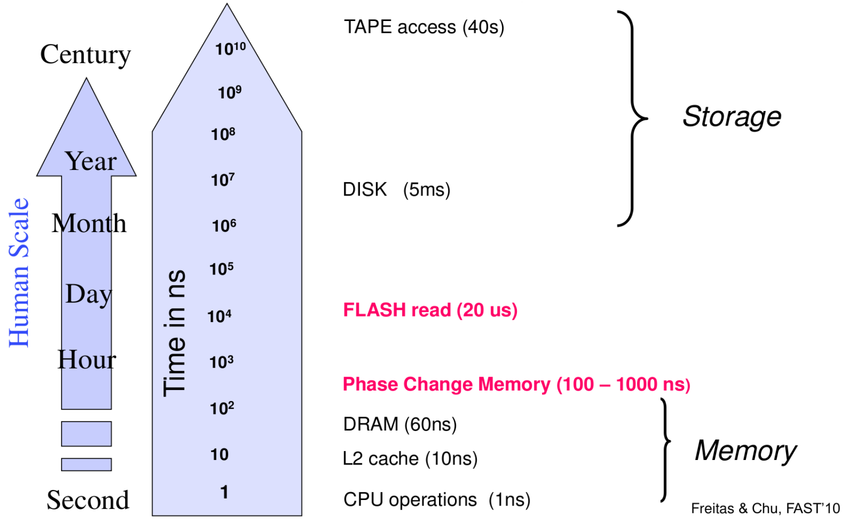



Figure by R. Freitas and L Chiu, IBM Almaden Labs, FAST'10

# Definitions

## IOPs

Input/Output Operations Per Second (read,write,open,close,seek)

## I/O Bandwidth

Quantity you read/write (think network bandwidth)

## Comparisons

| Device | Bandwidth (MB/s) | per-node | IOPs | per-node |
|---|---|---|---|---|
| SATA HDD | 100 | 100 | 100 | 100 |
| SSD HDD | 250 | 250 | 4000 | 4000 |
| SciNet | 5000 | 1.25 | 30000 | 7.5 |

# Storege Formats

## Formats

- ASCII
- Binary
- MetaData (XML)
- Databases
- Standard Library's (HDF5,NetCDF)

# ASCII

## American Standard Code for Information Interchange

Pros

- Human Readable
- Portable (architecture independent)

Cons

- Inefficient Storage
- Precision
- Expensive for Read/Write (conversions)

# Native Binary

100100100

Pros

- Efficient Storage (256 x floats @4bytes takes 1024 bytes)
- Efficient Read/Write (native)

Cons

- Have to know the format to read
- Portability (Endianness)

# ASCII vs. binary

## Writing 128M doubles

| Format | /scratch (GPCS) | /dev/shm (RAM) | /tmp (disk) |
|---|---|---|---|
| ASCII | 173s | 174s | 260s |
| Binary | 6s | 1s | 20s |

## Syntax

| Format | C/C++ | FORTRAN |
|---|---|---|
| ASCII | **`fprintf()`** | open(6,file='test',form='formatted') |
| | **`file <<`** | write(6,*) |
| Binary | **`fwrite()`** | open(6,file='test',form='unformatted') |
| | **`file.write()`** | write(6) |

# C++ Writing Binary

## Read

```
#include <fstream>
std::ifstream inFile ("data.in", std::ifstream::binary );
std::ifstream& read(char *, int );
```

## Write

```
#include <fstream>
std::ofstream outFile ("data.out", std::ofstream::binary );
std::ofstream& write(const char *, int n);
```

# C++ Writing Binary

```
#include <fstream>
int main() {
  int num=100; char a='t';
  char *obuffer = new char [num];
  char *ibuffer = new char [num];
  for (int i=0; i<num; i++) obuffer[i]=a;
  //--------- write to outfile -----------
  std::ofstream outfile ("file.bin", std::ofstream::binary);
  outfile.write (obuffer,num);
  outfile.close();
  //--------- read infile ----------------
  std::ifstream infile ("file.bin", std::ifstream::binary);
  infile.read (ibuffer,num);
  infile.close();
  delete [] ibuffer, obuffer;
  return 0;
}
```

# C++ Writing Binary

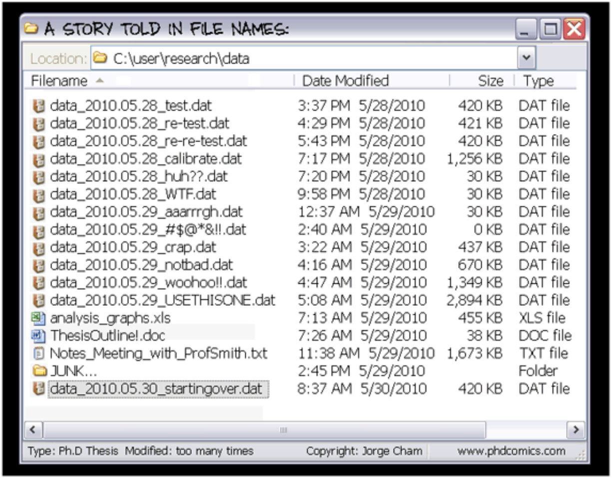
```
#include <fstream>
int main() {
  int num=100; double a=44.0;
  double *obuffer = new double [num];
  double *ibuffer = new double [num];
  for (int i=0; i<num; i++) obuffer[i]=a;
  //--------- write to outfile -----------
  std::ofstream outfile ("file.bin", std::ofstream::binary);
  outfile.write ((char *)obuffer,num*sizeof(double ));
  outfile.close();
  //--------- read infile ----------------
  std::ifstream infile ("file.bin", std::ifstream::binary);
  infile.read ((char *)ibuffer,num*sizeof(double ));
  infile.close();
  delete [] ibuffer, obuffer;
  return 0;
}
```

# Data Management

## File(s)

- Human-interpretable filenames lose their charm after few dozen files (or even after a few months pass)...
- Need to avoid thousands of files in a flat directory.
- A few big files are more efficient that many little ones.
- Keep parallel I/O in mind.
- Rigorously maintained metadata becomes essential.
- Possibly use a database or version control (i.e. git-annex).

# Data Management




http://www.phdcomics.com/comics/archive.php?comicid=1323

# Metadata

## What is Metadata?

Data about Data

- File System: size, location, date, owner, etc.
- App Data: File format, version, iteration, etc.

## Example: XML

```
<?xml version="1.0" encoding="UTF-8" ?>
<slice_data>
  <format>UTF1000</format>
  <verstion>6.8</version>
  <img src="slice1_2010.img" alt='Slice 1 of Data'/>
  <date> January 15th, 2010 </date>
  <loc> 47 23.516 -122 02.625 </loc>
</slice_data>
```

# "Standard" Formats

## File Formats

- CGNS (CFD General Notation System)
- IGES/STEP (CAD Geometry)
- HDF5 (Hierarchical Data Format)
- NetCDF (Network Common Data Format)
- disciplineX version

## Benefits

- Most provided with as libraries.
- Self Describing (imbedded metadata).
- Many are binary agnostic, so portable.
- Many support Parallel I/O and native FS support.
- Broader tool support (visualization, etc.)

# Databases

## Beyond flat files

- Very powerful and flexible storage approach
- Data organization and analysis can be greatly simplified
- Enhanced performance over seek/sort depending on usage
- Open Source Software
  - SQLite (serverless)
  - PostgreSQL
  - mySQL
  - mongoDB (NoSQL)

# Data Management Summary

## Summary

- Have a Plan for Data.
- Automate and reduce/post process on the fly.
- Start small and plan for scalability.