Big Data Analytics Introduction

Yoav Freund

UCSD / Computer Science and Engineering

Coordinates

- Class meets tue, thu, 3:00 – 4:50
- Demonstrations and homework will be based on **iPython notebooks**.
- Hadoop and Spark clusters will run on AWS (later on in the course).
- Class Web Page:
  - http://cseweb.ucsd.edu/classes/sp15/cse255-a
  - Or (different name for same place)
  - http://seed.ucsd.edu/mediawiki/index.php/BigDataAnalytics2015

Teaching staff

- Instructor: Prof. Freund
  - Office hours: Friday 1-3pm
  - Computer science building, office #4126

- TA: Dev Agarwal
  - Office hours: Wed. 11am 1pm
  - Computer Science Bulding, Lab #4223

Piazza

- Piazza will be our hub for communication.
- Register here: piazza.com/ucsd/spring2015/cse255
- Use Piazza to ask questions and to answer them.
- TAs and me will monitor Piazza and:
  - Answer questions
  - Evaluate answers
  - Make announcements.
  - Announcements that are relevant for all time will be “pinned”.
- The board is shared between MAS-DSE and CSE255. Increased traffic tends to help. If not: I will split them.
- Not a place to co-miserate. Try to be helpful to your fellow students.

Getting iPython

- Preferred method:
  - install on your personal computer:
  - Ubuntu and Mac OS are supported.
  - Otherwise you are on your own.
- Other method:
  - I can give you tools and permissions to run an iPython Notebook server on AWS.

Home work and grading

- Ten hours a week (in addition to class time)
- There will be a homework every two weeks. (4 in total).
- The homework is to be submitted as an iPython notebook.
- Each Homework will be 12.5% of the grade, the final will be 50% of the grade.
- Homeworks are for individuals, not groups.

GitHub

- Notebooks will be distributed using GitHub.
- Learn how to use git.
- Clone this repository onto your computer:
  - https://github.com/yoavfreund/UCSD_BigData_2015
- Experiment, make changes, and do your homework on your private repository.
- Recommended: create an account on GitHub and fork the repository.
- If you use Github for your homework – be sure that it is a private repository. Students can get free private repositories.

Get the course repository

- Get git:
  - Fedora: `yum install git-core`
  - Ubuntu: `apt-get install git`
  - Mac (macports):
    ```
    sudo port install git-core +svn +doc
    +bash_completion +gitweb
    ```
- Get the course git:
  - `git clone`
    https://github.com/yoavfreund/UCSD_BigData_2015.git

BIG DATA ANALYTICS

What is big data?

- Giga byte? Tera byte? Peta Byte? …
- Actual size depends on technology.
- The real bottleneck is communication, not storage.

Name	Equal to:	Size in Bytes
Bit	1 bit	1/8
Nibble	4 bits	1/2 (rare)
Byte	8 bits	1
Kilobyte	1,024 bytes	1,024
Megabyte	1,024 kilobytes	1,048,576
Gigabyte	1,024 megabytes	1,073,741,824
Terrabyte	1,024 gigabytes	1,099,511,627,776
Petabyte	1,024 terrabytes	1,125,899,906,842,624
Exabyte	1,024 petabytes	1,152,921,504,606,846,976
Zettabyte	1,024 exabytes	1,180,591,620,717,411,303,424
Yottabyte	1,024 zettabytes	1,208,925,819,614,629,174,706,176

Then there is the hypothetical "Googolbyte" which would be a number of bytes equal to a 10 followed by 100 zeroes.

Name	Example(s) of Size
Byte	A single letter, like "A."
Kilobyte	A 14-line e-mail. A pretty lengthy paragraph of text.
Megabyte	A good sized novel. Shelley's "Frankenstein" is only about four-fifths of a megabyte.
Gigabyte	The multi-player version of Diablo II, installed. About 300 MP3s. About 40 minutes of video at DVD quality (this varies, depending on maker). A CD holds about three-fourths of a gigabyte.
Terrabyte	About thirty and a half weeks worth of high-quality audio. Statistically, the average person has spoken about this much by age 25.
Petabyte	The amount of data available on the web in the year 2000 is thought to occupy 8 petabytes (theorized by Roy Williams).
Exabyte	In a world with a population of 3 billion, all information generated anually in any form would occupy a single exabyte. Supposedly, everything ever said by everyone who is or has lived on the planet Earth would take up 5 exabytes.
Zettabyte	Three hundred trillion MP3s; Two hundred billion DVDs. If every person living in the year 2000 had had a 180 gigabyte hard drive filled completely with data, all the data on all those drives would occupy 1 zettabyte.
Yottabyte	???

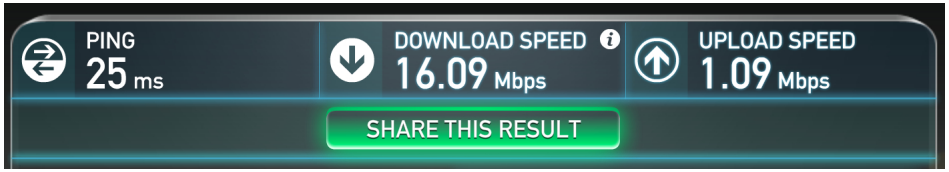
Moving Data Takes Time

- Seagate :
  - 4TB capacity
  - Retail price: $270.00
  - Up to 0.22GB/sec transfer rate (realistically .05-.1GB/sec).
- Time to copy disk1 to disk2:
  - 4,000GB/(0.22GB/sec) 5-20 hours.
- Using Solid State Disks: around 0.5GB/sec
  - Faster but much more expensive.
  - Transfer would take at least 2.2 hours
- For More info see:
  - http://www.tomshardware.com/charts/hard-drives-and-ssds,3.html

What is the speed of the internet?

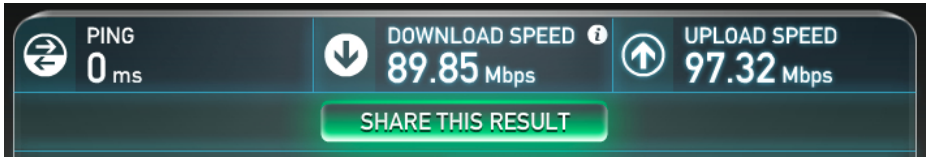
- From the site www.speedtest.net
- Measuring from home (Time Werner Cable)

2014

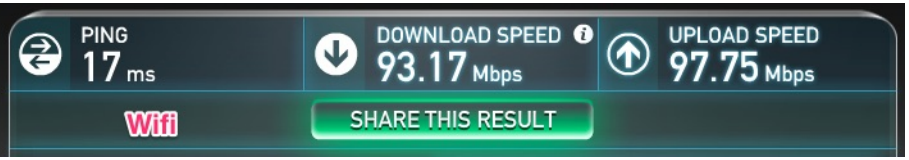


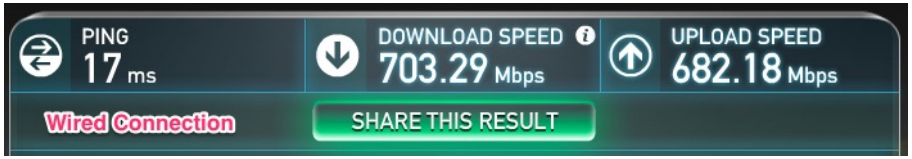
- Measuring from UCSD office:

2014



2015





The internet is slow and expensive

By Jose Vilches
On April 24, 2013, 11:30 A

Download speeds for residential lines

		Avg. Mbps	Change	Change
–	Global	2.9	5.0%	25%
1	South Korea	14.0	-4.8%	-13%
2	Japan	10.8	2.7%	19%
3	Hong Kong	9.3	3.4%	5.4%
4	Latvia	8.9	2.3%	20%
5	Switzerland	8.7	0.5%	20%
6	Netherlands	8.6	0.1%	3.3%
7	Czech Republic	8.1	7.0%	21%
8	United States	7.4	2.3%	28%
9	Sweden	7.3	7.4%	29%

Downloading 4Terabyte (4TB) At 10Mb/Sec takes about 40 days.

Residential: Uploading is typically 10 times slower.

Cable fluctuations high and reliability low

The price of communication speed

	Speed(download) (Mega Bits Per Second)	Price/month	Time to download 4TByte
Res-Cable	0-20 Mbps (variable)	$20 - $100	> 20 days
T1	1.544 Mbps (guaranteed)	$250 - $500	240 days
T3	43.232 Mbps	$4,000 - $16,000	8.6 days
OC-3	155 Mbps	$20,000 - $45,000	2.4 days

- Sources:
- http://www.infobahn.com/research-information.htm
- http://t1service.homestead.com/Cable_DSL_T1.html

Higher speeds exist (optical fiber) but prices are not public

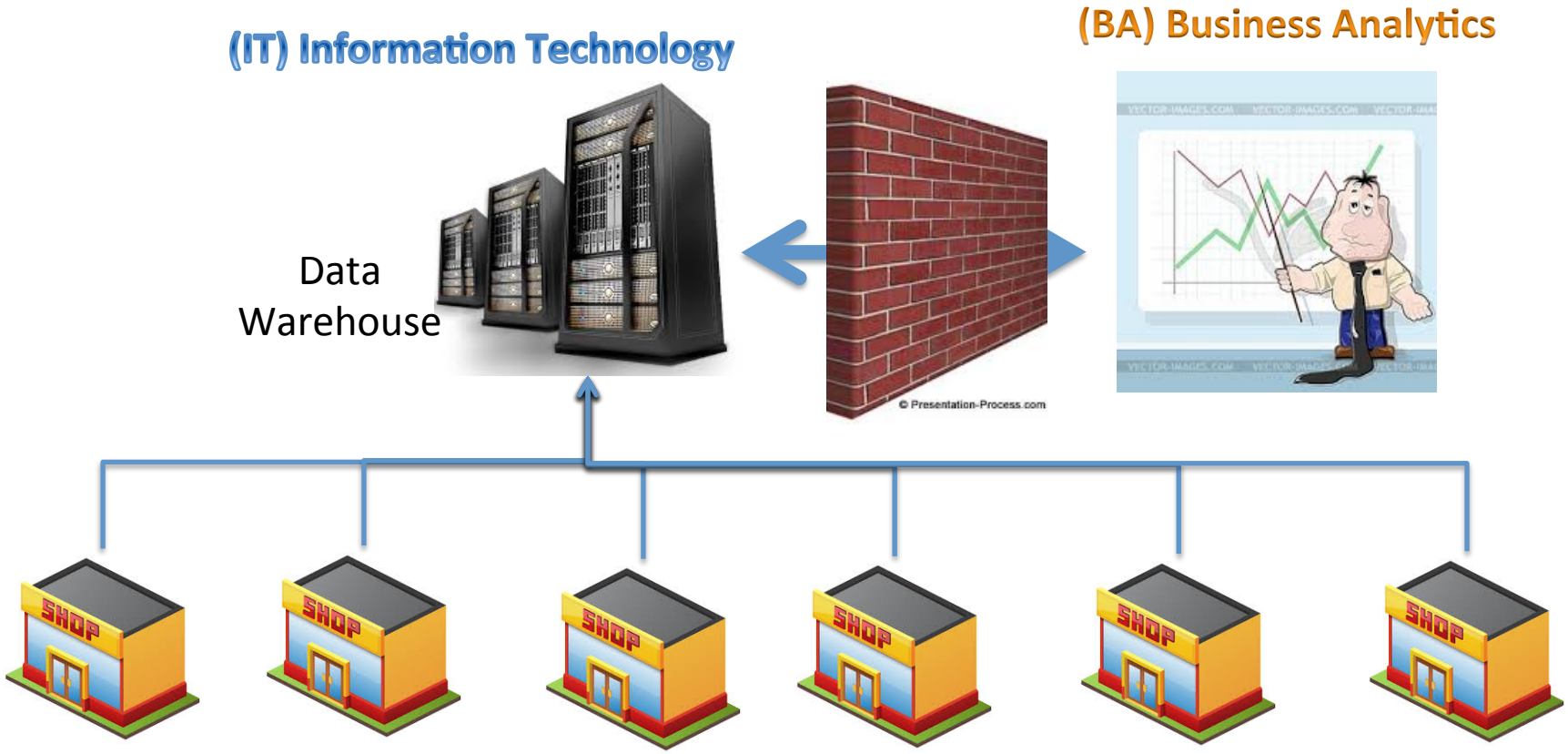
When Fedex is better than the internet.

- Suppose you want to transfer 4TB from San Diego to LA.
  - Regular line it will take 2-3 weeks.
  - Dedicated OC3 line 57 hours, $20,000/month
  - Fedex: < 24 hours, < $100.
    - We still need to read the disk when it is there…
  - Cloud computing companies offer services by which you upload data by mailing them the disk.

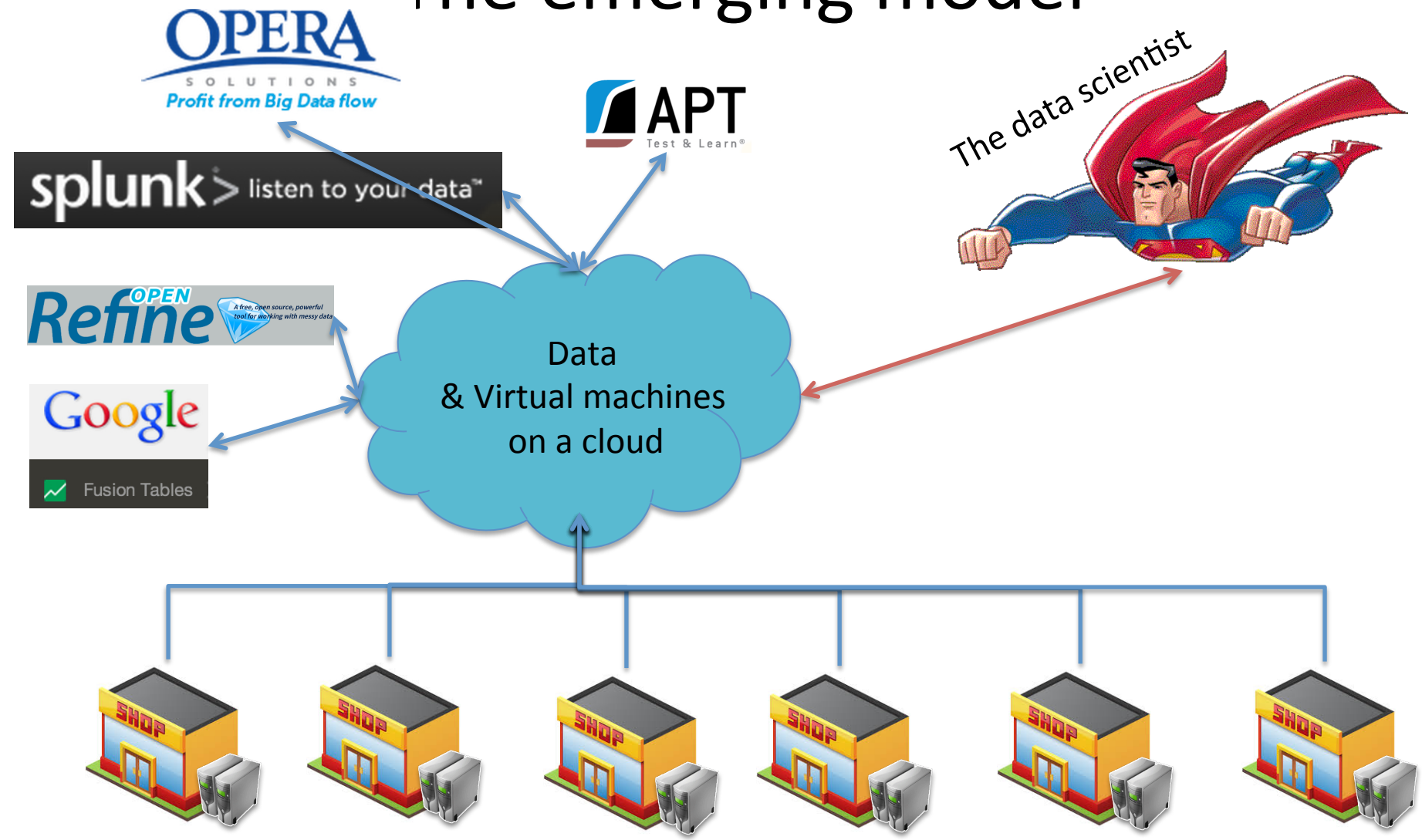
What is analytics?

- Larger and larger data feeds are available.
- The goal of analytics is to understand and model the system behind the data.

Analytics in a distributed retail chain
The traditional model



Analytics in a distributed retail chain
The emerging model



The power of predictive analytics

AMAZON PATENTED AN ANTICIPATORY SHIPPING SYSTEM THAT PREDICTS ORDERS

By **Mike Flacy** — January 18, 2014

Noticed by the Wall Street Journal, Amazon recently patented a new system that will help the retailer create predictive models to accurately forecast where an item will need to ship. Calling the system "anticipatory shipping," Amazon will collectively compile data such as product searches, page visits, wish list items, order history, overall time on page, items left in the shopping cart and return history to pre-ship items to closer warehouse locations or even directly to the eventual recipient. Amazon even plans to measure the length of time that a user's mouse cursor hovers over an item in order to predict an upcoming purchase.

- Actual patent: shipping a package without a final destination.
- This method can only work when there are many identical orders from one location/ city. Final address changed at UPS/USPS location.
- Supply chain management is a long standing practice, amazon is bringing it to the next level.
- Supply chain management is a prime big data application.

A big data problem

- Suppose we work for a chain with 10,000 retail stores.
- We group buyers by age / gender / item /…
- Grouping known ahead of time
- We want to know how much each group is spending at any time.
- Simple solution: send all of the purchase information to a center, and compute all there.
- Bad:
  - High communication bandwidth.
  - High computation load on the center.

Can we do better?

- Push the computation to the stores, send sums to the center.
- Yes, but how often should the store push the sum to the center?
- If update after each purchase – no savings in communication.
- Can we do better if we allow approximation?

Predictive distributed computation

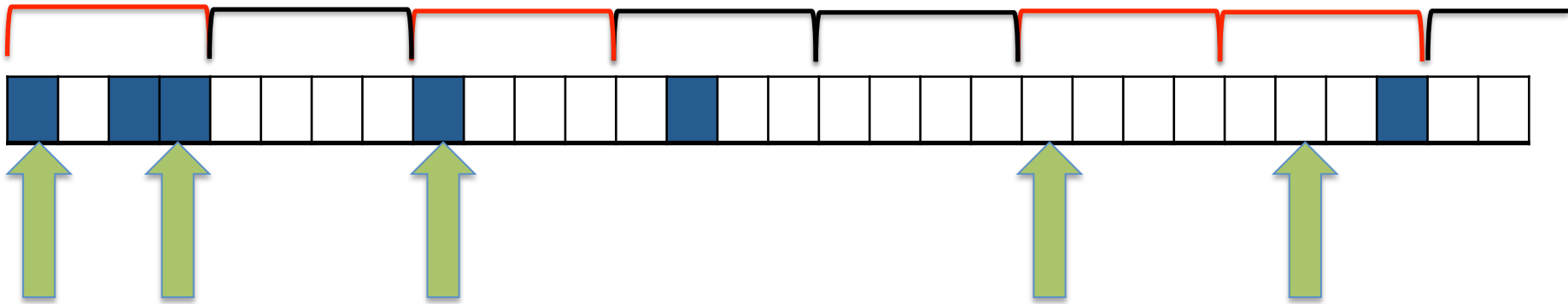
- Suppose we have, for each store, a **statistical model** for predicting future purchases.
- Model is known to both store and center.
- The computer in the store compares the actual purchases to the predictions and sends an update when the two diverge.
- Only “highly informative” bits are sent.
- Details later in the course.

Simple approach: count everything

- Read through the whole file and count all purchases larger than $1.
- Will take an hour (I/O limited)

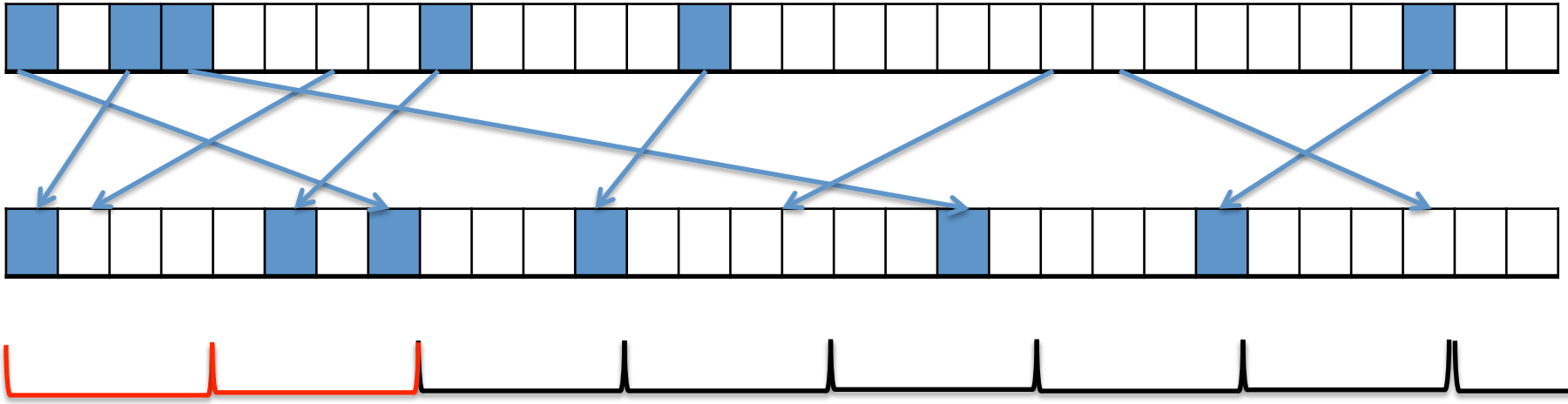
Random Sampling

- Select 1,000,000 entries at random and count the number of of >$1 purchases in selected location.
- Yields approximation of true number within ±0.1%. **Regardless of file size.**
- Takes about 1 minute.
- Reading a disk block at a random location takes about 10msec.
- In most blocks we use only one entry.
- We can count all entries each sampled blocks, but that will not improve accuracy if >$1 are clumped.



Permutation

- Pre-processing: randomly permute all the records in the file.
- Equivalent to sorting on a random key. Map-Reduce is based on efficient distributed sorting. (more on map-reduce in the next slide)
- Reading a consecutive block of 1000 records is statistically equivalent to sampling 1000 records – which requires reading 1000 blocks.
- Computation of fraction of >$1 purchases takes a fraction of a second.
- Not worth preprocessing price if just one query.
- Histogram of purchase amounts requires many queries.



Stratified sampling

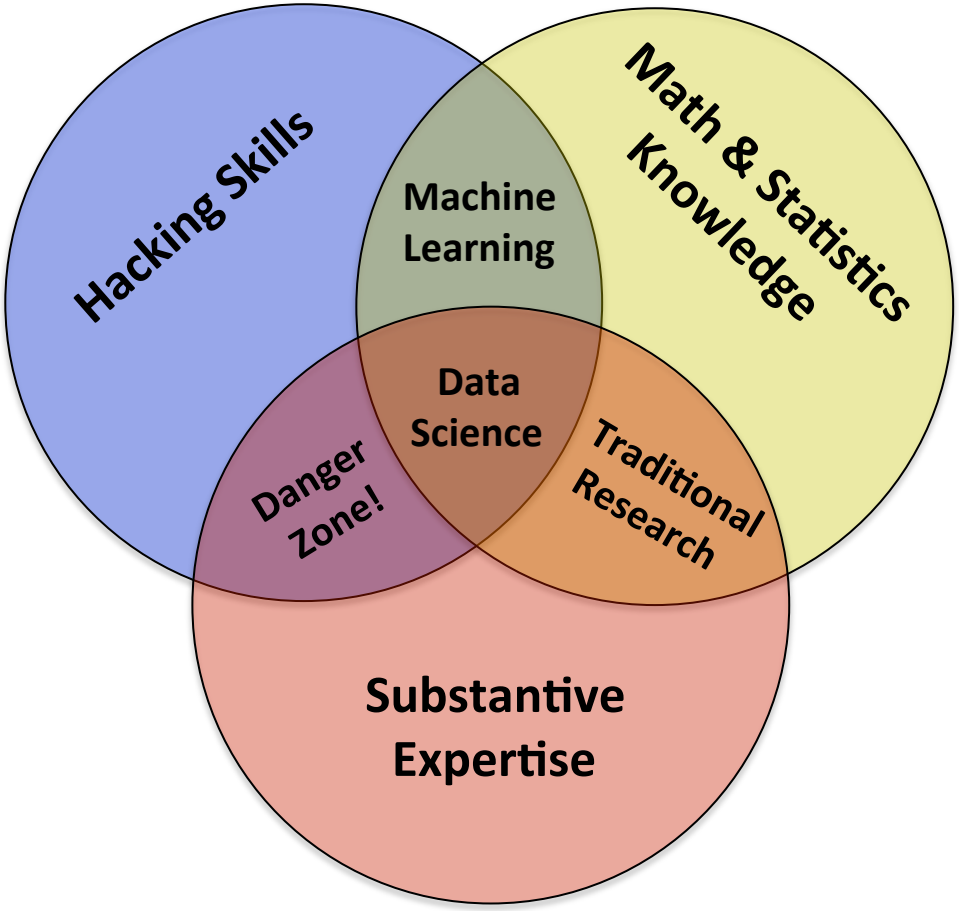
- We can do even better than permutation.
- Partition data into “strata” (Example: Age of customer)
- Condition: the fraction of positive examples in each strata is close to zero or to one.
- Randomly permute each strata separately.
- If condition holds: Number of samples needed to get given accuracy can be significantly decreased.

The system side: GFS and Map-Reduce

- Even when all the data is collected in a data warehouse, communication between computers remains the main cost driver.
- **The Google file system:** Replicate data and distribute it on a large number of small commodity machines connected by commodity communication network.
- **Map-Reduce:** programming paradigm for distributed computation. Hides complexity of system. promotes low communication.
- **HDFS and Hadoop:** the open-source implementation of GFS and map-reduce.

The education of a data scientist

Doing Data Science / Straight Talk from The Frontline / by Rachel Schutt & Cathy O'Neil



What this means for you?

- **Hackers:** you must review and understand the following concepts:
  - Events, Independence, random variable, expectation, variance, covariance, p-value.
  - There will be other concepts throughout the course.
- **Statisticians / physicists / biologists etc**. if most of your programming is in Matlab / R / SAS, then you need to review the following python concepts:
  - PIP Install, import X as Y, from X import M
  - Git, github
  - List comprehension (also dictionary/file)
  - Numpy, matplotlib
  - Iterators, generators, co-routines
  - Threads, processes.
  - Raising can catching exceptions.
  - Classes, class methods, “self.” __init__, inheritence
  - Using python scripts, argparse, if __name__==“__main__”

Substantive Experience

- Diverse application areas:
  - Insurance
  - Supply chain management.
  - Targeted marketing.
  - Smart electric grid.
  - Production
  - Investigative reporting.
  - Hospitals.
  - Education
  - Cyber warfare.
  - Scientific research…
- Cannot expect student or teacher to be/become expert in all areas.
- Alternative: learn how to communicate effectively with domain experts.
- Communication of technical results to non-technical audience is **key**.
- Use “live documents”

Literate Computing

- “Literate Programming” **- Donald Knuth** 1992. – Programs as literature.
- **Fernando Perez**, 2013: “Insight, not numbers: from literate programming to literate computing”. Data analysis as literature.
- Data scientist needs to communicate effectively with programmers, statisticians, and decision makers.
- Documents that combine computer code, interactive figures and textual explanations improves the communication between people with different backgrounds.
- It is insufficient for a data scientist to write code that only a computer can understand. Needs to create a complete document which programmers, statisticians decision makers and computers can all understand.
- Answer: Ipython notebooks.

END OF INTRODUCTION

The AWS/Ipython/Git ecosystem

- **Amazon Web Services:** Cloud-based computation platform that scales.
- **Ipython Notebooks:** An environment for literate computing.
- **MarkDown:** Plain text that can be rendered nicely as a web page or a document.
- **Git and Github:** A peer to peer version control system.

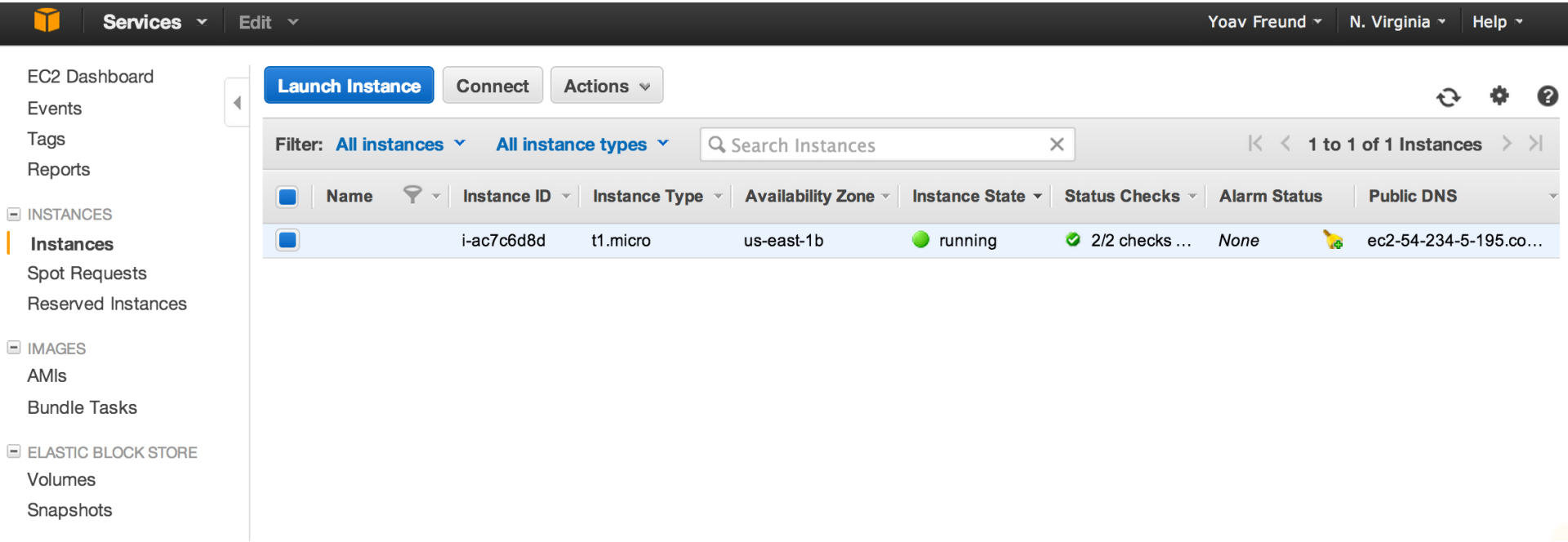
What is AWS?

- Data centers, owned and run by Amazon from which you can rent “virtual computers” on a minute-by-minute basis.
  - AWS > 80% of the cloud computing market
  - ~3,000 people work in AWS – a very big IT team.
- Cost: $0.01-$4.00 per instance per hour.
- Zero up-front investment – cost scales with load.

The parts of AWS we will use:

- EC2 – the compute platform
- S3 – Large-scale storage.
- EMR – hadoop in the cloud.
- Administrative:
  - User accounts and account linking.(set it up for yourself)
  - Key-pairs: security without passwords. (setup for yourself)
  - Security Groups: control what IP's can connect to which ports. (Setup for yourself)
  - AMI's : Disk images that define a complete configuration (Get this from me.).

Console to Amazon/EC2 instances



Create Your own AWS account

- Follow directions on web page:
  - http://seed.ucsd.edu/mediawiki/index.php/Creating_an_AWS_account
- Need to use credit card, but it will not be charge up to $100 (additional funds depend on effective use of resources).
- Establish AWS credentials and launch a trial instance.

AWSCredentials.py

- Enter your AWS Credentials in this file.
- Needs only be done once.

```
### AWS credentials: ####
# Change entries here to match your own #
# Get values for the first two entries from the "Security crediential" Tab
# Under in the menu under your name.
aws_access_key_id='Key ID'
aws_secret_access_key='Key'
# Get the Keypair in the EC2 Dashboard page.
keyPairFile="<~/.ssh/KeyFilename.pem>" # name of file keeping local key
key_name="key name" # name of keypair (not name of file where key is stored)
# Set the security group On the EC2 page (You will need to add IP addresses if
# you want to connect from a place previously unauthorized.
security_groups=['GroupName']
### End of AWS credentials ####
```

Demo LaunchIpythonServer.py

- -h to get help.
- Without parameters – demonstrate ssh
- Output list.
- Launch Ipython Collection
  - 1. Programming
  - 2. MarkDown.
- Kill notebook servers.

Really Quick Intro To Git

Hasan Veldstra
<hasan@hypernumbers.com>

quick poll: Git users, SVN users.

hypernumbers

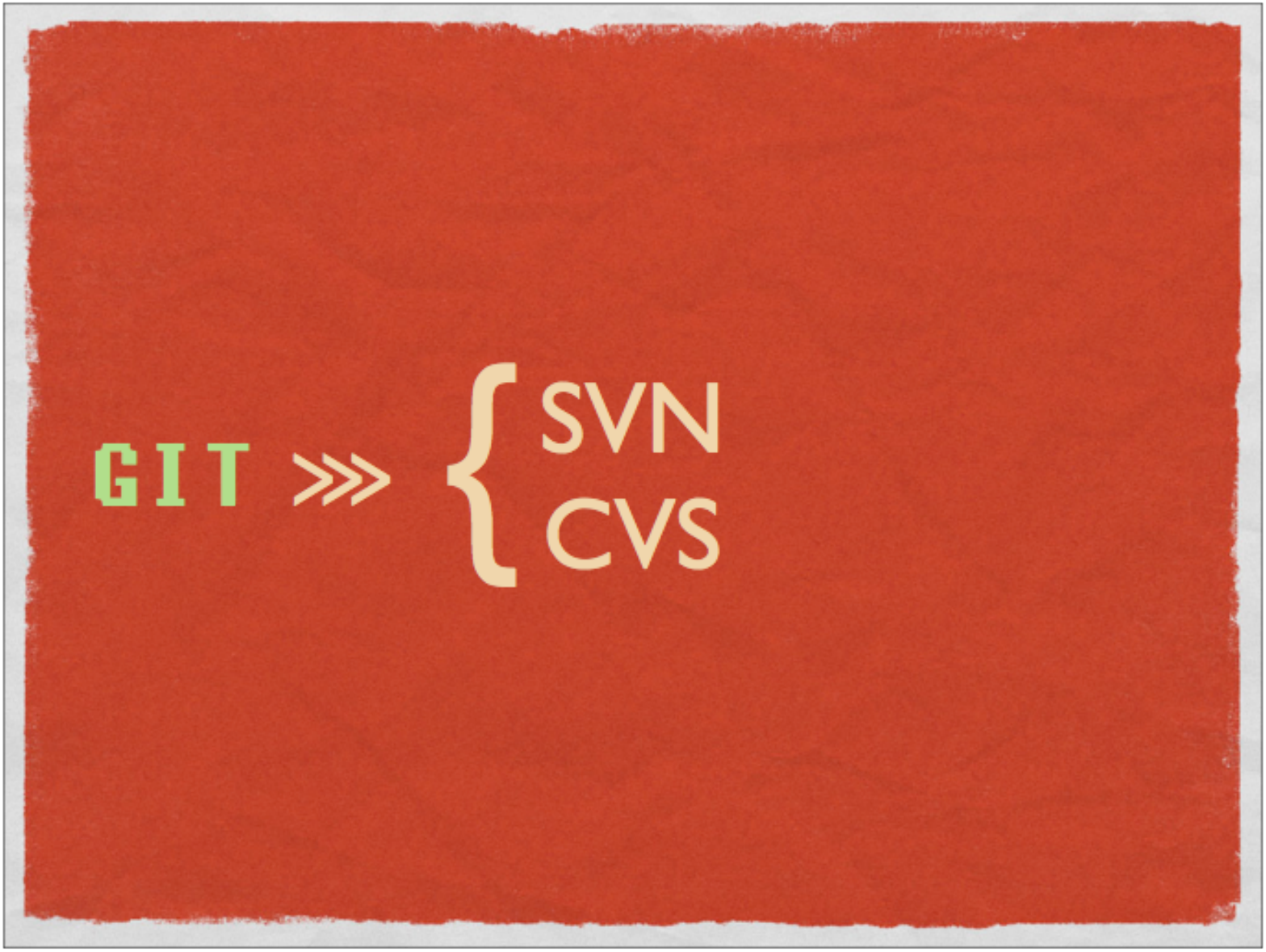
like CVS or SVN, only much better.
fundamental difference – distributed.

source control system

the logo: git

an artistic impression:

SVN
CVS


GIT
SVN
CVS

roadmap, disclaimers &c

why it's cool, how it works, how to get started right now. basic stuff – if you like what you see, there's lots of detailed documentation elsewhere.

i am not an expert, just sharing bits i've learned.

Linux kernel

why Git?

why Git?

- makes stuff cheap
  - any workflow
    - fast
  - small
    - easy to learn

Git has acquired a reputation for being somewhat difficult to get started with and understand but that's lies.

```
$ cd (project directory)
$ git init
$ (create some files)
$ git add .
$ git commit -m 'start the project'
```

distributed

distributed

=

no central repository

distributed

=

no central repository

(superset of SVN &c)

centralized (SVN):

repository

centralized (SVN):

repository

centralized (SVN):

repository

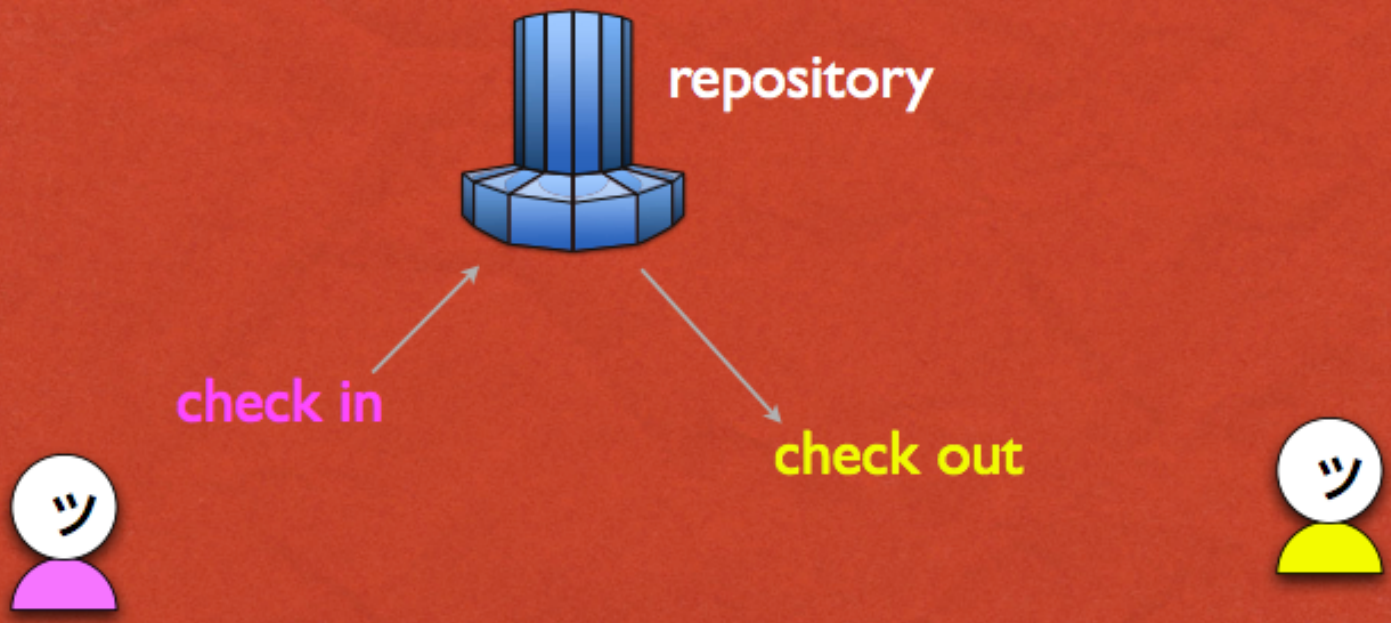
devs “check out” the code to create a “working copy”

centralized (SVN):

repository

check in

centralized (SVN):



centralized (SVN):

repository

check in

check out

make changes

centralized (SVN):

repository

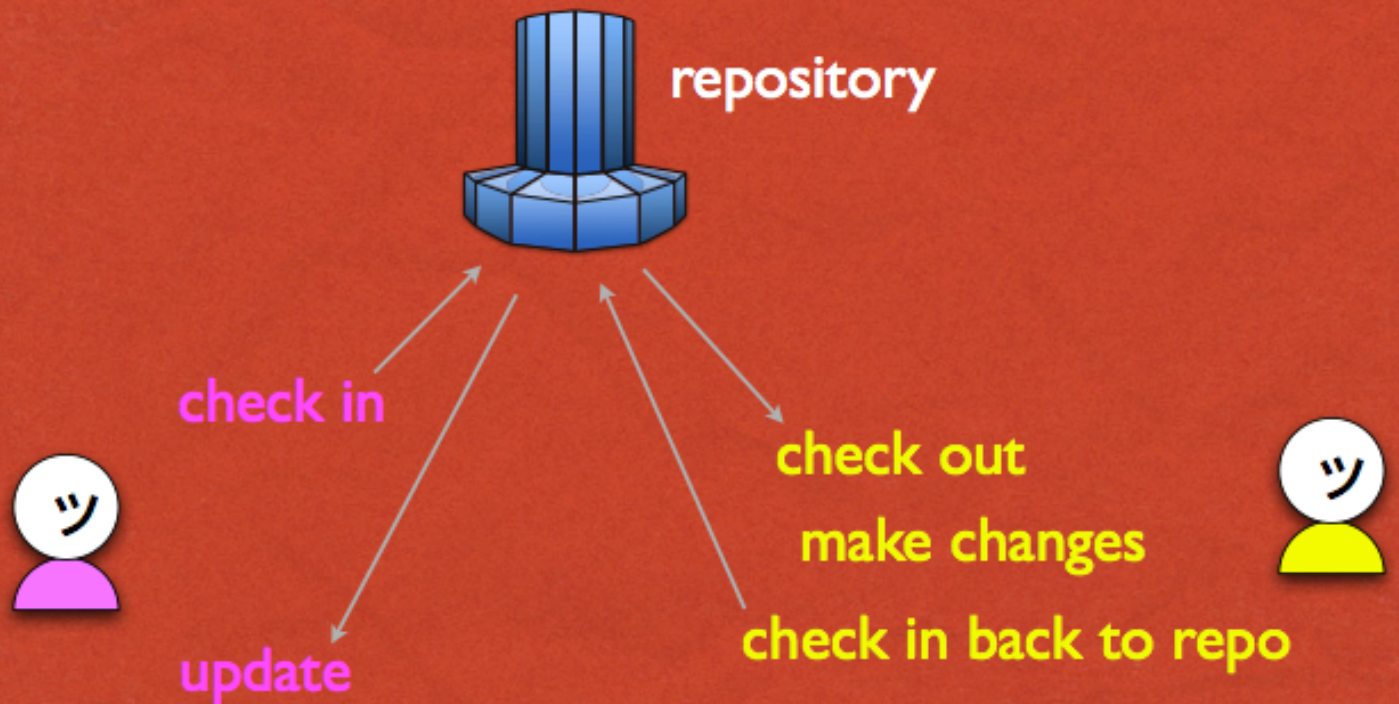
check in

check out

make changes

check in back to repo

centralized (SVN):



centralized (SVN):

repository

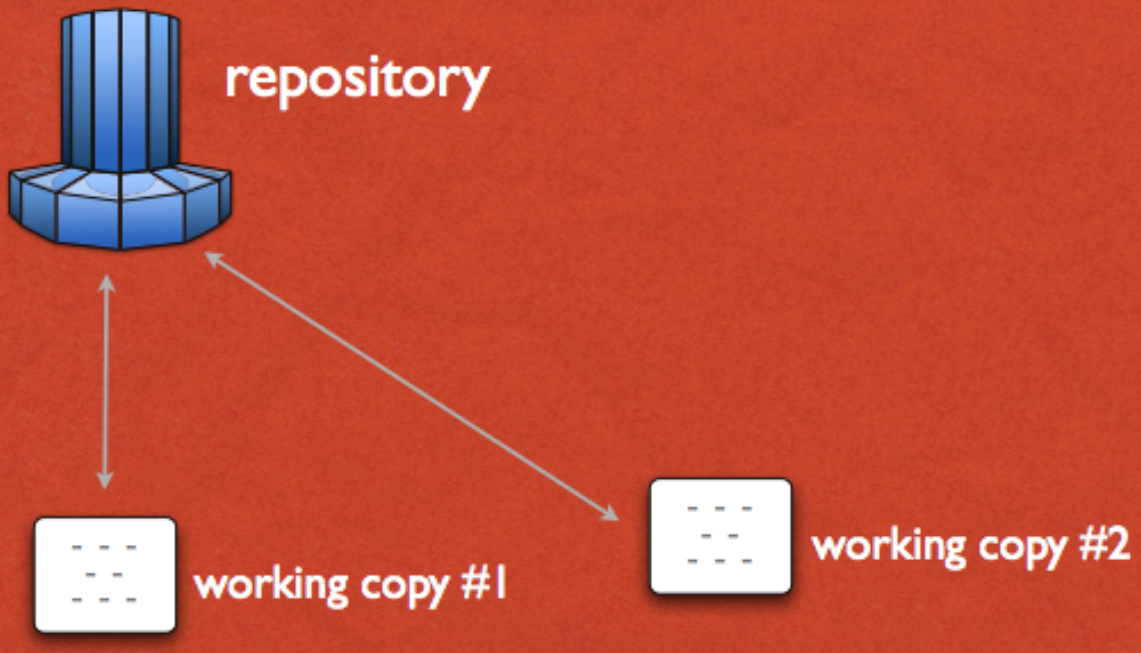
one-to-many relationship

centralized (SVN):

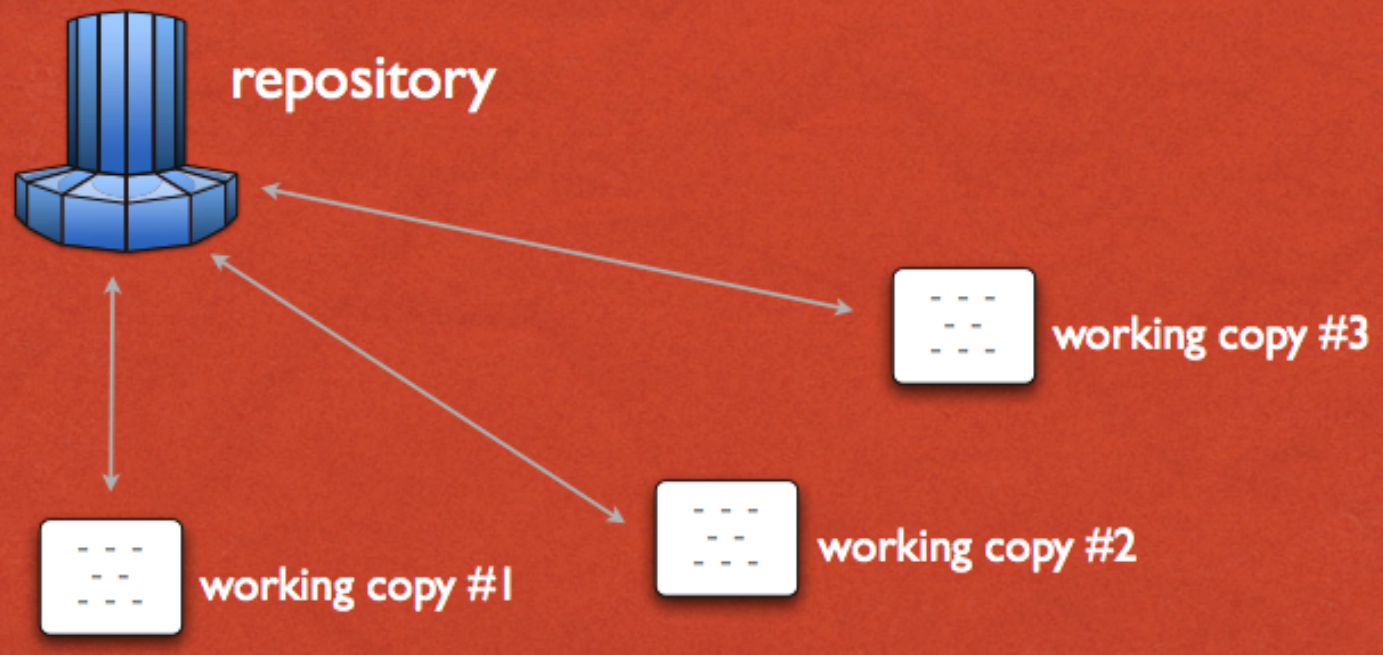
repository

working copy #1

centralized (SVN):



centralized (SVN):



centralized (SVN):

- working copies are inferior

centralized (SVN):

- working copies are inferior
- & can't talk to each other

centralized (SVN):

- working copies are inferior
- & can't talk to each other
- you gotta be online

centralized (SVN):

- working copies are inferior
- & can't talk to each other
  - you gotta be online
- single point of failure

distributed (`Git`):

distributed (`Git`):

~~checkout~~

distributed (`Git`):

~~checkout~~

clone

we don't check out, we clone
clone = full history, can do anything original can

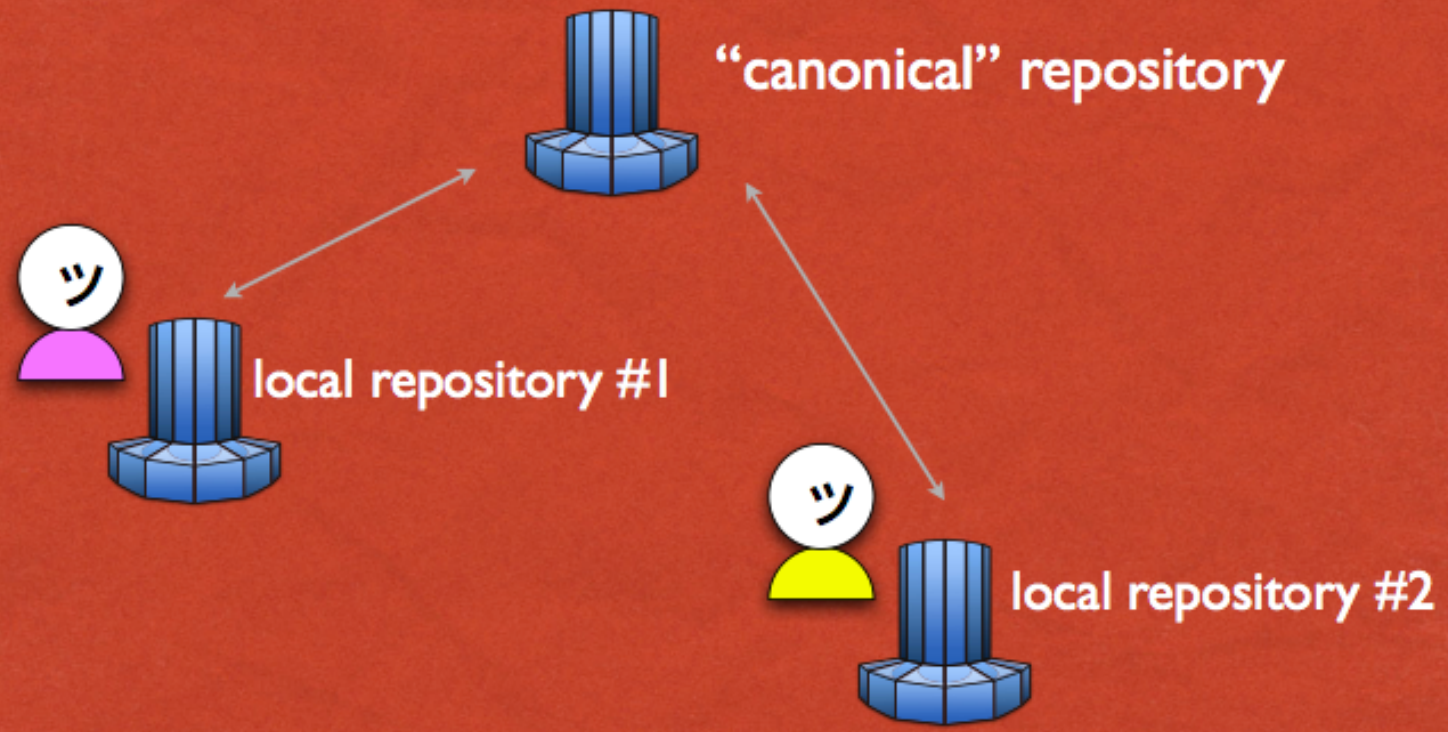
distributed (`Git`):

"canonical" repository

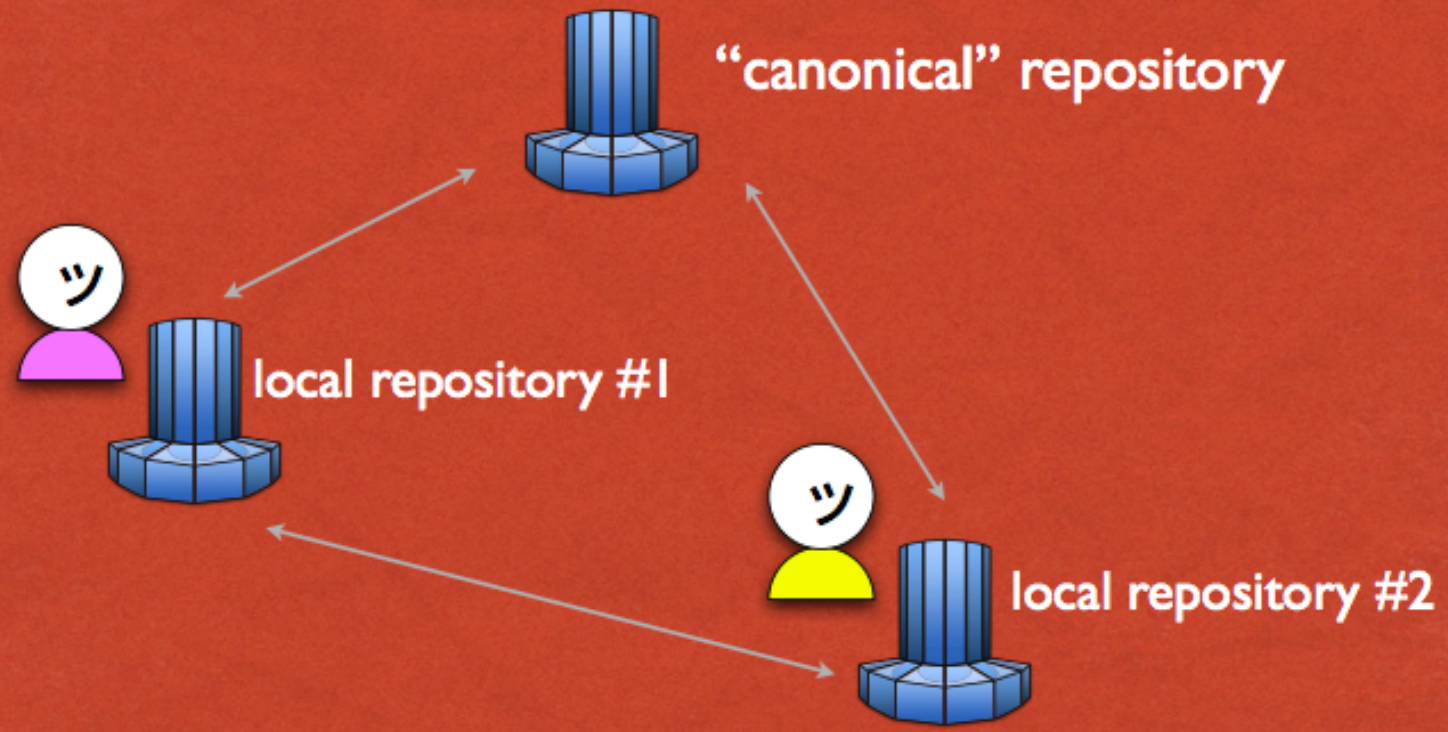
"canonical" because it's a social convention, not a technical requirement

distributed (`Git`):

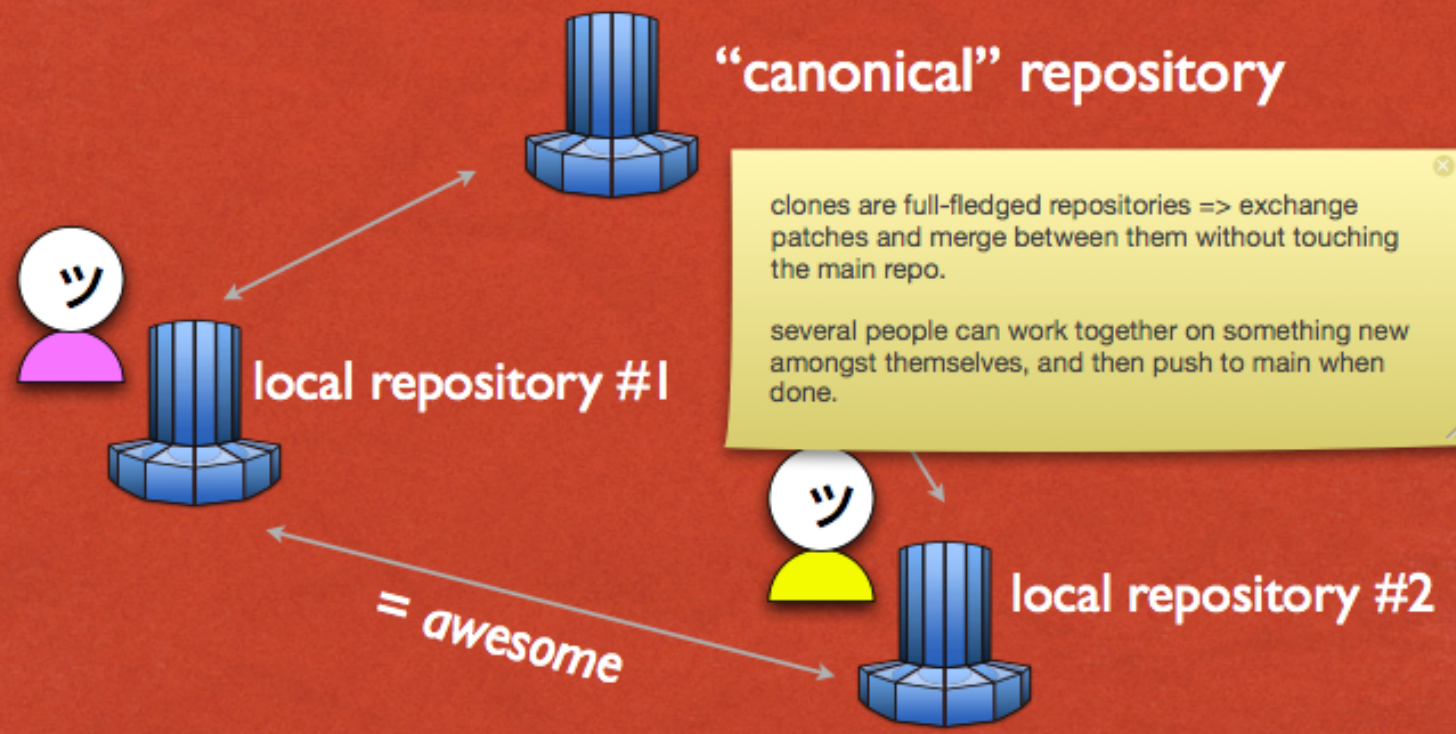
distributed (`Git`):



distributed (`Git`):



distributed (`Git`):



staging area

"canonical" repository

like a shelf on which to put things that'll make up the next commit. incremental building of. "chunk commits" = awesome. "tangled working copy" problem gone.

local repository

working copy

object database, trees & blobs that represent the repo (file contents,

staging area

stuff on your disk tracked by the repo

as many as you want
independent of each other
private
FAST operations (create, check out, merge, delete)

* for new ideas
* for bugfixes/features
* master for production, one for testing, several for day-to-day work

changes the way you work
can manage to do some of this with other systems, but is PITA.

branching

Github

- Github is a web site that supports public repositories
  - Free as long as you allow the world to clone you.
- Create an account.
- Go to `UCSD_BigData` and `fork` it. Now you have your own copy of the class scripts.
- To get it on your computer you need to clone it.
- You can `push` to your repositories and to repositories on which you are a “collaborator”.

For next class

- Install and play with LaunchNotebookServer.py
  - `Git pull` to get latest version
- Create a github account and fork UCSD_BigData
- Register yourself on Piazza.
- Go Through notebooks:
  - `LaunchPythonNotebook –c ipython`
    - `Part 1 - Running Code.ipynb`
    - `Part 2 - Basic Output.ipynb`
    - `Part 3 - Plotting with Matplotlib.ipynb`
    - `Part 4 - Markdown Cells.ipynb`
    - `Part 5 - Rich Display System.ipynb`
  - `LaunchPythonNotebook –c weather`
  - `LaunchPythonNotebook –c hrojas-learn-pandas`

In remaining time

- Demo GitHub
- Demo weather analysis notebook.