

Google Cloud

# Scientific Computing in the Clouds

Karan Bhatia, Google

May 1, 2017

# Investing to meet University and research needs

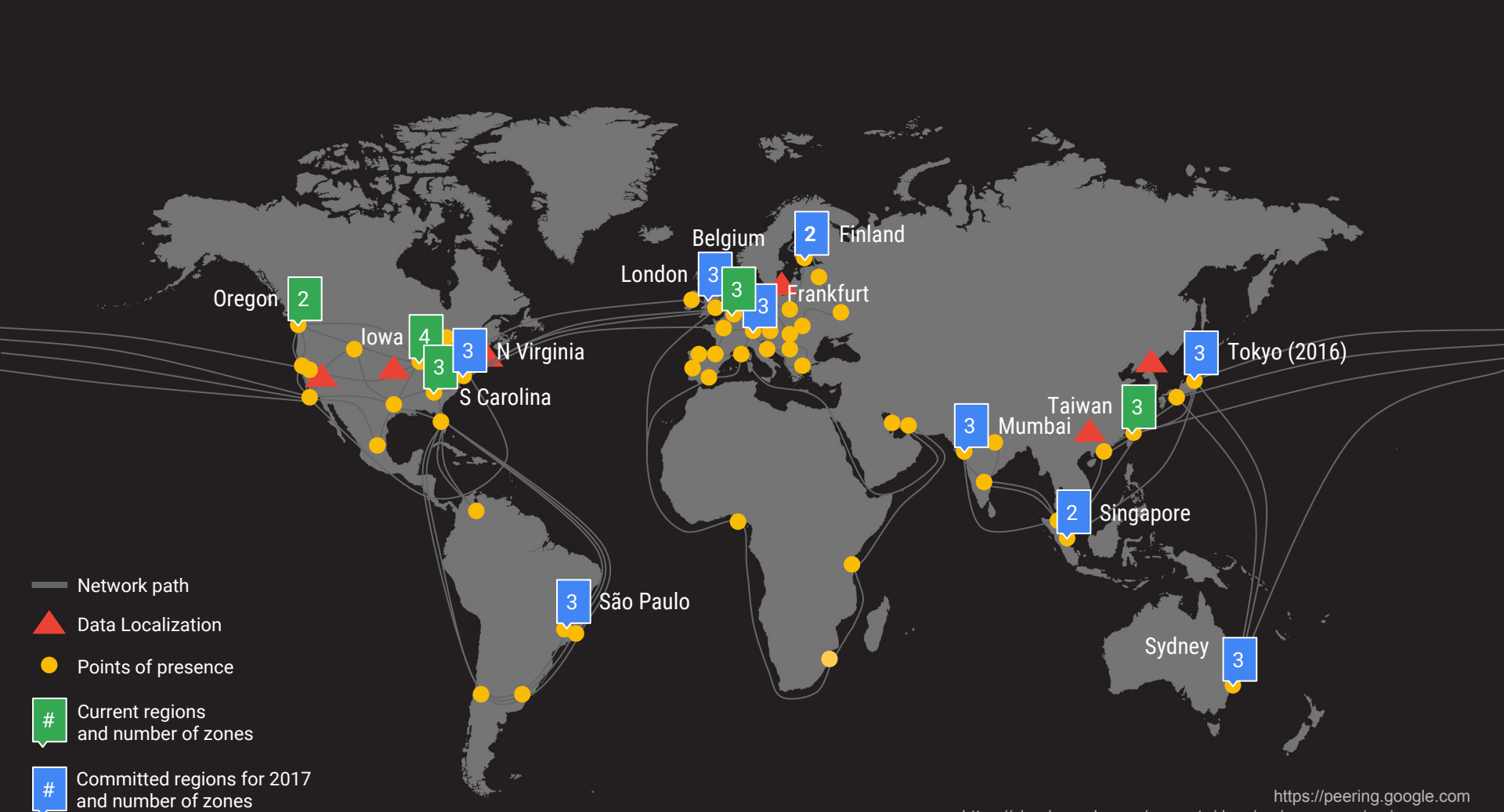
# 1 Billion

End users served by GCP customers

# \$29.4 Billion

Google's trailing 3 Year CAPEX investment





— Network path

▲ Data Localization

● Points of presence

# Current regions and number of zones

# Committed regions for 2017 and number of zones

# Agenda

- Big Compute
- Big Data
- Programs
- Patterns

# Big Compute

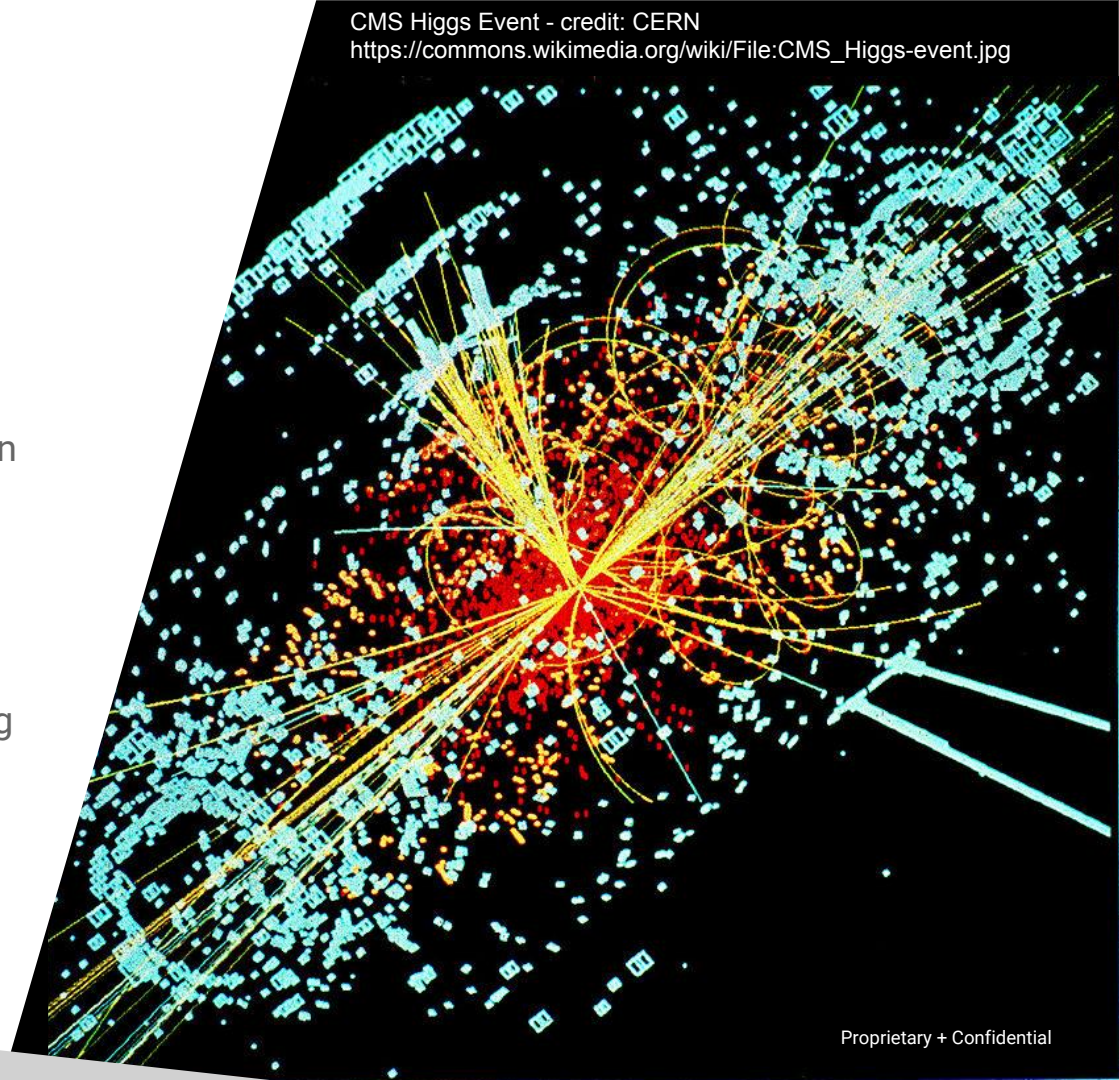


# SC16 CMS Demonstrator

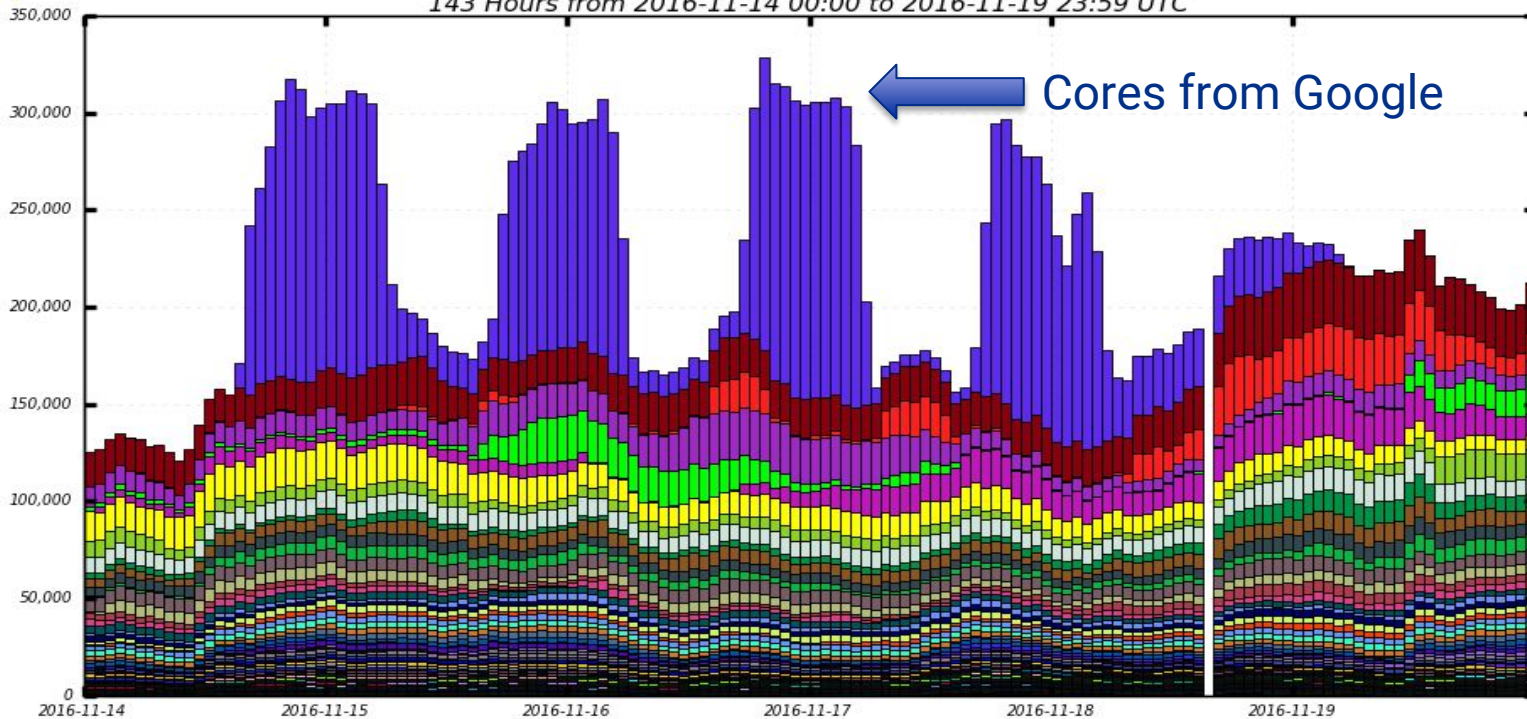
Target: generate 1 Billion events in  
48 hours during Supercomputing 2016 on  
Google Cloud via HEPCloud

35% filter efficiency = stage out 380  
million events → 150 TB output

Double the size of global CMS computing  
resources



Running Job Cores  
143 Hours from 2016-11-14 00:00 to 2016-11-19 23:59 UTC



T3\_US\_HEP\_Cloud  
T3\_US\_NotreDame

T1\_US\_FNAL  
T2\_CH\_CERN  
T2\_US\_Caltech

T0\_CH\_CERN  
T2\_DE\_DESY  
T2\_US\_Purdue

T2\_US\_Wisconsin  
T2\_US\_Florida  
T2\_US\_MIT

T2\_CH\_CERN\_HLT  
T1\_IT\_CNAF  
T2\_US\_UCSD





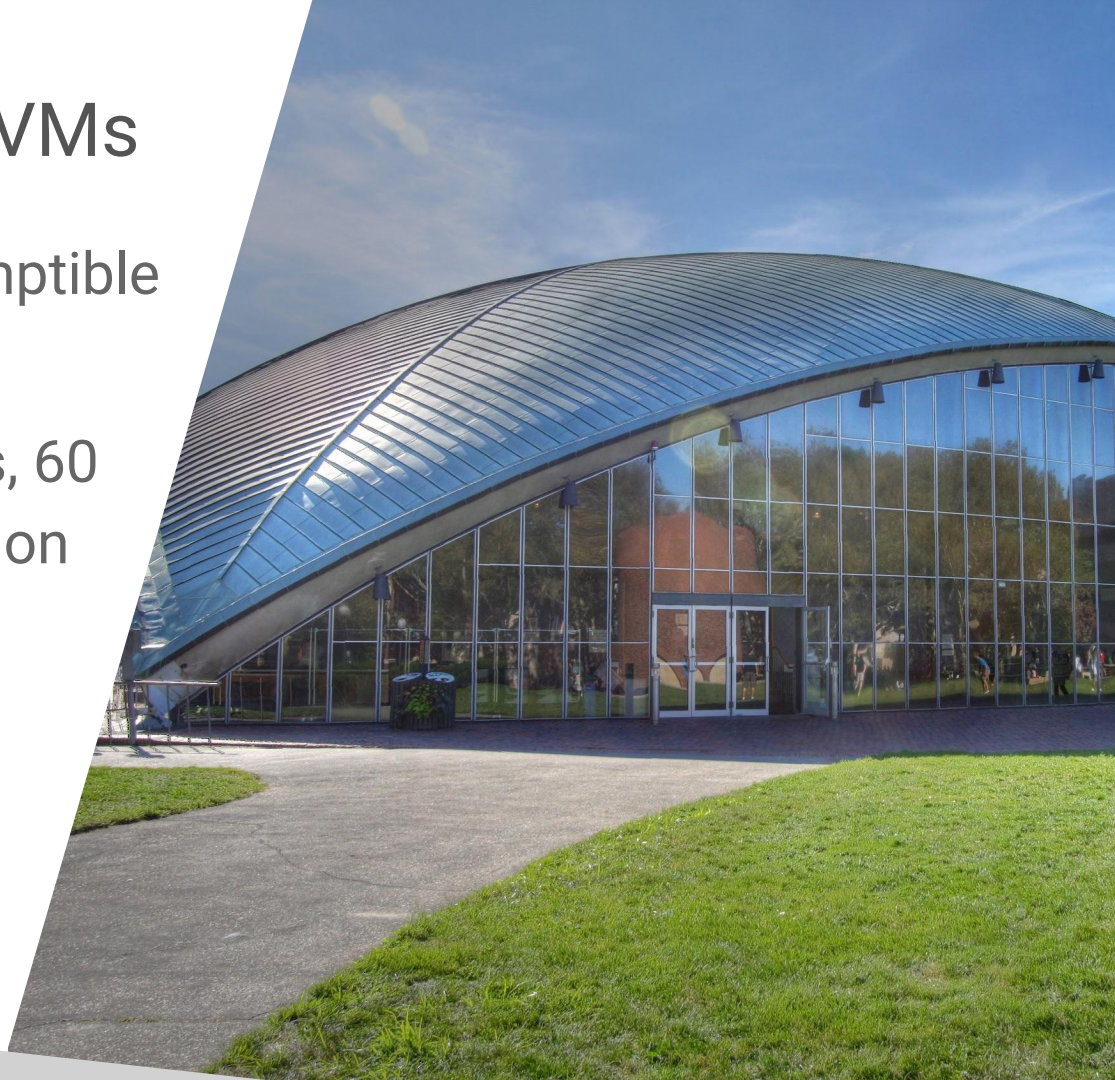
# MIT Research w/ VMs

220,000 cores on preemptible VMs

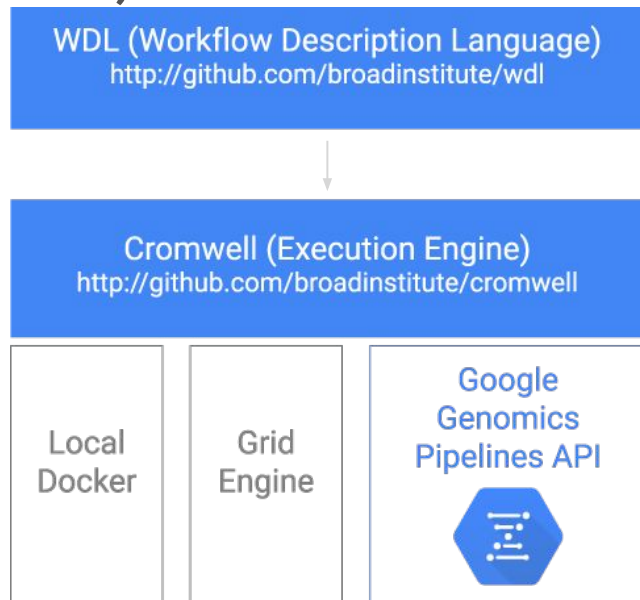
2,250 32-core instances, 60 CPU-years of computation in a single afternoon

Answers in hours v. months

Products used: Google Compute Engine, Cloud Storage, DataStore



# Broad Firecloud: WDL, Cromwell and Google Genomics



A full stack for use by the community!  
See [software.broadinstitute.org/wdl](http://software.broadinstitute.org/wdl)

**WDL:** an external DSL used by computational biologists to express the analytical pipelines

**Cromwell:** a scalable, robust engine for executing WDL against pluggable backends including local, Docker, Grid Engine or ...

**Google Genomics Pipelines API:** co-developed by Broad and Google Genomics, a scalable Docker-as-a-Service with data scheduling

# Pipeline definition

```
{
  "name": "samtools index",
  "description": "Run samtools index to generate a BAM index file",
  "inputParameters": [
    {
      "name": "inputFile",
      "localCopy": {
        "disk": "data",
        "path": "input.bam"
      }
    },
    {
      "name": "outputFile",
      "localCopy": {
        "disk": "data",
        "path": "output.bam.bai"
      }
    }
  ],
  "resources": {
    "minimumCpuCores": 1,
    "minimumRamGb": 1,
    "disks": [
      {
        "name": "data",
        "type": "PERSISTENT_HDD"
        "sizeGb": 200,
        "mountPoint": "/mnt/data",
      }
    ]
  },
  "docker": {
    "imageName": "quay.io/cancercollaboratory/dockstore-tool-samtools-index",
    "cmd": "samtools index /mnt/data/input.bam /mnt/data/output.bam.bai"
  }
}
```

# Create, run, monitor, and kill pipelines

## Create

```
$ gcloud alpha genomics pipelines create --pipeline-json-file PIPELINE-FILE.json --pipeline-json-file samtools_index.json  
Created samtools index, id: PIPELINE-ID
```

## Run

```
$ gcloud alpha genomics pipelines run --pipeline_id PIPELINE-ID \  
--logging gs://YOUR-BUCKET/YOUR-DIRECTORY/logs \  
--inputs inputFile=gs://genomics-public-data/gatk-examples/example1/NA12878_chr22.bam \  
--outputs outputFile=gs://YOUR-BUCKET/YOUR-DIRECTORY/output/NA12878_chr22.bam.bai  
Running: operations/OPERATION-ID
```

## Status

```
$ gcloud alpha genomics operations describe OPERATION-ID
```

## Kill

```
$ gcloud alpha genomics operations cancel OPERATION-ID
```

# DSUB (google genomics pipelines)



Features Business Explore Pricing

This repository

Search

[Sign in](#) or [Sign up](#)

[goolegenomics](#) / [dsub](#)

Watch

14

★ Star

16

Fork

4

<> Code

🔔 Issues 11

🔗 Pull requests 0

📁 Projects 0

📊 Pulse

📈 Graphs

Branch: master ▾

[dsub](#) / [README.md](#)

Find file

Copy path

```
./dsub \  
--project my-cloud-project \  
--zones "us-*" \  
--logging gs://my-bucket/logs \  
--input BAM=gs://genomics-public-data/1000-genomes/bam/HG00114.mapped.ILLUMINA.bwa.GBR.low_covera  
--output BAI=gs://my-bucket/HG00114.mapped.ILLUMINA.bwa.GBR.low_coverage.20120522.bam.bai \  
--image quay.io/cancerlaboratory/dockstore-tool-samtools-index \  
--command 'samtools index ${BAM} ${BAI}' \  
--wait
```

# ElastiCluster

aims to provide a user-friendly command line tool to create, manage and setup computing clusters hosted on cloud infrastructures like [Amazon's Elastic Compute Cloud EC2](#), [Google Compute Engine](#), or a private [OpenStack](#) cloud. Its main goal is to get your compute cluster up and running with just a few commands.

[Read the Documentation](#)[Install ElastiCluster](#)

## How it works

The architecture of ElastiCluster is quite simple: a configuration file defines a set of cluster configurations and information on how to access a specific cloud webservice.

Using the command line (or, very soon, a simple API), you can start a cluster. ElastiCluster will connect to the desired cloud, start the virtual machines and wait until they are accessible via SSH.



After all the virtual machines are up and running, ElastiCluster will use [Ansible](#) to configure them.

## Features

ElastiCluster provides automated setup of:

- HPC batch-queuing clusters running [SLURM](#), [Grid Engine](#), or [TORQUE+MAUI](#);
- [Spark](#) / [Hadoop](#) clusters with HDFS and Hive/SQL;
- distributed storage clusters using [GlusterFS](#), [Ceph](#), or [OrangeFS](#)
- Useful add-on tools like [Ganglia](#) for monitoring or [Jupyter/IPython](#) for teaching or interactive programming use.
- ... or anything that you can install with an [Ansible](#) playbook!

## Demo Video

Demo: Elasticcluster deploying a SLURM clus..  

resize SLURM cluster

add 1 worker node 



# Lessons

- Integration with third-party workload manager vs roll your own vs something in between
  - HTCondor, Slurm, Google Genomics Pipelines, ssh
  - Managed instance groups
- On-premise + hybrid vs on-cloud
- Cost optimizations
  - Preemptible vms and custom machine types
  - Per-minute billing
- Networking is a key differentiator, public peering + internet2 member



# Intel Skylake

- Significant “per core” performance improvements
- **Intel® Advanced Vector Extension 512** (Intel® AVX-512)
  - 2x flops/second
- Accelerated IO with Intel® Omni-Path Architecture (Fabric)
- Integrated Intel® QuickAssist Technology (crypto & compression offload)
- Intel® Resource Director Technology (Intel® RDT) for Efficiency & TCO



## Google Cloud Platform Blog

Product updates, customer stories, and tips and tricks on Google Cloud Platform

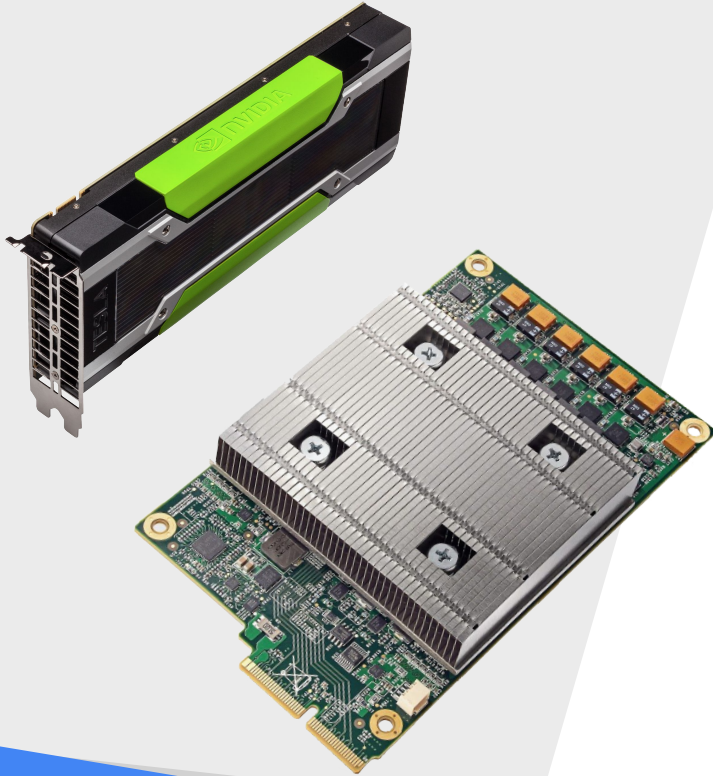
### Google Cloud Platform is the first cloud provider to offer Intel Skylake

Friday, February 24, 2017

By Urs Hölzle, Senior Vice President, Google Cloud Infrastructure

I'm excited to announce that [Google Cloud Platform](#) (GCP) is the first cloud provider to offer the next generation Intel Xeon processor, codenamed Skylake.

# Hardware Accelerated



- Available Today: **NVIDIA K80 GPU**
- Coming Soon: **Tensor Processing Unit (TPU)**
- Custom ASIC built and optimized for TensorFlow
- Used in production at Google for over 16 months
- 7 years ahead of GPU performance per watt

# In-Datacenter Performance Analysis of a Tensor Processing Unit™

Norman P. Jouppi, Cliff Young, Nishant Patil, David Patterson, Gaurav Agrawal, Raminder Bajwa, Sarah Bates, Suresh Bhatia, Nan Boden, Al Borchers, Rick Boyle, Pierre-luc Cantin, Clifford Chao, Chris Clark, Jeremy Coriell, Mike Daley, Matt Dau, Jeffrey Dean, Ben Gelb, Tara Vazir Ghaemmaghami, Rajendra Gottipati, William Gulland, Robert Hagmann, C. Richard Ho, Doug Hogberg, John Hu, Robert Hundt, Dan Hurt, Julian Ibarz, Aaron Jaffey, Alek Jaworski, Alexander Kaplan, Harshit Khaitan, Daniel Killebrew, Andy Koch, Naveen Kumar, Steve Lacy, James Laudon, James Law, Diemthu Le, Chris Leary, Zhuyuan Liu, Kyle Lucke, Alan Lundin, Gordon MacKean, Adriana Maggiore, Maire Mahony, Kieran Miller, Rahul Nagarajan, Ravi Narayanaswami, Ray Ni, Kathy Nix, Thomas Norrie, Mark Omernick, Narayana Penukonda, Andy Phelps, Jonathan Ross, Matt Ross, Amir Salek, Emad Samadiani, Chris Severn, Gregory Sizikov, Matthew Snelham, Jed Souter, Dan Steinberg, Andy Swing, Mercedes Tan, Gregory Thorson, Bo Tian, Horia Toma, Erick Tuttle, Vijay Vasudevan, Richard Walter, Walter Wang, Eric Wilcox, and Doe Hyun Yoon

*Google, Inc., Mountain View, CA USA*

Email: {jouppi, cliffy, nishantpatil, davidpatterson} @google.com

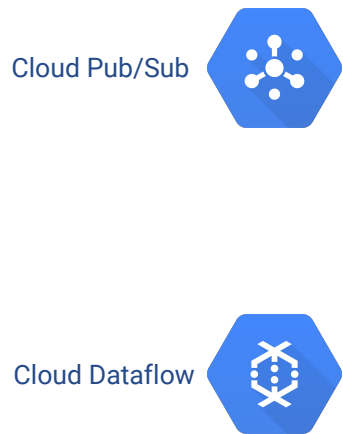
*To appear at the 44th International Symposium on Computer Architecture (ISCA), Toronto, Canada, June 26, 2017.*

## Abstract

Data

# Data Prep (beta)

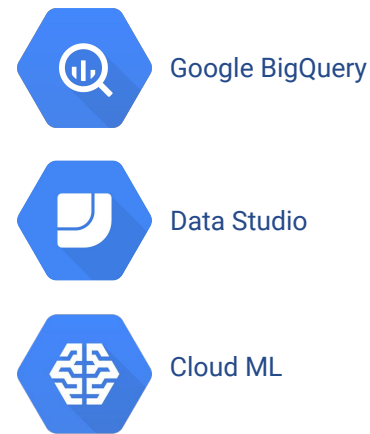
## 1. Ingest Data



## 2. Instantly Prepare Data



## 3. Analyze Data





# Cloud Dataprep

## Instant Data Exploration

Visually explore and interact with data in seconds. Instantly understand data distribution and patterns. There is no need for one to write code. You can prepare data with a few clicks.

## Intelligent Data Cleansing

Cloud Dataprep automatically identifies data anomalies and helps you to take corrective actions fast. Get data transformation suggestions based on your usage pattern. Standardize, structure, and join datasets easily with a guided approach.

## Serverless

Cloud Dataprep is a serverless service, so you do not need to create or manage infrastructure.

## Seriously Powerful

Cloud Dataprep is built on top of powerful Google Cloud Dataflow service. Cloud Dataprep is auto-scalable and can easily handle processing massive data sets.



## Supports Common Data Sources of Any Size

Process diverse datasets - structured and unstructured. Transform data stored in CSV, JSON, or relational Table formats. Prepare datasets of any size, megabytes to terabytes, with equal ease.

Grid Columns Full Dataset - 785.29kB 5 Columns 20,000 Rows 3 Data Types

Preview

#	column2	column3	#	column4	#	column5
0 - 1		13 Categories				
0		.AZ				
0		.CA				
0		.AK				
0		.NM				
0		.WA				
0		.KS				
0		.OK				
0		.OK				
0		.MN				
0		.MN				
0		.MN				
0		.MN				
1		.OK				
1		.OK				
1		.OK				
1		.KS				

SUGGESTIONS

Delete rows where `sourcerownumber() == 4`

#	column2	column3
0		.NM
0		.AZ
0		.CA

Affects all columns, 1 row

Switch to editor

transformation

Column required

column6

column6

Between two positions

starting from 5

ending at 7

Cancel Add to Recipe

Grid Columns Full Dataset - 785.29kB 5 Columns 20,000 Rows 3 Data Types

# column5

Overview Patterns

SUMMARY

TOP VALUES

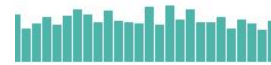
1 - 0 Run Job

Columns: All Transformed - 3 Columns

Source to be dropped Preview

#	column4	#	column5	ABC	column6	ABC	column1	#	column7
	1.42B - 1.45B		0 - 499		10,000 Categories		1,000 Categories		152 - 9M
0	.1424860400		.37		.1LOCWA4000790		.1LOC		4000790
0	.1430020400		.495		.1LOCKS5000950		.1LOC		5000950
0	.1438298400		.395		.1LOCOK6000228		.1LOC		6000228
0	.1438380400		.72		.1LOCOK7000310		.1LOC		7000310
0	.1436414400		.320		.1LOCMN8000344		.1LOC		8000344
0	.1438167400		.266		.1LOCMN9000097		.1LOC		9000097
0	.1438271020		.195		.1LOCMN9000097		.1LOC		9000097
0	.1437008760		.351		.1LOCMN8000344		.1LOC		8000344
0	.1438623160		.429		.1LOCOK7000310		.1LOC		7000310
0	.1438557900		.96		.1LOCOK6000228		.1LOC		6000228
0	.1430173580		.164		.1LOCKS5000950		.1LOC		5000950
0	.1425260440		.460		.1LOCWA4000790		.1LOC		4000790

- 57
- 56
- 56
- 55
- 54
- 54
- 54
- 54
- 54
- 54
- 54
- 53
- 53



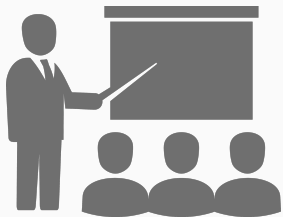
MISMA

None

OUTLIE

None

Google cloud computing  
can help universities transform



Teaching



Faculty in  
select  
countries

+



Teaching  
university  
courses

+



In computer  
science or  
related fields

# Funding Agency Partnerships

- National Science Foundation
  - [BIGDATA](#)
- National Institutes of Health
  - [Data Commons](#)



# Google Cloud Platform Free Tier



Google BigQuery

Fully managed, petabyte scale, analytics data warehouse.

1 TB

Queries per month



Google Container Engine

One-click container orchestration via Kubernetes clusters, managed by Google.

Basic Cluster

Sized 5 nodes or fewer



Google Cloud Storage

Best in class performance, reliability, and pricing for all your storage needs.

5 GB-Months

Regional storage



Google App Engine

Platform for building scalable web applications and mobile backends.

28

Instance hours per day

5 GB

Cloud Storage





# Google Cloud Public Datasets Program

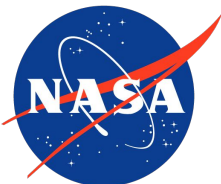
## **Mission:**

Facilitate the onboarding of datasets into Google Cloud products



# Next

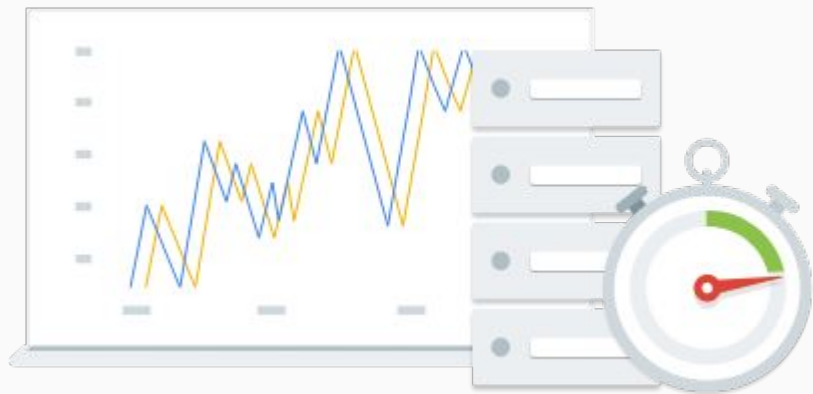
42+ datasets



# You can contribute too!

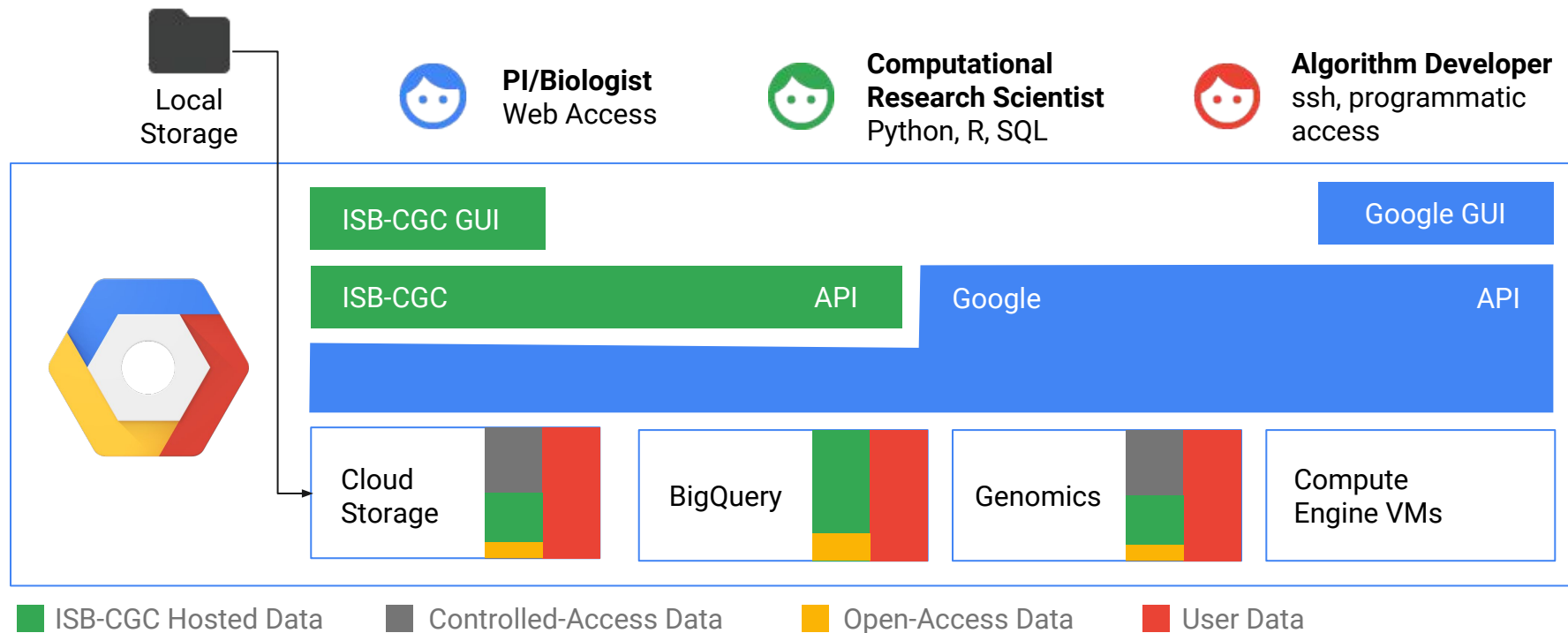
**Visit:** <https://cloud.google.com/public-datasets/>

**Email:** [bq-public-data@google.com](mailto:bq-public-data@google.com)



# Themes / Patterns for Scientific Computing

# Extending the Cloud APIs



# FireCloud

A cloud-based cancer genomics analysis platform with co-located TCGA data



## Use FireCloud

Cloud-based analysis services



## Documentation

Detailed user guide, tutorials and resources



## Forum

Ask here for help with questions and bug reports



## Blog

Announcements and progress updates

### Cloud credits now available!

[Details and application forms](#)

### User Survey

[Please fill out this survey](#)

### Latest release: Apr 13, 2017

[Release Notes](#)

Modeled after the Broad Institute's Firehose analysis infrastructure, FireCloud **democratizes data access and facilitates collaboration** by providing a **robust, scalable platform accessible to the community at large**. Using the elastic compute capacity of Google Cloud, FireCloud **empowers analysts, tool developers and production managers** to perform **large-scale analysis**, engage in **data curation**, and **store or publish results**.

Users can upload their **own analysis methods** to workspaces or run the Broad Institute's **best practice tools and pipelines**. FireCloud also includes tutorial workspaces as well as **carefully curated open and controlled-access TCGA workspaces** that users can clone.

FireCloud **supports the mission of TCGA** by provisioning workspaces with curated data and best practice tools and pipelines. This will empower researchers across the globe to **explore TCGA data in novel and innovative ways**, and create **new opportunities for cancer research**.



# Copy Number segments

The goal of this notebook is to

This table contains all available T Genome Wide SNP6 array, as of recent archives (egbroad.mit. types was downloaded from the Each of these segmentation files During ETL the sample identifier the SDRF file in the associated n

In order to work with BigQuery, the name(s) of the table(s) you a

```
import gcp.bigquery as bq
cn_BQtable = bq.Table
```

From now on, we will refer to thi table name each time.

Let's start by taking a look at the

bigquery schema --ta

name	type
ParticipantBarcode	STRING
SampleBarcode	STRING
SampleTypeLetterCode	STRING
AliquotBarcode	STRING
Study	STRING
Platform	STRING
Chromosome	STRING
Start	INTEGER
End	INTEGER
Num_Probes	INTEGER
Segment_Mean	FLOAT

Unlike most other molecular data microRNAs, this data is produced sizes and positions of these segn

Now we'll use matplotlib to create some simple visual

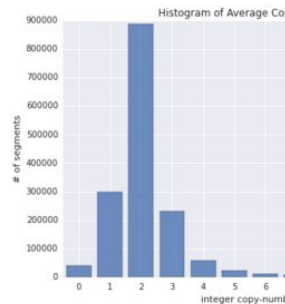
```
import numpy as np
import matplotlib.pyplot as plt
```

For the segment means, let's invert the log-transform

```
%%sql --module getCNhist
```

```
SELECT
  lin_bin,
  COUNT(*) AS n
FROM (
  SELECT
    Segment_Mean,
    (2.*POW(2,Segment_Mean)) AS lin_
    INTEGER(((2.*POW(2,Segment_Mean))
  FROM
    $t
  WHERE
    ((End-Start+1)>1000 AND Sample
  GROUP BY
    lin_bin
  HAVING
    ( n > 2000 )
  ORDER BY
    lin_bin ASC
```

```
CNhist = bq.Query(getCNhist,t=cn_BQc
bar_width=0.80
plt.bar(CNhist['lin_bin']+0.1,CNhist
plt.xticks(CNhist['lin_bin']+0.5,CN
plt.title('Histogram of Average Copy
plt.ylabel('# of segments');
plt.xlabel('integer copy-number');
```



The histogram illustrates that the vast majority of the either side representing deletions (left) and amplifications (right)

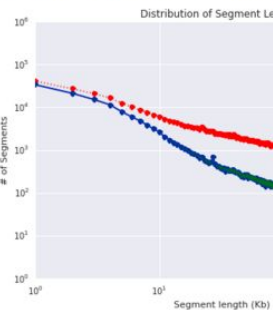
```
bin
ORDER BY
bin ASC
```

```
%%sql --module getSLhist_1k_amp
```

```
SELECT
  bin,
  COUNT(*) AS n
FROM (
  SELECT
    (END-Start+1) AS segLength,
    INTEGER((END-Start+1)/1000) AS b
  FROM
    $t
  WHERE
    (END-Start+1)<1000000 AND Sample
  GROUP BY
    bin
  ORDER BY
    bin ASC
```

```
SLhistDel = bq.Query(getSLhist_1k_de
SLhistAmp = bq.Query(getSLhist_1k_
```

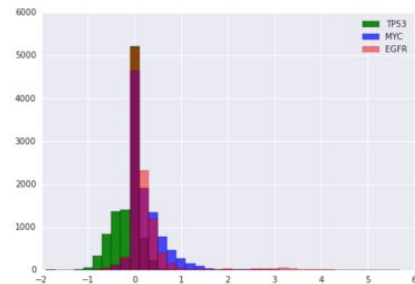
```
plt.plot(SLhist_1k['bin'],SLhist_1k[
plt.plot(SLhistDel['bin'],SLhistDel[
plt.plot(SLhistAmp['bin'],SLhistDel[
plt.xscale('log');
plt.yscale('log');
plt.xlabel('Segment length (kb)');
plt.ylabel('# of segments');
```



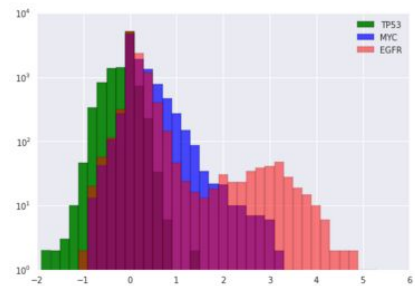
The amplification and deletion distributions are nearly from this graph that a majority of the segments less th lengths >100Kb are copy-number neutral.

And now we'll take a look at histograms of the average copy-number for these three genes. TP53 (in green) shows a significant number of partial deletions (CN<0), while MYC (in blue) shows some partial amplifications -- more frequently than EGFR, while EGFR (pale red) shows a few extreme amplifications (log2(CN/2) > 2). The final figure shows the same histograms on a semi-log plot to bring up the rarer events.

```
binwidth = 0.2
binVals = np.arange(-2+(binwidth/2.), 6-(binwidth/2.), binwidth)
plt.hist(tp53CN['avgCN'],bins=binVals,normed=False,color='green',alpha=0.9,label='TP53');
plt.hist(mycCN ['avgCN'],bins=binVals,normed=False,color='blue',alpha=0.7,label='MYC');
plt.hist(egfrCN['avgCN'],bins=binVals,normed=False,color='red',alpha=0.5,label='EGFR');
plt.legend(loc='upper right');
```



```
plt.hist(tp53CN['avgCN'],bins=binVals,normed=False,color='green',alpha=0.9,label='TP53');
plt.hist(mycCN ['avgCN'],bins=binVals,normed=False,color='blue',alpha=0.7,label='MYC');
plt.hist(egfrCN['avgCN'],bins=binVals,normed=False,color='red',alpha=0.5,label='EGFR');
plt.yscale('log');
```



# TensorFlow



- World's most popular ML framework
- Developer friendly yet performance optimized
- **Powers over 100 Google services**
- Managed infrastructure with Cloud ML
- Tutorials at <https://www.tensorflow.org>

## Linear Regression

VS

## Neural Network

```
1 import tensorflow as tf
2
3 #Define input feature columns
4 sq_footage = tf.contrib.layers.real_valued_column("sq_footage")
5 feature_columns = [sq_footage]
6
7 #Define input function
8 def input_fn(feature_data,label_data=None):
9     return {"sq_footage":feature_data}, label_data
10
11 #Instantiate Linear Regression Model
12 estimator = tf.contrib.learn.LinearRegressor(
13     feature_columns=feature_columns,
14     optimizer=tf.train.FtrlOptimizer(learning_rate=100))
15
16 #Train
17 estimator.fit(
18     input_fn=lambda:input_fn(tf.constant([1000,2000]),
19                               tf.constant([100000,200000])),
20     steps=100)
21
22 #Predict
23 estimator.predict(input_fn=lambda: input_fn(tf.constant([3000])))
```

```
1 import tensorflow as tf
2
3 #Define input feature columns
4 sq_footage = tf.contrib.layers.real_valued_column("sq_footage")
5 feature_columns = [sq_footage]
6
7 #Define input function
8 def input_fn(feature_data,label_data=None):
9     return {"sq_footage":feature_data}, label_data
10
11 #Instantiate Neural Network Model
12 estimator = tf.contrib.learn.DNNRegressor(
13     feature_columns=feature_columns, hidden_units=[10,10])
14
15
16 #Train
17 estimator.fit(
18     input_fn=lambda:input_fn(tf.constant([1000,2000]),
19                               tf.constant([100000,200000])),
20     steps=100)
21
22 #Predict
23 estimator.predict(input_fn=lambda: input_fn(tf.constant([3000])))
```

Thank you!