

GraphLab: What's Next



Yucheng Low

Chief Architect

GraphLab

<http://www.istc-cc.cmu.edu/>

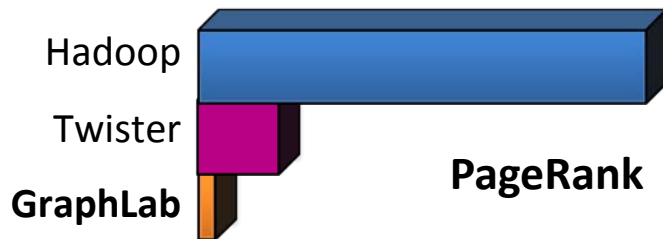
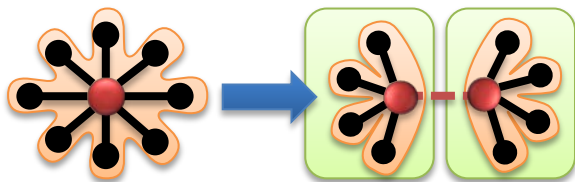


A Tail of Two Projects

GraphLab₂

Distributed Graph Processing System

How Fast Can we Go?



PageRank

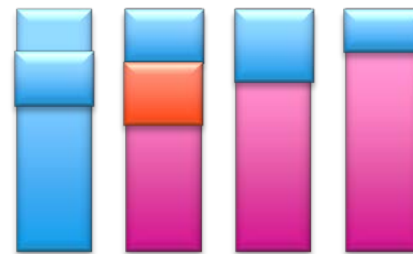


Triangle Count

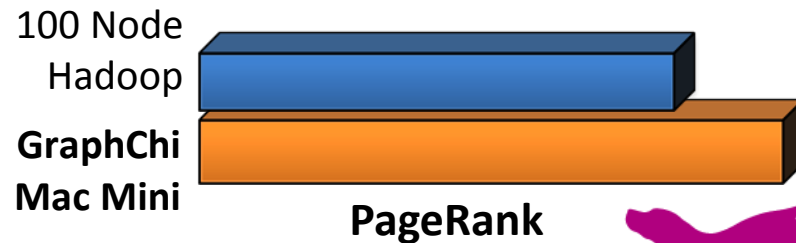


Disk/SSD Graph Processing System

How Large Can we Go?



20B edges on one Laptop



PageRank

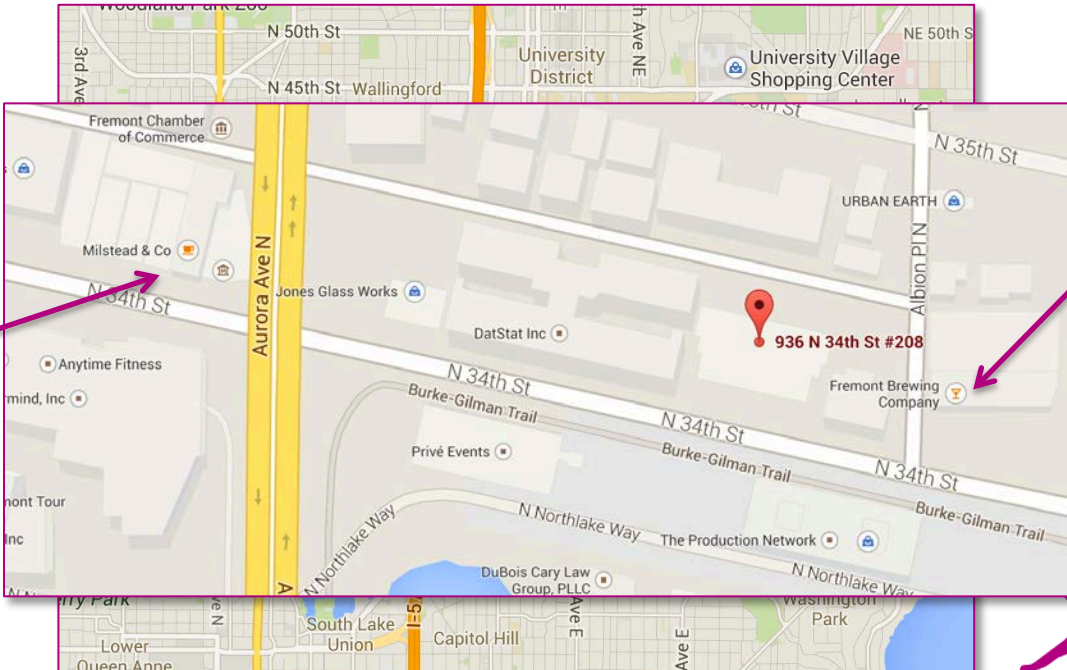




GraphLab

Unleash Data Science

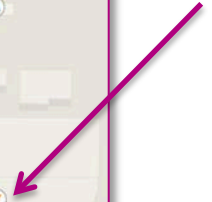
May 2013: **GraphLab Co** came into existence



Coffee Shop



Brewery



Currently Proprietary,
Open sourcing in near future.



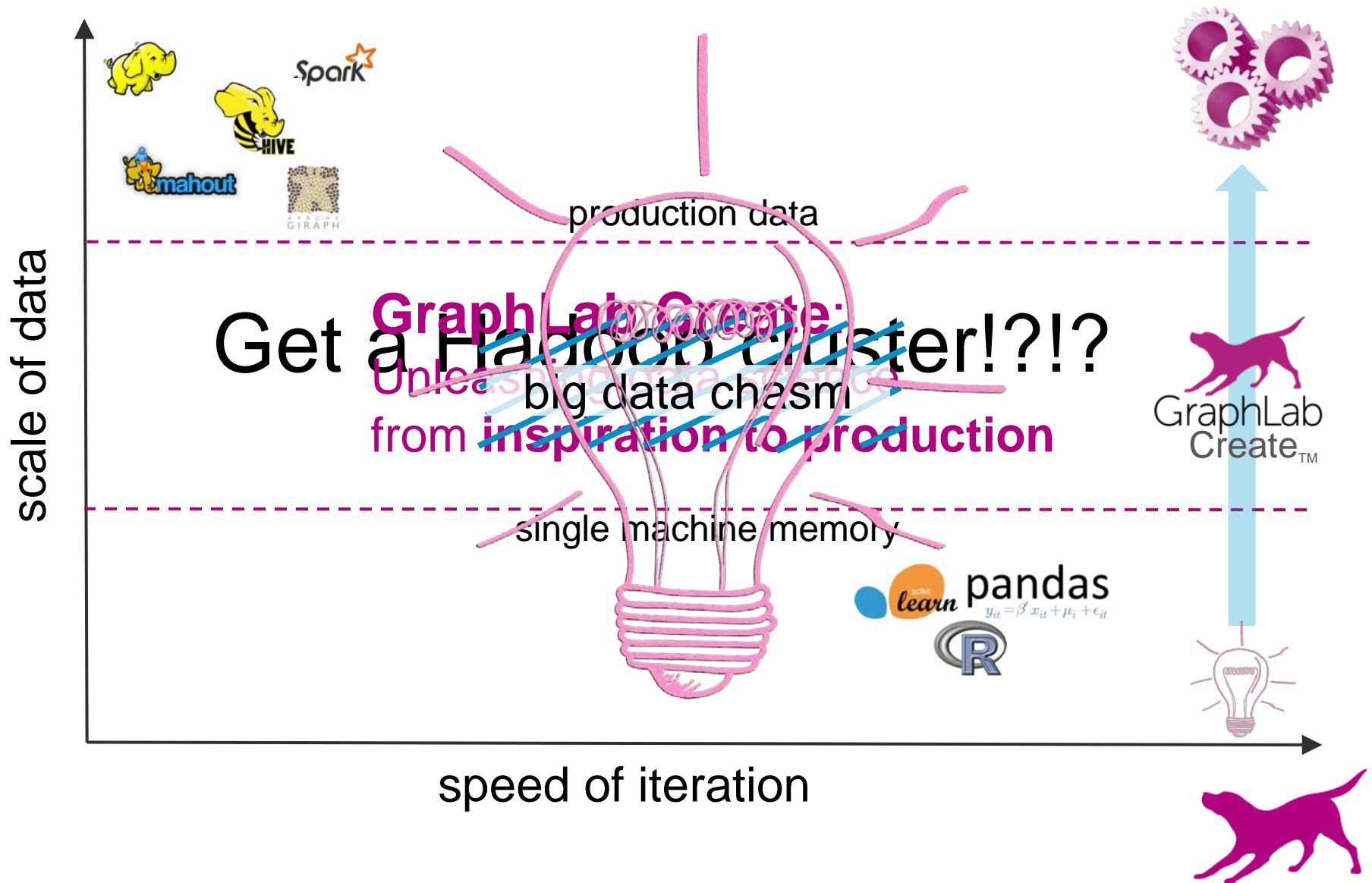
GraphLab
Create™

From Inspiration To Production

Get v0.9.1 at <http://graphlab.com/>





Crossing the Big Data Chasm

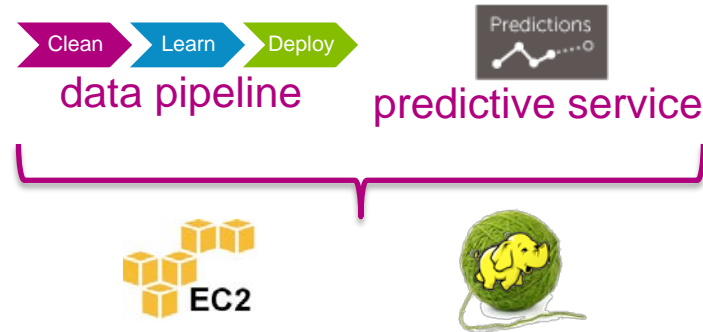


Data scientist: inspiration to production

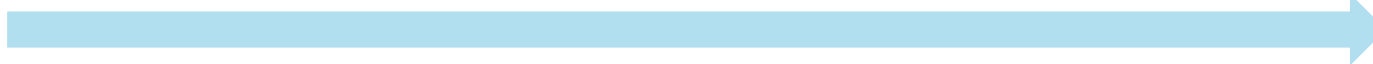
Prototype

 Use my laptop
Variety of data
Not toy data scales
 Language I love
Iterate quickly

Production



Monitor



GraphLab Create

Analyze **big data** on **one machine**
graphs, tables, text, images
in **Python**
doesn't have to fit in memory

Distribute in production
with same code
on EC2, Yarn,...

GraphLab Canvas: Monitor & visualize
from prototype to production



What folks are saying about GraphLab Create

“The ease of use and scalable performance, which is not limited by the memory of the machine, are allowing us to **innovate and advance at an astonishing pace.**”

- Andrew Bruce, Senior Director, Data Science, Zillow

“Graphlab Create provides us with an end-to-end **efficient framework ... both tabular and graph data** generated by the activity of our users.”

- Baldo Faeita, Social Computing Lead, Adobe Systems

ExxonMobil

Zillow



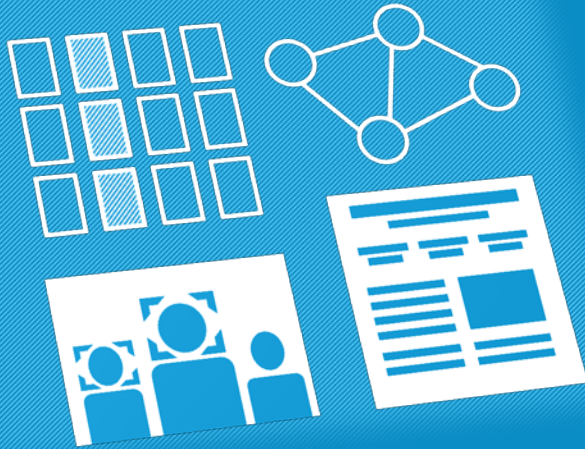
PANDORA



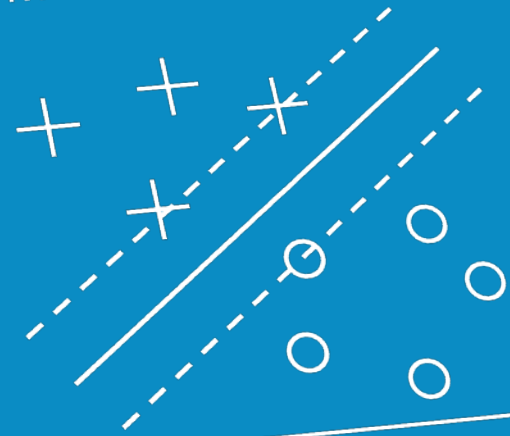
crosswise



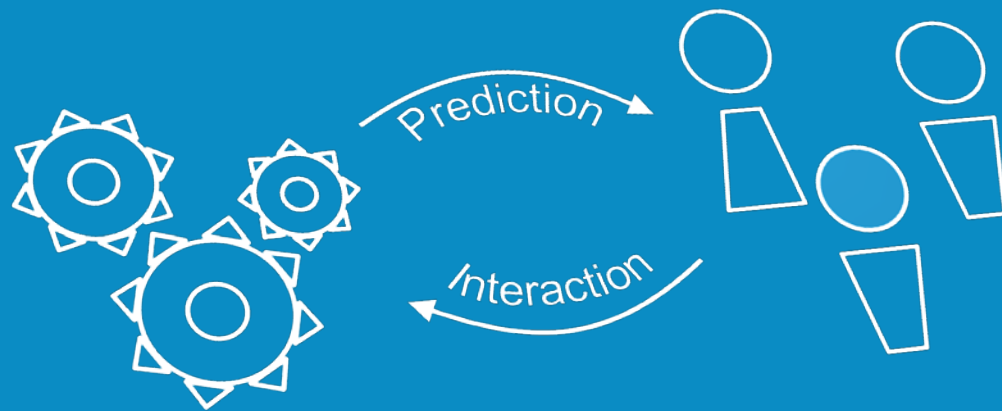
Scalable Data Representation



Robust Machine Learning



Predictive Apps in Production



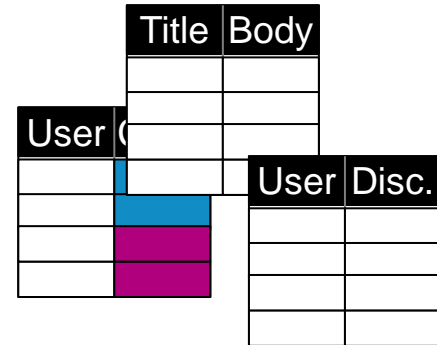
SFrame and SGraph

Built *by* data scientists,
for data scientists.

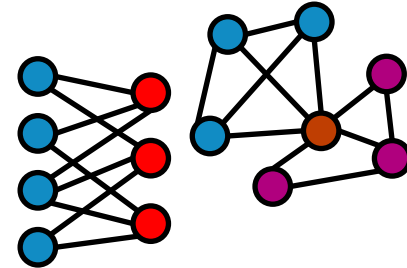
Building on decades of database
and systems research.



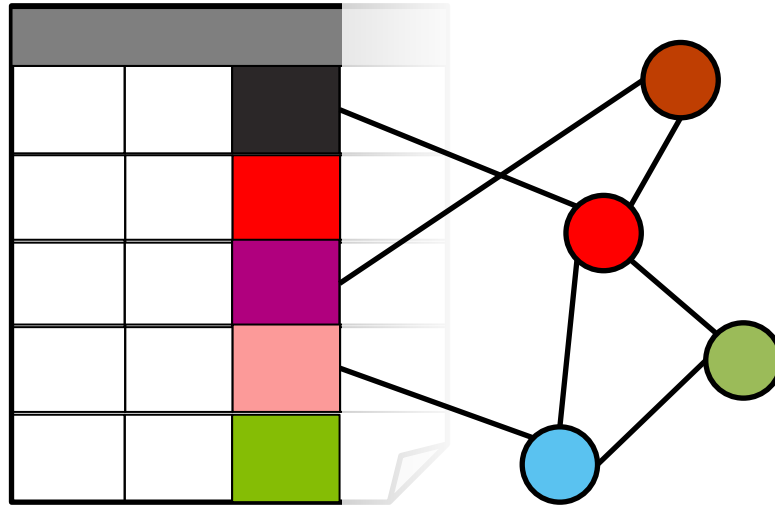
SFrame: Scalable Tabular Data Manipulation



SGraph: Scalable Graph Manipulation



Enabling users to **easily** and **efficiently** translate between both representations to get the best of both worlds.



SFrame Python API Example



SFrame Querying

Supports most typical SQL SELECT operations using a Pythonic syntax.

SQL

```
SELECT Book.title AS title, COUNT(*) AS authors
FROM Book
JOIN Book_author ON Book.isbn = Book_author.isbn
GROUP BY Book.title;
```

SFrame Python

```
Book.join(Book_author, on='isbn')
    .groupby('title', {'authors':gl.aggregate.COUNT})
```



SFrame Design

- **Graceful Degradation as 1st principle**
 - Disk/SSD backed
- **Rich Datatypes**
 - Strong schema types: int, double, string...
 - Weak schema types: list, dictionary
- **Columnar Architecture**
 - Easy feature engineering + Vectorized feature operations.
 - Immutable columns + Lazy evaluation
 - Statistics + visualization + sketches

Scales to >10K columns, Billions of rows on one machine.



Demo



SGraph

- **SFrame backed** graph representation. Inherits SFrame properties.
 - Data types, External Memory, Columnar, compression, etc.
- Layout optimized for batch **external memory computation**.
- GraphChi-inspired architecture, scales to billions of edges on one machine.

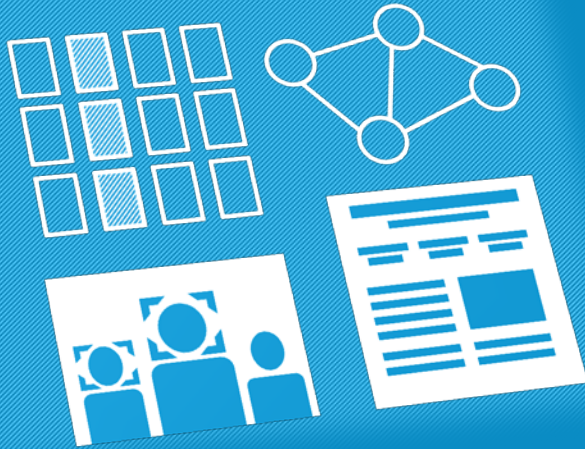


Deep Integration of SFrames and SGraphs

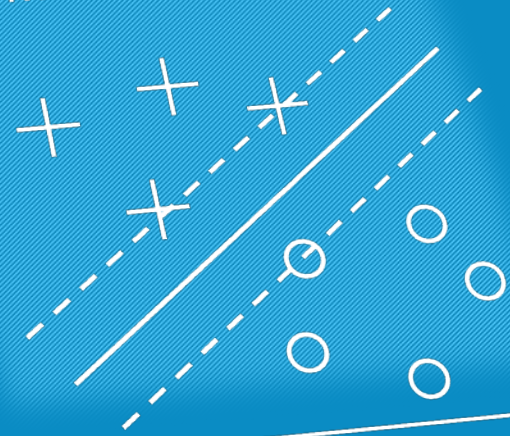
- Seamless interaction between graph data and table data.
- Queries can be performed easily across graph and tables.



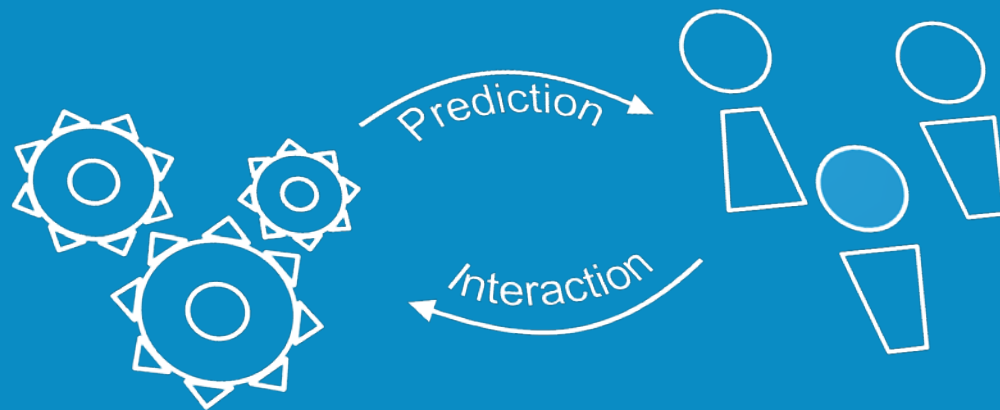
Scalable Data Representation



Robust Machine Learning



Predictive Apps in Production



Most ML toolkits don't focus on the real challenges

Tools out there	Real needs
Bag of algorithms	

GraphLab Create: Robust ML & graph analytics
state-of-the-art scaling and accuracy
focused on solving tasks, automatically



What We Have

Supervised Learning

(Sparse/Dense regimes)

- Logistic Regression
- L1/L2 Reg. Linear Regression
- SVM
- Boosted Decision Trees
- **Random Kitchen Sink of the above**
- **Hash Kernel of the above**

Recommender

(Sparse/Dense feature regimes)

- Matrix Factorization
- Matrix Factorization w/ Features
- **Factorization Machine**
- **Ranking version of all Factorization methods**
- Item Similarity
 - Jacard
 - Pearson
 - Cosine

Internal Convex Solvers

(Sparse/Dense regimes)

- GD
- SGD
- FISTA
- LBFGS
- Newton

Graph

- PageRank
- KCore
- CC
- Shortest Path
- Triangle Count
- Graph Color

Nearest Neighbor

(Sparse/Dense regimes)

- Brute Force
- Ball Tree
- **LSH varieties**

Text

- Topic Modeling

Clustering

- kMeans
- **Hierarchical**

Deep Learning

- **CUDA Accelerated NN**

Sketching

- Hyperloglog
- SpaceSaving
- CountSketch
- Quantile

More Coming!



Scalable Machine Learning

Garbage In Garbage Out.

- Unknown



Scalable Machine Learning

Garbage In Guidance Out.

- Sethu Raman
VP Eng. GraphLab



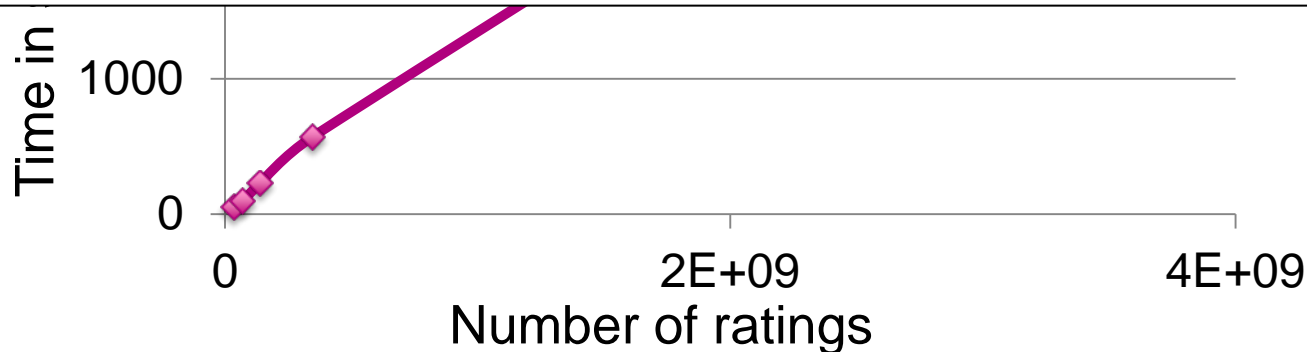
Recommender using matrix factorization

skttop
5000
4000

3.5B ratings
660M users
~ 1 hour

“...during my time as Zynga's lead architect for big data, I found my way to GraphLab. I was astounded at the **dramatic savings, on the order of 500x...**”

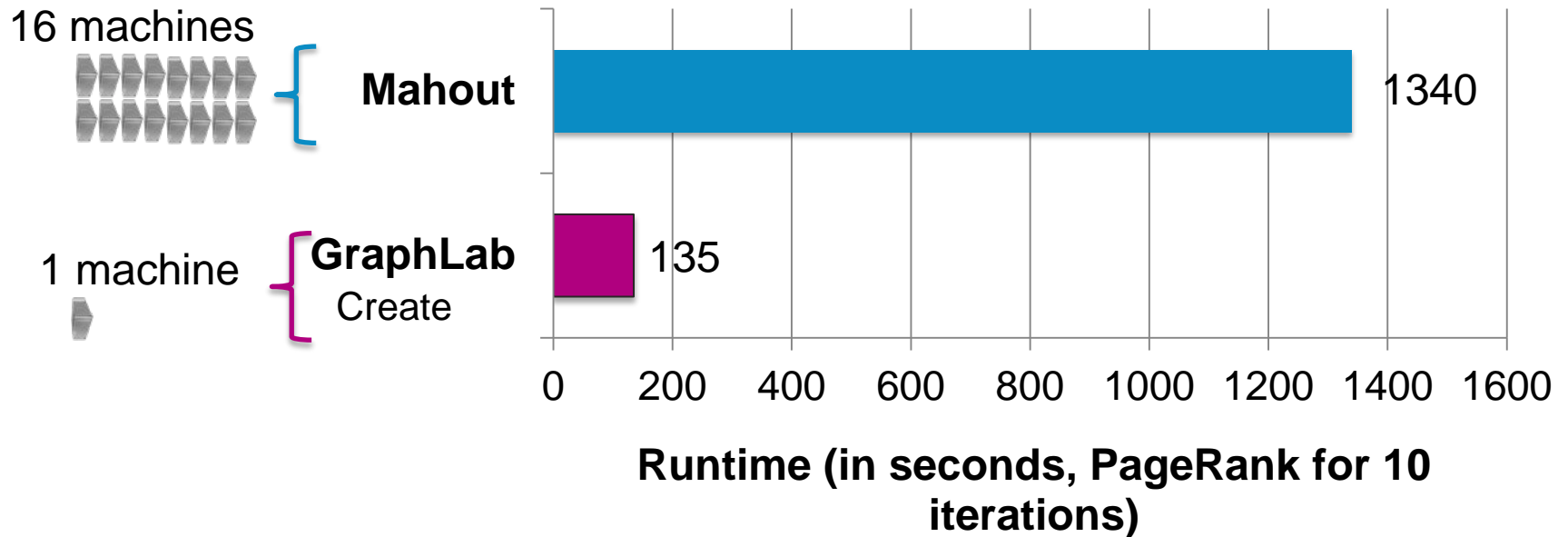
- Mohan Reddy, Chief Architect, The Hive LLC.



Amazon ratings data: 35M ratings, 6.6M users, 2.5M products
Replicated synthetically WRT users to evaluate scaling



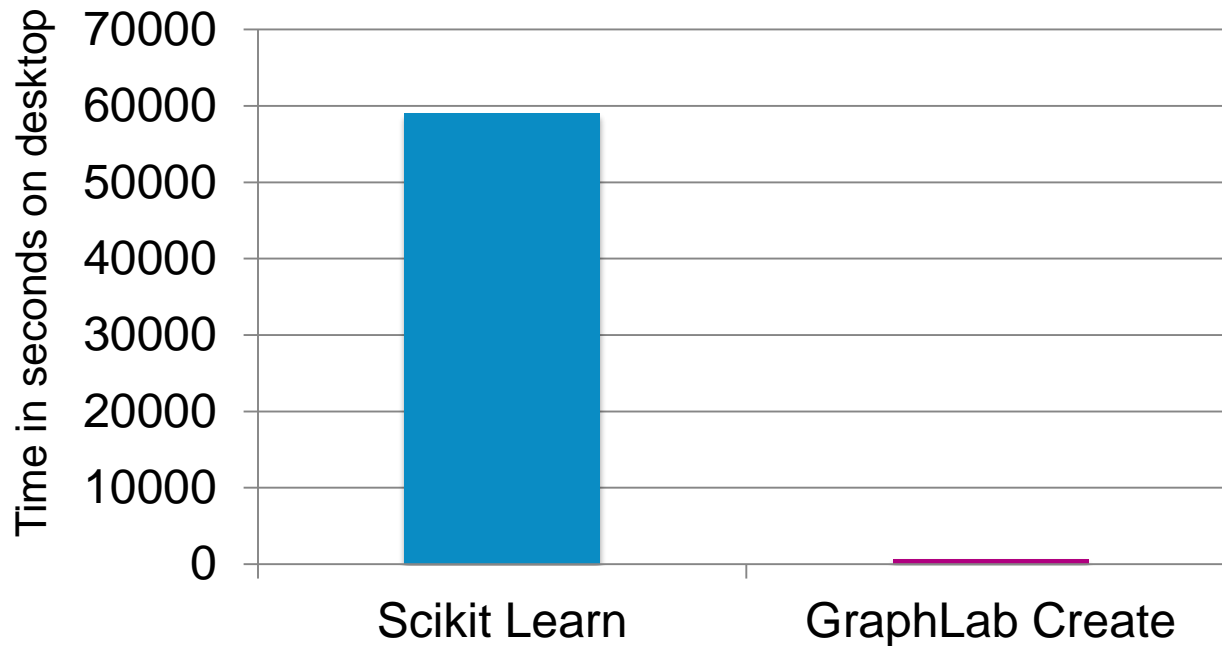
Finding influencers in the Live-Journal graph



GraphLab on 1 machine is 10x faster than Mahout on 16 machines



Logistic regression benchmark

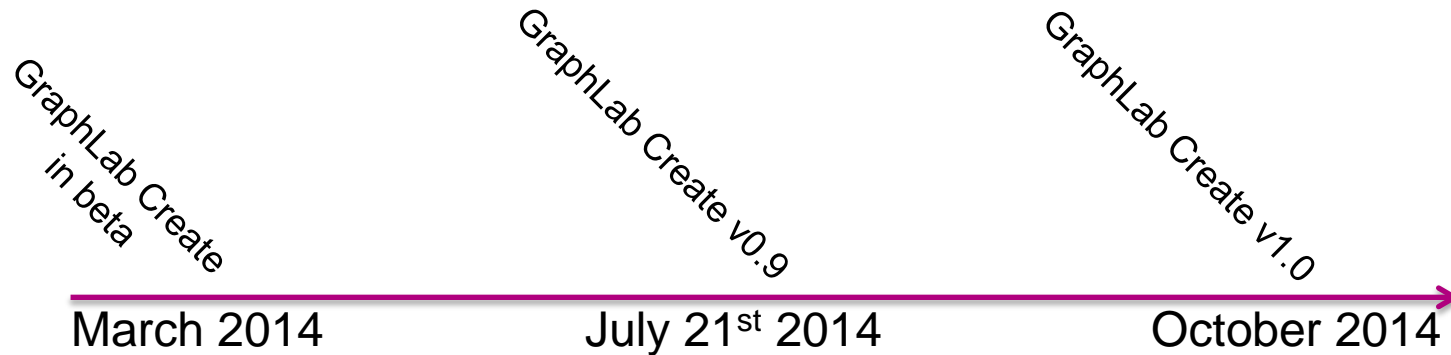


Orders of magnitude faster

KDD Cup data: predict student performance on math problems based on interactions with tutoring system
8.4M data points, 20M features, 2.4GB compressed



GraphLab Create Roadmap



Scalable data structures
Tables, graphs, text
Robust ML algorithms
GraphLab Canvas
Data pipelines

New ML algorithms
More data types
Predictive services
Monitoring in production
SDK

100+ companies participated in beta program
Already used in production
Extremely positive feedback

Every feature since March in response to customer requests
Please keep them coming!



Commitment to open-source

- We have been committed to open-source for 6 years
 - PowerGraph, GraphChi,...
 - Our focus now is on GraphLab Create
- We are inspired by companies like MongoDB & ElasticSearch
 - Open-source core
 - Provide value-add tools, such as monitoring & management

Our users can be successful by just using open-source version



GraphLab Create: Unleashing data science from inspiration to production



Have we forgotten a family member?

GraphLab₂

**Distributed Graph
Processing System**

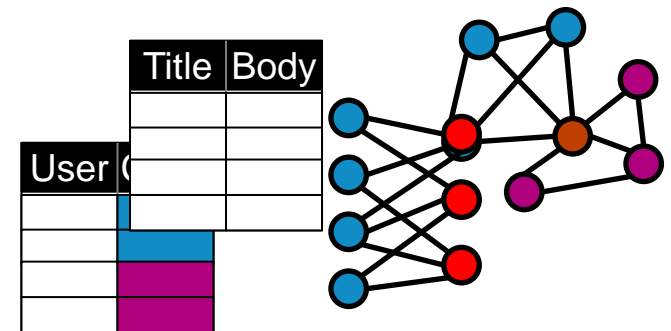
How Fast Can we Go?



**Disk/SSD graph
processing System**

How Large Can we Go?

?



Return of Distributed ML



Different Reasons to Distribute

- IO Bound ML
 - ex: 10TB of data, low compute per element
 - Batch Optimization, etc
 - Classical “Big Data” Tasks
- Compute Bound ML
 - Small working set (GBs), high compute
 - Probabilistic Inference, Non-Convex Optimization, etc.
 - Classical “Super Computer” Tasks



There is not one unique abstraction for Distributed ML

Iterative Dataflow



Dense Iterative

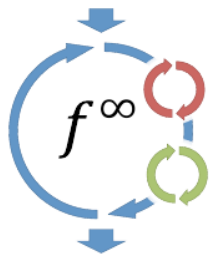


Supercomputer

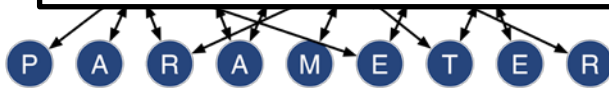


Charm++

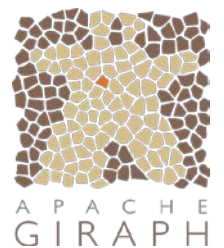
Logic Programming



Bloom



Combinatorial
BLAS



Faster? More Accurate?

Single Machine
Implementation of
Complex Algorithm



Distributed
Implementation of
Simple Algorithm

Matrix Factorization For Recommender

SGD

Gets Better Answers

Optimization

- Auto Stepsize Tuning
- AdaGrad and other varieties

Model

- user-item features w/quadratic interaction
- Adaptive Negative Sampling

ALS

Easy to Distribute



Many Things We Don't Know How to Distribute (well)

- Mixed Dense-Sparse Optimization (SGD?/Coordinate Descent?)
- High Order Tensor Factorization (Factorization Machine)
- Many Probabilistic Graphical Models
- etc.



Distributed ML API

Provide an architecture which enables GraphLab and other researchers to attack distributed ML.

“STL” for Distributed ML

Generic Data Structures

DMap<key, value> → Parameter Server
DGraph<v,e> → PowerGraph
DArray<value_type> → Data set
DMultiMap<key, value> → ?Connection
Machine?
etc.

Traits

SequentialReadable
RandomReadable
RandomWritable
EventualConsistent
etc.

Generic Algorithms

copy
for_each
reduce
transform
vertex_apply



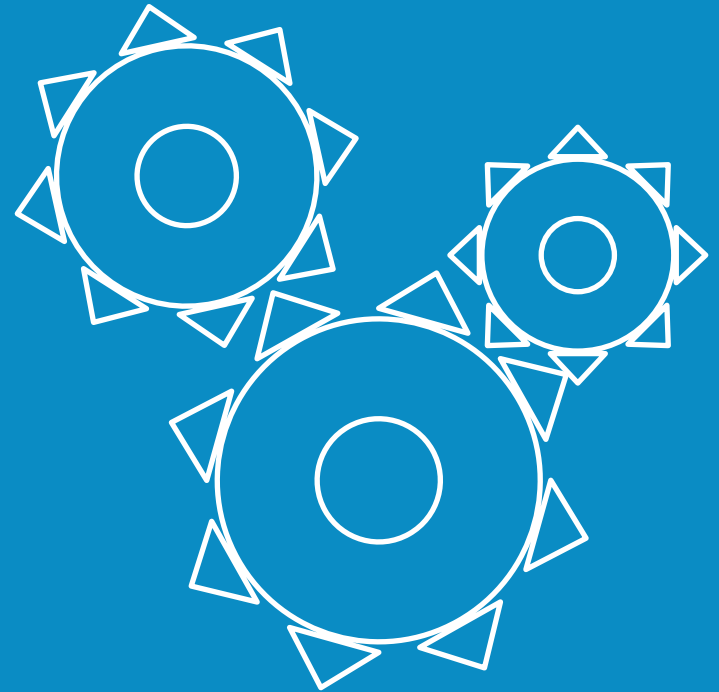
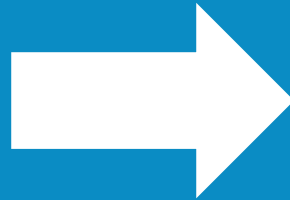
```
double AsyncSGD(gl::DArray<T> data) {  
    // a parameter server  
    gl::DMap<int, double> params;  
    while(1) {  
        gl::for_each(data, [](auto t) {  
            // modifies parameters  
            // asynchronously  
            update_sgd_step(params, t);  
        });  
        auto loss = gl::reduce(data, [](auto t) {  
            return loss(params, t);  
        });  
        if (converged(loss)) return loss;  
    }  
}
```



Very Initial Work

Come chat if you are interested.





`pip install graphlab-create`

`jobs@graphlab.com`

`@graphlabteam`

