

Building Data Products with Hadoop

Sam Shah

LinkedIn

February 2, 2011



WHAT IS A DATA PRODUCT?

A product whose mission is to allow interaction with data in an interesting and meaningful way.

DATA PRODUCTS AT LINKEDIN

People you may know



Jay Kreps, Principle Engineering Manger at LinkedIn



[+](#) Connect



Jeremy Gillick, Senior Web Developer at LinkedIn



[+](#) Connect



Albert Wang, User Experience Design at LinkedIn



[+](#) Connect

[See more »](#)

DATA PRODUCTS AT LINKEDIN

Profile Stats Pro

Last 90 Days Aug 17, 2009 - Nov 17, 2009 [Setting](#)

Who's Viewed My Profile

TODAY



David Feldman 
Director, User Experience Design
San Francisco Bay Area | Internet
In Common: [1 shared connection](#)



Someone in the Art & Design function in the Computer Software Industry



Kendra Shimmell 
Design Research & User Experience Design, Director
Columbus, Ohio Area | Design
In Common: [1 shared connection](#) [3 shared groups](#)

YESTERDAY




Yevgeniy Urdenko 
Experienced Electronic Hardware Designer
San Francisco Bay Area | Consumer Electronics
In Common: [1 shared connection](#)



Someone in the Art & Design function in the Finance Industry from San Francisco Bay Area



Tom Schafer, MBA (OD), MLS (HR) 
Organization Design, Organizational Development,
Organizational Effectiveness, Learning, Talent &
Leadership Development
Dallas/Fort Worth Area | Management Consulting

LAST WEEK



Software Engineer at Microsoft

Trends

Visits [Appear in Search](#)

Total Visits 76



[Get tips for a better profile](#)

Top Search Keywords

1. Design	45%
2. UI design	21%
3. User Experience	11%
4. Flash	8%
5. Flex	3%
6. Action Script 3	1%

Viewers by Geography



Top Country

Visits

DATA PRODUCTS AT LINKEDIN

Viewers of this profile also viewed...

[Russell Jurney](#)

Data Prodineer at LinkedIn

[Eric Tschetter](#)

Squirrelly One

[Florian Leibert](#)

Software Engineer, Research at Twitter

[Lili Wu](#)

Senior Software Engineer at LinkedIn

[Sam Rash](#)

Software Engineer at Facebook

[Jay Kreps](#)

Principal Engineer and Engineering...

[David Phillips](#)

Software Engineer

[Anmol Bhasin](#)

Engineering Manager and Senior...

[Baq Haidri](#)

Senior Software Engineer at LinkedIn

[Pierre-Alexandre Meyer](#)

Software Engineer - Analytics at Ning

Professionals that recommend this product also recommend...

- [LinkedIn InMaps](#)
- [LinkedIn Custom Groups](#)
- [LinkedIn Company Pages](#)
- [LinkedIn Jobs](#)
- [LinkedIn Mobile](#)

People Who Viewed This Job Also Viewed

- [Marketing Manager, Customer Engagement at LinkedIn](#)
- [Marketing Manager, Online Channels at LinkedIn](#)
- [Relationship Manager, SMB at LinkedIn](#)
- [Research Associate at LinkedIn](#)
- [Product Marketing Lead, Corporate Recruiting Solutions](#)

Look before you leap
○○○○○○

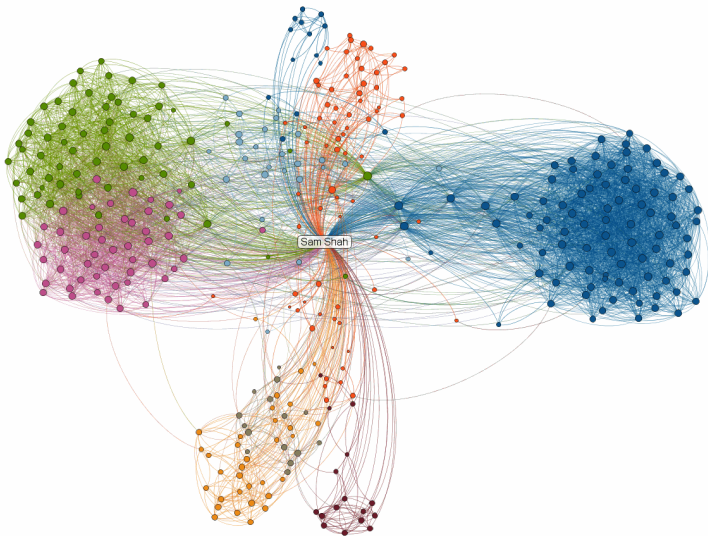
Tall oaks grow from little acorns
○○○○

Don't put the cart before the horse
○○

A stitch in time saves nine
○○○

Half a loaf is better than none
○○○○○○

DATA PRODUCTS AT LINKEDIN



OVERVIEW

- ▶ Take something that runs once...
 - ▶ ...and run it multiple times
 - ▶ ...and serve it at scale
 - ▶ ...and iterate quickly

OVERVIEW

- ▶ Take something that runs once...
 - ▶ ...and run it multiple times
 - ▶ ...and serve it at scale
 - ▶ ...and iterate quickly

Using off-the-shelf open source components

WHO AM I?

Sam Shah



Sr. Software Engineer at LinkedIn

San Francisco Bay Area | Computer Software



Current

- **Sr. Software Engineer at LinkedIn**

Past

- Platform Engineer (Analytics) at Ning
- Research Assistant at University of Michigan

Education

- University of Michigan

Recommendations

1 person has recommended Sam

Connections

393 connections

Websites

- [Personal Website](#)

Public Profile

<http://www.linkedin.com/in/shahsam>

Look before you leap

○○○○○○

Tall oaks grow from little acorns

○○○○

Don't put the cart before the horse

○○

A stitch in time saves nine

○○○

Half a loaf is better than none

○○○○○○

PROVERBS

Look before you leap

Tall oaks grow from little acorns

Don't put the cart before the horse

A stitch in time saves nine

Half a loaf is better than none

LOOK BEFORE YOU LEAP...

Do you have a data product?

- ▶ Involves ingenuity & creativity
- ▶ Research & analysis

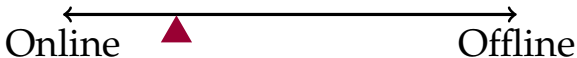
LOOK BEFORE YOU LEAP...

Do you have a data product?

- ▶ Involves ingenuity & creativity
- ▶ Research & analysis

Let's build "People You May Know"

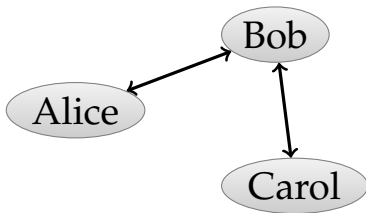
ONLINE VS. OFFLINE



- ▶ Sub-second processing
- ▶ Harder to scale
- ▶ Must handle failures gracefully
- ▶ Computationally intensive
- ▶ Easier to scale
- ▶ Easier to tolerate failures
- ▶ Iterate quickly

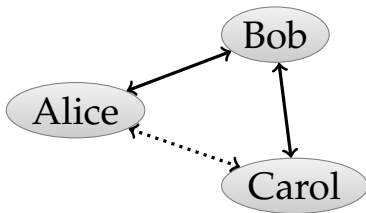
PEOPLE YOU MAY KNOW

How do people know each other?



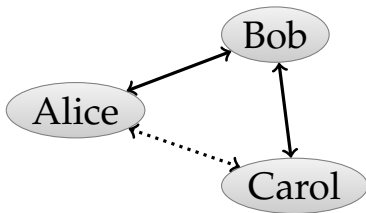
PEOPLE YOU MAY KNOW

How do people know each other?



PEOPLE YOU MAY KNOW

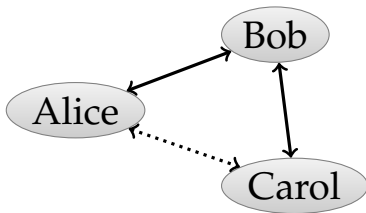
How do people know each other?



Known as triangle closing

PEOPLE YOU MAY KNOW

How do people know each other?



Known as triangle closing

$P(\text{know person}) \propto \# \text{ people known in common}$

TRIANGLE CLOSING IN PIG

```

/* connections contains (source_id, dest_id) bidirectional pairs */
connections = LOAD 'connections' USING AvroStorage();
by_member = GROUP connections BY source_id;
by_member = FOREACH by_member GENERATE
    generatePairs(connections.dest_id) AS (id1,id2);

fof = GROUP by_member BY (id1, id2);
fof = FOREACH fof GENERATE
    flatten(group) AS (source_id, dest_id),
    COUNT(by_member) AS common_neighbors;

STORE fof INTO 'closed-triangles' USING AvroStorage();

```

TRIANGLE CLOSING IN PIG

```

/* connections contains (source_id, dest_id) bidirectional pairs */
connections = LOAD 'connections' USING AvroStorage();
by_member = GROUP connections BY source_id;
by_member = FOREACH by_member GENERATE
    generatePairs(connections.dest_id) AS (id1,id2);

fof = GROUP by_member BY (id1, id2);
fof = FOREACH fof GENERATE
    flatten(group) AS (source_id, dest_id),
    COUNT(by_member) AS common_neighbors;

STORE fof INTO 'closed-triangles' USING AvroStorage();

```

TRIANGLE CLOSING IN PIG

```

/* connections contains (source_id, dest_id) bidirectional pairs */
connections = LOAD 'connections' USING AvroStorage();
by_member = GROUP connections BY source_id;
by_member = FOREACH by_member GENERATE
    generatePairs(connections.dest_id) AS (id1,id2);

fof = GROUP by_member BY (id1, id2);
fof = FOREACH fof GENERATE
    flatten(group) AS (source_id, dest_id),
    COUNT(by_member) AS common_neighbors;

STORE fof INTO 'closed-triangles' USING AvroStorage();

```

TRIANGLE CLOSING IN PIG

```

/* connections contains (source_id, dest_id) bidirectional pairs */
connections = LOAD 'connections' USING AvroStorage();
by_member = GROUP connections BY source_id;
by_member = FOREACH by_member GENERATE
    generatePairs(connections.dest_id) AS (id1,id2);

fof = GROUP by_member BY (id1, id2);
fof = FOREACH fof GENERATE
    flatten(group) AS (source_id, dest_id),
    COUNT(by_member) AS common_neighbors;

STORE fof INTO 'closed-triangles' USING AvroStorage();

```

TRIANGLE CLOSING IN PIG

```

/* connections contains (source_id, dest_id) bidirectional pairs */
connections = LOAD 'connections' USING AvroStorage();
by_member = GROUP connections BY source_id;
by_member = FOREACH by_member GENERATE
    generatePairs(connections.dest_id) AS (id1,id2);

fof = GROUP by_member BY (id1, id2);
fof = FOREACH fof GENERATE
    flatten(group) AS (source_id, dest_id),
    COUNT(by_member) AS common_neighbors;

STORE fof INTO 'closed-triangles' USING AvroStorage();

```

TRIANGLE CLOSING IN PIG

```

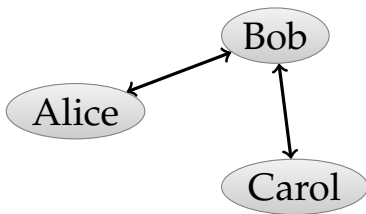
/* connections contains (source_id, dest_id) bidirectional pairs */
connections = LOAD 'connections' USING AvroStorage();
by_member = GROUP connections BY source_id;
by_member = FOREACH by_member GENERATE
    generatePairs(connections.dest_id) AS (id1,id2);

fof = GROUP by_member BY (id1, id2);
fof = FOREACH fof GENERATE
    flatten(group) AS (source_id, dest_id),
    COUNT(by_member) AS common_neighbors;

STORE fof INTO 'closed-triangles' USING AvroStorage();

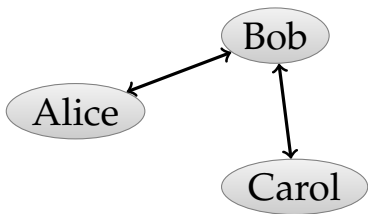
```

TRIANGLE CLOSING EXAMPLE



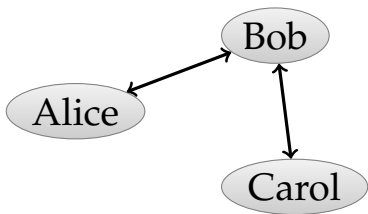
1. $\{A \rightarrow B, B \rightarrow A, B \rightarrow C, C \rightarrow B\}$

TRIANGLE CLOSING EXAMPLE



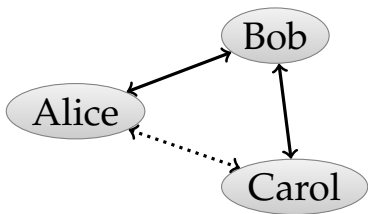
1. $\{A \rightarrow B, B \rightarrow A, B \rightarrow C, C \rightarrow B\}$
2. $A : \{B\}, B : \{A, C\}, C : \{B\}$

TRIANGLE CLOSING EXAMPLE



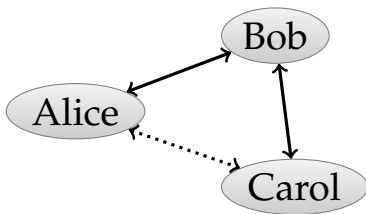
1. $\{A \rightarrow B, B \rightarrow A, B \rightarrow C, C \rightarrow B\}$
2. $A : \{B\}, B : \{A, C\}, C : \{B\}$

TRIANGLE CLOSING EXAMPLE



1. $\{A \rightarrow B, B \rightarrow A, B \rightarrow C, C \rightarrow B\}$
2. $A : \{B\}, B : \{A, C\}, C : \{B\}$
3. $B : \{A \rightarrow C, C \rightarrow A\}$

TRIANGLE CLOSING EXAMPLE



1. $\{A \rightarrow B, B \rightarrow A, B \rightarrow C, C \rightarrow B\}$
2. $A : \{B\}, B : \{A, C\}, C : \{B\}$
3. $B : \{A \rightarrow C, C \rightarrow A\}$
4. $\{(A \rightarrow C, 1); (C \rightarrow A, 1)\}$

Look before you leap

○○○○●

Tall oaks grow from little acorns

○○○○

Don't put the cart before the horse

○○

A stitch in time saves nine

○○○

Half a loaf is better than none

○○○○○

OUR WORKFLOW



triangle-closing

Look before you leap

○○○○●

Tall oaks grow from little acorns

○○○○

Don't put the cart before the horse

○○

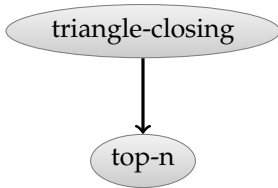
A stitch in time saves nine

○○○

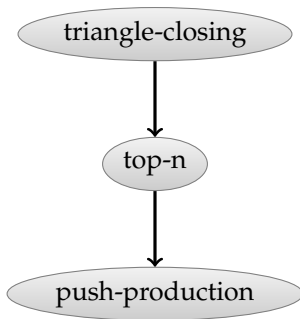
Half a loaf is better than none

○○○○○

OUR WORKFLOW



OUR WORKFLOW



Look before you leap
○○○○○

Tall oaks grow from little acorns
○○○○

Don't put the cart before the horse
○○

A stitch in time saves nine
○○○

Half a loaf is better than none
○○○○○

PROVERBS

Look before you leap

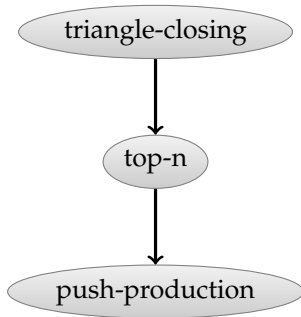
Tall oaks grow from little acorns

Don't put the cart before the horse

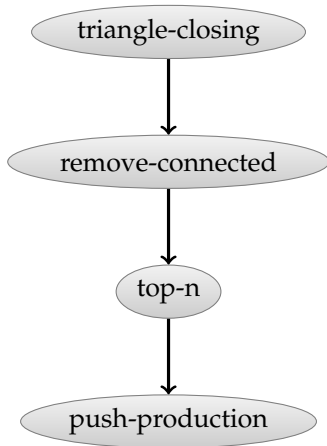
A stitch in time saves nine

Half a loaf is better than none

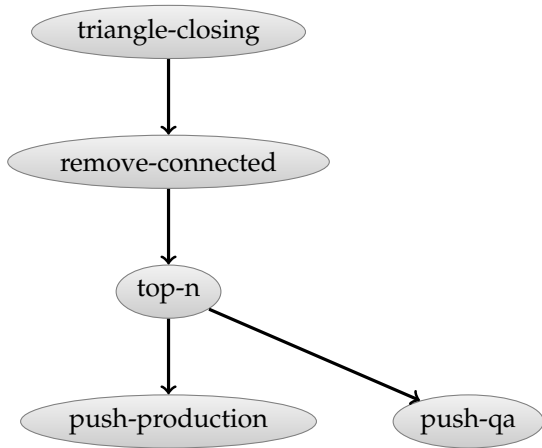
OUR WORKFLOW



OUR WORKFLOW



OUR WORKFLOW



Look before you leap

○○○○○○

Tall oaks grow from little acorns

●○○○

Don't put the cart before the horse

○○

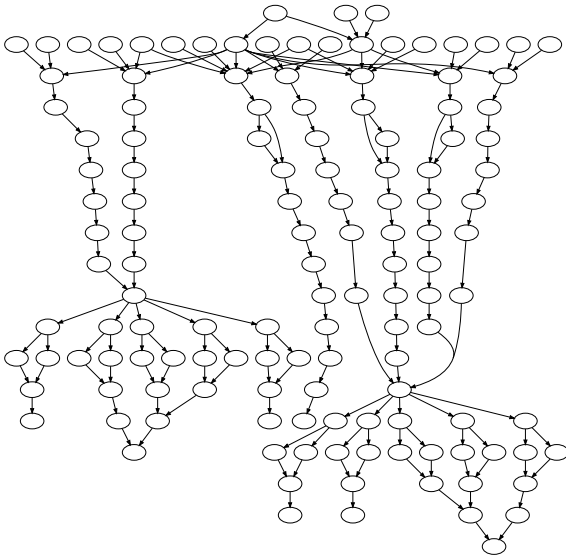
A stitch in time saves nine

○○○

Half a loaf is better than none

○○○○○○

PYMK WORKFLOW



WHAT WE NEED...

- ▶ Dependency management
- ▶ Diverse job types
- ▶ Scheduling
- ▶ Monitoring
- ▶ Configuration
- ▶ Retry/restart on failure
- ▶ Resource locking
- ▶ Log collection
- ▶ Historical information

WHAT WE NEED...

- ▶ Dependency management
- ▶ Diverse job types
- ▶ Scheduling
- ▶ Monitoring
- ▶ Configuration
- ▶ Retry/restart on failure
- ▶ Resource locking
- ▶ Log collection
- ▶ Historical information



Azkaban



Oozie

EXAMPLE AZKABAN JOB SPEC

```
type=pig
pig.script=top-n.pig

param.N=50

dependencies=triangle-closing
```

Look before you leap
○○○○○○

Tall oaks grow from little acorns
○○●

Don't put the cart before the horse
○○

A stitch in time saves nine
○○○

Half a loaf is better than none
○○○○○○



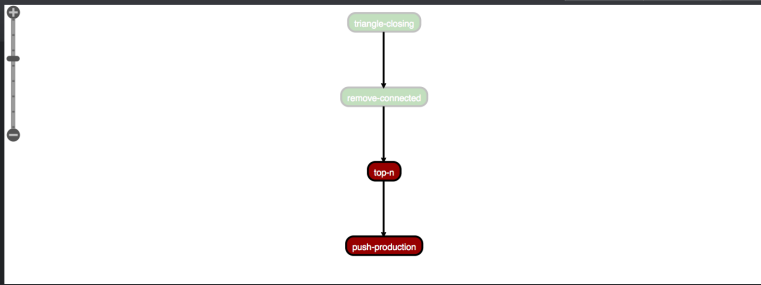
[Home](#) [Create Job](#) [Upload Job](#) [History](#) [HDFS](#)

Flow Instance

Name: push-production Flow ID: 2304

Search

Start Time	End Time	Period
02-01-2011 11:29:04	02-01-2011 11:29:10	0 minutes



Execute

Current Time: 02-01-2011 14:05:49 PST

Look before you leap

○○○○○○

Tall oaks grow from little acorns

○○●

Don't put the cart before the horse

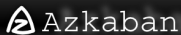
○○

A stitch in time saves nine

○○○

Half a loaf is better than none

○○○○○○



Home

Running push-production

Scheduled Jobs

Name	Next Execution	Period	
push-production	02-02-2011 12:00:00 PST	1 day, 0 minutes	Remove

Executing Jobs

Name	Start Time	Period	
push-production	02-01-2011 13:57:34 (0 minutes ago)	-	Cancel

All Jobs

Run immediately (with dependencies) [Run](#)

Schedule to run at : on and repeat every [Schedule](#)

»

Look before you leap

○○○○○○

Tall oaks grow from little acorns

○○●

Don't put the cart before the horse

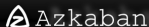
○○

A stitch in time saves nine

○○○

Half a loaf is better than none

○○○○○○

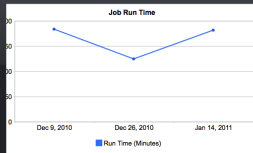


Home Create Job Upload Job History HDFS

Job Details

Details Logs

Previous Runtimes



Job History

Name	Started	Ended	Elapsed	Completed Successfully?	Log
pymk-triangle-closing	01-14-2011 13:59:35 PST	01-14-2011 17:01:52 PST	3 hours, 2 minutes	yes	log
pymk-triangle-closing	12-26-2010 23:21:44 PST	12-27-2010 01:27:19 PST	2 hours, 5 minutes	yes	log
pymk-triangle-closing	12-16-2010 08:06:33 PST	12-16-2010 13:06:23 PST	4 hours, 59 minutes	no	log
pymk-triangle-closing	12-10-2010 21:57:07 PST	12-10-2010 21:57:31 PST	0 minutes	no	log
pymk-triangle-closing	12-09-2010 20:39:28 PST	12-09-2010 23:44:20 PST	3 hours, 4 minutes	yes	log

Edit Run Run with Dependencies

Current Time: 02-01-2011 14:20:29 PST

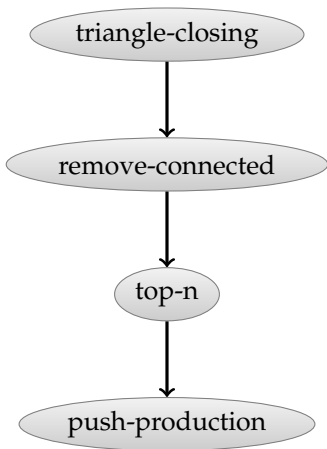
Look before you leap
○○○○○○

Tall oaks grow from little acorns
○○○○

Don't put the cart before the horse
○○

A stitch in time saves nine
○○○

Half a loaf is better than none
○○○○○○



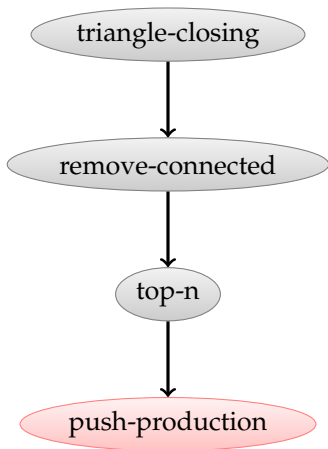
Look before you leap
○○○○○○

Tall oaks grow from little acorns
○○○○

Don't put the cart before the horse
○○

A stitch in time saves nine
○○○

Half a loaf is better than none
○○○○○○



Look before you leap
○○○○○

Tall oaks grow from little acorns
○○○○

Don't put the cart before the horse
○○

A stitch in time saves nine
○○○

Half a loaf is better than none
○○○○○

PROVERBS

Look before you leap

Tall oaks grow from little acorns

Don't put the cart before the horse

A stitch in time saves nine

Half a loaf is better than none

STORAGE SYSTEM CONSIDERATIONS

- ▶ Cost of data load?
- ▶ Data stored per node? Response time?
- ▶ Fail-over?
- ▶ How to transfer?
- ▶ Versioning & rollback?

STORAGE SYSTEM CONSIDERATIONS

- ▶ Cost of data load?
- ▶ Data stored per node? Response time?
- ▶ Fail-over?
- ▶ How to transfer?
- ▶ Versioning & rollback?



Voldemort

VOLDEMORT RO EXTENSIONS

- ▶ Partition data over cluster of machines
- ▶ Build store offline in Hadoop
- ▶ Fast query times
- ▶ Replication with automatic failover
- ▶ Versions new datasets & allows rollback

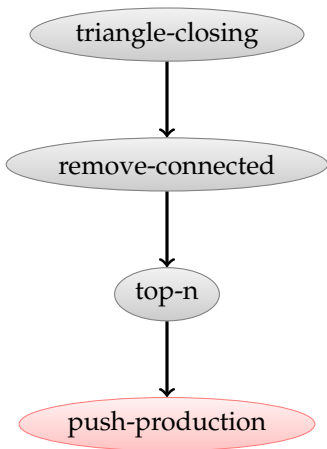
Look before you leap
○○○○○○

Tall oaks grow from little acorns
○○○○

Don't put the cart before the horse
○○

A stitch in time saves nine
○○○

Half a loaf is better than none
○○○○○○



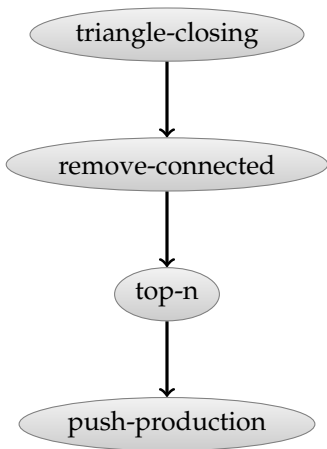
Look before you leap
○○○○○○

Tall oaks grow from little acorns
○○○○

Don't put the cart before the horse
○○

A stitch in time saves nine
○○○

Half a loaf is better than none
○○○○○○



Look before you leap
○○○○○○

Tall oaks grow from little acorns
○○○○

Don't put the cart before the horse
○○

A stitch in time saves nine
○○○

Half a loaf is better than none
○○○○○○

PROVERBS

Look before you leap

Tall oaks grow from little acorns

Don't put the cart before the horse

A stitch in time saves nine

Half a loaf is better than none

VERIFICATION

Data issues notoriously hard to solve

- ▶ Check your invariants
- ▶ Check your invariants
 - ▶ Before & after processing
 - ▶ Statistical checks helpful too

VERIFICATION

Data issues notoriously hard to solve

- ▶ Check your invariants
- ▶ Check your invariants
 - ▶ Before & after processing
 - ▶ Statistical checks helpful too

What if connections data contained duplicates?

DATA QUALITY

- ▶ Create dataset with explanatory info
 - ▶ Push to QA cluster
 - ▶ Viewer applications—EXPLAIN command
- ▶ Versioning
 - ▶ Have ability to rollback quickly
- ▶ Unit tests!

PROFILE YOUR JOBS

```
static class TheReducer extends /* ... */ {  
    @Override  
    public void configure(JobConf job) {  
        myData = processFromDistributedCache();  
        /* ... */  
    }  
    /* ... */  
}
```

```
conf.setCombinerClass(TheReducer.class);
```

- ▶ Ran in production for a while

PROFILE YOUR JOBS

```

static class TheReducer extends /* ... */ {
  @Override
  public void configure(JobConf job) {
    myData = processFromDistributedCache();
    /* ... */
  }
  /* ... */
}

```

```

conf.setCombinerClass(TheReducer.class);

```

- ▶ Ran in production for a while
- ▶ $m \cdot r$ reads from distributed cache
 - ▶ $m = 5000, r = 2000 \implies$ ten million calls!
 - ▶ $\approx 50\%$ overhead of job

Look before you leap
○○○○○○

Tall oaks grow from little acorns
○○○○

Don't put the cart before the horse
○○

A stitch in time saves nine
○○○

Half a loaf is better than none
○○○○○○

PROVERBS

Look before you leap

Tall oaks grow from little acorns

Don't put the cart before the horse

A stitch in time saves nine

Half a loaf is better than none

IMPROVING PERFORMANCE

- ▶ **Symmetry**
 - ▶ Alice knows Carol, Carol knows Alice

IMPROVING PERFORMANCE

- ▶ **Symmetry** 50% speedup
 - ▶ Alice knows Carol, Carol knows Alice

TRIANGLE CLOSING IN PIG

```
/* connections contains (source_id, dest_id) bidirectional pairs */
connections = LOAD 'connections' USING AvroStorage();
by_member = GROUP connections BY source_id;
by_member = FOREACH by_member GENERATE
    generatePairs(connections.dest_id) AS (id1,id2);

fof = GROUP by_member BY (id1, id2);
fof = FOREACH fof GENERATE
    flatten(group) AS (source_id, dest_id),
    COUNT(by_member) AS common_neighbors;

STORE fof INTO 'closed-triangles' USING AvroStorage();
```

Barack Obama 2nd

President of the United States of America



Washington D.C. Metro Area | Government Administration



Current

- **President at United States of America**

Past

- US Senator at US Senate (IL-D) 
- State Senator at Illinois State Senate
- Senior Lecturer in Law at University of Chicago Law School 

Education

- Harvard University
- Columbia University in the City of New York
- Occidental College

Connections

500+ connections

Websites

- [White House website](#)
- [Join Barack's LinkedIn Group](#)
- [BarackObama.com](#)

Public Profile

<http://www.linkedin.com/in/barackobama>

IMPROVING PERFORMANCE

- ▶ **Symmetry** 50% speedup
 - ▶ Alice knows Carol, Carol knows Alice
- ▶ **Limiting**
 - ▶ Ignore members with $> k$ connections

IMPROVING PERFORMANCE

- ▶ **Symmetry** 50% speedup
 - ▶ Alice knows Carol, Carol knows Alice
- ▶ **Limiting**
 - ▶ Ignore members with $> k$ connections
- ▶ **Sampling**
 - ▶ Randomly choose up to k connections

IMPROVING PERFORMANCE

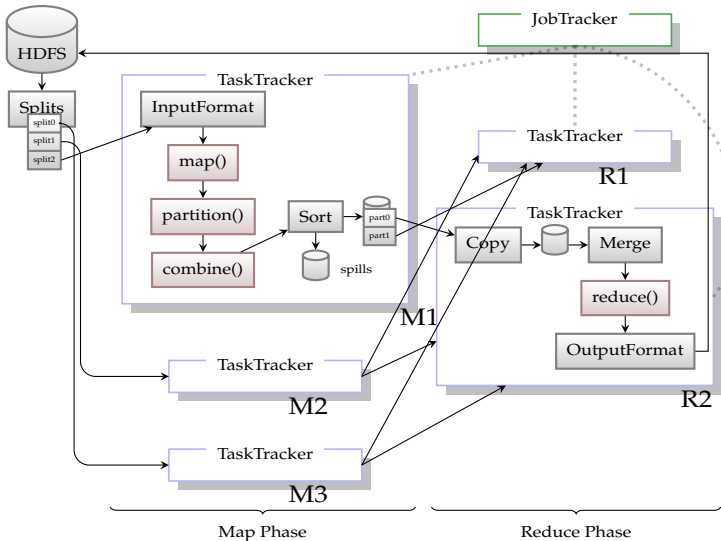
- ▶ **Symmetry** Structural (incl. Incremental)
 - ▶ Alice knows Carol, Carol knows Alice
- ▶ **Limiting** Bounding
 - ▶ Ignore members with $> k$ connections
- ▶ **Sampling** Probabilistic
 - ▶ Randomly choose up to k connections

IMPROVING PERFORMANCE

- ▶ **Symmetry** Structural (incl. Incremental)
 - ▶ Alice knows Carol, Carol knows Alice
- ▶ **Limiting** Bounding
 - ▶ Ignore members with $> k$ connections
- ▶ **Sampling** Probabilistic
 - ▶ Randomly choose up to k connections

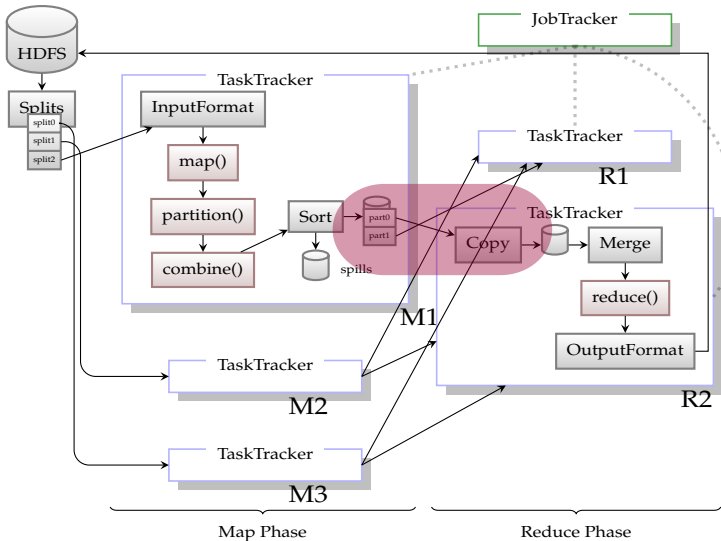
PROBABILISTIC FILTERING

- ▶ Biggest bottleneck is intermediate I/O



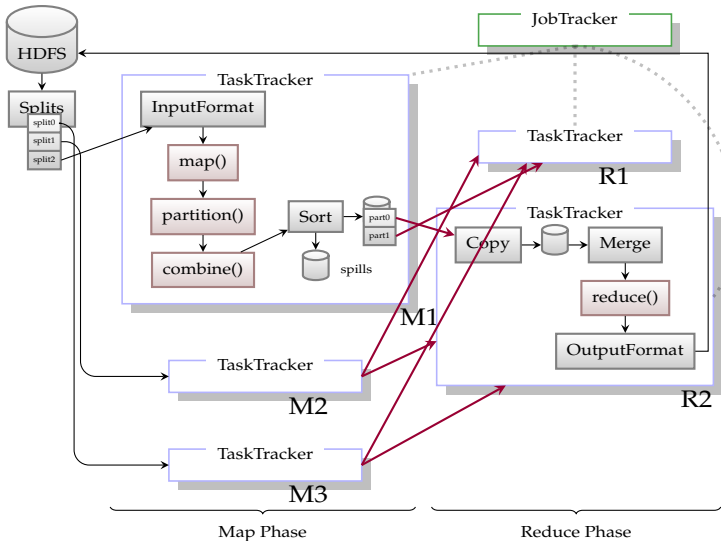
PROBABILISTIC FILTERING

- ▶ Biggest bottleneck is intermediate I/O

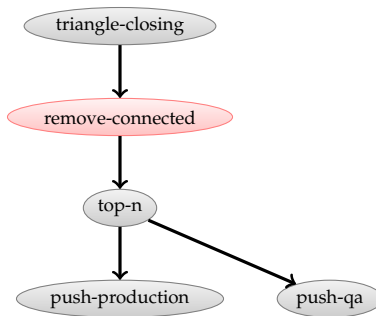


PROBABILISTIC FILTERING

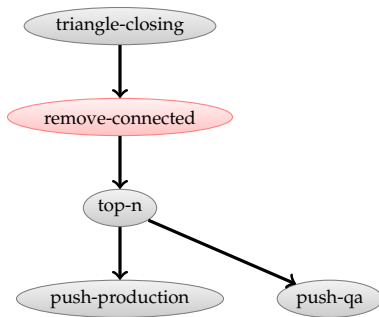
- ▶ Biggest bottleneck is intermediate I/O



PROBABILISTIC FILTERING



PROBABILISTIC FILTERING



- ▶ Build Bloom filter of existing connections
 - ▶ No false negatives
 - ▶ Some false positives
- ▶ Filter map-side

TODAY...

- ▶ Take something that runs once...
 - ▶ ...and run it multiple times
 - ▶ ...and serve it at scale
 - ▶ ...and iterate quickly

SNA TEAM

<http://sna-projects.com>
<http://sna-projects.com/blog>

Abhishek Gupta
Adam Smyczek
Adil Aijaz
Albert Wang
Alex Feinberg
Alexis Pribula
Ali Imam
Anmol Bhasin
Baoqiu Cui
Baq Haidri
Cheng-Tao Chu
Chris Conrad
Chris Riccomini
Christian Posse

DJ Patil
Daniel Tukelang
David McCutcheon
Eric Tschetter
Ethan Zhang
Fatih Emekci
Hao Yan
Heyning Cheng
Igor Perisic
Jakob Homan
Janet Ryu
Jay Kreps
Jeffrey Schang
Jill Chen

Jingwei Wu
Jiong Wang
John Wang
Jonathan Koren
Joseph Adler
Joshua Hartman
Jun Rao
Lili Wu
Lin Guo
Mathieu Bastian
Matthew Hayes
Micah Alpern
Monica Rogati
Neha Narkhede

Parul Jain
Paul Ogilvie
Peter Skomoroch
Ramesh Dommeti
Richard Park
Ron Bekkerman
Roshan Sumbaly
Rui Wang
Russell Journey
Sam Shah
Shannon Zhang
Utku Irmak
Wei Shen
Yasuhiro Matsuda