From Academia to Industry

Perspectives on research directions in large scale computation geared to industrial impact

P. Oscar Boykin | oscar@twitter.com | @posco

#hpdc

- Life in Academia
- Life in Industry
- The mismatch
- My wish-list problems
- Impacting Industry

In Academia:

- Assistant Prof. in ECE at Univ. of Fl.
- P2P Networks
- Self-configuring “grids”
- Virtual Networking
- ad-hoc resource discovery
- CPU-bound distributed computing

“Customers” in Academia

- Colleagues: review your papers
- Collaborators: systems researchers often work with some domain expert.
- Funding agencies: feed the triangle (funds -> students -> papers -> funds ...)
- Value ideas, generally put less weight on well engineered solutions.

These customers don’t place very similar demands to Industrial customers

Industry

Customers in Industry

- People paying to use your product, they will abandon if you do poor work.
- Teams dependent on your systems to deliver to the above. They can abandon too.
- New systems need to save time, money, and/or effort.

Why my work is not applicable at Twitter

- Constant code change: self configuring doesn't lower cost, but increases risk.

- IO-bound processing: my academic work was focused on smaller data, with bigger compute. We have HUGE data, but easy compute.

- Little attempt to understand the pain points of industry, assumption of unsophistication.

Ideas are cheap

- Large numbers of PhDs are employed by Twitter, Facebook, Google, etc.
- Everyone has ideas, most of them good, but our systems are still too immature to make ideas a bottleneck.
- Never-the-less: Academia could help solve more industry problems.

At Twitter

see code: http://github.com/twitter

- Co-developed, @scalding, a Scala Fuctional Programming API for Hadoop. Enables easy deploy of giant (90 or more) Map/Reduce pipelines.

- Leveraged abstract algebra (Monoids!) to isolate logic from systems in streaming compute and Matrix systems.

- Implemented sketching algorithms as Monoids to encourage fast+cheap approximations: github.com/twitter/algebird

Twitter Scale

Tens of Thousands Machines

Linux 2.6.39 mostly

Huge Hadoop Clusters for Analytics (tens of k jobs/day)

100TB+ of data digested daily

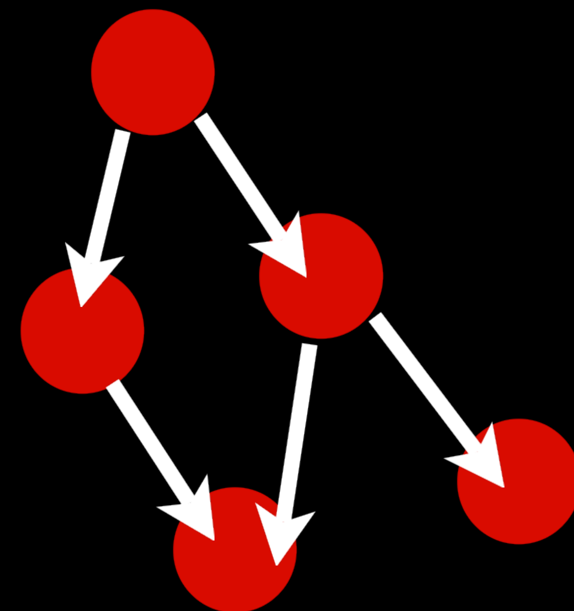
Real systems do real work.

My Wish-list

1: Data Pipelines

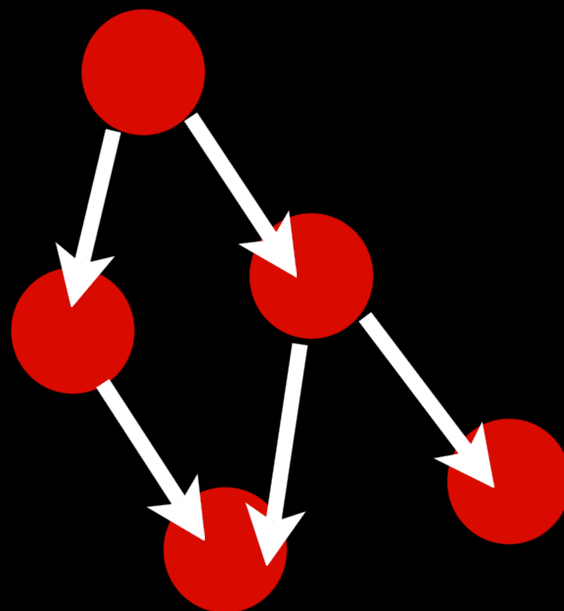
Elegant Model for Big Data Pipelines

- ~100 Terabytes a day
- Sometimes there are errors at ingest that can be repaired
- Some nodes are functions (which have bugs and are fixed)
- Sometimes we need to backfill



Elegant Model for Big Data Pipelines

- Time looks special, why? How to model?
- Immutable/write-once is appealing, but what about bugs?
- How to deal with outages in the pipeline? How to deal with priorities in recovery? Total Loss?
- How to alert when input

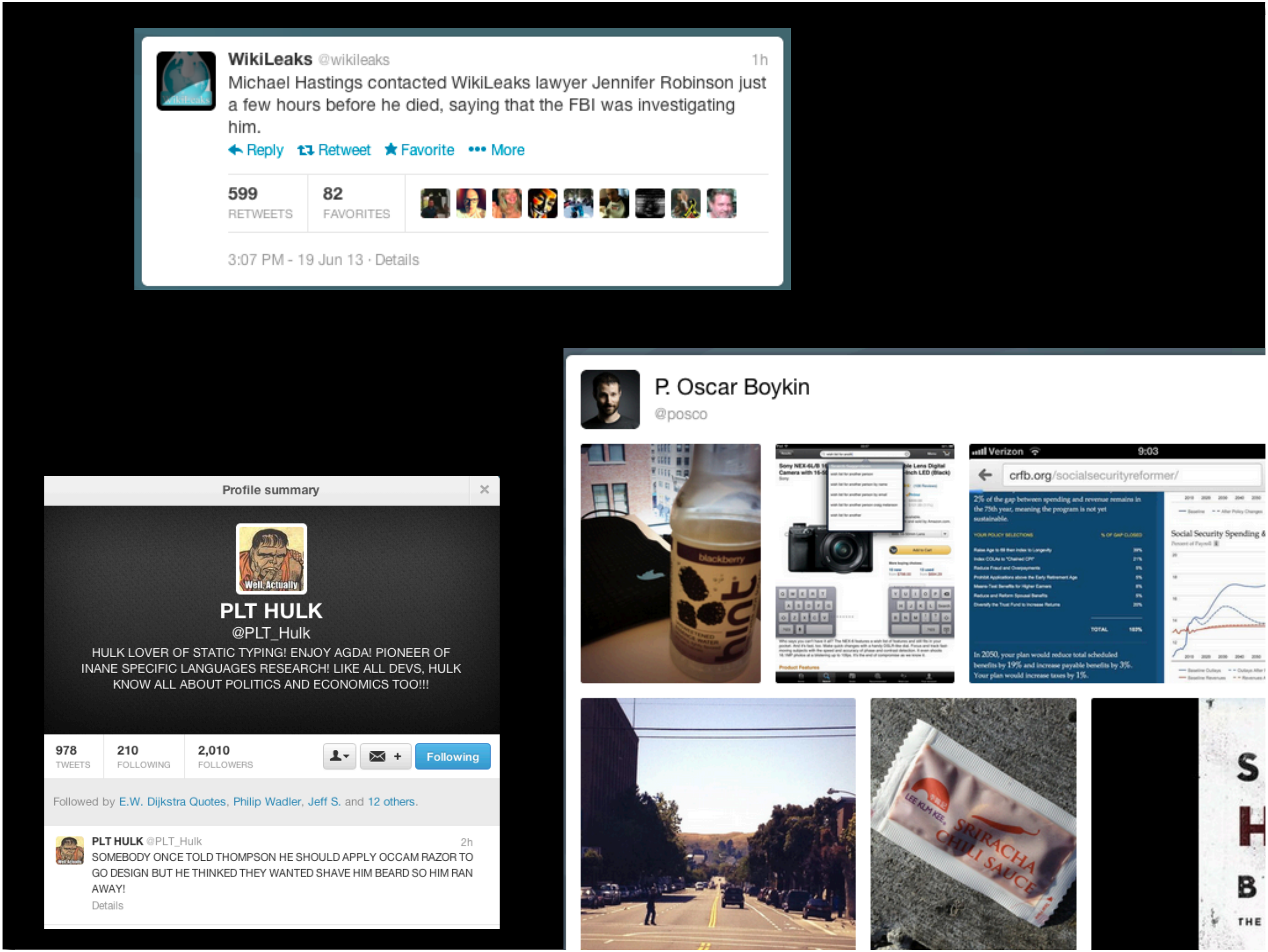


IDEAS ARE CHEAP!

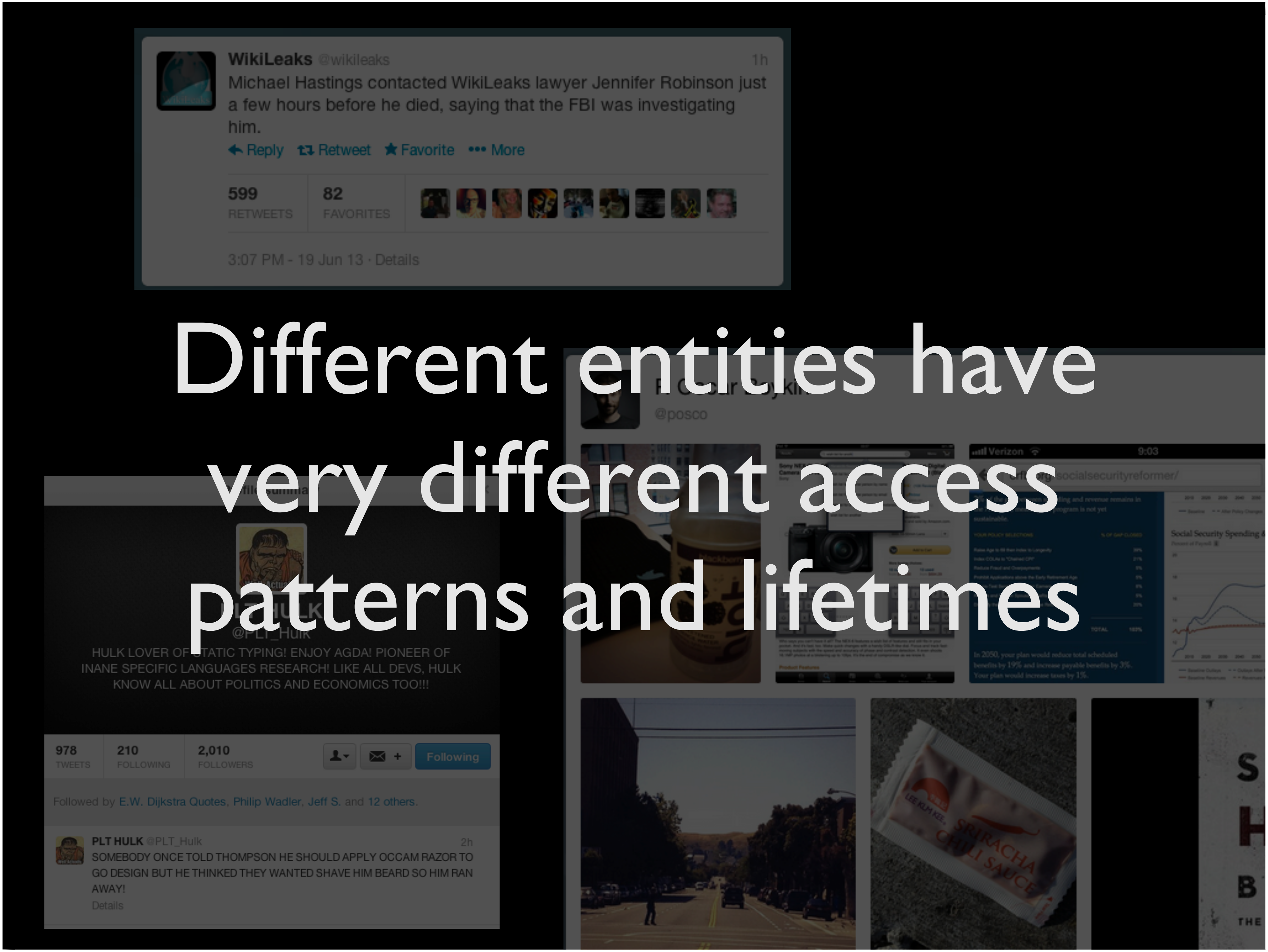
Give Us

- Theory, proofs, Iron-clad and large-scale experiments
- ***Real (and usable) code,*** i.e. @Amplab at Berkeley
- Collaboration: sending summer interns.
- Employees: PhD grads to solve hardcore systems problems with your research.

2: Storing and Serving Big Data


WikiLeaks @wikileaks 1h
Michael Hastings contacted WikiLeaks lawyer Jennifer Robinson just a few hours before he died, saying that the FBI was investigating him.
Reply Retweet Favorite More
599 RETWEETS
82 FAVORITES
3:07 PM - 19 Jun 13 · Details
P. Oscar Boykin
@posco
Verizon 9:03
crfb.org/socialsecurityreformer/
Social Security Spending &
SRIRACHA CHILI SAUCE
LEE KUM KEE
Profile summary
Well, Actually
PLT HULK
@PLT_Hulk
HULK LOVER OF STATIC TYPING! ENJOY AGDA! PIONEER OF INANE SPECIFIC LANGUAGES RESEARCH! LIKE ALL DEVS, HULK KNOW ALL ABOUT POLITICS AND ECONOMICS TOO!!!
978 TWEETS
210 FOLLOWING
2,010 FOLLOWERS
Following
Followed by E.W. Dijkstra Quotes, Philip Wadler, Jeff S. and 12 others.
PLT HULK @PLT_Hulk 2h
SOMEBODY ONCE TOLD THOMPSON HE SHOULD APPLY OCCAM RAZOR TO GO DESIGN BUT HE THINKED THEY WANTED SHAVE HIM BEARD SO HIM RAN AWAY!
Details

Different entities have very different access patterns and lifetimes

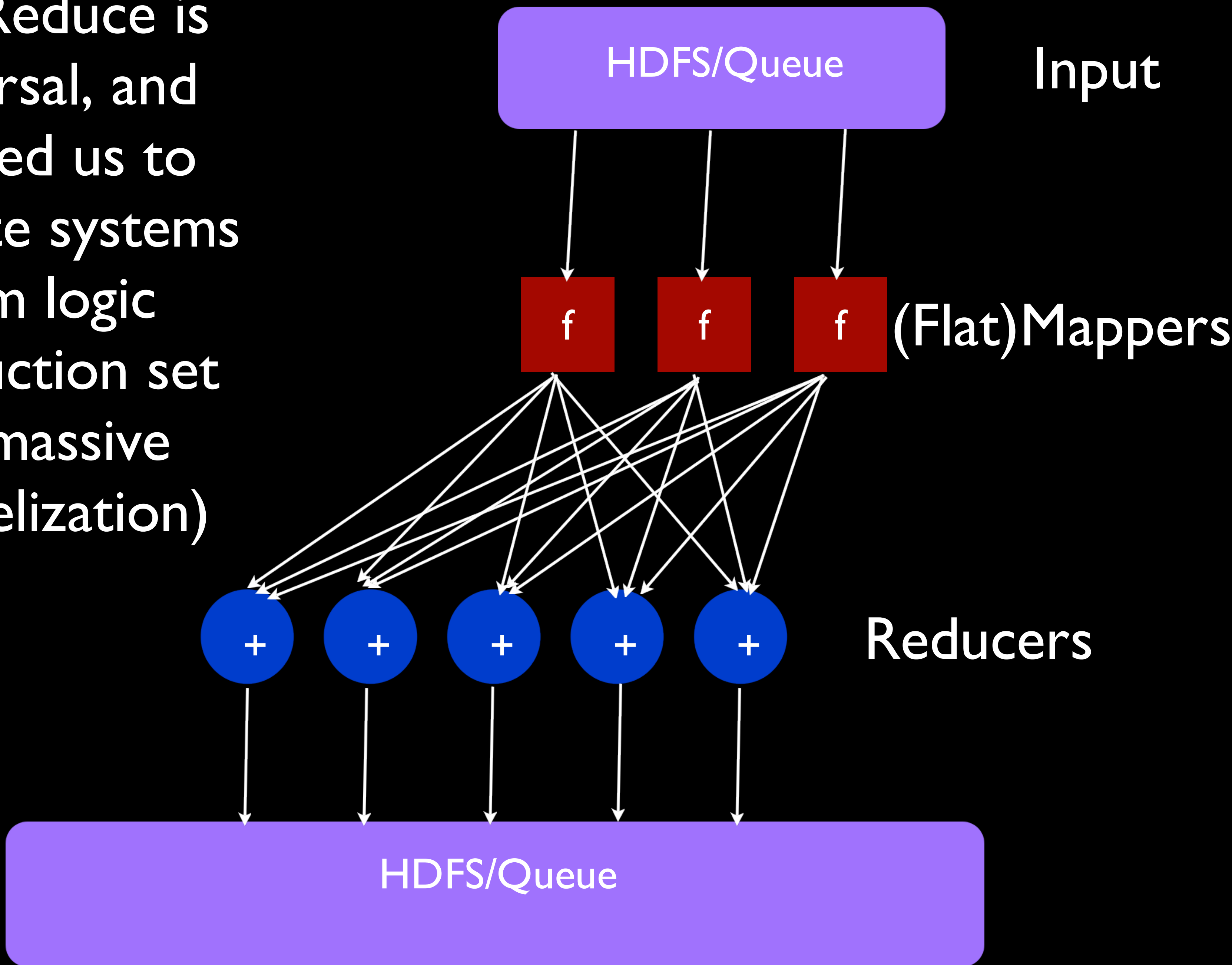


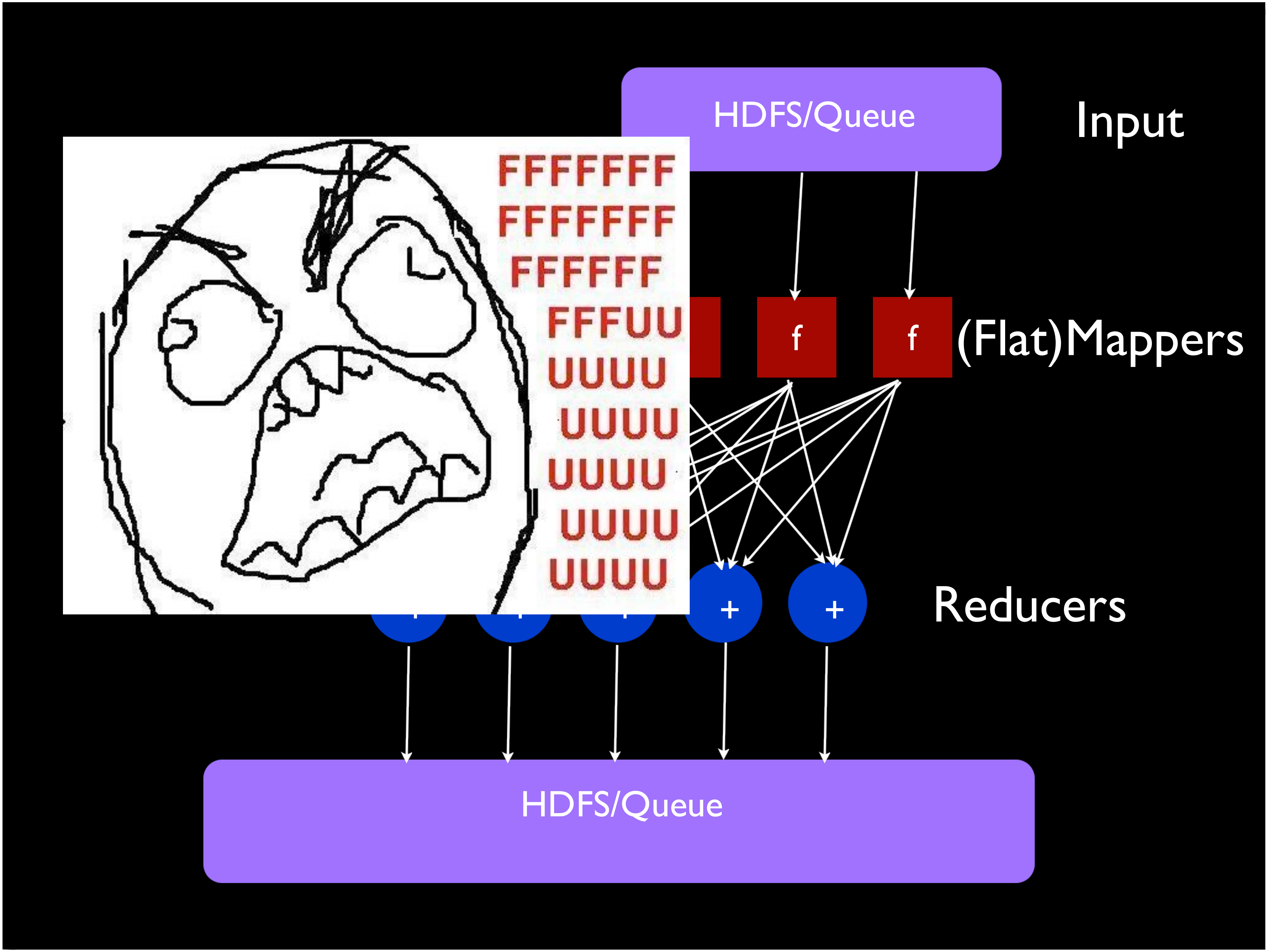
Storage Problems

- How to leverage the latency hierarchy? cache, main memory, networked memory, in data-center SSD, in DC hard-disk, cross-DC disk, tape?
- Can we find laws or approximations to govern relative allocation given power-laws everywhere?
- Can we get erasure coding as widely deployed as compression?
- Theory to give general combination of sampling/sketching algorithms to give approximate answers fast (e.g. memcache) and detailed answers as we dig deeper (like a progressive JPEG, exact data on slow, cheap storage, memcache that

3. System/Function Abstractions

Map/Reduce is universal, and allowed us to separate systems from logic (instruction set for massive parallelization)




HDFS/Queue
Input
FFFFFFF
FFFFFFF
FFFFFF
FFFUU
UUUU
UUUU
UUUU
UUUU
UUUU
f
f
(Flat)Mappers
+
+
Reducers
HDFS/Queue

MapReduce is Good Enough? If All You Have is a Hammer, Throw Away Everything That's Not a Nail!

Jimmy Lin
University of Maryland
jimmylin@umd.edu

Version of September 12, 2012

ABSTRACT

Hadoop is currently the large-scale data analysis "hammer" of choice, but there exist classes of algorithms that aren't "nails", in the sense that they are not particularly amenable to the MapReduce programming model. To address this, researchers have proposed MapReduce extensions or alternative programming models in which these algorithms can be elegantly expressed. This essay espouses a very different position: that MapReduce is "good enough", and that

addresses the limitation. The algorithms are expressed in this new framework, and, of course, experiments show substantial (an order of magnitude!) performance improvements over Hadoop.

This essay espouses a very different position, that MapReduce is "good enough" (even if the current Hadoop implementation could be vastly improved). While it is true that a large class of algorithms are not amenable to MapReduce implementations, there exist alternative solutions to the same underlying problems that *can* be easily imple-

Better system/function abstractions

- More CLEAN separation of logic and systems that allow optimizations, flexibility in deploy, and generality.
- Graph systems (e.g. GraphLab, Giraph) are not standard. Why? Lack of generality, bad implementations? Poor evangelism of value?
- Can we be less coarse than Map/Reduce, but be more coarse than MPI?
- Can we make a “Hadoop for Linear Algebra” (matrices, tensors, vectors, products, sums, regressions) that is comfortable to program, scalable and performant? Minimal

Impacting Industry

Timescale

- Short term: 1 week, Near term: 1-2 months, Long term: 3-6 months
- Have to deliver something of value soon.

Biggest Surprise

- Scale: 10 TB memcache? no problem, 20 TB intermediate output in Hadoop? ok.
- Code velocity: everything is in constant flux. Always replacing the engine with the car in motion. Constant deploys.
- Subtleties of small problems: graph of code versioning (diamond deps FFFUUUU, all tools are immature), serialization.

Send your students.
Do a sabbatical.
Attack industrially
relevant problems.

Thank you!
@posco
http://twitter.com/posco
oscar@twitter.com