Relax (Consistency)…
It's Only Analytics.

Christopher Ré
Stanford CS

Secure Internet of Things Workshop
Stanford, Aug 11

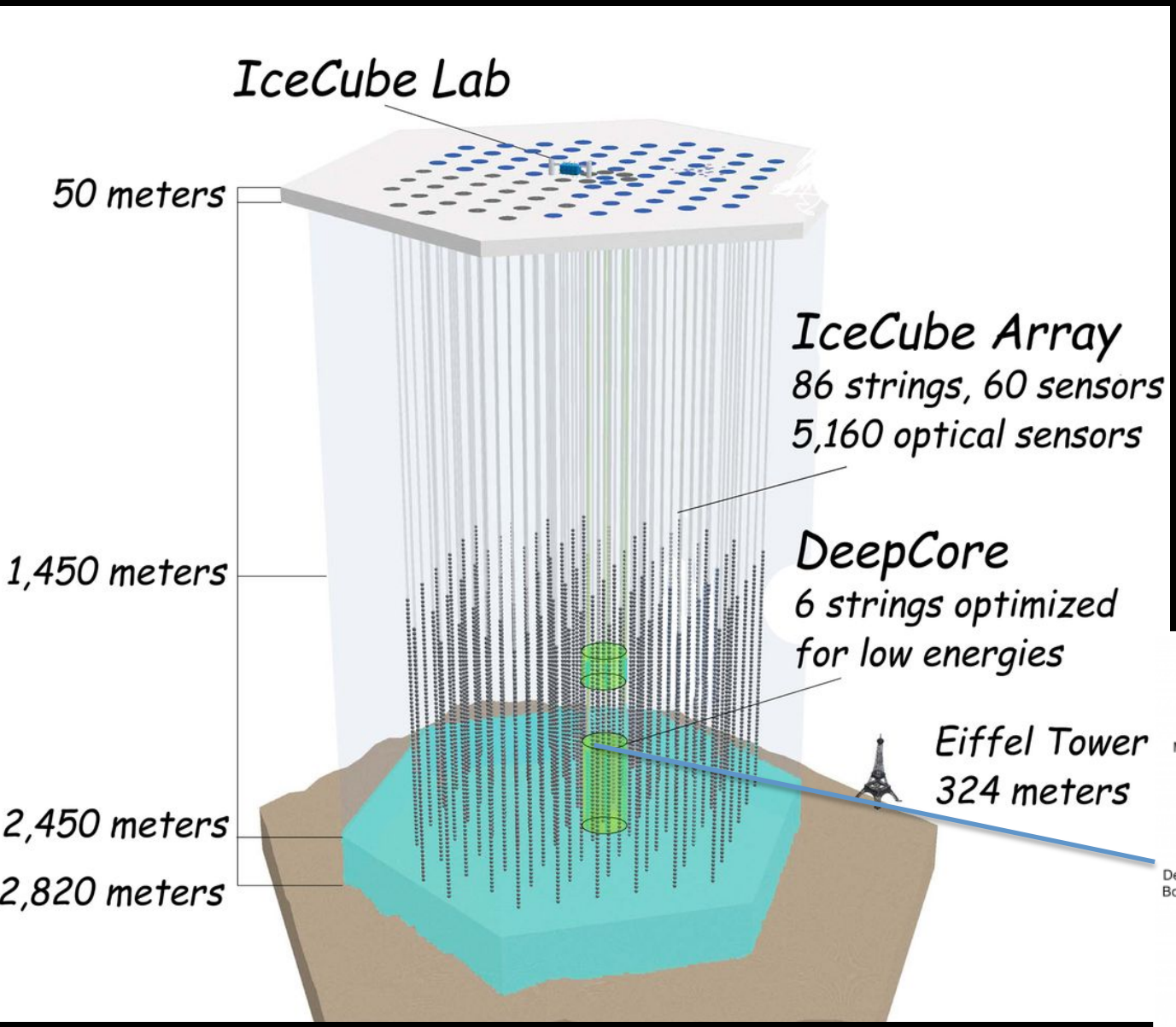
Some applications of **data analytics + sensing***

** Shamelessly biased by my group's work…*
** An excuse to make some technical points…*

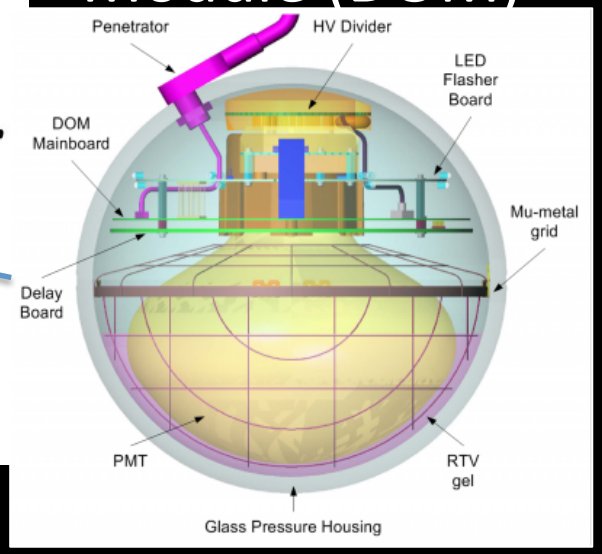
Analytics for **Big** Science and **Little** Dogs

Analytics for **Big** Science and **Little** Dogs

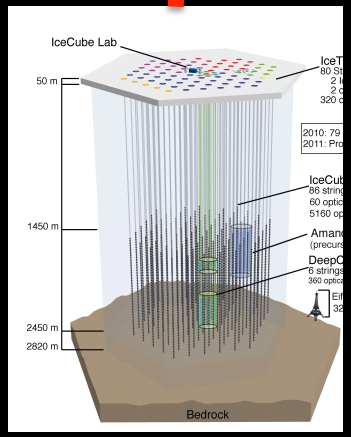
The IceCube Detector



Digital Optical Module (DOM)



Workflow of IceCube



In Madison: deeper analysis

Via satellite: interesting readings

At pole: *"Is interesting?"*

In Ice: Detection occurs

A Key Step: Recovering Track

Mark Wellons, Ben Recht, **Ré**, Francis Halzen, and Gary Hill

In real time ~ 4 μs

Mark's code on the Pole

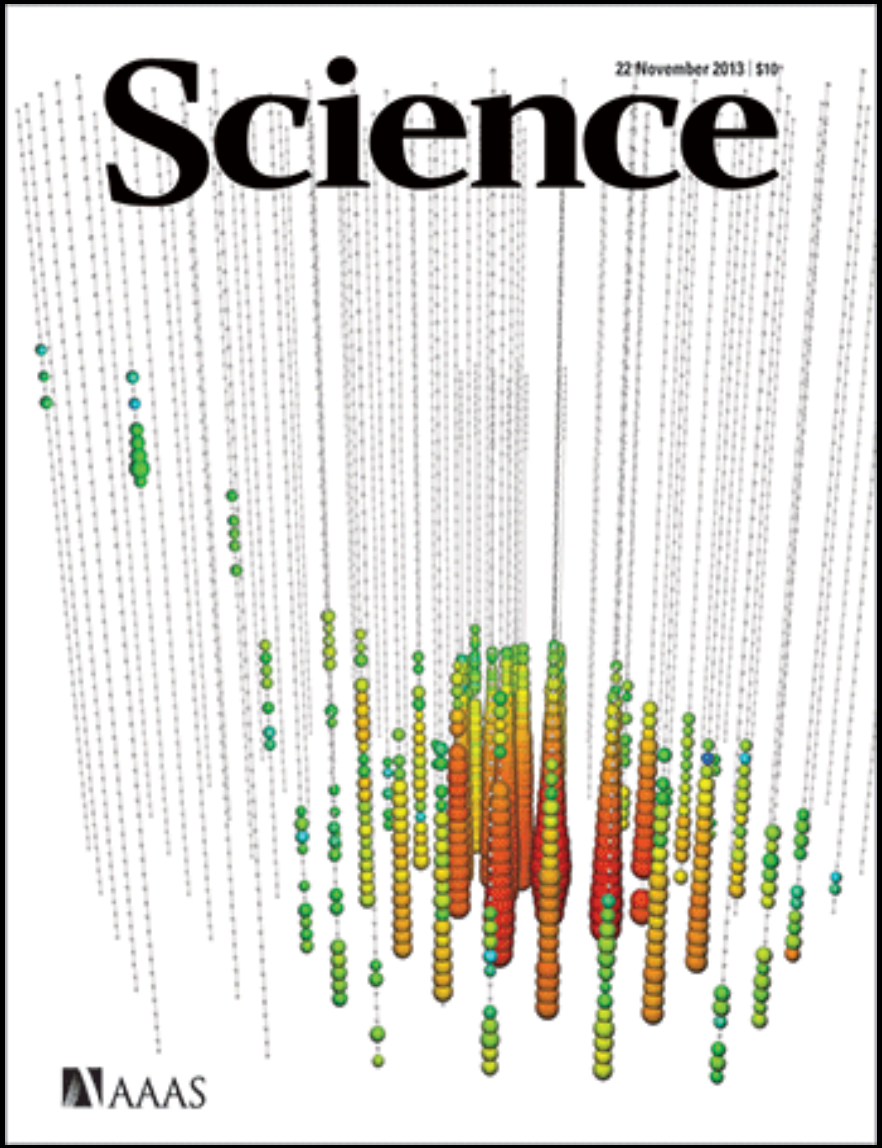
3KHz or 250M events/day

*Little physics but **combine** filters and **simple regression***

10% fewer false negatives and 1% more neutrinos

Hard part: Figuring out the filters

Mark got a Science paper!



Cosmic neutrinos named Physics World 2013 Breakthrough of the Year

physicsworld.com

1st Takeaway

In analytics + sensing,
the ***sizzle*** may be new algorithms, but
the ***steak*** is the application stack.

Analytics for **Big** Science and **Little** Dogs

Cortana: Microsoft's Digital Assistant

AI breakthrough: Microsoft's 'Project Adam' identifies dog breeds, points to future of machine learning

"…using a technology called, of all things, ***Hogwild!"***

http://www.wired.com/2014/07/microsoft-adam/
http://www.geekwire.com/2014/artificial-intelligence-breakthrough-microsofts-project-adam-identifies-dog-breeds/

What the heck is **Hogwild!?**

Microsoft

Google

MADlib

facebook

ORACLE®

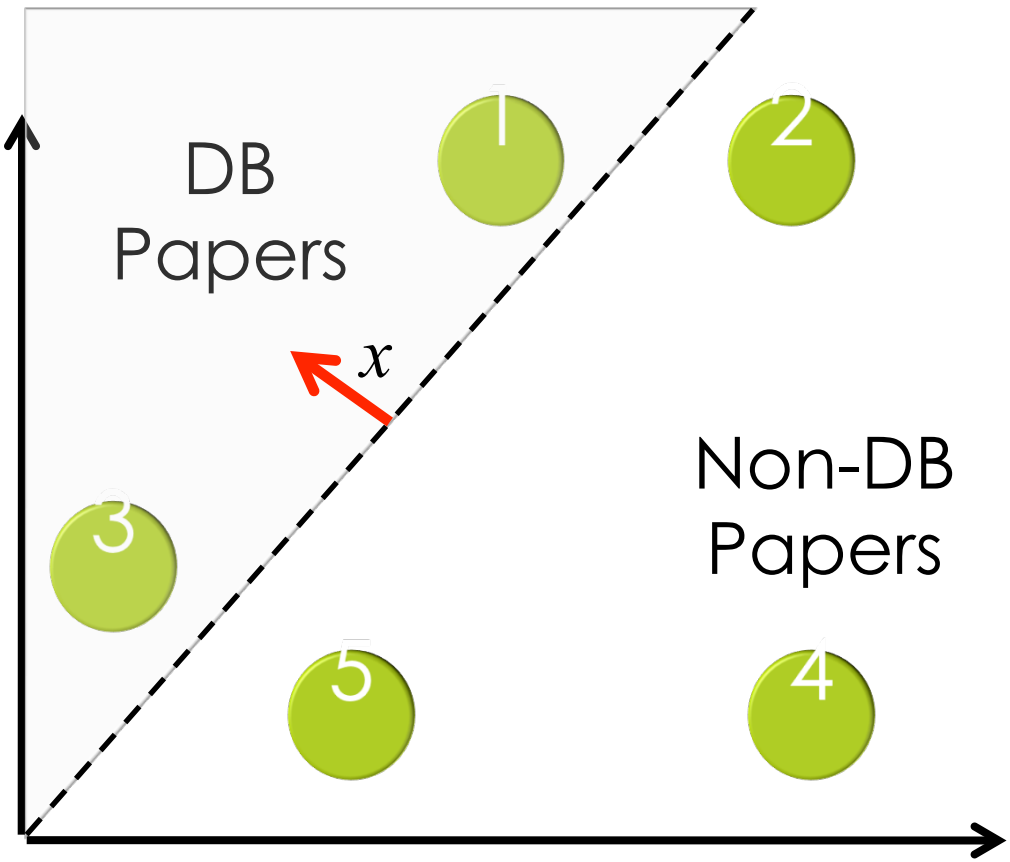
GREENPLUM IS NOW Pivotal

Example: Linear Models

Label papers as DB Papers or Non-DB Papers

1. Map papers to R^d

2. Classify via plane



How do we pick a good plane, x?

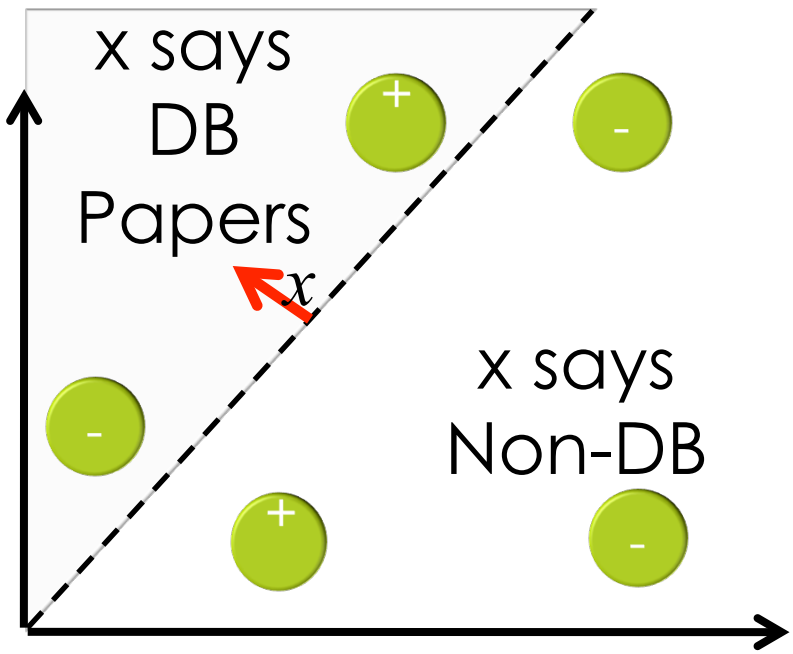
Example: Linear Models

Input: *Labeled* papers. Each point labeled as DB paper (+) or not (-)

<u>Idea:</u> score each plane

$$\min_x \sum_{i=1}^{N} f(x, y_i)$$

y_i is a paper vector and its label



f is convex function in x

e.g. squared distance (least squares), hinge loss (svm), log loss (logistic regression)

Separable Inverse Problems

Experts: f and P are ***convex.***

$$\min_x P(x) + \sum_{i=1}^{N} f(x, y_i)$$

x	the model
y_i	a data item
f	scores the error
P	Enforces prior

Paper Classification: y_i is the paper with its label, and x is a vector of weights

Neutrinos: x is neutrino path and y_i is a DOM reading

Natural Language (Conditional Random Fields), Recommendations (Low-rank factorizations), Logistic regression, Lasso, etc.

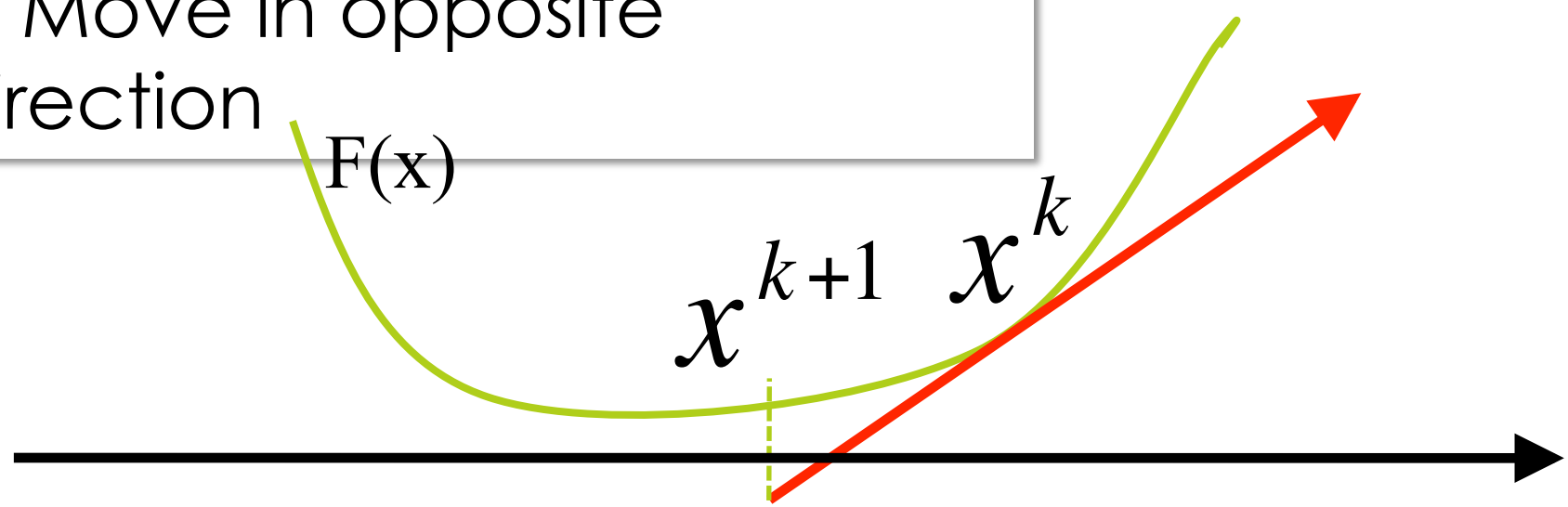
General statistical data analysis setup.

a

Background: Gradient Methods

$$F(x) = \sum_{i=1}^{N} f(x, y_i) \qquad x^{k+1} = x^k - \alpha \nabla F(x^k)$$

Gradient Methods: Iterative
1. Start at current x^k,
2. Take gradient at x^k, and
3. Move in opposite direction



Background: Incremental Gradient Methods

[Robbins & Monro, 1951]

$$F(x) = \sum_{i=1}^{N} f(x, y_i)$$

Gradient Methods:

1. Start at current x,
2. ***Approximate*** gradient at x by selecting *j in [N]*

$$G(x, y_j) = N \nabla f(x, y_j) \qquad \nabla F(x) \approx G(x, y_j)$$

3. Move in opposite direction

$$x^{k+1} = x^k - \alpha G(x^k, y_j)$$

Single *data item* to approximate gradient

Incremental Gradient Methods

Why use iGMs? iGMs converge to an optimal for many problems, but the real reason is:

iGMs are *fast.*

$$x+ = -\alpha G(x^k, y_j)$$

But iGM seems inherently sequential…

So...
what the heck is **Hogwild!?**

Hogwild! Overview

[**Niu**, Recht, Ré, & Wright NIPS 11]

Often, after preprocessing the *data fit on a single machine (a few TB)*.

How to parallelize iGMs in shared-memory?

Prior Work

-*Master/Worker* (Bertsekas and Tsitsiklis 85),
-*Round Robin* (Langford et al 09),
-*Average Runs* (Zinkevich et al 10),
-*Average Gradients* (Xiao 10), etc.

May fail to get speedup due to *lock contention*

We propose: Go Hogwild! and no locking!

Hogwild!

Goal to minimize:

$$F(x) = \sum_{i=1}^{N} f(x, y_i)$$

NO Locking

Each processor does:

1. Pick randomly a j in {1,…,N}
2. *Approximate* derivative of F wrt x using $f(x,y_j)$
3. Move in opposite direction (gradient step)
4. Write back components of x.

Issue: *All the processes write to one shared x!*

Problem: partial writes, use an old initial value of x.

This is crazy. Could this work?

Yes, you can go Hogwild!

	Dataset	Size (GB)	Time (s)	Speedup
SVM	RCV1	0.9	10	4.5
Matrix Completion	Netflix	1.5	301	5.3
	KDD Cup	3.9	878	5.2
	JUMBO	30	9,454	6.8
Cut Problems	DBLife	0.003	230	8.8
	Abdomen	18	1,181	4.1

Speed up to one percent of optimal.

10 Processors used for work.
(2 processors for data movement)

2nd Takeaway

Relaxing **consistency** to be **architecturally aware** can be a big performance win.

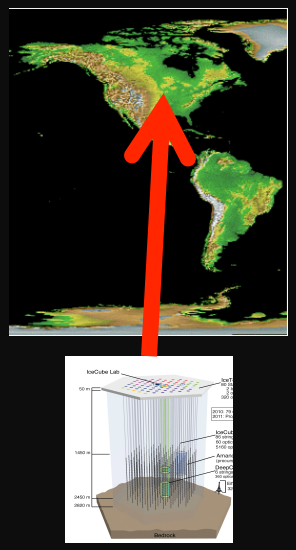
*NB: There is theory here.
[NIPS11 (Hogwild! for SGD) and ICML14 (SCD)
NIPS12, NIPS13, COLT12, and more…]

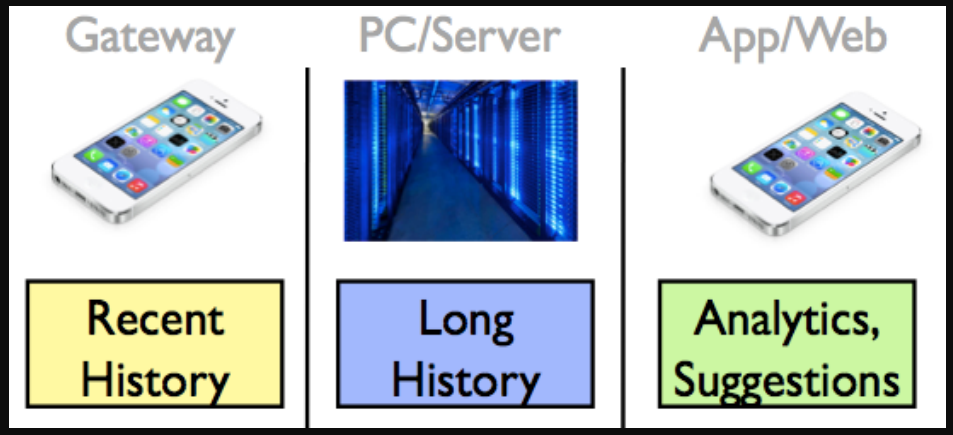
Main Idea: Statistical algorithms have new models of ***consistency*** *that allow new tradeoffs.*

What Changes in IoT?

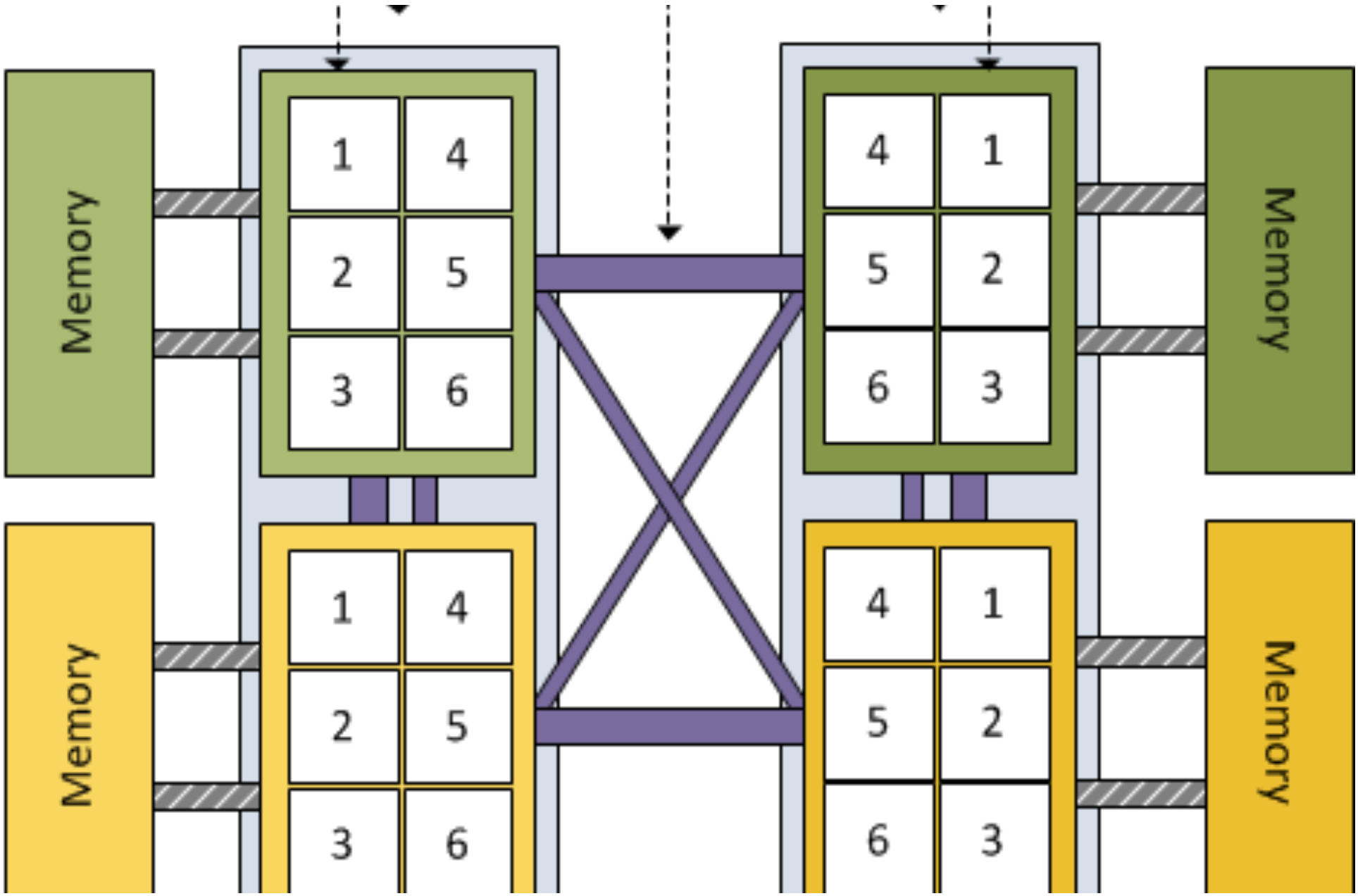
Cost of Communication and Energy, Failure modes, Heterogeneity, Security….

Well, we (sort of) have a start on **communication…**





Simplified 4 socket.

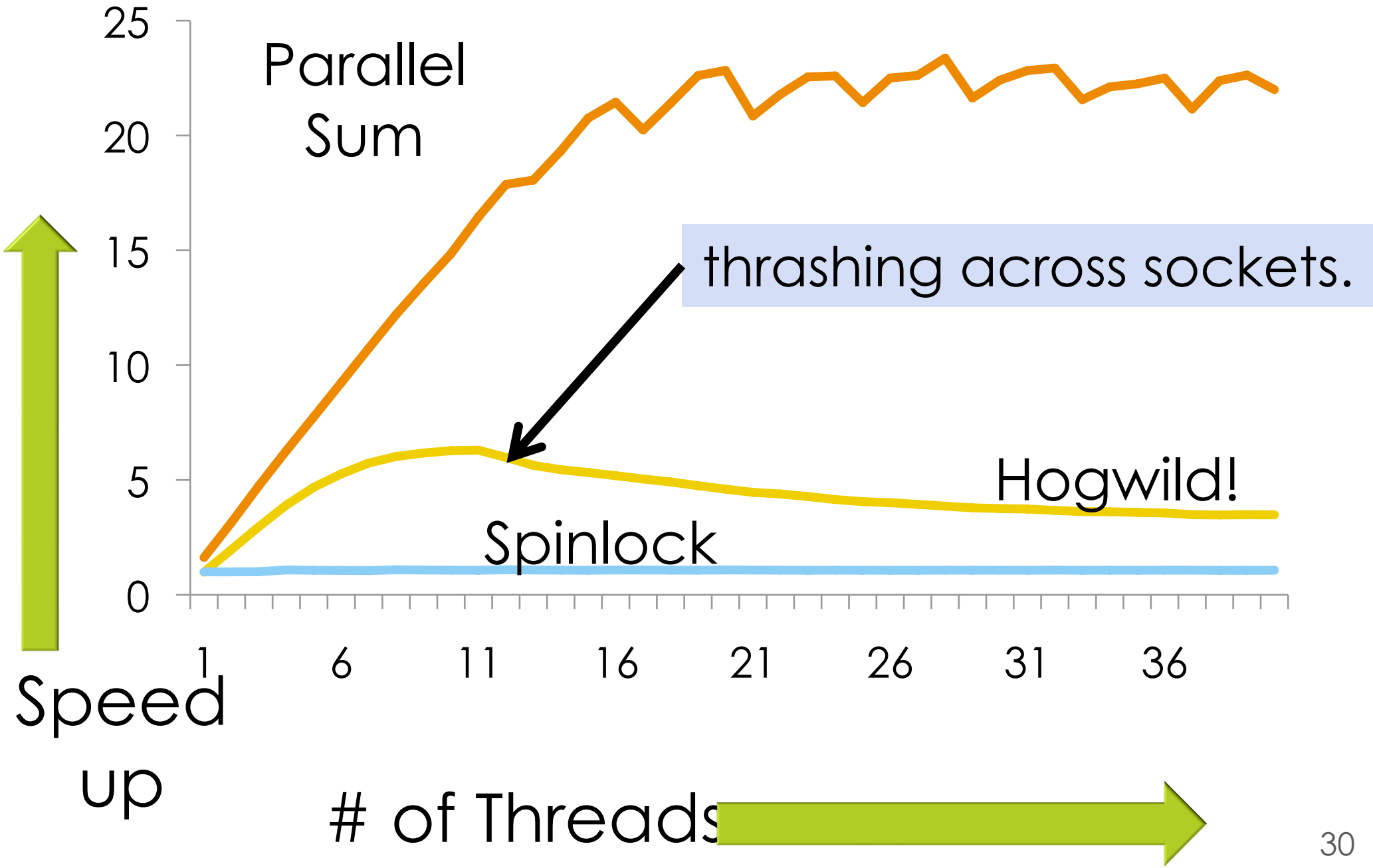


*Let's try a **tough** problem on 4 sockets (10 cores/socket)*

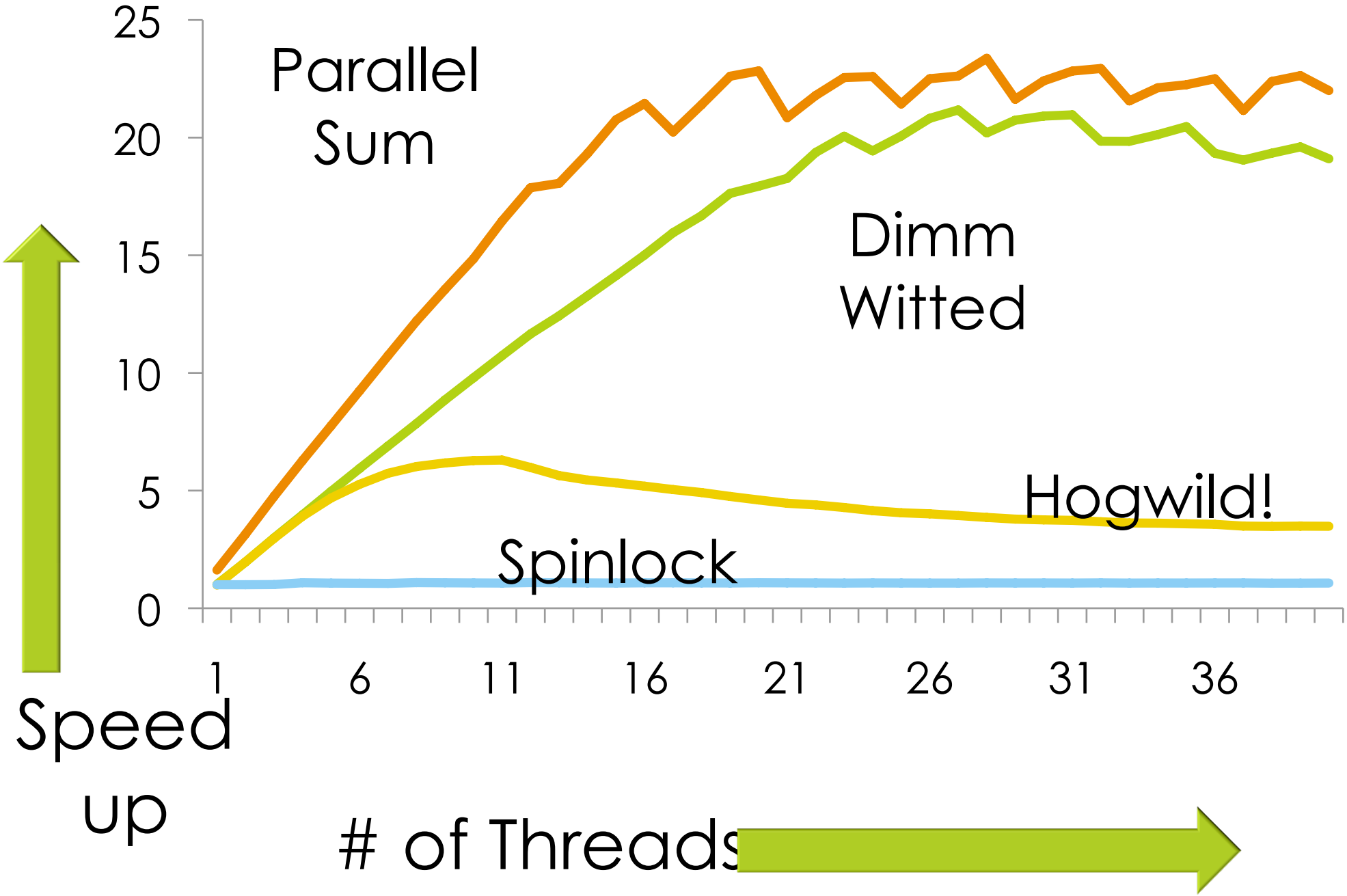
Linear programming relaxation.

NIPS13 : LP Relaxation for NP-Hard problems
ICML14 : Asynchronous SCD.

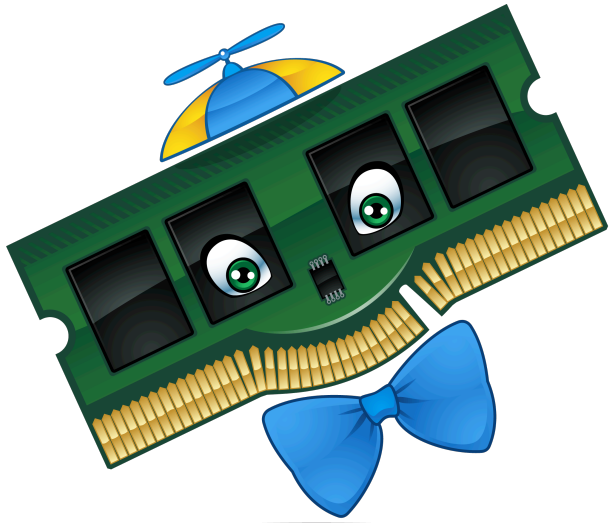
Constraints require communication



What about multiple sockets?



NUMA-awareness is key.



DIMMWitted

Communication aware (NUMA)-aware layout that ***further relaxes*** consistency of the algorithm.

Balance: *Statistical v. Hardware Efficiency*.

Statistical versus Hardware Efficiency

Relaxing consistency results in new tradeoffs.

1. Access methods
   - {Row, Column, Row-col}
2. Model Replication
   - {Core, Node, Machine}
3. Data Replication
   - {Full, Importance, Shard}

Can be 100x faster than classical choices

DimmWitted: A Study of Main-Memory Statistical Analytics. VLDB14.

One hammer with some questions?

Can relaxing **consistency** help

- Improve performance?
- Decrease communication?
- Improve energy efficiency?
- Help deal with failures?
- Exploit heterogeneous hardware? Cup holders?

3 Wrap-up thoughts

1. In IoT, ***the steak is the application stack***.

2. **Analytics might be the unfair advantage**
   e.g, cheaper, faster, less resources, more robust.

3. ***Relaxed consistency*** is one hammer for 2.