MapReduce and MPI

Steve Plimpton
Sandia National Labs

SOS 17 - Intersection of HPC & Big Data
March 2013

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

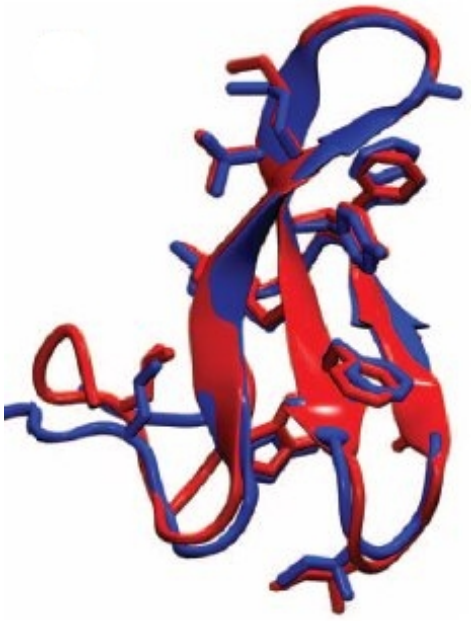
Part 1: MapReduce for HPC and big data

Tiankai Tu, et al (DE Shaw), *Scalable Parallel Framework for Analyzing Terascale MD Trajectories*, SC 2008.

Part 1: MapReduce for HPC and big data

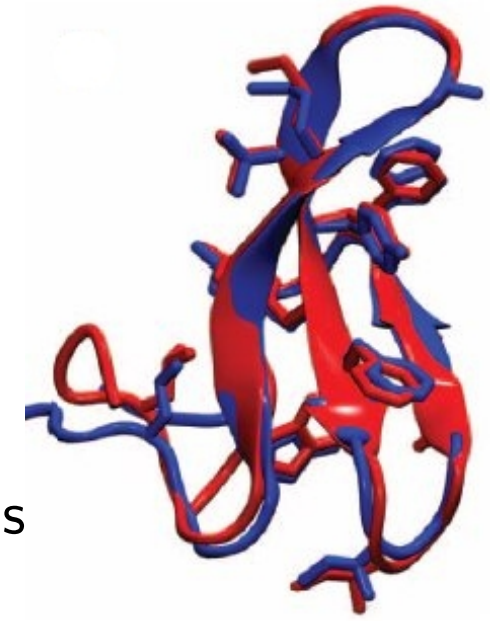
Tiankai Tu, et al (DE Shaw), *Scalable Parallel Framework for Analyzing Terascale MD Trajectories*, SC 2008.

- 1M atoms, 100M snapshots $\Rightarrow$ **3 Pbytes**
- Stats on where each atom traveled
  - near-approach to docking site
  - membrane crossings



Part 1: MapReduce for HPC and big data

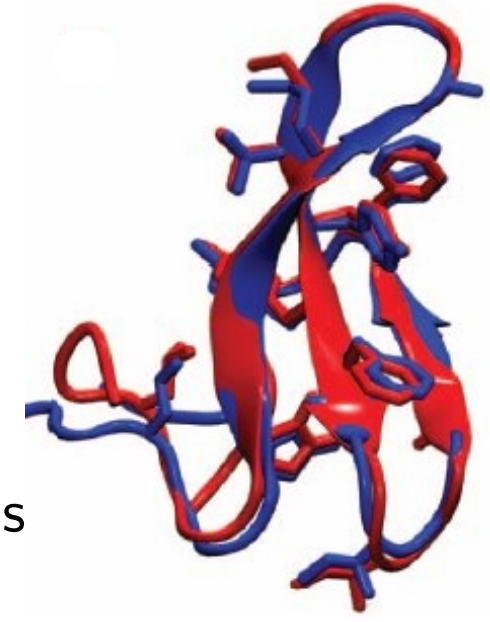
Tiankai Tu, et al (DE Shaw), *Scalable Parallel Framework for Analyzing Terascale MD Trajectories*, SC 2008.



- 1M atoms, 100M snapshots $\Rightarrow$ **3 Pbytes**
- Stats on where each atom traveled
  - near-approach to docking site
  - membrane crossings
- Data is stored **exactly wrong** for this analysis
- MapReduce solution:
  1. **map**: read snapshot, emit key = ID; value = (time, xyz)
  2. **communicate**: aggregate all values with same ID
  3. **reduce**: order the values, perform analysis

Part 1: MapReduce for HPC and big data

Tiankai Tu, et al (DE Shaw), *Scalable Parallel Framework for Analyzing Terascale MD Trajectories*, SC 2008.



- 1M atoms, 100M snapshots $\Rightarrow$ **3 Pbytes**
- Stats on where each atom traveled
  - near-approach to docking site
  - membrane crossings
- Data is stored **exactly wrong** for this analysis
- MapReduce solution:
  1. **map**: read snapshot, emit key = ID; value = (time, xyz)
  2. **communicate**: aggregate all values with same ID
  3. **reduce**: order the values, perform analysis
- **Key point**: extremely parallel comp + MPI_All2all comm

Why is MapReduce attractive?

- **Plus**:
  - write only the code that only you can write
  - write zero parallel code (no parallel debugging)
  - out-of-core for free
- **Plus/minus** (features!):
  - ignore data locality
  - load balance thru random distribution
    - key hashing = slow global address space
  - maximize communication (all2all)
- **Minus**:
  - have to re-cast your algorithm as a MapReduce

Why is MapReduce attractive?

- **Plus**:
  - write only the code that only you can write
  - write zero parallel code (no parallel debugging)
  - out-of-core for free
- **Plus/minus** (features!):
  - ignore data locality
  - load balance thru random distribution
    - key hashing = slow global address space
  - maximize communication (all2all)
- **Minus**:
  - have to re-cast your algorithm as a MapReduce

Good programming model for big data analyst:
not maximal performance, but **minimal human effort**

MapReduce software

- Hadoop:
  - parallel HDFS, fault tolerance
  - extra big-data goodies (BigTable, etc)
  - no one runs it on huge HPC platforms (as far as I know)

MapReduce software

- Hadoop:
  - parallel HDFS, fault tolerance
  - extra big-data goodies (BigTable, etc)
  - no one runs it on huge HPC platforms (as far as I know)

- MR-MPI: **http://mapreduce.sandia.gov**
  - MapReduce on top of MPI
  - Lightweight, portable, C++ library with C API
  - Out-of-core on big iron if each proc can write scratch files
  - No HDFS (parallel file system with data redundancy)
  - No fault-tolerance (blame it on MPI)

What could you do with MapReduce at Peta/Exascale?

- **Post-simulation analysis** of big data output:
  - on HPC platform, don't have to move your data
  - computations needing info from entire time series
  - trajectories, flow fields, acoustic noise estimation

What could you do with MapReduce at Peta/Exascale?

- **Post-simulation analysis** of big data output:
  - on HPC platform, don't have to move your data
  - computations needing info from entire time series
  - trajectories, flow fields, acoustic noise estimation
- **Matrix operations**:
  - matrix-vector multiply (PageRank kernel)
  - tall-skinny QR (D Gleich, P Constantine)
    - simulation data $\Rightarrow$ cheaper surrogate model
    - 500M x 100 dense matrix $\Rightarrow$ 30 min on 256-core cluster

What could you do with MapReduce at Peta/Exascale?

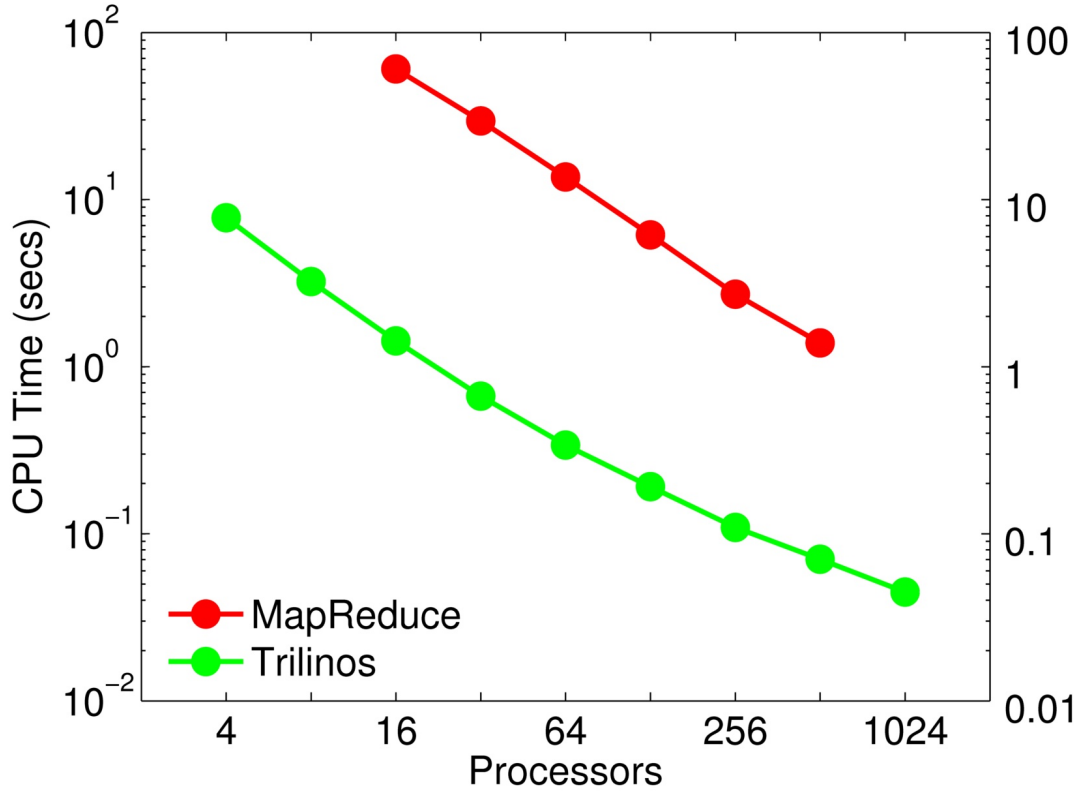
- **Post-simulation analysis** of big data output:
  - on HPC platform, don't have to move your data
  - computations needing info from entire time series
  - trajectories, flow fields, acoustic noise estimation
- **Matrix operations**:
  - matrix-vector multiply (PageRank kernel)
  - tall-skinny QR (D Gleich, P Constantine)
    - simulation data $\Rightarrow$ cheaper surrogate model
    - 500M x 100 dense matrix $\Rightarrow$ 30 min on 256-core cluster
- Graph algorithms:
  - vertex ranking via **PageRank** (460)
  - connected components (250)
  - triangle enumeration (260)
  - single-source shortest path (240)
  - **sub-graph isomorphism** (430)

What could you do with MapReduce at Peta/Exascale?

- **Post-simulation analysis** of big data output:
  - on HPC platform, don't have to move your data
  - computations needing info from entire time series
  - trajectories, flow fields, acoustic noise estimation
- **Matrix operations**:
  - matrix-vector multiply (PageRank kernel)
  - tall-skinny QR (D Gleich, P Constantine)
    - simulation data $\Rightarrow$ cheaper surrogate model
    - 500M x 100 dense matrix $\Rightarrow$ 30 min on 256-core cluster
- Graph algorithms:
  - vertex ranking via **PageRank** (460)
  - connected components (250)
  - triangle enumeration (260)
  - single-source shortest path (240)
  - **sub-graph isomorphism** (430)
- Machine learning: classification, clustering, ...
- Win the TeraSort benchmark

No free lunch: PageRank (matvec) performance

Cray XT3, 1/4 billion edge sparse, highly irregular matrix



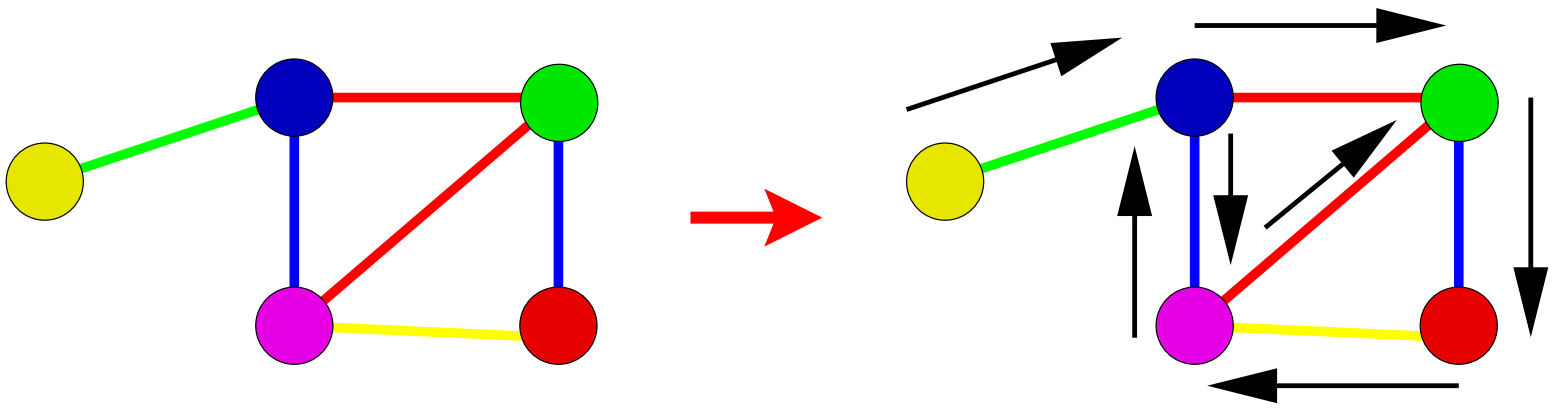
- MapReduce communicates matrix elements
- But recall: load-balance, out-of-core for **free**

Sub-graph isomorphism for data mining

- Data mining, **needle-in-haystack** anomaly search
- Huge semantic graph with **labeled vertices, edges**
- **SGI** = find all occurrences of small target graph

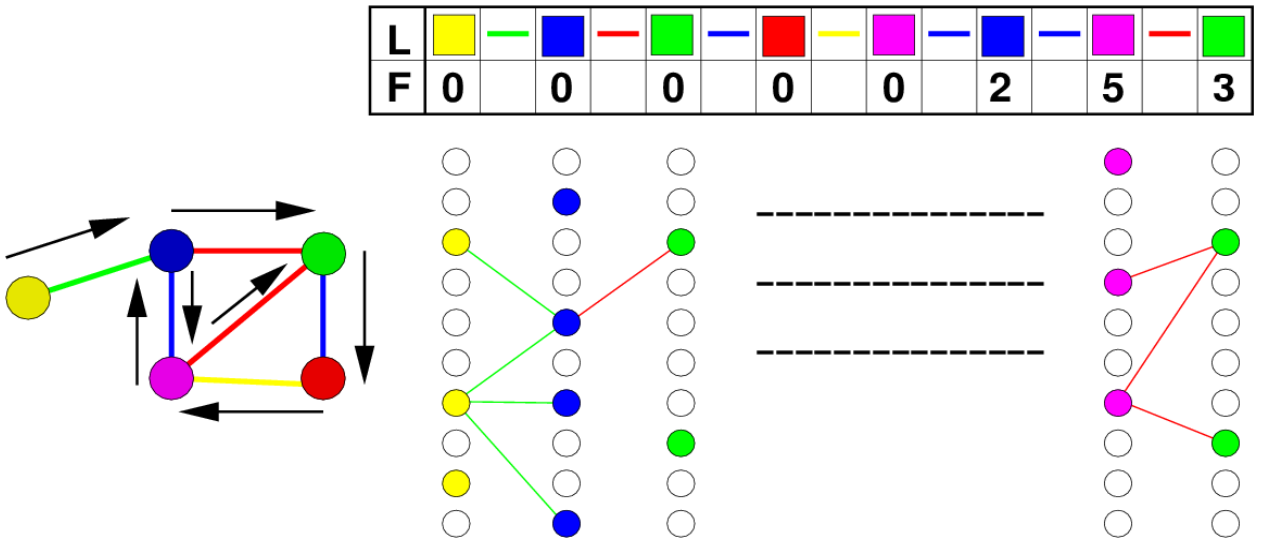
Sub-graph isomorphism for data mining

- Data mining, **needle-in-haystack** anomaly search
- Huge semantic graph with **labeled vertices, edges**
- **SGI** = find all occurrences of small target graph



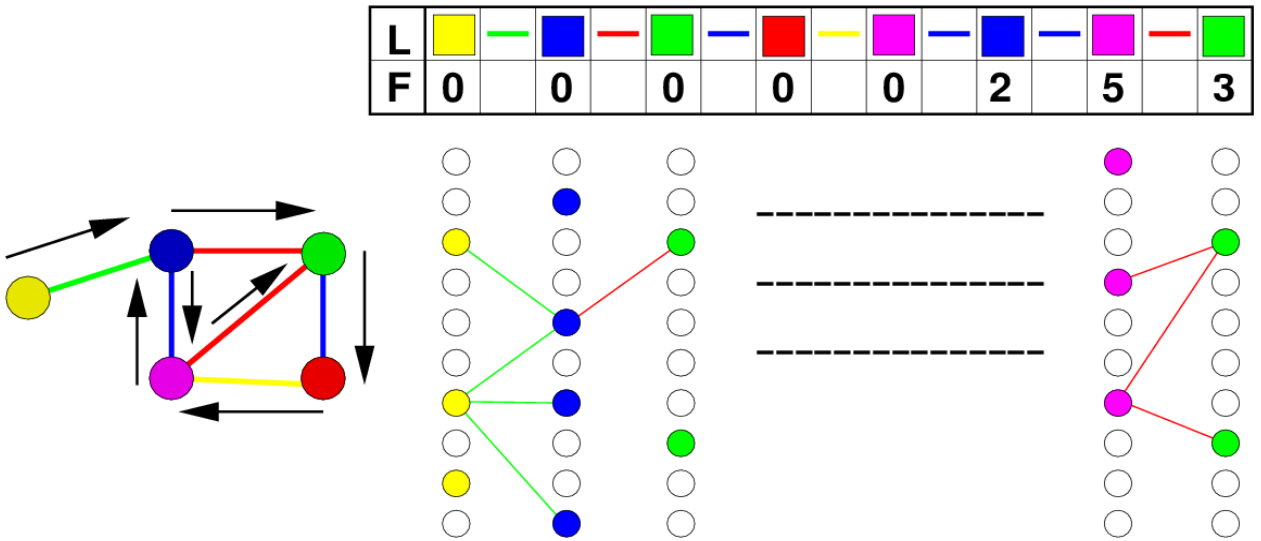
L															
F	0		0		0		0		0		2		5		3

MapReduce algorithm for sub-graph isomorphism



- One MR object per column of bipartite graph
- Iterate from left to right, keying on colored vertices
- Generate list of **candidate walks**, one edge at a time
- Caveat: list can explode due to delayed constraints
- But: **430 lines of code**, no MPI, out-of-core graphs

MapReduce algorithm for sub-graph isomorphism

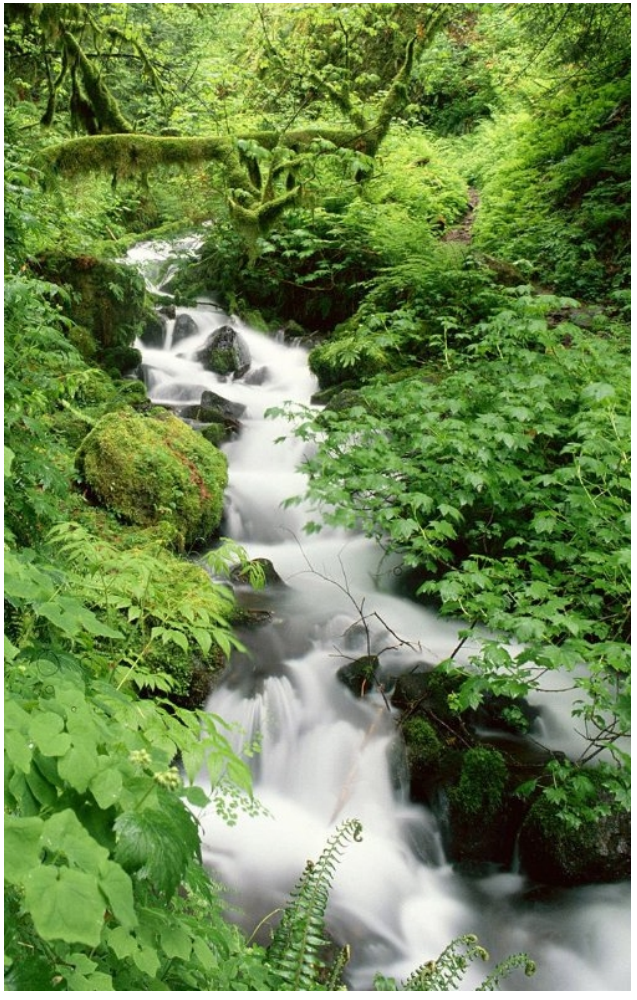


- One MR object per column of bipartite graph
- Iterate from left to right, keying on colored vertices
- Generate list of **candidate walks**, one edge at a time
- Caveat: list can explode due to delayed constraints
- But: **430 lines of code**, no MPI, out-of-core graphs

Example: 18 Tbytes $\Rightarrow$ 107B edges $\Rightarrow$ 573K matches in 55 minutes on 256 cores

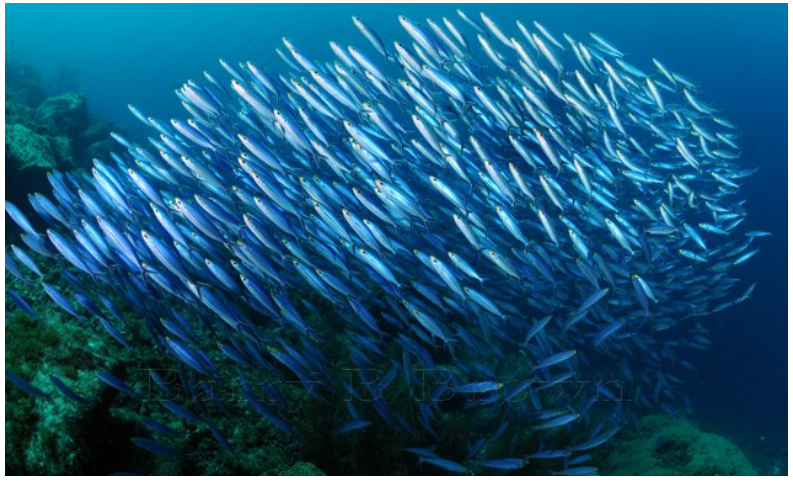
Streaming data

- Continuous, real-time data
- Stream = small datums at high rate
- **Resource-constrained processing**:
  - only see datums once
  - compute/datum $<$ stream rate
  - only store state that fits in memory
  - age/expire data
- Pipeline model is attractive:
  - datums flow thru **compute processes** running on cores
  - hook processes together to perform analysis
  - **split stream** to enable shared or distributed-memory parallelism



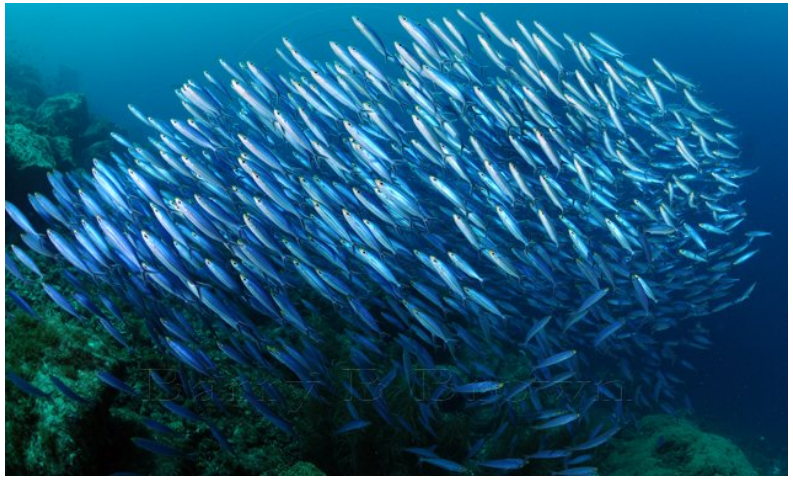
Streaming software

- IBM InfoSphere (commercial)
- Twitter Storm (open-source)
- PHISH: **http://www.sandia.gov/∼sjplimp/phish.html**
  - Parallel Harness for Informatic Stream Hashing
  - phish swim in a stream
  - runs on top of MPI or sockets (zeroMQ)



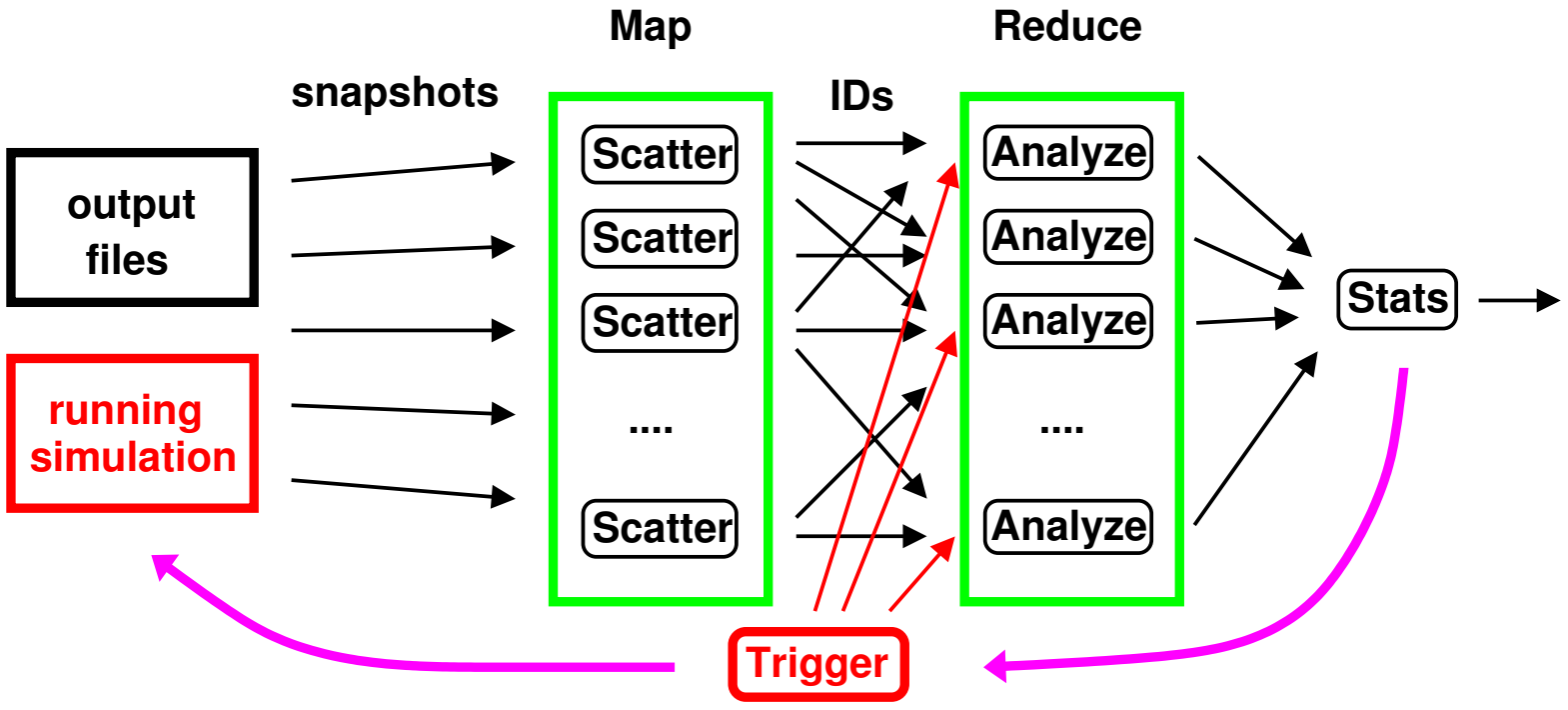
Streaming software

- IBM InfoSphere (commercial)
- Twitter Storm (open-source)
- PHISH: **http://www.sandia.gov/~sjplimp/phish.html**
  - Parallel Harness for Informatic Stream Hashing
  - phish swim in a stream
  - runs on top of MPI or sockets (zeroMQ)



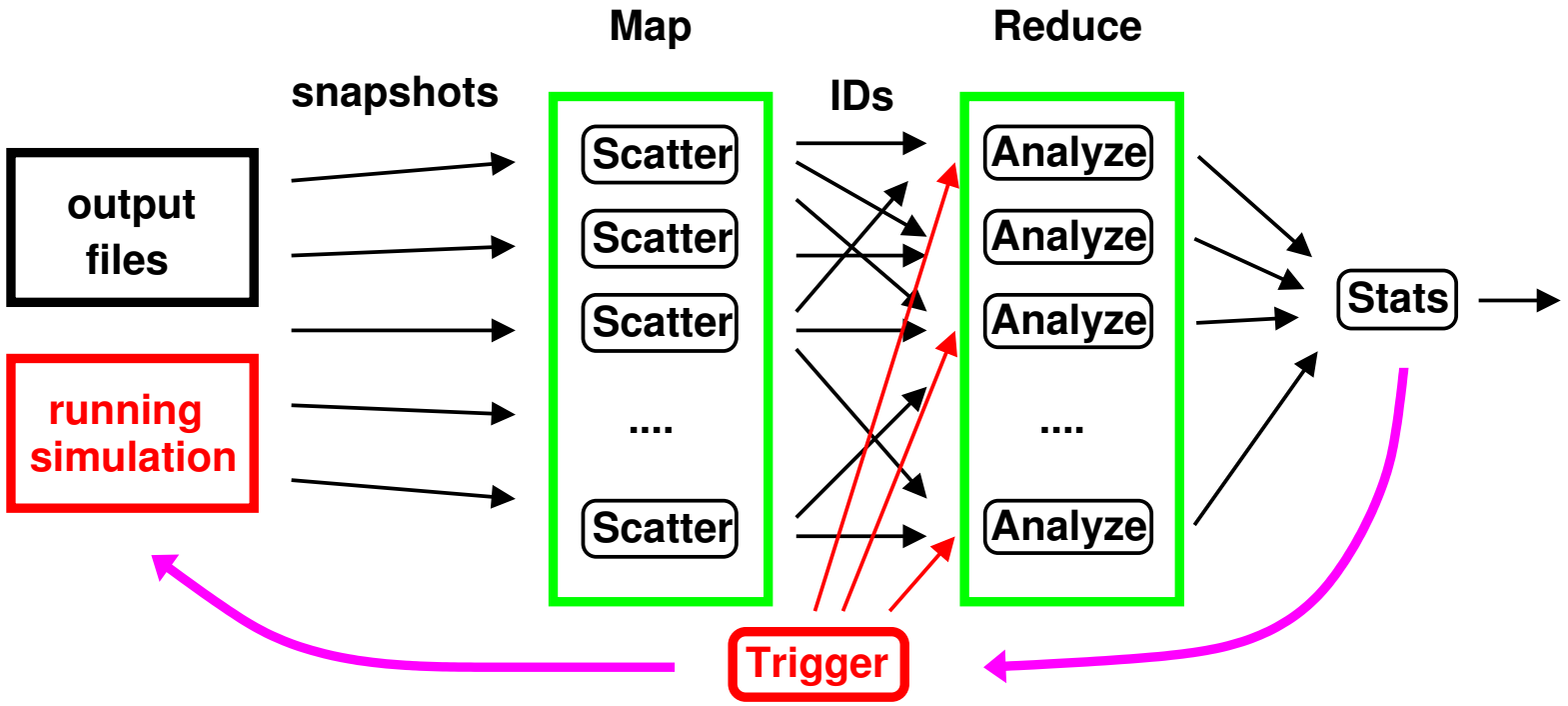
- Key point: **zillions of small messages** flowing thru processes

PHISH net for real-time analysis of big data



- Data source could be experiment or simulation
- A **streaming MapReduce** is now fine-grained and continuous
- Could add user interactions for **simulation steering**

PHISH net for real-time analysis of big data



- Data source could be experiment or simulation
- A **streaming MapReduce** is now fine-grained and continuous
- Could add user interactions for **simulation steering**
- Graph algorithms can operate on stream of edges
- 1024 nodes of HPC: 150M edges/sec for hashed all2all

Part 2: Intersection of HPC & Big Data?

Part 2: Intersection of HPC & Big Data?

Φ (empty set)

Part 2: Intersection of HPC & Big Data?

Φ (empty set)
ϵ (tiny)

Part 2: Intersection of HPC & Big Data?

Φ (empty set)
ϵ (tiny)

- Defining HPC in a **broad way**
  - rack of servers + cheap interconnect is not traditional HPC
  - Higgs talk is a good example
- Defining big data in **narrow way**
  - scientific data is only a tiny fraction of big data

Part 2: Intersection of HPC & Big Data?

Φ (empty set)
ϵ (tiny)

- Defining HPC in a **broad way**
  - rack of servers + cheap interconnect is not traditional HPC
  - Higgs talk is a good example
- Defining big data in **narrow way**
  - scientific data is only a tiny fraction of big data
- How many **Top50 machines** owned by "big data" companies?
- If companies/govt spent $200B on big data today, would they buy a Top10 petascale machine?
- Would they use HPC if you **gave the machines away**?

Part 2: Intersection of HPC & Big Data?

Φ (empty set)
ϵ (tiny)

- Defining HPC in a **broad way**
  - rack of servers + cheap interconnect is not traditional HPC
  - Higgs talk is a good example
- Defining big data in **narrow way**
  - scientific data is only a tiny fraction of big data
- How many **Top50 machines** owned by "big data" companies?
- If companies/govt spent $200B on big data today, would they buy a Top10 petascale machine?
- Would they use HPC if you **gave the machines away**?
  - tried that at Sandia
  - gave a decommissioned HPC machine to intelligence groups
  - barely used for big data problems

Three reasons why intersection is small

- Using HPC platform and MPI in **non-optimal way**:
  - little computation
  - ignoring data locality
  - all2all (MapReduce)
  - tiny messages (streaming)
  - lots of I/O

Three reasons why intersection is small

- Using HPC platform and MPI in **non-optimal way**:
  - little computation
  - ignoring data locality
  - all2all (MapReduce)
  - tiny messages (streaming)
  - lots of I/O
- Big data for **science vs informatics** is different:
  - **Sci**: compute bound; **Info**: memory/disk bound
  - **Sci**: precise computations; **Info**: inexact/agile/one-off
  - **Sci**: big data is an output; **Info**: big data is an input
  - **Sci**: simulation is valuable, data is not; **Info**: inverse

Three reasons why intersection is small

- Using HPC platform and MPI in **non-optimal way**:
  - little computation
  - ignoring data locality
  - all2all (MapReduce)
  - tiny messages (streaming)
  - lots of I/O
- Big data for **science vs informatics** is different:
  - **Sci**: compute bound; **Info**: memory/disk bound
  - **Sci**: precise computations; **Info**: inexact/agile/one-off
  - **Sci**: big data is an output; **Info**: big data is an input
  - **Sci**: simulation is valuable, data is not; **Info**: inverse
- HPC sells what big data customers **don't need**:
  - scientific simulations need CPUs and network
  - big data needs disks and I/O

Olympic price metric: gold vs silver vs bronze

- Gold = ORNL Jaguar
- Aluminum = bioinformatics cluster at Columbia U
- Plywood = racks of cheap cores & disks (Facebook, Walmart)

Olympic price metric: gold vs silver vs bronze

- Gold = ORNL Jaguar
- Aluminum = bioinformatics cluster at Columbia U
- Plywood = racks of cheap cores & disks (Facebook, Walmart)

Medal:	$$	$/PByte	GBs/PB	TB/core	PB/Pflop
Gold:	$100M	$10M	24	0.044	5
Aluminum:	$2.5M	$2.5M	20	0.25	40
Plywood:	scalable	$0.3M	100	1+	100+

Olympic price metric: gold vs silver vs bronze

- Gold = ORNL Jaguar
- Aluminum = bioinformatics cluster at Columbia U
- Plywood = racks of cheap cores & disks (Facebook, Walmart)

Medal:	$$	$/PByte	GBs/PB	TB/core	PB/Pflop
Gold:	$100M	$10M	24	0.044	5
Aluminum:	$2.5M	$2.5M	20	0.25	40
Plywood:	scalable	$0.3M	100	1+	100+

- No one wants to pay **gold prices** to do big data computing
- Big data informatics done on aluminum and plywood
- 90% of Jaguar price is for hardware informatics barely uses

Exascale car salesman

Exascale car salesman

Big data customer

Big data customer

Exascale car salesman - the green solution

Exascale car salesman - the green solution

Exascale car salesman - the hybrid model

Exascale car salesman - the hybrid model

Same machine for HPC simulations and big data?

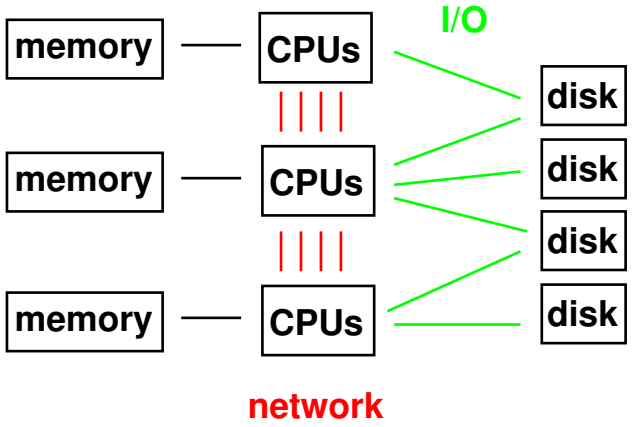
- Convince data owners HPC calculates **something they can't**
  - if computation is O(N), they can do it
  - can HPC add value for $O(N \log N)$ or $O(N^2)$ (T Schulthess)

Same machine for HPC simulations and big data?

- Convince data owners HPC calculates **something they can't**
  - if computation is O(N), they can do it
  - can HPC add value for O($N \log N$) or O(N^2) (T Schulthess)
- An architecture suggestion
  - caveat: I have zero architectural savvy ...

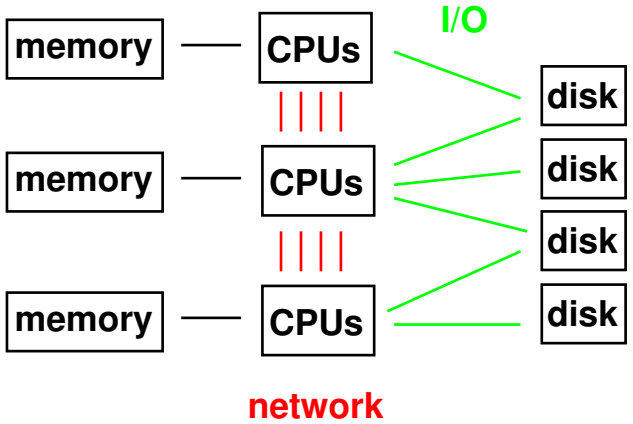
Same machine for HPC simulations and big data?

- Convince data owners HPC calculates **something they can't**
  - if computation is O(N), they can do it
  - can HPC add value for $O(N \log N)$ or $O(N^2)$ (T Schulthess)
- An architecture suggestion
  - caveat: I have zero architectural savvy ...



Same machine for HPC simulations and big data?

- Convince data owners HPC calculates **something they can't**
  - if computation is O(N), they can do it
  - can HPC add value for O($N \log N$) or O(N^2) (T Schulthess)
- An architecture suggestion
  - caveat: I have zero architectural savvy ...



- **Idea**: add cheap CPUs to each disk, let disks do MapReduce
- **Q**: what moves data between disks?
  fast network or something else?
- **Q**: Can disk-centric informatics run at same time
  as CPU-centric simulation?

One hybrid machine ...

One hybrid machine to rule them all ...

Thanks & links

Sandia **collaborators**:

- Karen Devine (MR-MPI)
- Tim Shead (PHISH)
- Todd Plantenga, Jon Berry, Cindy Phillips (graph algorithms)

Open-source packages (BSD license):

- **http://mapreduce.sandia.gov** (MapReduce-MPI)
- **http://www.sandia.gov/∼sjplimp/phish.html** (PHISH)

Papers:

- Plimpton & Devine, *"MapReduce in MPI for large-scale graph algorithms"*, Parallel Computing, 37, 610 (2011).
- Plimpton & Shead, *"Streaming data analytics via message passing"*, submitted to JPDC (2012).