Platforms for Data Intensive Research

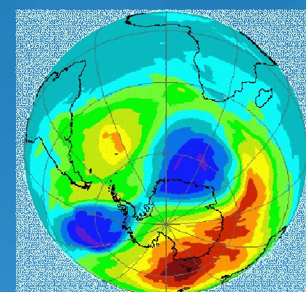
Dryad & DryadLINQ
Windows Azure

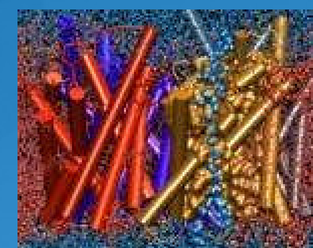
Roger Barga
Architect, Cloud Computing Futures Group
Microsoft Research (MSR)

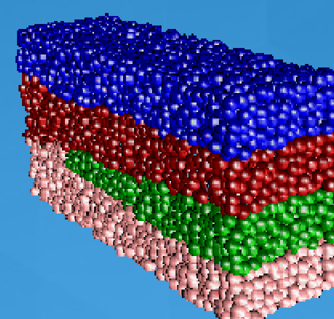
Science 2020

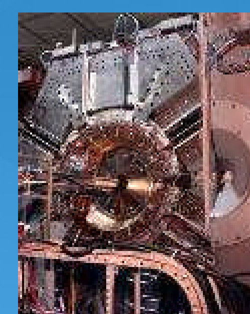
“In the last two decades advances in computing technology, from processing speed to network capacity and the Internet, have revolutionized the way scientists work.

From sequencing genomes to monitoring the Earth's climate, many recent scientific advances would not have been possible without a parallel increase in computing power - and with technologies such as the quantum computer edging towards reality, *what will the relationship between computing and science bring us over the next 15 years?”*









http://research.microsoft.com/towards2020science

Sapir–Whorf: Context and Research

Sapir–Whorf Hypothesis (SWH)

Language influences the habitual thought of its speakers

Scientific computing analog

Available systems shape research agendas

Consider some past examples

- Cray-1 and vector computing
- VAX 11/780 and UNIX
- Workstations and Ethernet
- PCs and the web
- Inexpensive clusters and Grids

Today's examples

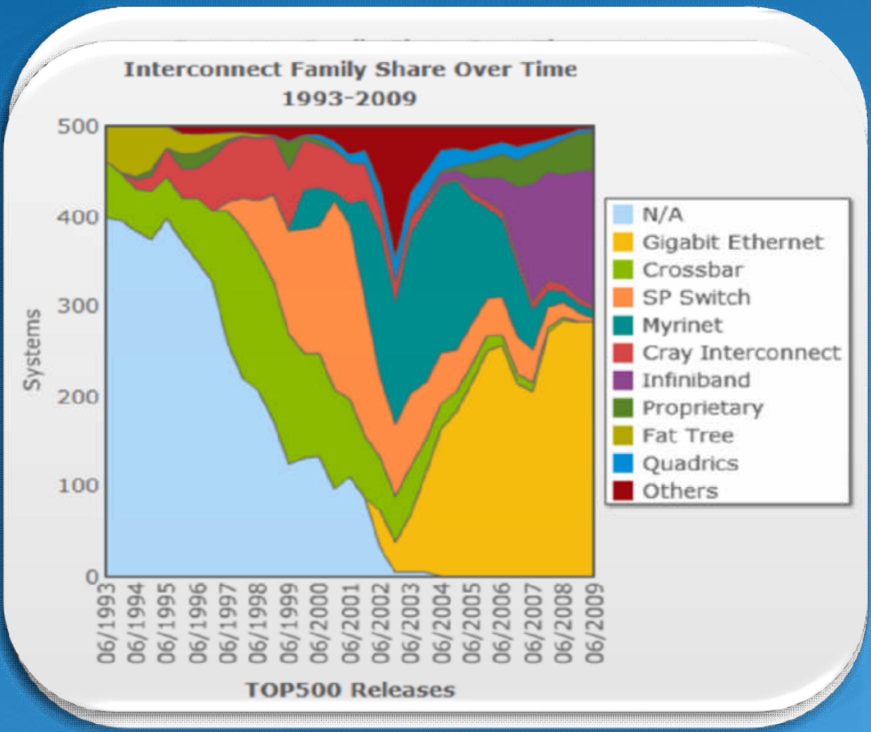
- multicore, sensors, clouds and internet scale services

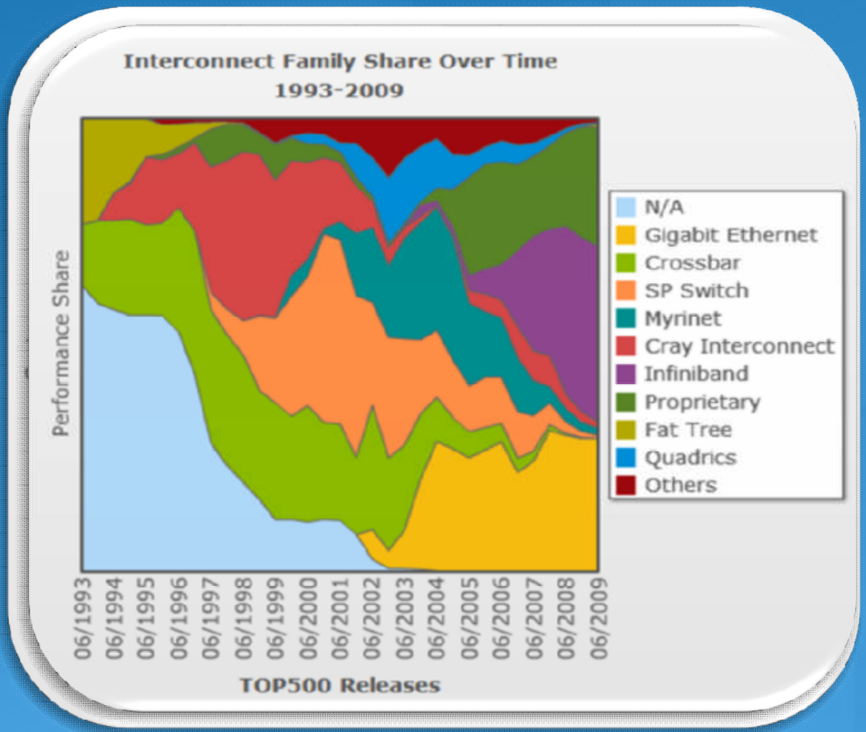
Today's Truisms (2009)

- Bulk computing is almost free
  … but applications and power are not
  & programming large systems is hard
- Inexpensive sensors are ubiquitous
  … but data fusion remains difficult
  & our ability to collect outpaces our ability to analyze
- Moving lots of data is {still} hard
  … because we're missing trans-terabit/sec networks
- People are really expensive!
  … robust software remains extremely labor intensive
- Our political/technical approaches must change
  … or we risk solving irrelevant problems

The Pull of Economics …

- Moore's "Law" favored consumer commodities
  - Specialized processors and systems faltered
  - "Killer micros" and industry standard blades led
  - Inexpensive clusters now dominate





The Pull of Economics …

- Moore's "Law" favored consumer commodities
  - Specialized processors and systems faltered
  - "Killer micros" and industry standard blades led
  - Inexpensive clusters now dominate
- Today's economics
  - Manycore processors/accelerators
  - Software as a service/cloud computing
  - Multidisciplinary data analysis and fusion
- This is the driving change in technical computing
  - Just as "killer micros" and inexpensive clusters

Cloud Economics

When applications are hosted

- Even sequential ones are embarrassingly parallel
- Few dependencies among users
- Unprecedented economies of scale

Moore's benefits accrue to platform owner

2x processors →

- ½ servers (+ ½ power, space, cooling …)
- Or 2X service at the same cost

Tradeoffs not entirely one-sided due to

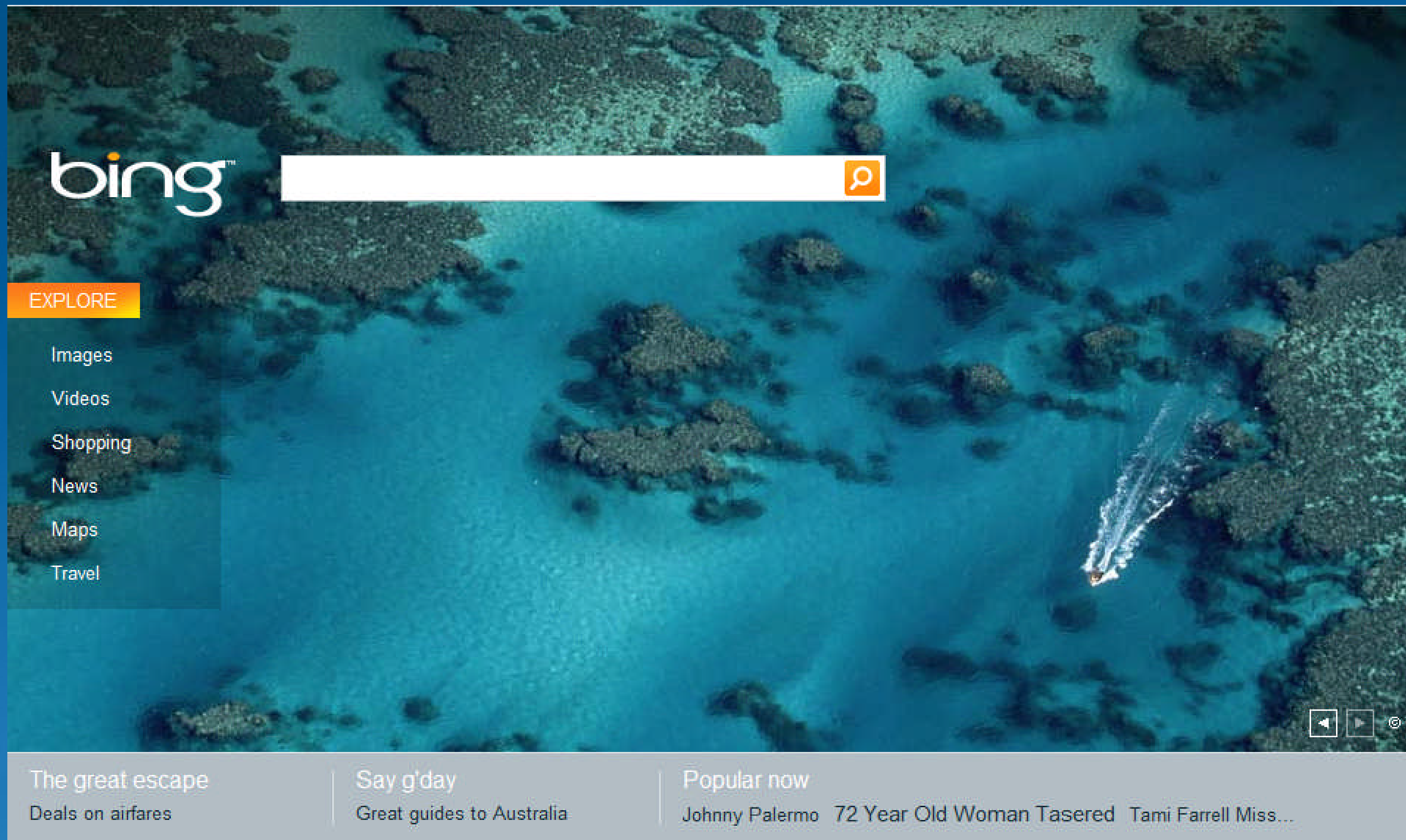
- Latency, bandwidth, privacy, off-line considerations
- Capital investment, security, programming problems

New Software Architecture

New platforms…

Insights: Not Just FLOPS Or Bytes

Software + Data + Services = Insights



When someone wants to find information on their favorite musician by submitting an internet search, they unleash the power of several hundred processors operating over terabytes of data. Why then can't a scientist seeking a cure for cancer invoke large amounts of computation over a terabyte-size database of DNA microarray data at the click of a button?

Randy Bryant (CMU) May 2007, with permission.

Microsoft's Dryad

- Running on >> 10^4 machines
- Analyzing > 10Pb data daily
- Runs on clusters > 3000 machines
- Handles jobs with > 10^5 processes each
- Used by >> 100 developers
- Rich platform for data analysis

Programming at Scale

3000 node cluster ($2.5k - $4k/per node)

- 12,000 cores (36×10^{12} cycles/sec)
- 48 terabytes of RAM
- 9 petabytes of persistent storage

But, very hard to utilize

- Hard to program 12,000 cores
- Something breaks every day
- Challenge to deploy and manage

Challenges of Large Scale Computing

Scalability

Adding load to a system should not cause outright failure but rather a graceful decline.

Reliability

Total system must support graceful decline in application performance rather than a full halt

Recoverability

If nodes fail, their workload must be picked up by functioning units.

Consistency

Concurrent operations or partial internal failures will not lead to externally visible nondeterminism.

Ability to Get Started Quickly

Developers can use their existing skills in a familiar environment

Distributed Data-Parallel Computing

- Nodes talk to each other as little as possible (shared nothing)
- Programmer is not allowed to communicate between nodes
- Data is spread throughout machines in advance, computation happens where it's stored.
- Master program divvies up tasks based on location of data, detects failures and restarts, load balances, etc…

Simple Programming Model

Terasort, well known benchmark, time to sort time 1 TB data [J. Gray 1985]

- Sequential scan/disk = 4.6 hours
- DryadLINQ provides simple but powerful programming model
- Only few lines of code needed to implement Terasort

```
DryadDataContext ddc = new DryadDataContext(fileDir);
DryadTable<TeraRecord> records =
     ddc.GetPartitionedTable<TeraRecord>(file);
var q = records.OrderBy(x => x);
q.ToDryadPartitionedTable(output);
```

LINQ

Microsoft's Language **IN**tegrated **Q**uery

Available in Visual Studio 2008

A set of operators to manipulate datasets in .NET

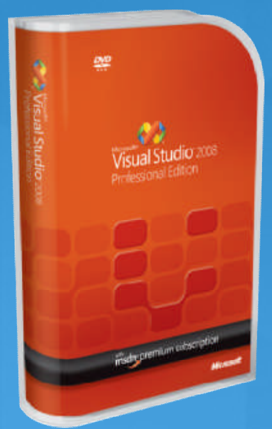
- Support traditional relational operators
  - Select, Join, GroupBy, Aggregate, etc.

Data model

- Data elements are strongly typed .NET objects
- Much more expressive than SQL tables

Extremely extensible

- Add new custom operators
- Add new execution providers



DryadLINQ Operators

LINQ operators

- Where, Select, SelectMany, OrderBy, GroupBy, Join, GroupJoin, Aggregate, Distinct, Concat, Union, Intersect, Except, Count, Contains, Sum, Min, Max, Average, Any, All, Skip, SkipWhile, Take, TakeWhile, …

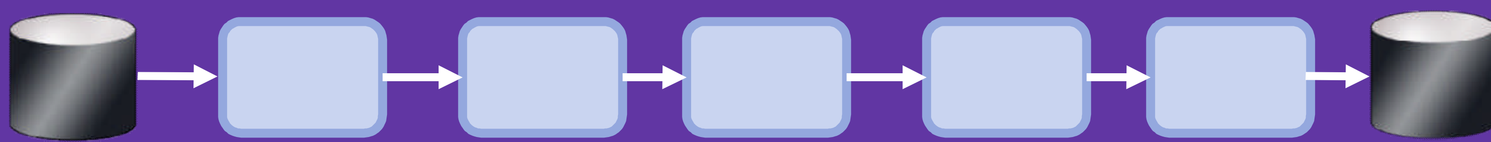
Operators introduced by DryadLINQ

- HashPartition, RangePartition, Apply, Fork, Materialize

Dryad Generalizes Unix Pipes

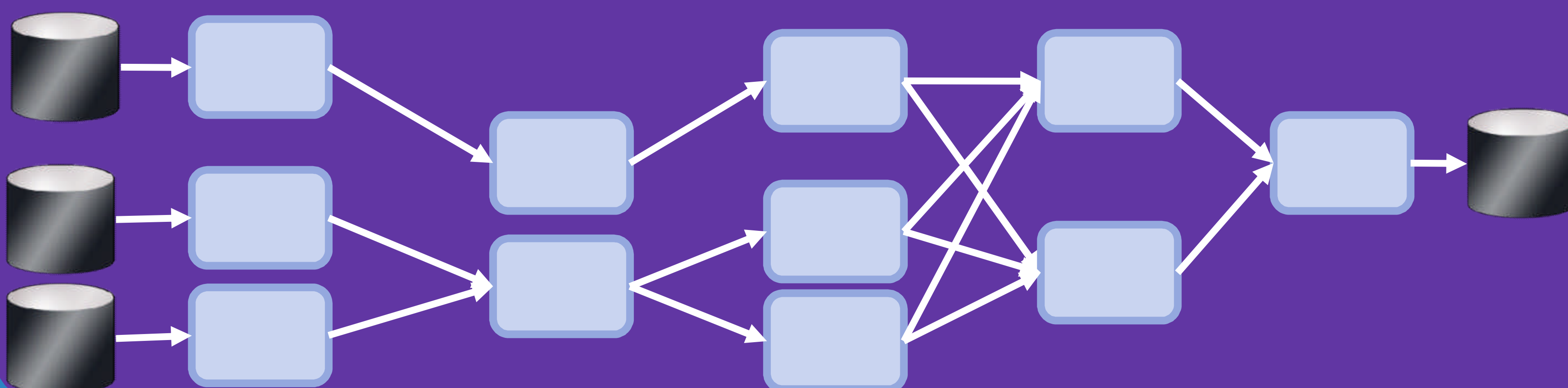
Unix Pipes: 1-D

grep | sed | sort | awk | perl

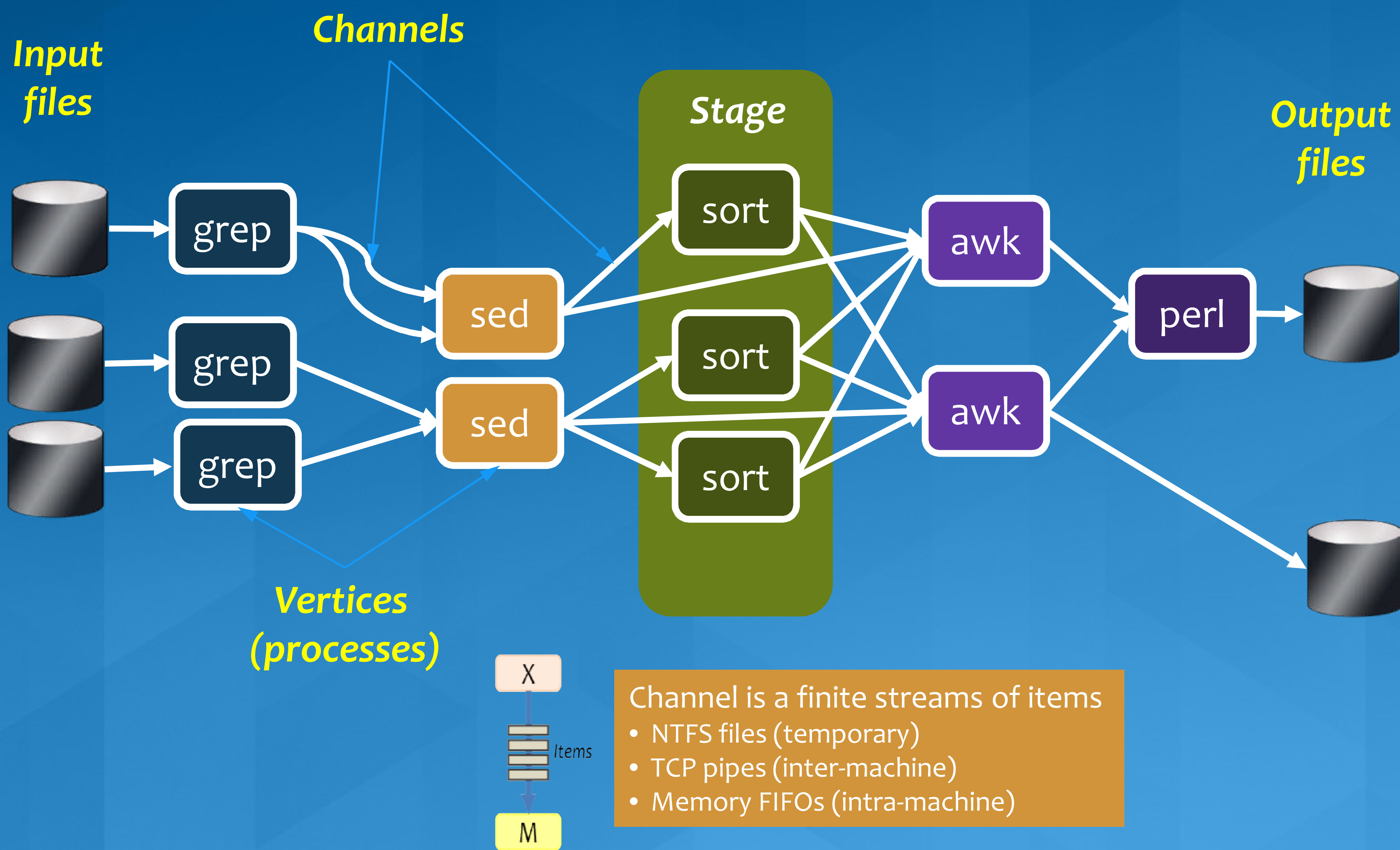


Dryad: 2-D, multi-machine, virtualized

grep^{1000} | sed^{500} | sort^{1000} | awk^{500} | perl^{50}

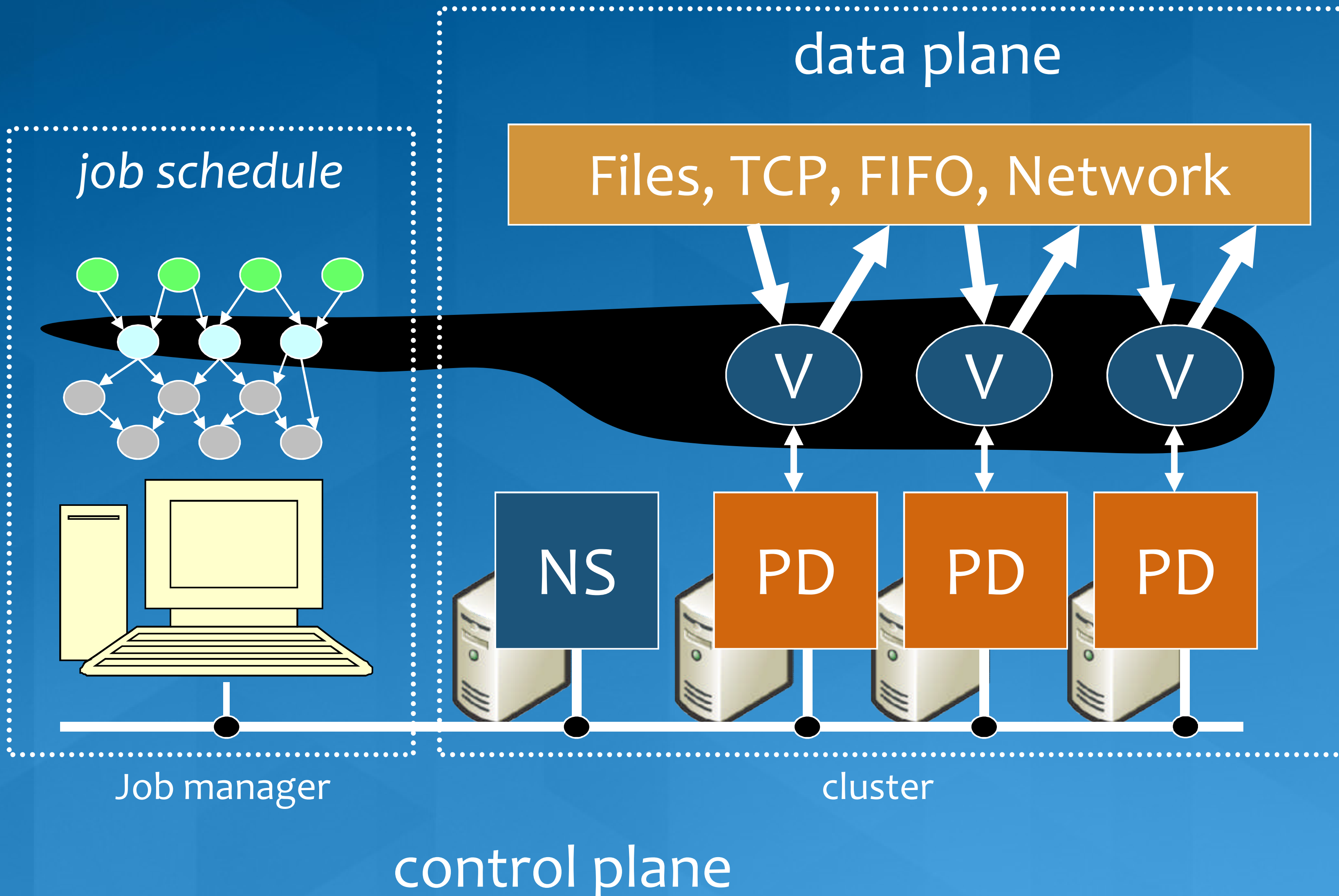


Dryad Job Structure

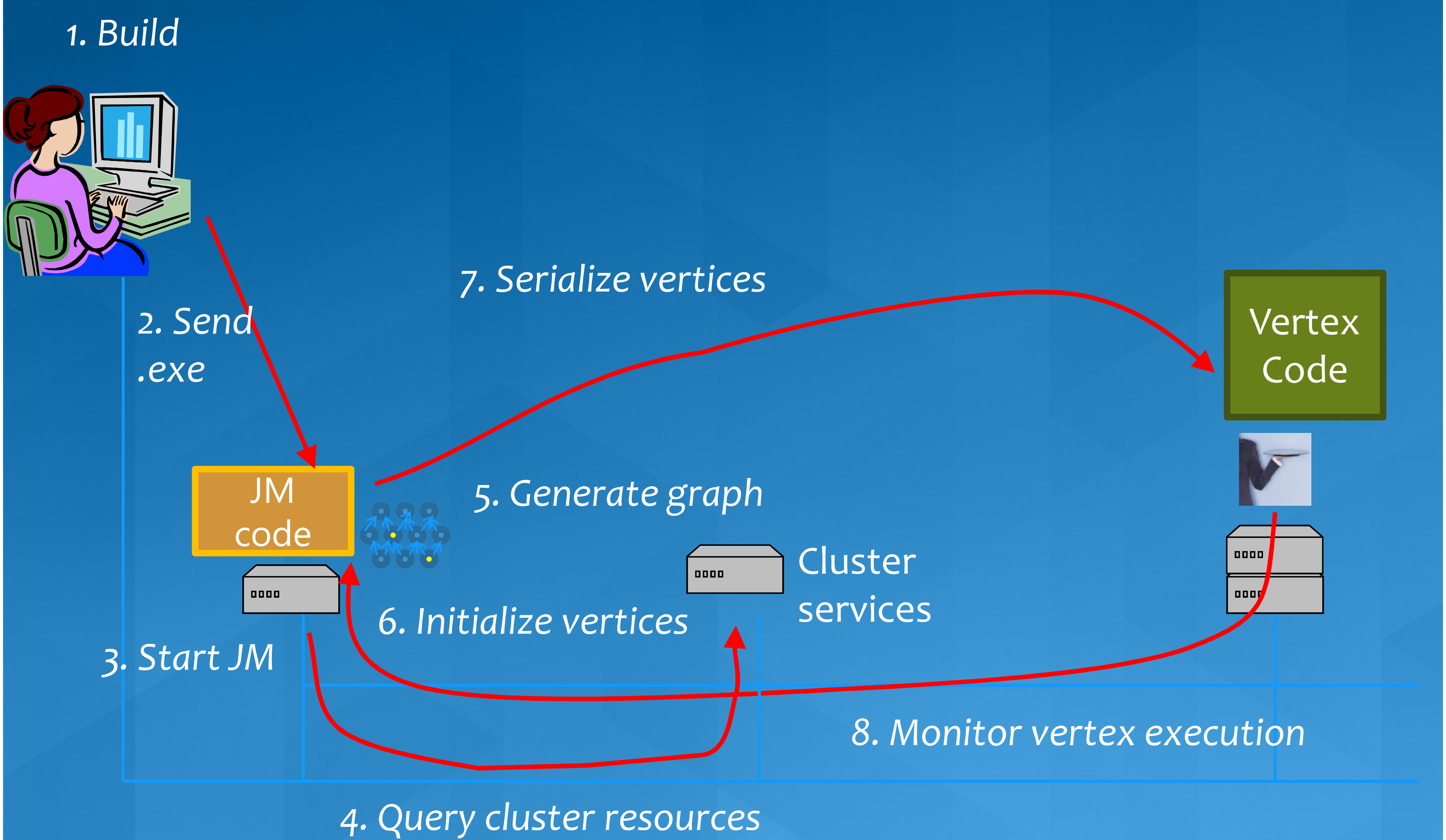


Channel is a finite streams of items
- NTFS files (temporary)
- TCP pipes (inter-machine)
- Memory FIFOs (intra-machine)

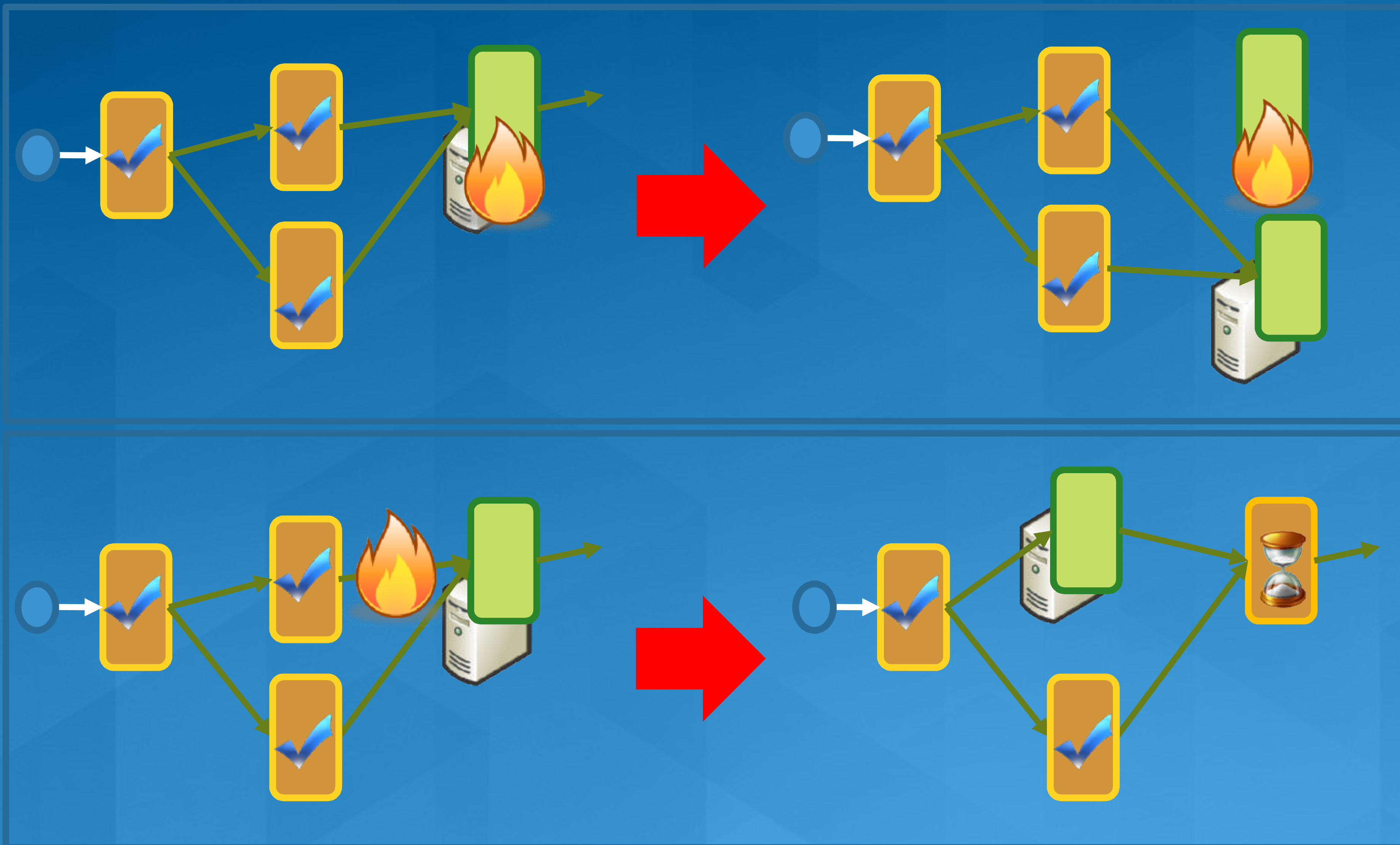
Dryad System Architecture



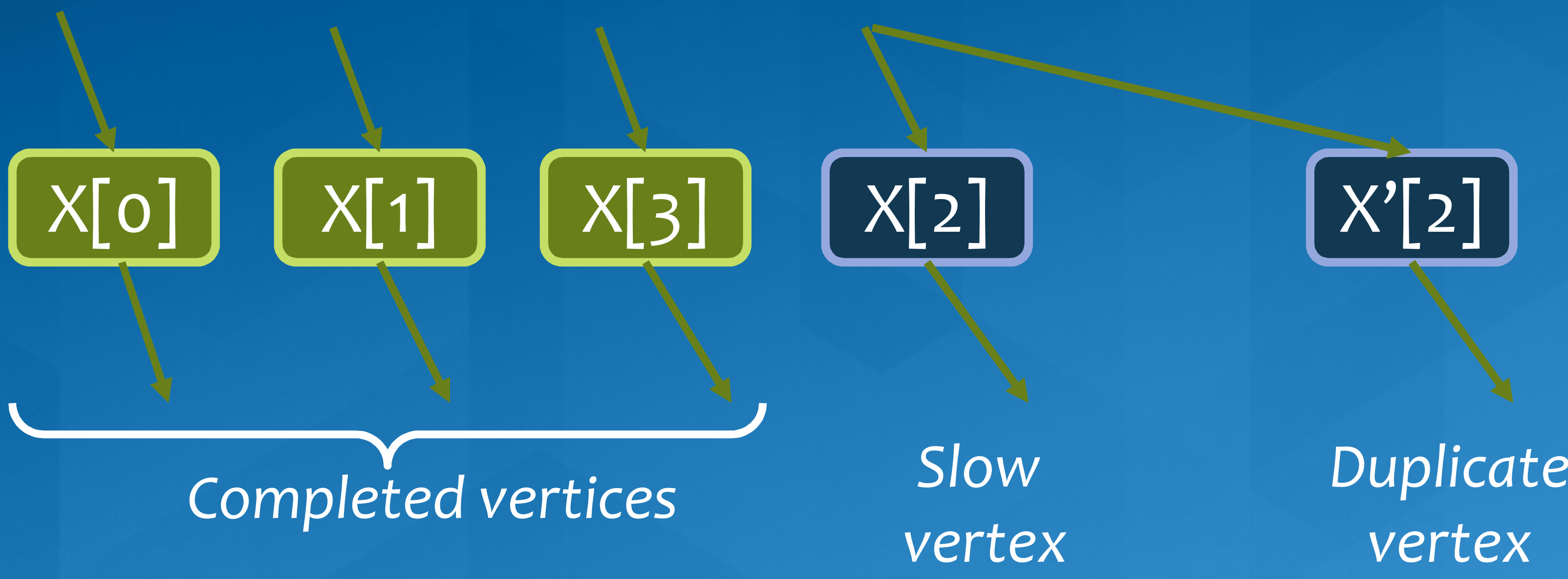
Dryad Job Staging



Fault Tolerance



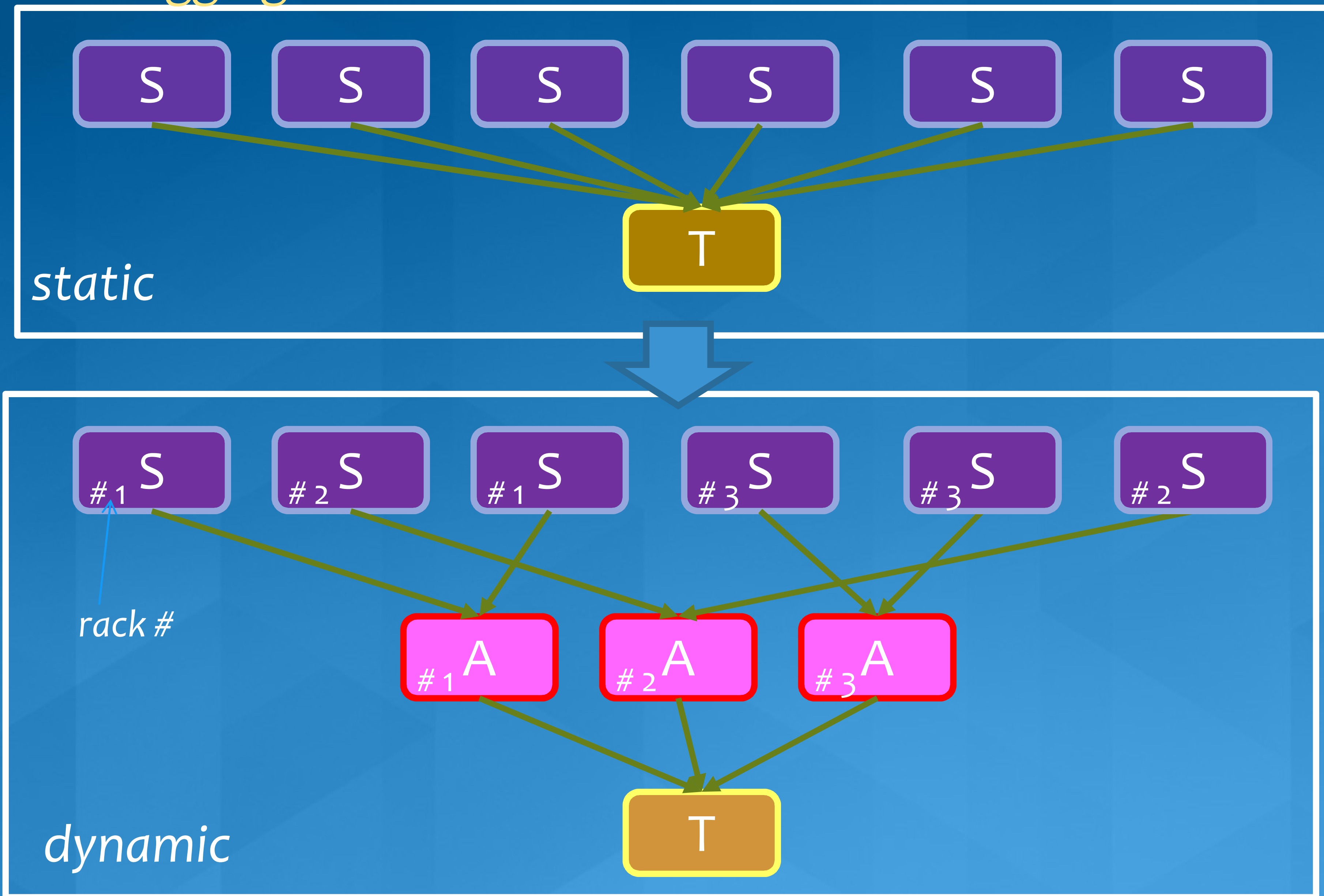
Dynamic Graph Rewriting



Duplication Policy = f(running times, data volumes)

Dryad

Dynamic Aggregation



Example: Histogram

```
public static IQueryable<Pair> Histogram(
 IQueryable<LineRecord> input, int k)
{
  var words = input.SelectMany(x => x.line.Split(' '));
  var groups = words.GroupBy(x => x);
  var counts = groups.Select(x => new Pair(x.Key, x.Count())
  var ordered = counts.OrderByDescending(x => x.count);
  var top = ordered.Take(k);
 return top;
}
```

“A line of words of wisdom”
[“A”, “line”, “of”, “words”, “of”, “wisdom”]
[[“A”], [“line”], [“of”, “of”], [“words”], [“wisdom”]]
[{“A”, 1}, {“line”, 1}, {“of”, 2}, {“words”, 1}, {“wisdom”, 1}]
[{“of”, 2}, {“A”, 1}, {“line”, 1}, {“words”, 1}, {“wisdom”, 1}]
[{“of”, 2}, {“A”, 1}, {“line”, 1}]

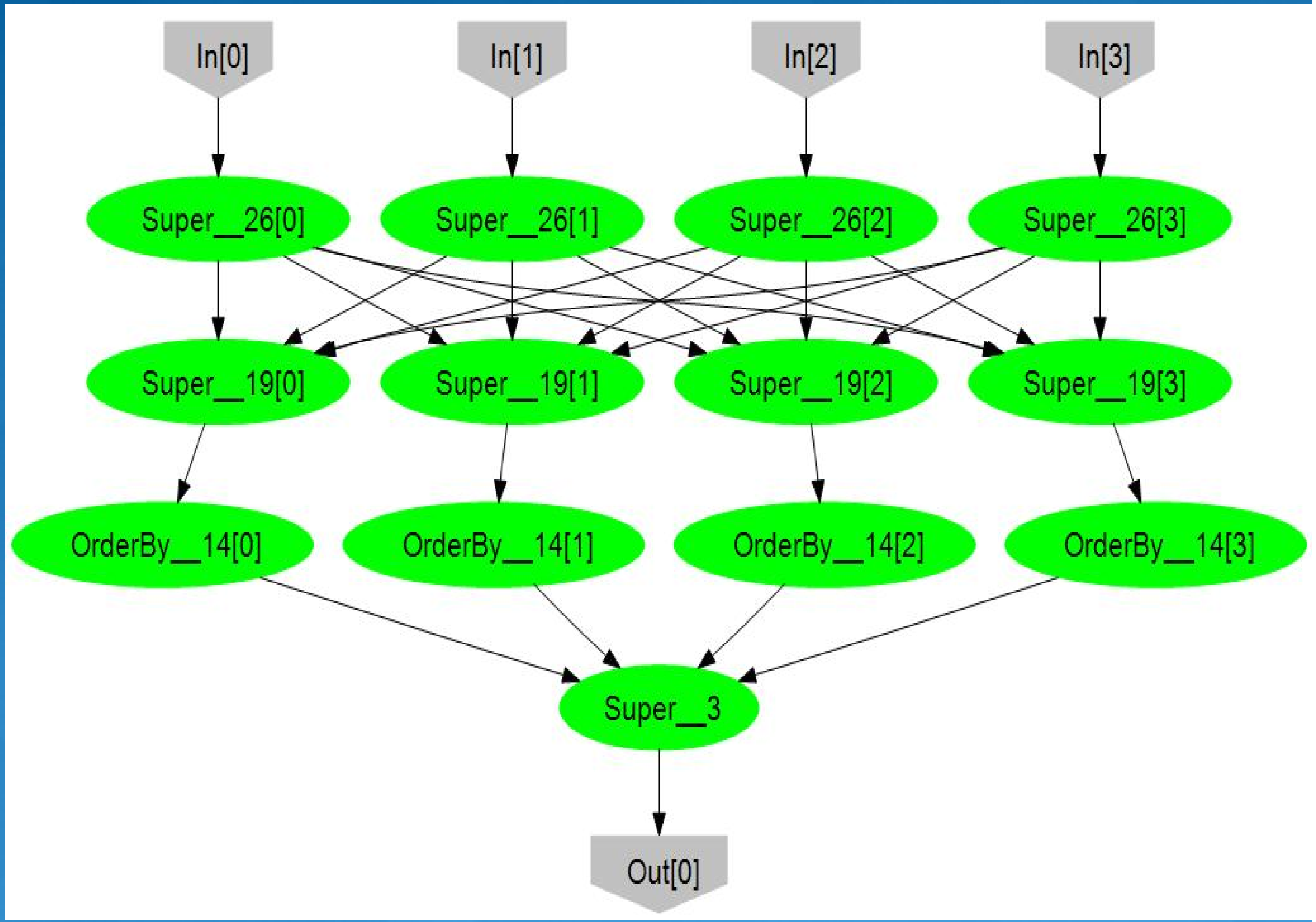
Histogram Plan

SelectMany
HashDistribute

Merge
GroupBy
Select

OrderByDescending
Take

MergeSort
Take



Dryad Scheduler is a State Machine

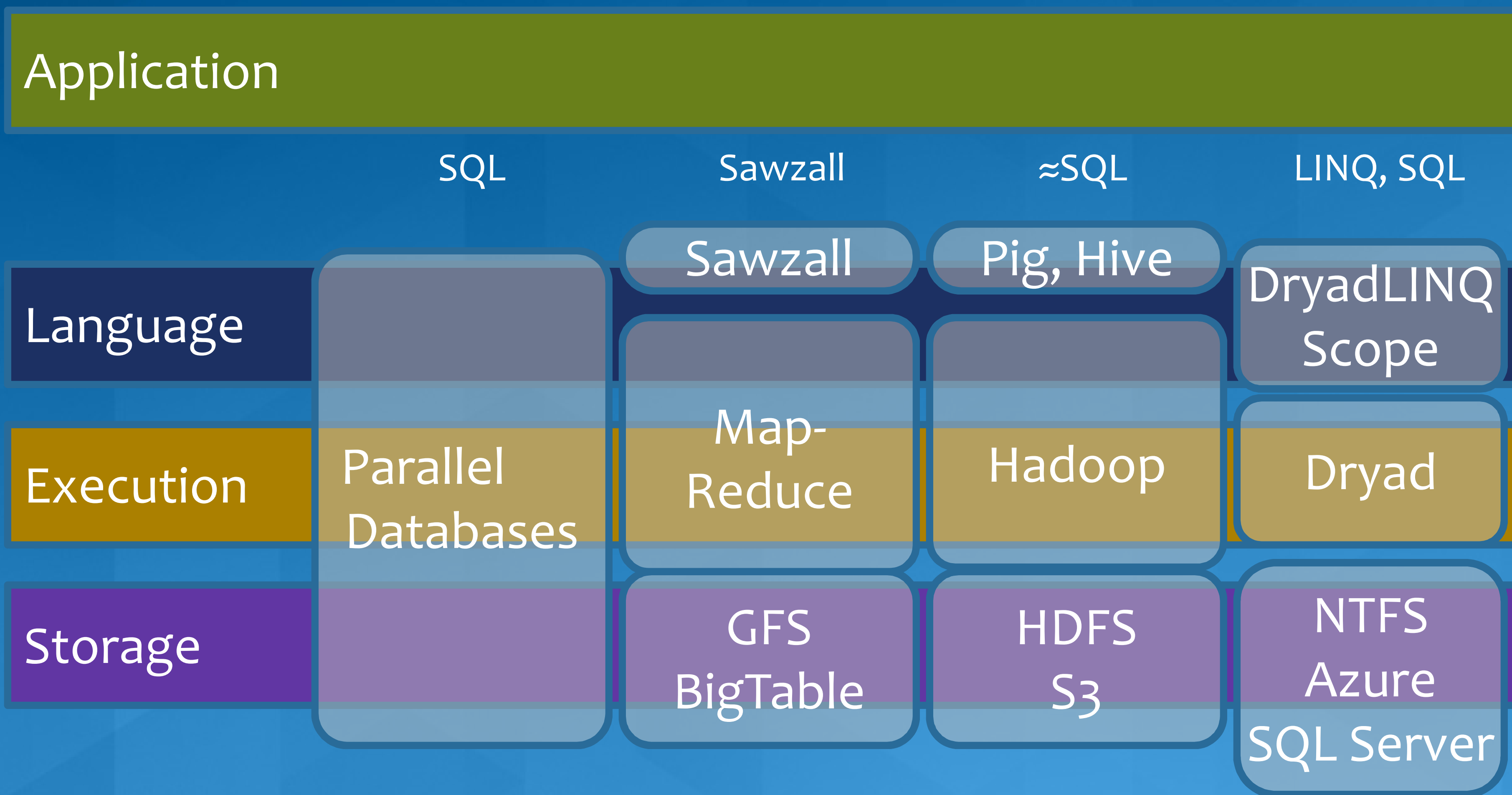
Static optimizer builds execution graph

- Vertex can run anywhere once all its inputs are ready.

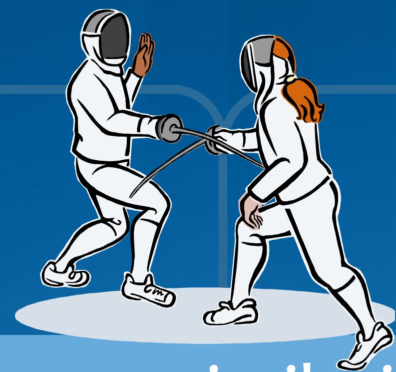
Dynamic optimizer mutates running graph

- Distributes code, routes data;
- Schedules processes on machines near data;
- Adjusts available compute resources at each stage;
- Automatically recovers computation, adjusts for overload
  - If A fails, run it again;
  - If A's inputs are gone, run upstream vertices again (recursively);
  - If A is slow, run a copy elsewhere and use output from one that finishes first.
- Masks failures in cluster and network;

Dryad in Context



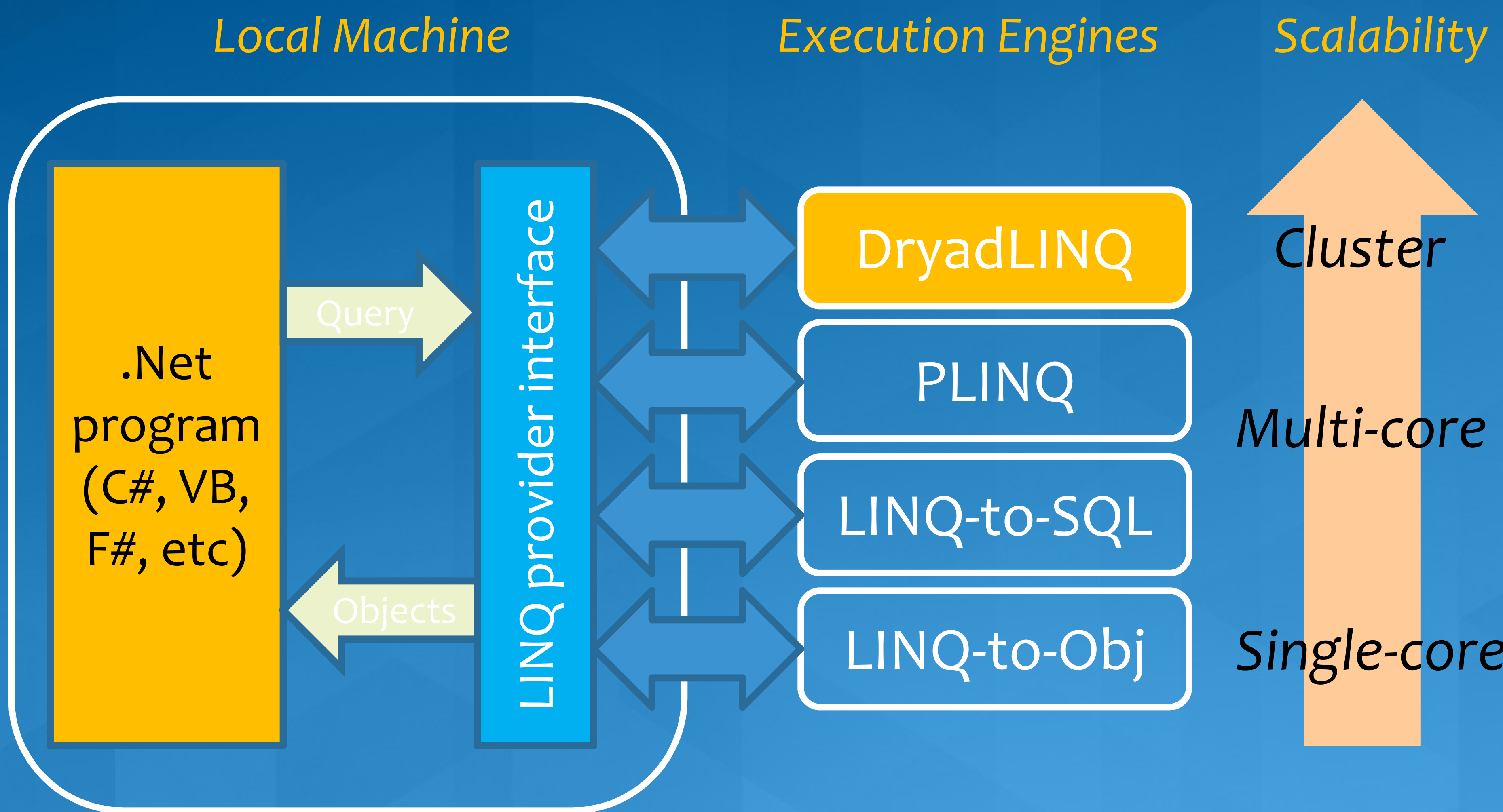
Dryad vs. Map-Reduce



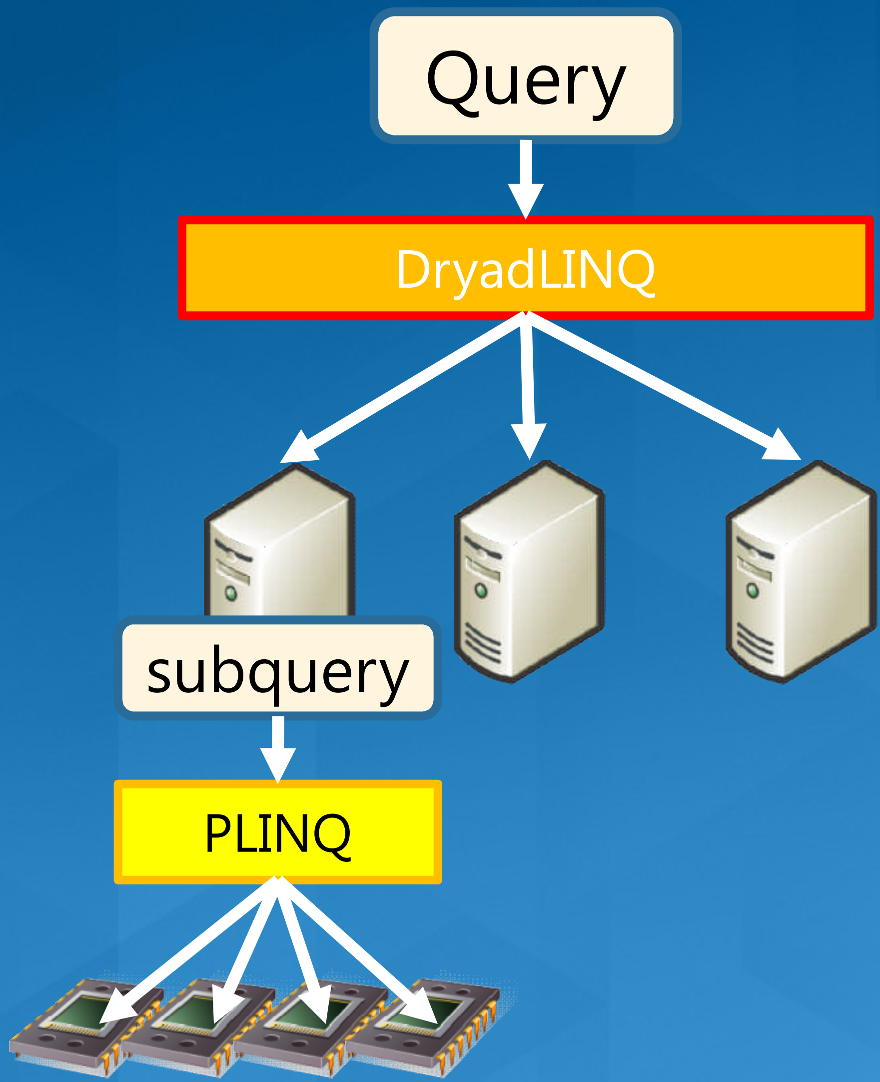
many similarities

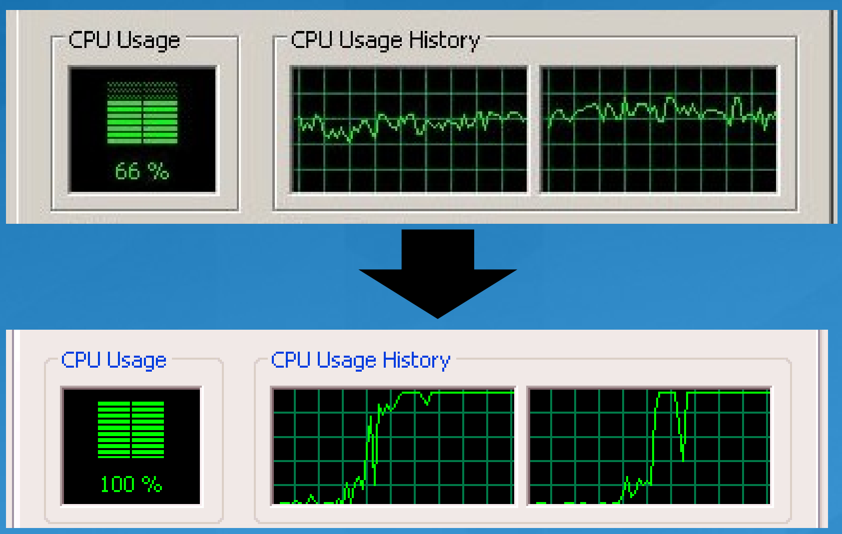
Dryad	Map-Reduce
• Execution layer • Job = arbitrary DAG • Plug-in policies • Program=graph gen.	• Exe + app. model • Map+sort+reduce • Few policies • Program=map+reduce
• Complex (features) • New (< 4 years) • Still growing • Internal (pending)	• Simple • Mature (> 4 years) • Widely deployed • Hadoop

Combining Query Providers

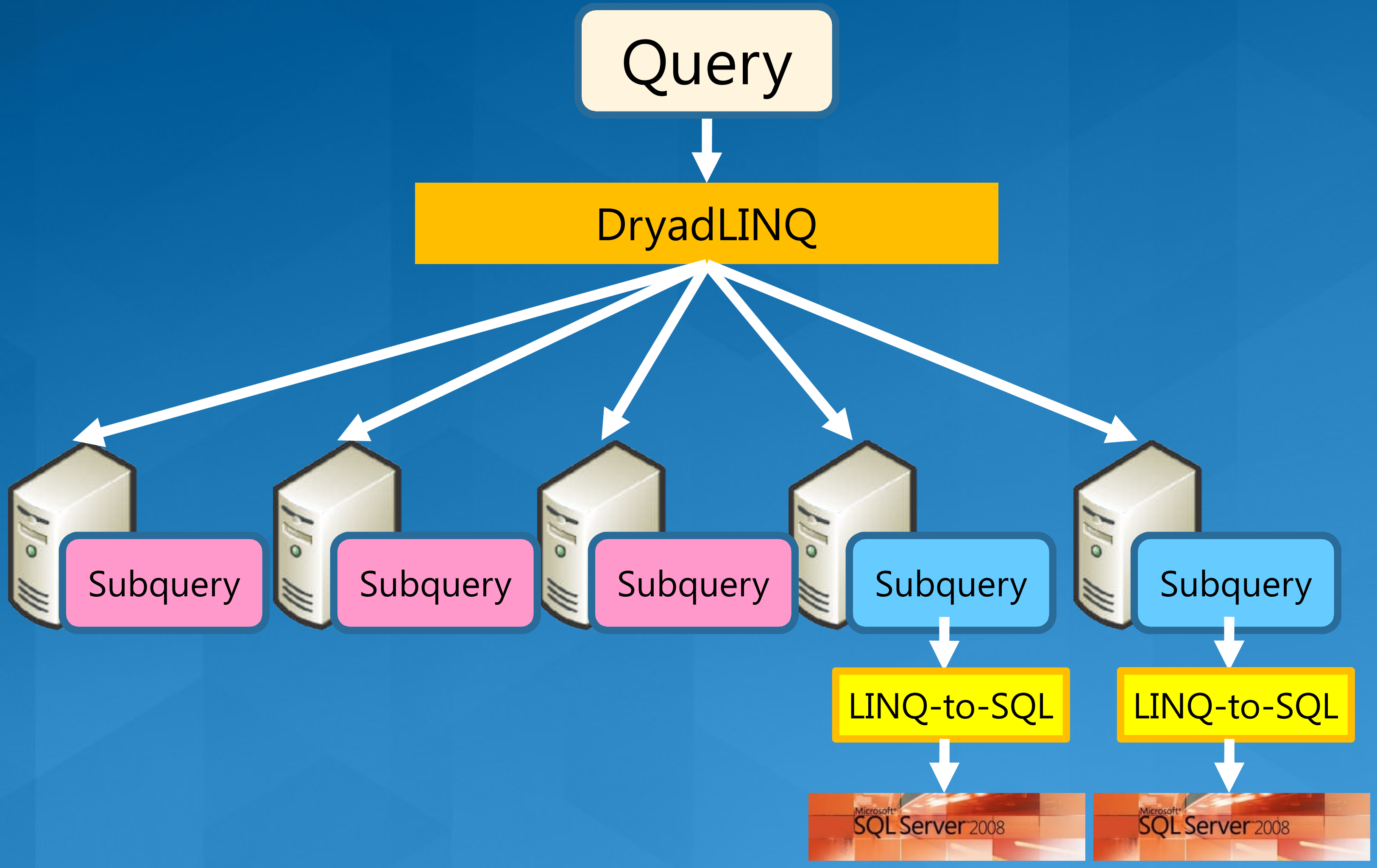


Combining with PLINQ





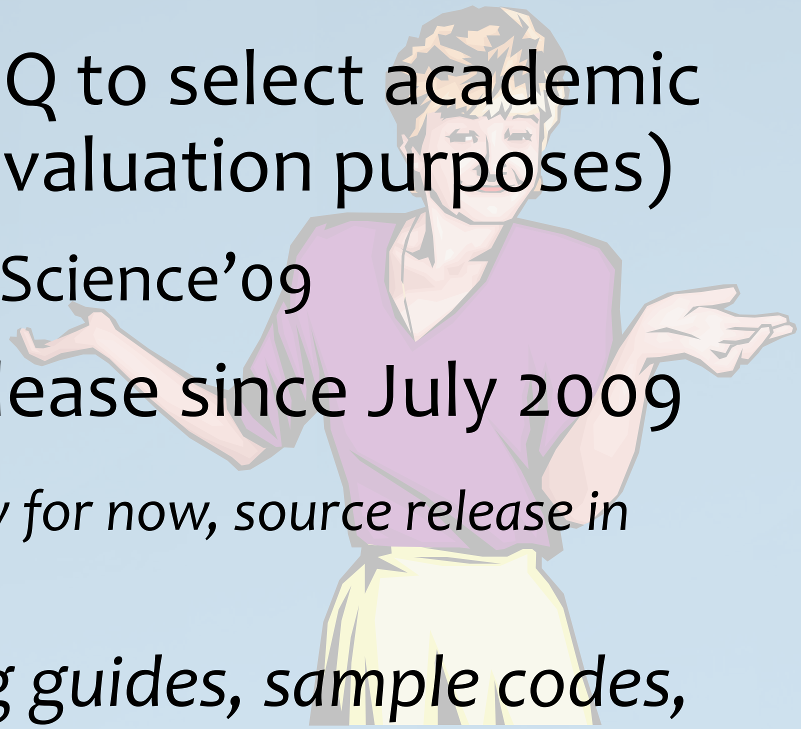
Combining with LINQ-to-SQL



Sample applications written using DryadLINQ	Class
Distributed linear algebra	Numerical
Accelerated Page-Rank computation	Web graph
Privacy-preserving query language	Data mining
Expectation maximization for a mixture of Gaussians	Clustering
K-means	Clustering
Linear regression	Statistics
Probabilistic Index Maps	Image processing
Principal component analysis	Data mining
Probabilistic Latent Semantic Indexing	Data mining
Performance analysis and visualization	Debugging
Road network shortest-path preprocessing	Graph
Botnet detection	Data mining
Epitome computation	Image processing
Neural network training	Statistics
Parallel machine learning framework infer.net	Machine learning
Distributed query caching	Optimization
Image indexing	Image processing
Web indexing structure	Web graph

"What's the point if I can't have it?"

- Glad you asked
- We offered Dryad+DryadLINQ to select academic partners (alpha release for evaluation purposes)
  - See the proceedings of IEEE eScience'09
- Broad academic/research release since July 2009
  - Dryad and DryadLINQ (*binary for now, source release in planning)*
  - *With tutorials, programming guides, sample codes, libraries, and a community site.*
  - *Copies available here on USB drives*
  - *http://research.microsoft.com/en-us/collaboration/tools/dryad.aspx*




What *is* a "cloud computing"?
"… data as a service… "
"cloud computing journal reports that… "
"… everything as a service…"
"… software as a service… "

So What is Cloud Computing?...

Using a remote data center to manage scalable, reliable, on-demand access to application services and data.

Scalable means

- millions of simultaneous users
- Exploiting thousand-fold parallelism

Reliable means 5 “nines” on-demand available right now

Each data center is
11.5 times
the size of a football field

So What is Cloud Computing?...

Unprecedented economies of scale. Approx costs for a med size center,1K servers, and large, 50K server center.

Technology	Cost in Medium-sized Data Center	Cost in Very Large Data Center	Ratio
Network	\$95 per Mbps/ month	\$13 per Mbps/ month	7.1
Storage	\$2.20 per GB/ month	\$0.40 per GB/ month	5.7
Admin	~140 servers/ Administrator	>1000 Servers/ Administrator	7.1

Economies of Scale

Electricity
Put Datacenters at Cheap Power

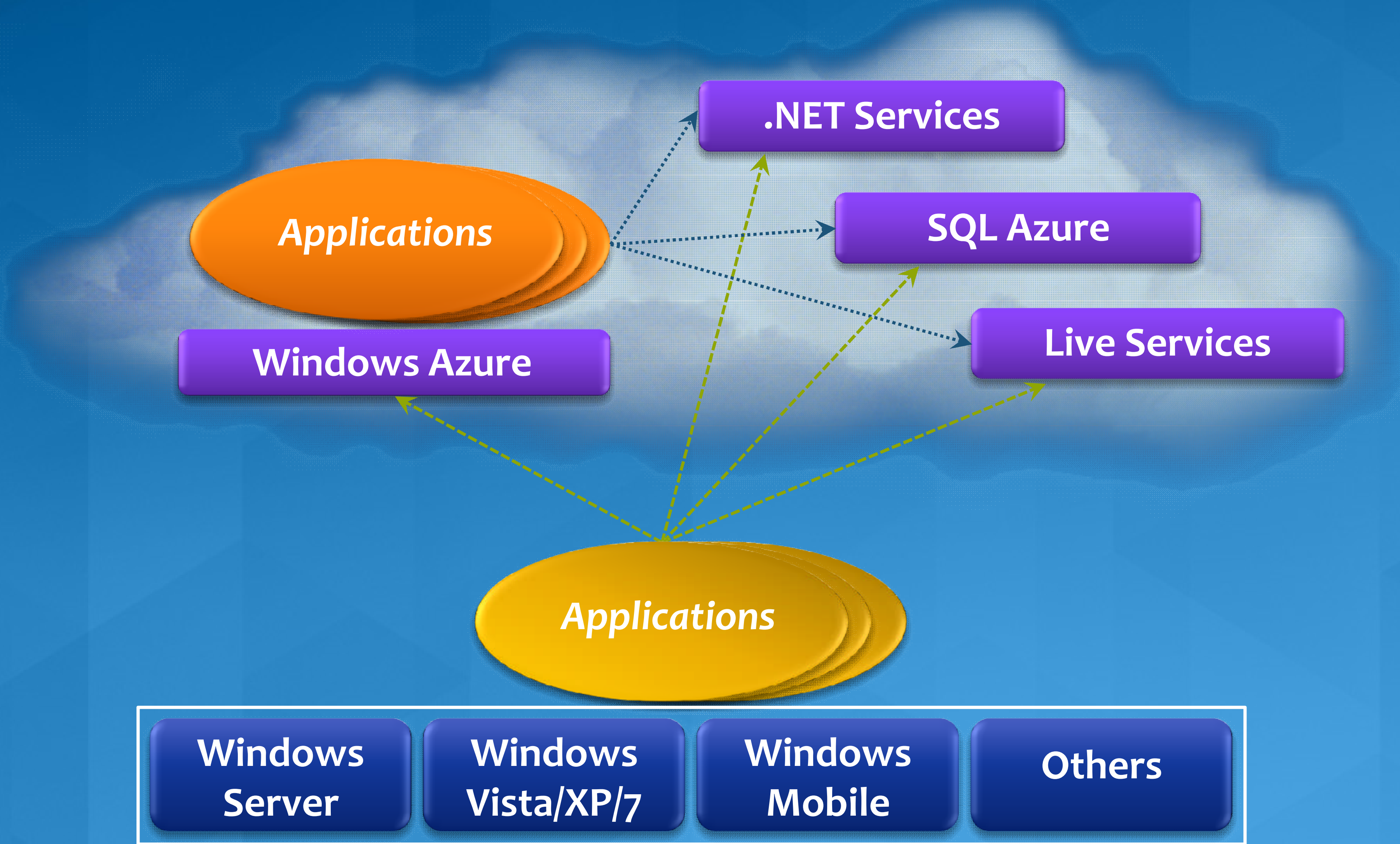
Operations
Standardize Automate Ops

Network
Put Datacenters on Main Trunks

Hardware
Containerized Low-Cost Servers

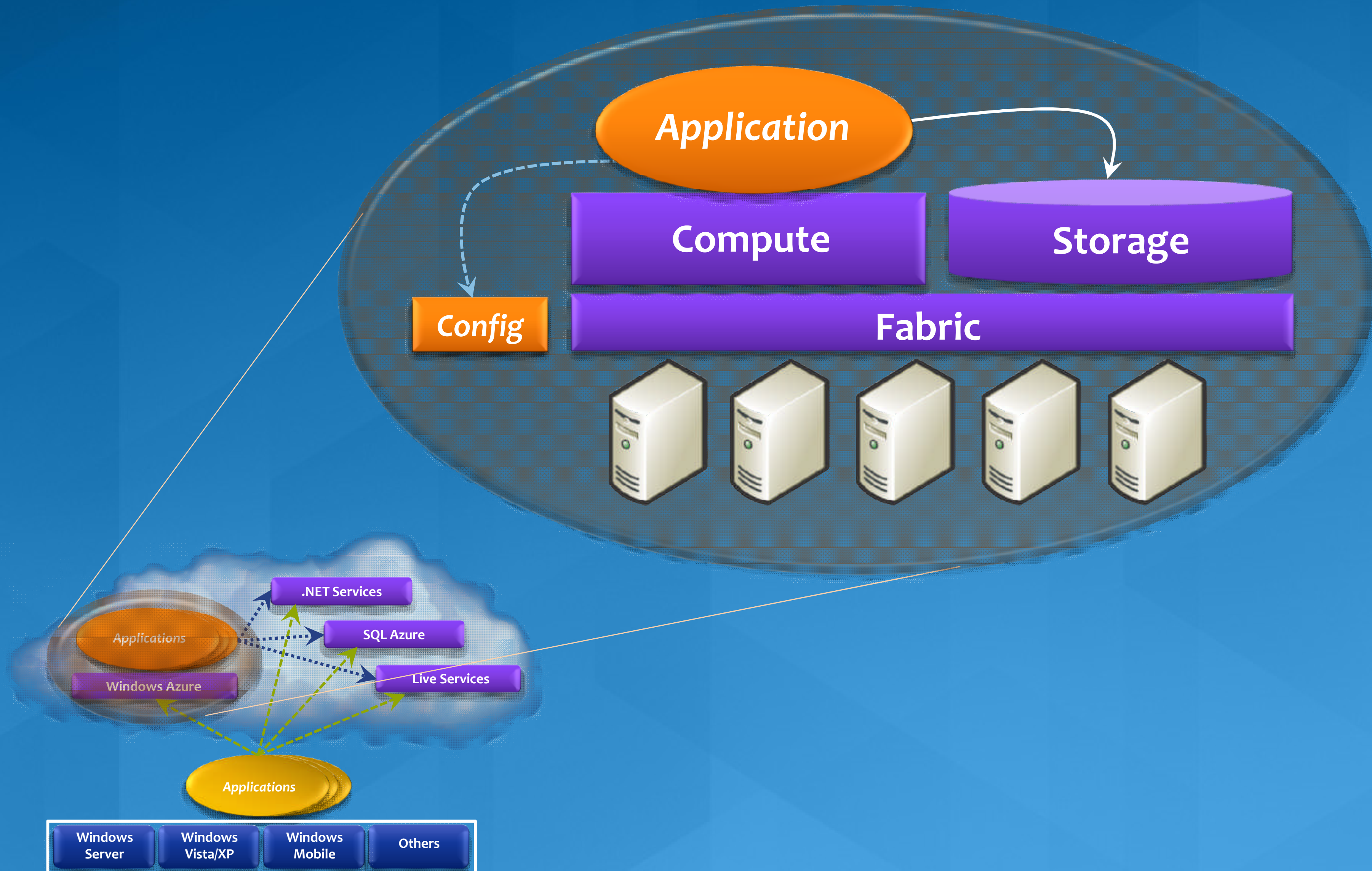
Modern Data Center: *Containers Separating Concerns*

The Windows Azure Platform



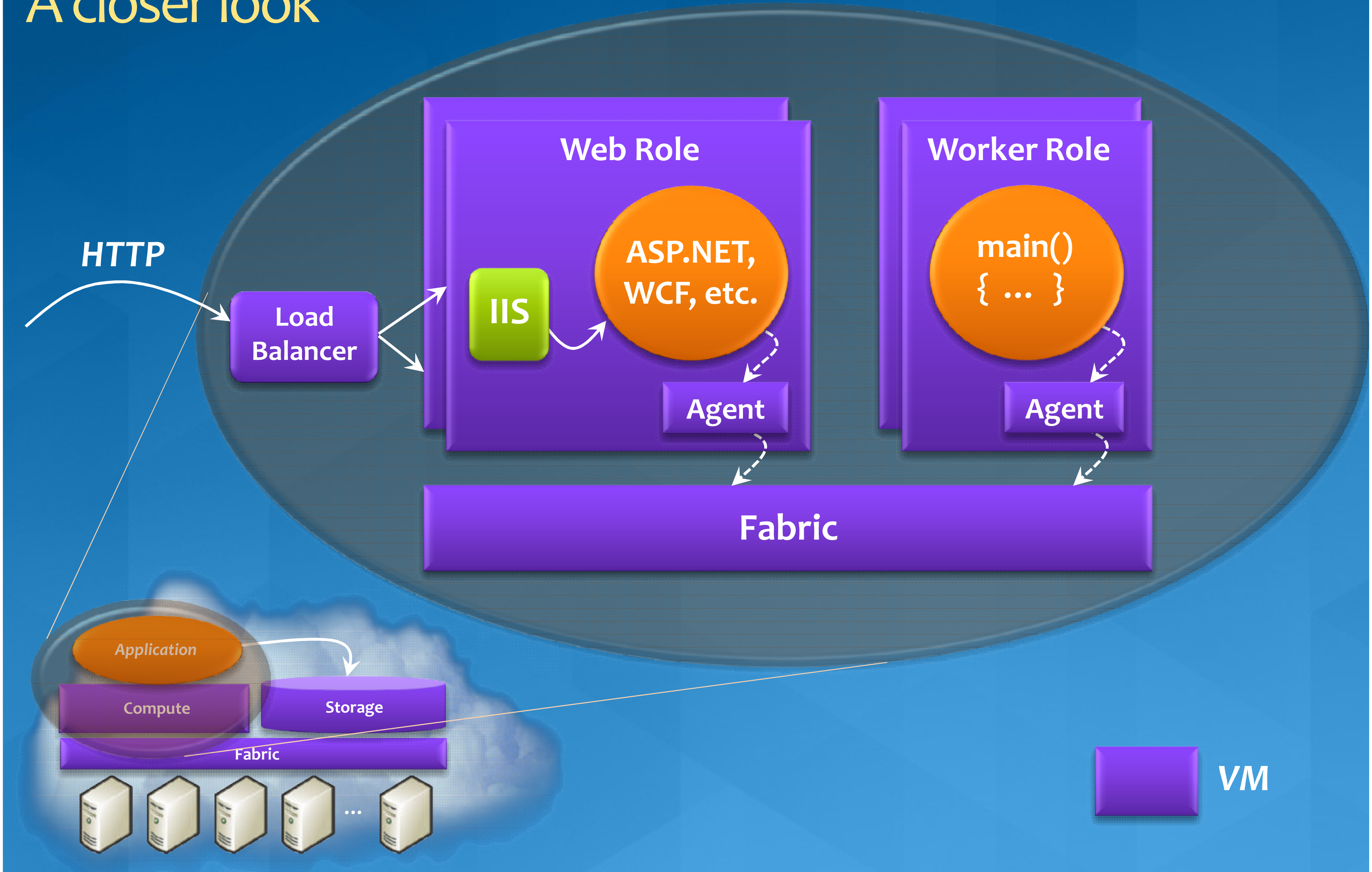
Windows Azure

An illustration



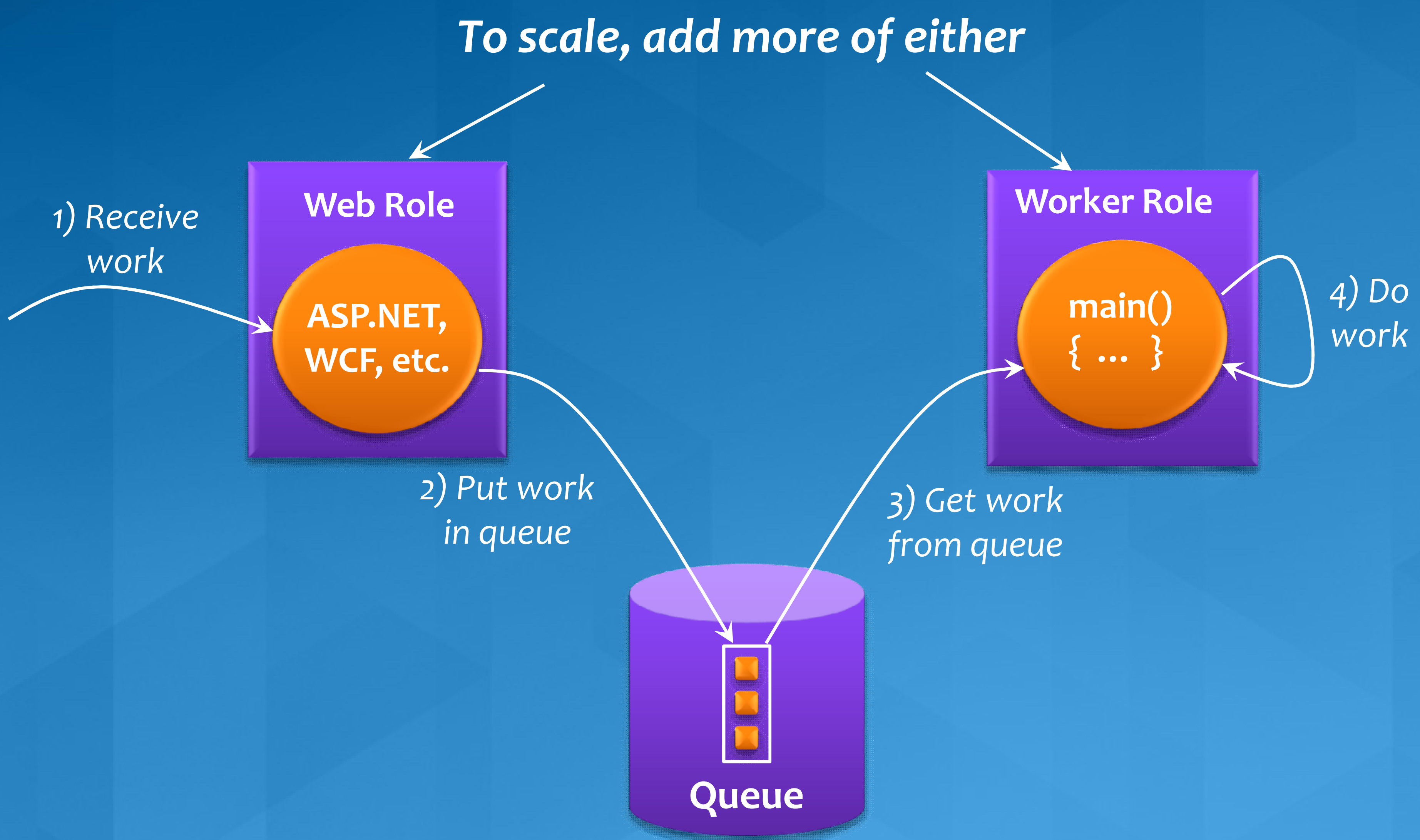
Windows Azure Compute Service

A closer look



The Suggested Application Model

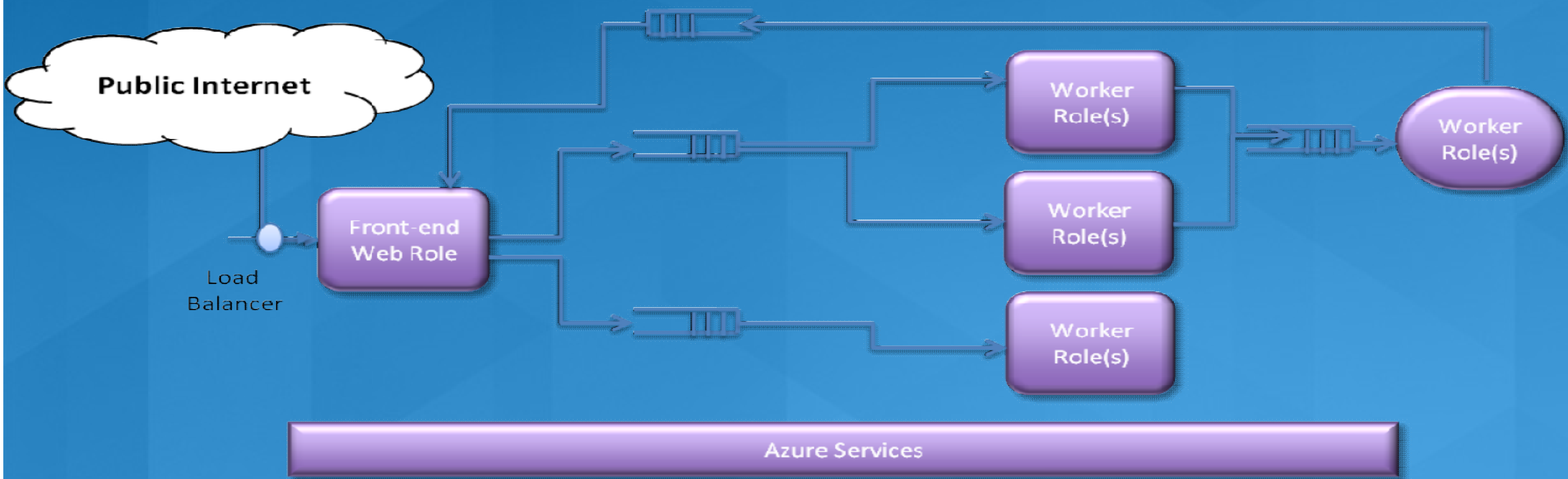
Using queues



Scalable, Fault Tolerant Applications on Azure

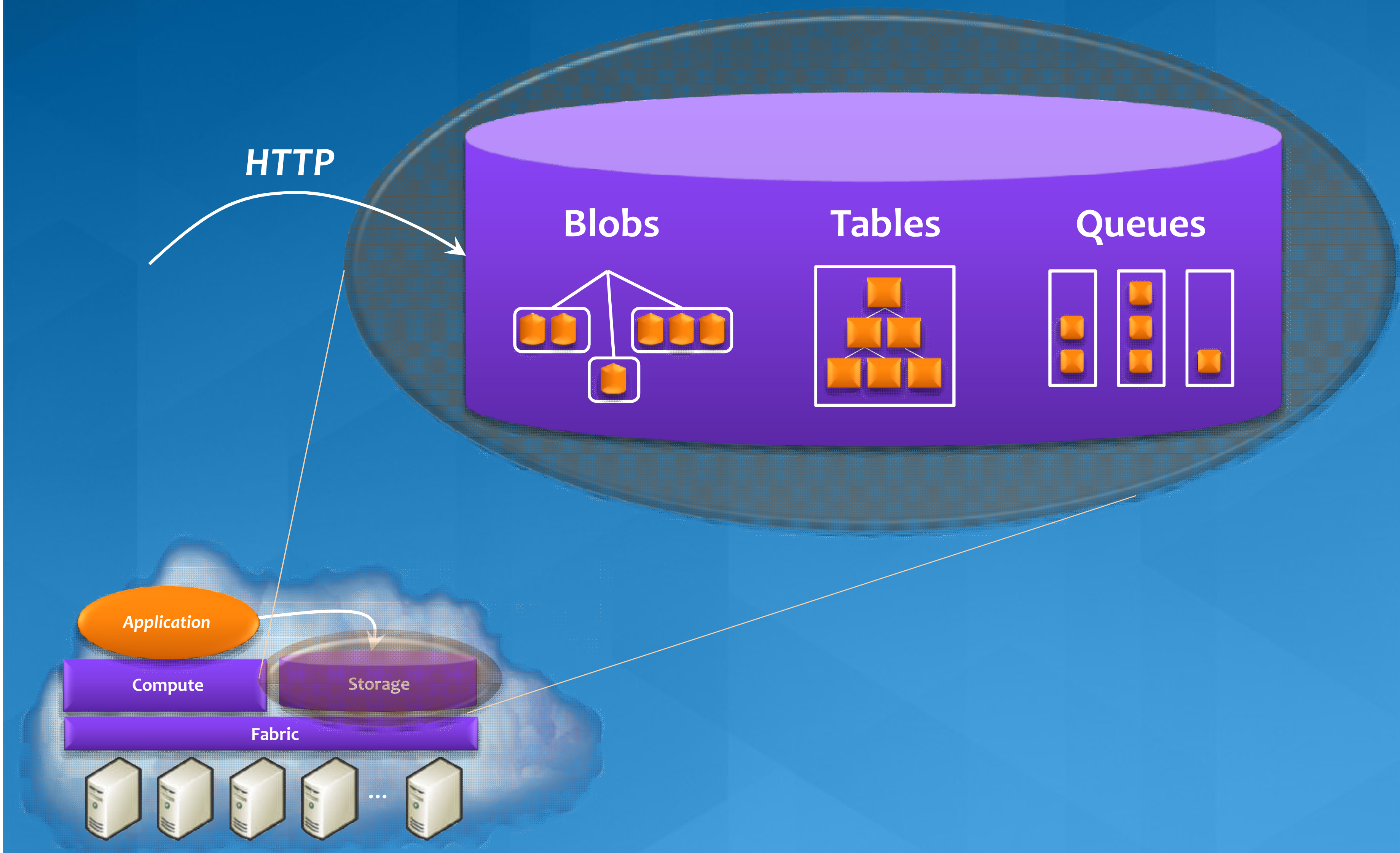
Queues are the application glue

- Queues decouple different parts of application, making it easier to scale app parts independently;
- Flexible resource allocation, different priority queues and separation of backend servers to process different queues.
- Queues mask faults in worker roles.



Windows Azure Storage Service

A closer look



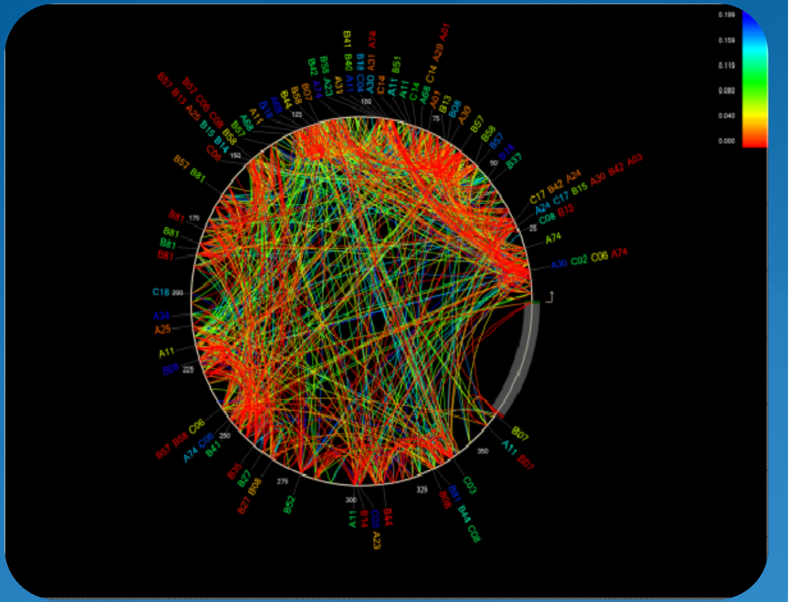
Science Example: *PhyloD as an Azure Service*

Statistical tool used to analyze DNA of HIV from large studies of infected patients

PhyloD developed by MSR and has been highly impactful

Small but important group of researchers

- *100's of HIV and HepC researchers actively use it*
- *1000's of research communities rely on results*



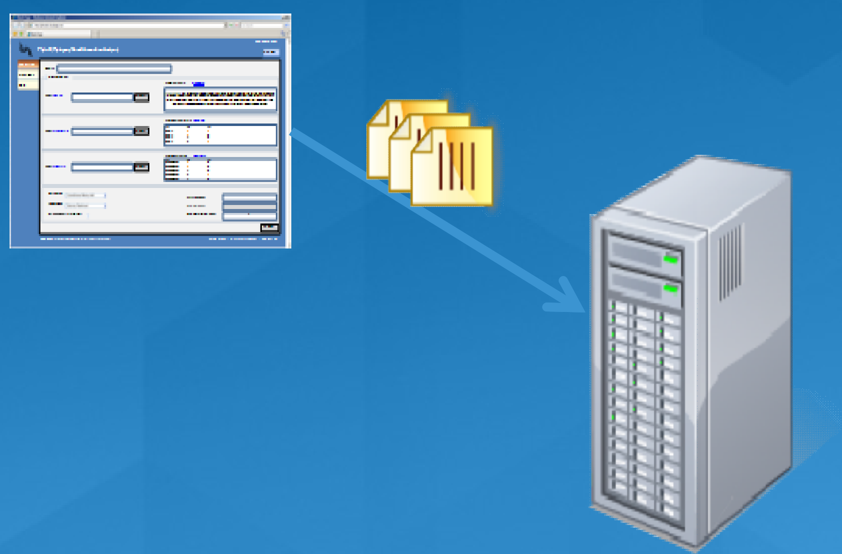
Cover of PLoS Biology November 2008

Typical job, 10 – 20 CPU hours
Extreme jobs require 1K – 2K CPU hours

PhyloD Demonstration

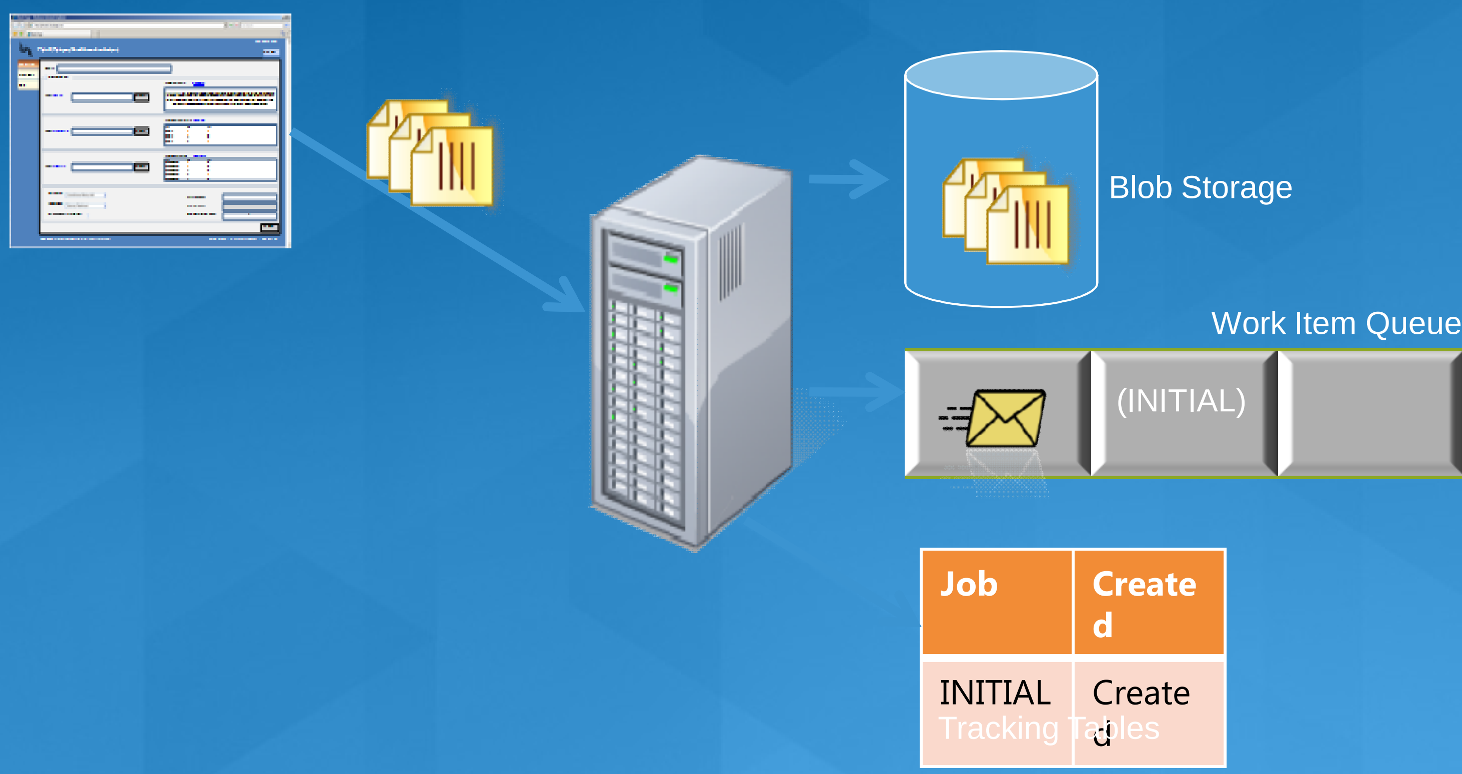
PhyloD as an Azure Service

- The web role provides an interface to the clients. Worker roles perform actual computation. Web role and worker roles share information using blobs, queue and tables.



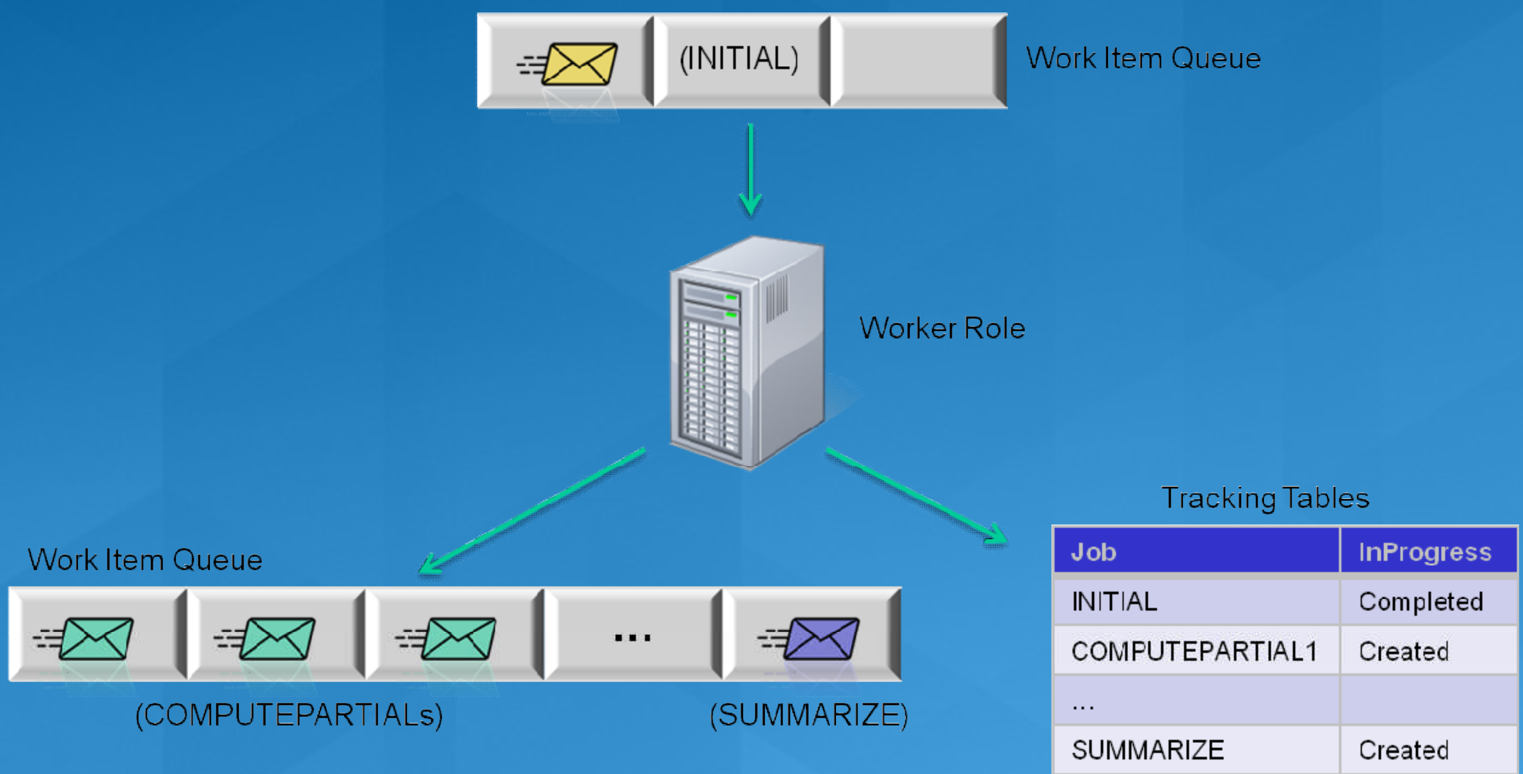
PhyloD as an Azure Service

- Web role copies input tree, predictor and target files to blob storage, enqueues INITIAL work item and updates tracking tables.



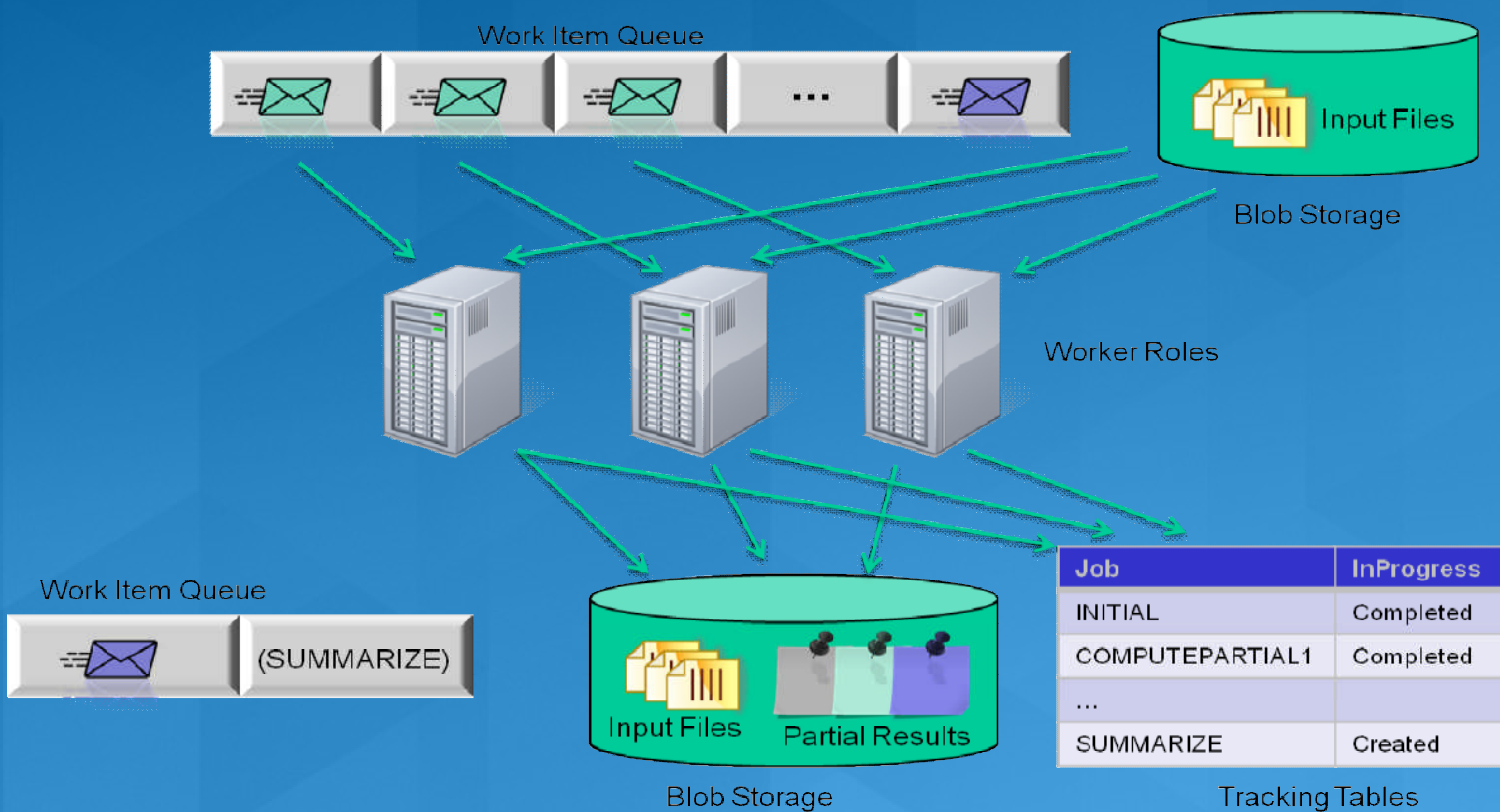
PhyloD as an Azure Service

- Worker role enqueues a COMPUTEPARTIAL work item for each partition of the input problem followed by a SUMMARIZE work item to aggregate the partial results and finally updates the tracking tables.



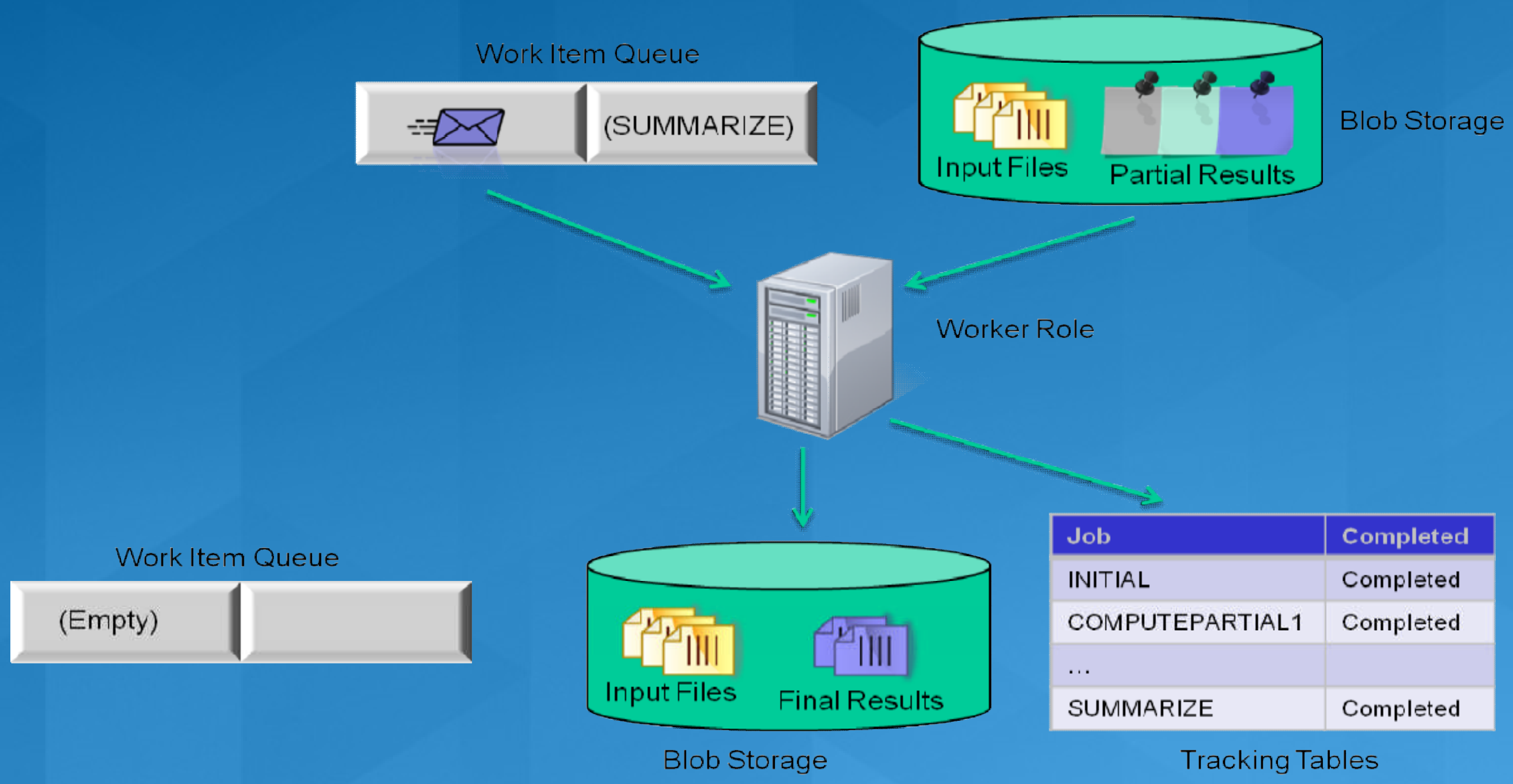
PhyloD as an Azure Service

- Worker role copies the input files to its local storage, computes p-values for a subset of the allele-codon pairs, copies the partial results back to blob storage and updates the tracking tables.



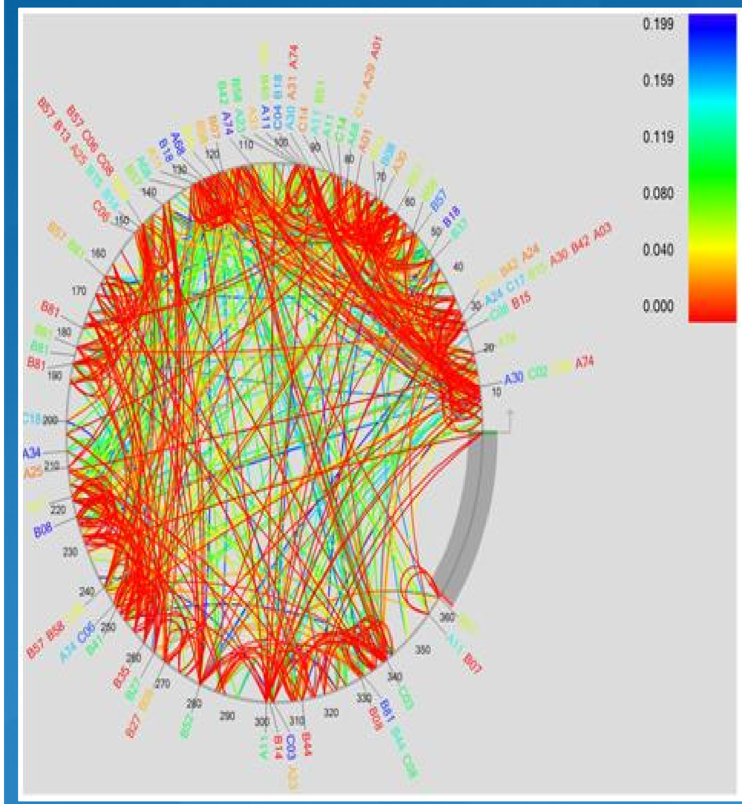
PhyloD as an Azure Service

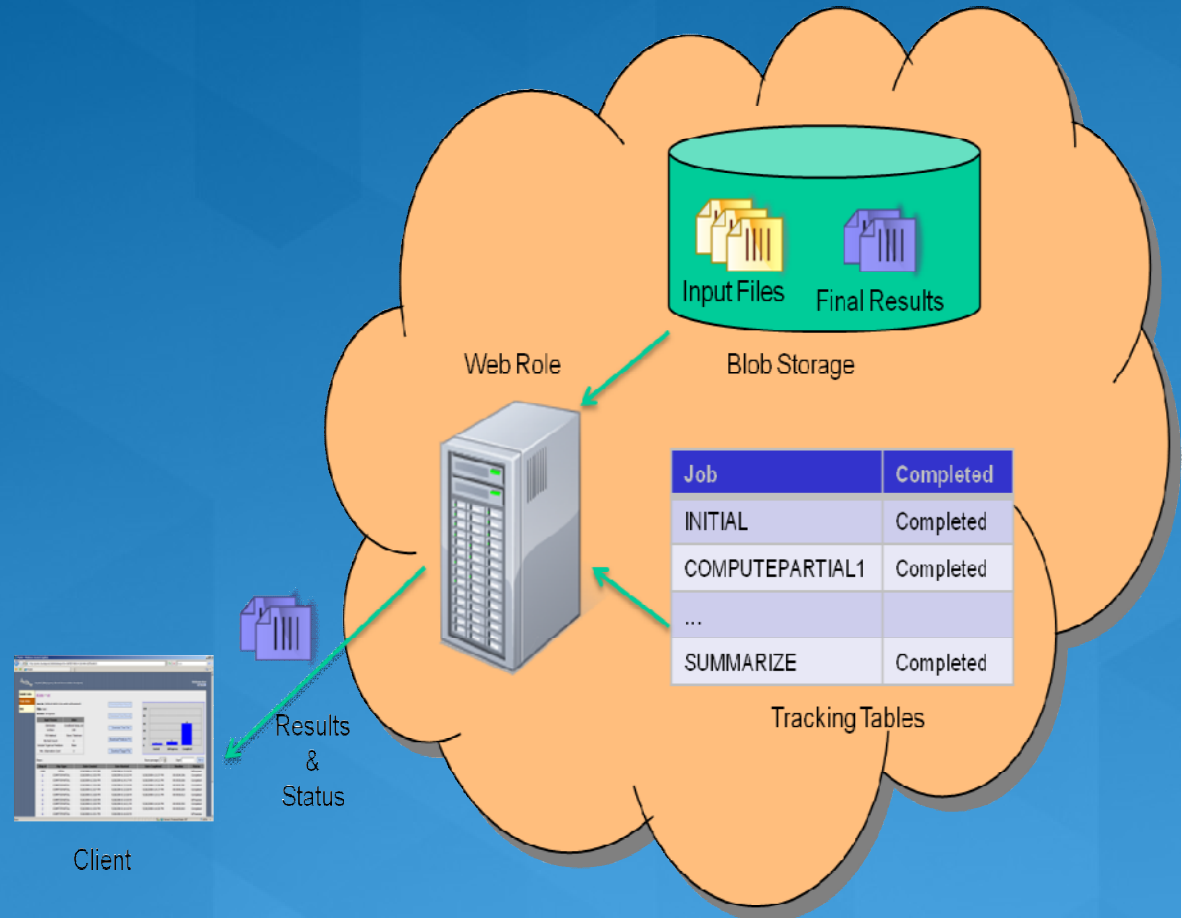
- Worker role copies input files and COMPUTEPARTIAL outputs to local storage, computes q-values for each allele-codon pair, copies results to the blob storage and updates tracking tables.



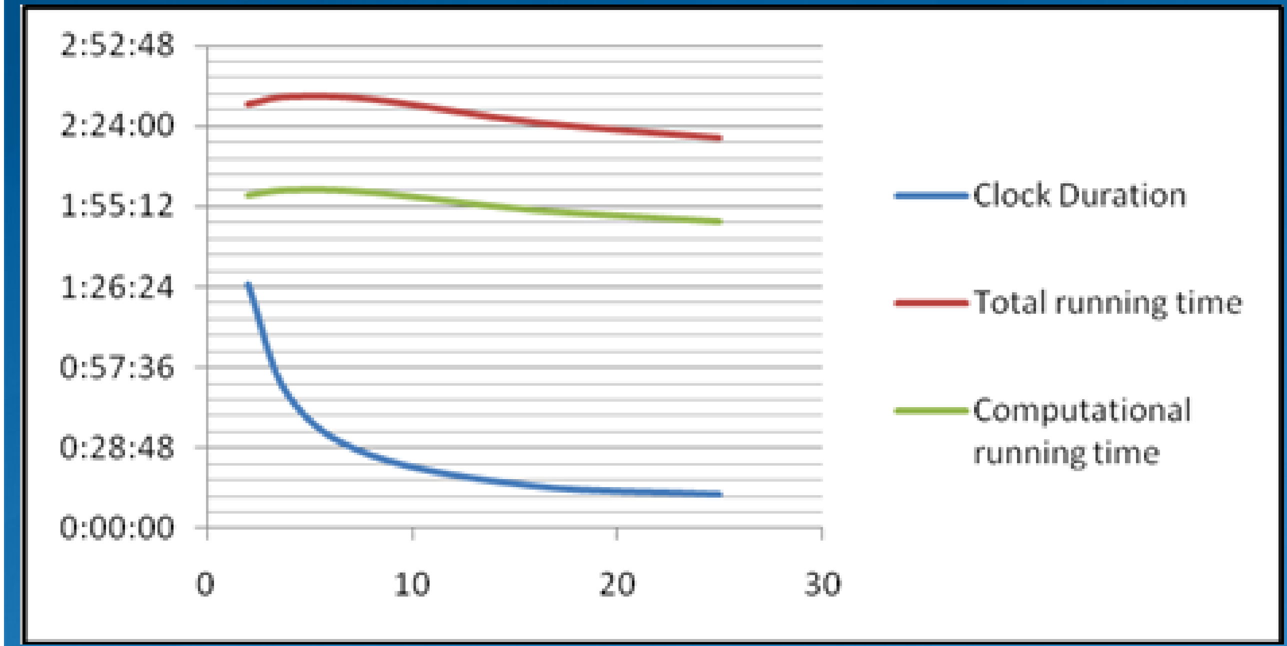
PhyloD as an Azure Service

- Web role serves the final results from blob storage and status reports from tracking tables.

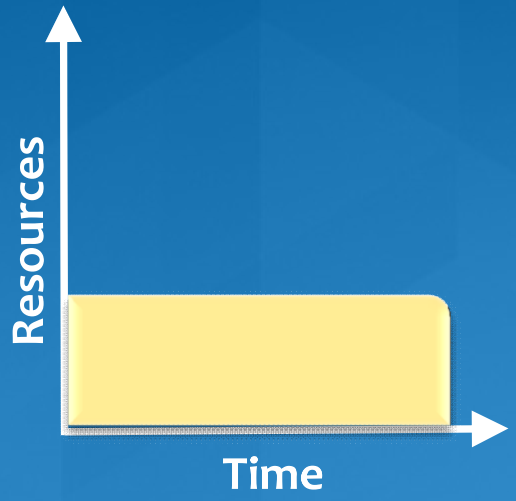




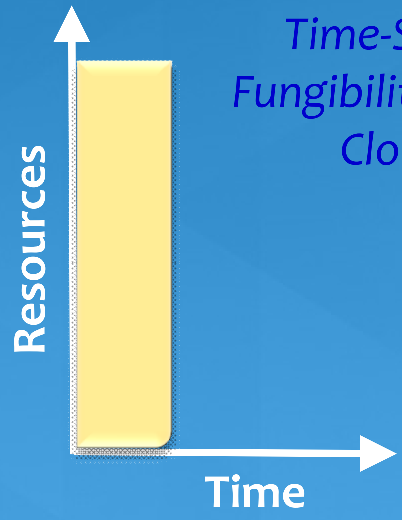
PhyloD as an Azure Service



Workers	Clock Duration	Total running time	Computational running time
25	0:12:00	2:19:39	1:49:43
16	0:15:00	2:25:12	1:53:47
8	0:26:00	2:33:23	2:00:14
4	0:47:00	2:34:17	2:01:06
2	1:27:00	2:31:39	1:59:13



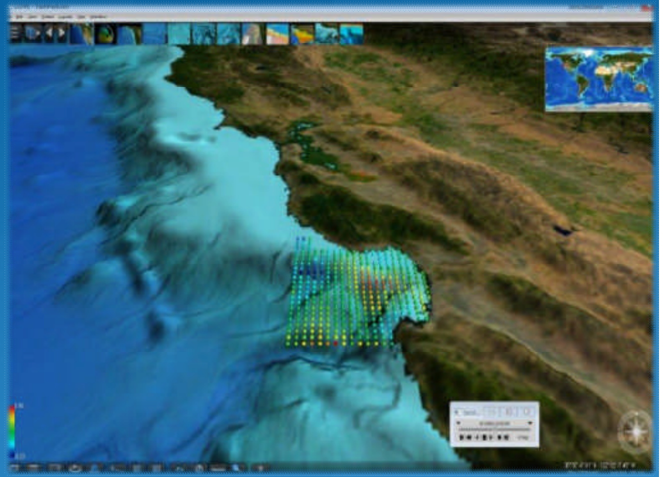
Time-Space Fungibility In the Cloud



Reference Data on Azure

Ocean Science data on Azure SDS

- Two terabytes of coastal and model data

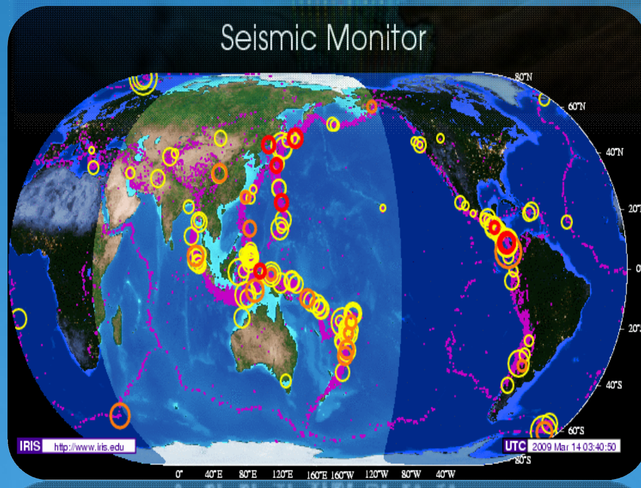


CompFi data on Azure SDS

- BATS, daily tick data for stocks (10 years)
- XBRL call report for banks (10,000 banks)

Storing select seismic data on Azure, NSF consortium that collects and distributes global seismological data.

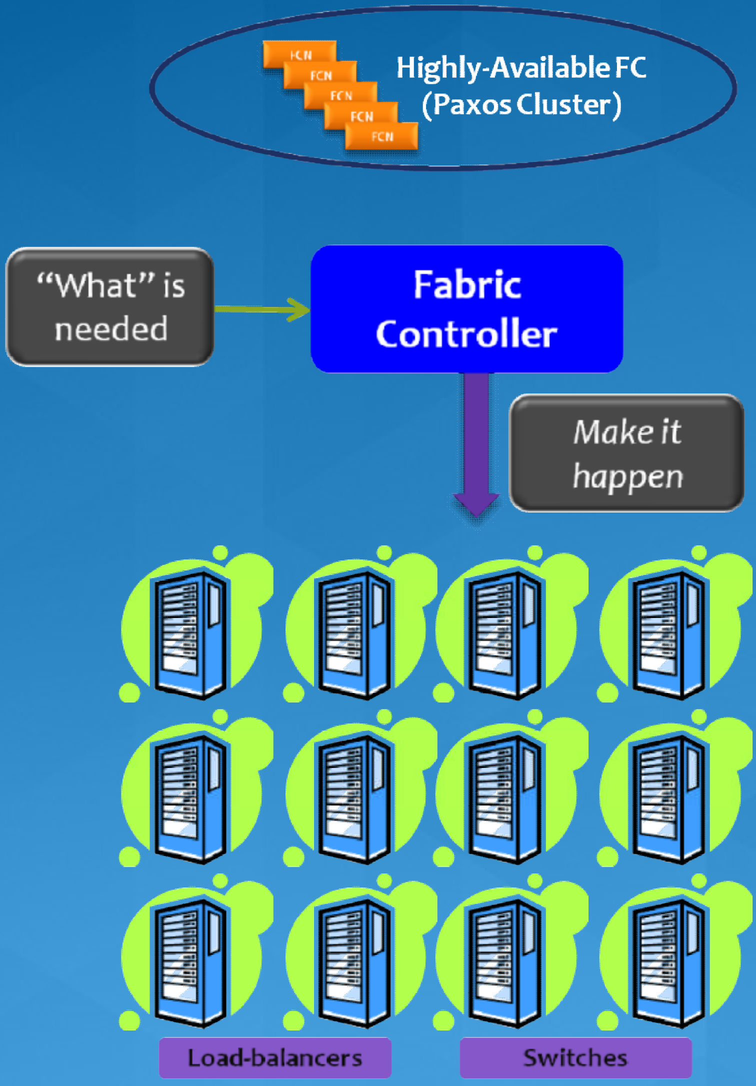
- Data sets requested worldwide
- Includes images, seismograms, events…



Windows Azure Compute Fabric

Fabric Controller

- Owns all data center hardware
- Uses inventory to host services
- Deploys applications to free resources
- Maintains the health of those applications
- Maintains health of hardware
- Manages the service life cycle starting from bare metal

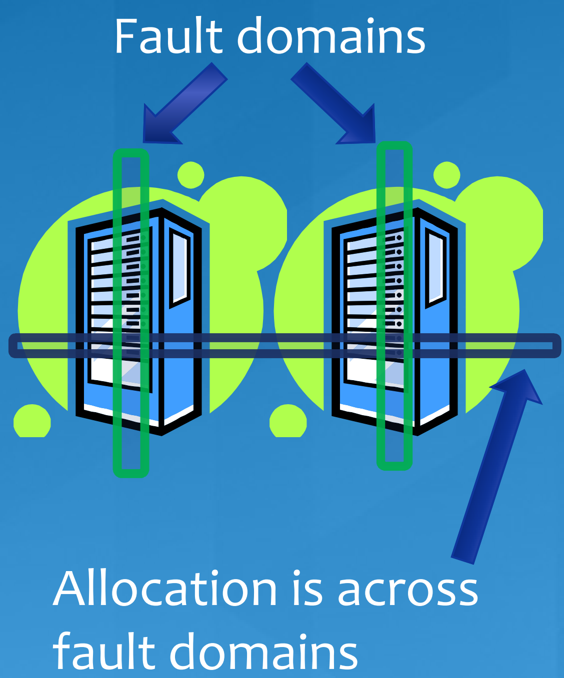


Windows Azure Compute Fabric

Fault Domains

Purpose: Avoid single points of failures

- Unit of a failure
  - Examples: Compute node, a rack of machines
- System considers fault domains when allocating service roles
- Service owner assigns number required by each role
  - Example: 10 front-ends, across 2 fault domains

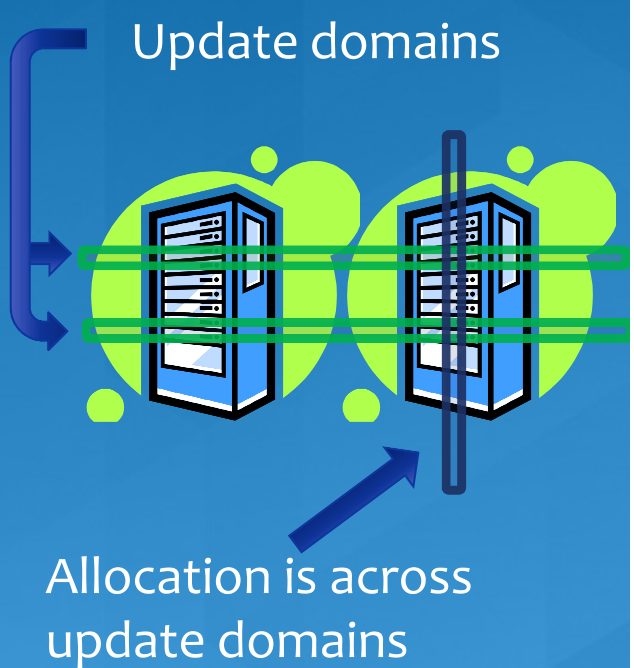


Windows Azure Compute Fabric

Update Domains

Purpose: ensure the service stays up while undergoing an update

- Unit of software/configuration update
  - Example: set of nodes to update
- Used when rolling forward or backward
- Developer assigns number required by each role
  - Example: 10 front-ends, across 5 update domains



Windows Azure Compute Fabric

The FC Keeps Your Service Running

Windows Azure FC monitors the health of roles

- FC detects if a role dies
- A role can indicate it is unhealthy
  - *Current state of the node is updated appropriately*
  - *State machine kicks in again to drive us back into goals state*

Windows Azure FC monitors the health of host

- If the node goes offline, FC will try to recover it

If a failed node can't be recovered, FC migrates role instances to a new node

- A suitable replacement location is found
- Existing role instances are notified of config change

Windows Azure Compute Fabric

Behind the Scenes Work

Windows Azure provisions and monitors hardware

- Compute nodes, TOR/L2 switches, LBs, access routers, and node OOB control elements

Hardware life cycle management

- Burn-in tests, diagnostics, and repair
- Failed hardware taken out of pool
  - Application of automatic diagnostics
  - Physical replacement of failed hardware

Capacity planning

- On-going node and network utilization measurements
- Proven process for bringing new hardware capacity online

The Cloud Empowers the Long Tail of Research

Research Funding

1. Have good idea
2. Write proposal
3. Wait 6 months
4. If successful, wait 3 months to get $$$
5. Install Computers
6. Start Work

Science Start-ups

1. Have good idea
2. Write Business Plan
3. Ask VCs to fund
4. If successful...
5. Install Computers
6. Start Work

Cloud Computing Model

1. Have good idea
2. Grab nodes from Cloud provider
3. Start Work
4. Pay for what you actually used

Poised to reach a broad class of new users

Slide compliments of Paul Watson, University of Newcastle (UK)

Emergence of a Fourth Research Paradigm

Thousand years ago – **Experimental Science**

- Description of natural phenomena

Last few hundred years – **Theoretical Science**

- Newton's Laws, Maxwell's Equations…

Last few decades – **Computational Science**

- Simulation of complex phenomena

Today – **Data-Intensive Science**

- Scientists overwhelmed with data sets from a variety of different sources
  - Data captured by instruments, sensor networks
  - Data generated by simulations
  - Data generated by computational models

Astronomy was one of the first disciplines to embrace data-intensive science with the Virtual Observatory (VO), enabling highly efficient access to data and analysis tools at a centralized site. The image shows the Pleiades star cluster form the Digitized Sky Survey combined with an image of the moon, synthesized within the WorldWide Telescope

Takeaways

Challenges facing research in science & technical computing

- Our ability to collect data outpaces our ability to analyze
- Develop, manage, maintain research services, OpEx >> CapEx

The Economics Are Changing towards Cloud Computing

- Big Data centers Offer Big Economies of Scale
- Cloud Computing Transfers Risks Away from Providers

The Application Model for Cloud Computing Is Evolving

- Dryad, Hadoop!, cloud computing platforms such as Azure
- Advantages to being "Close to the Metal" versus Advantages to programming against a Higher Level
- *Just because the infrastructure scales doesn't mean the app will !*

Many Obstacles to Ubiquitous Cloud Computing

- The Economic Forces Will Dominate the Obstacles
- There's Too Much to Gain… It Will Grow!