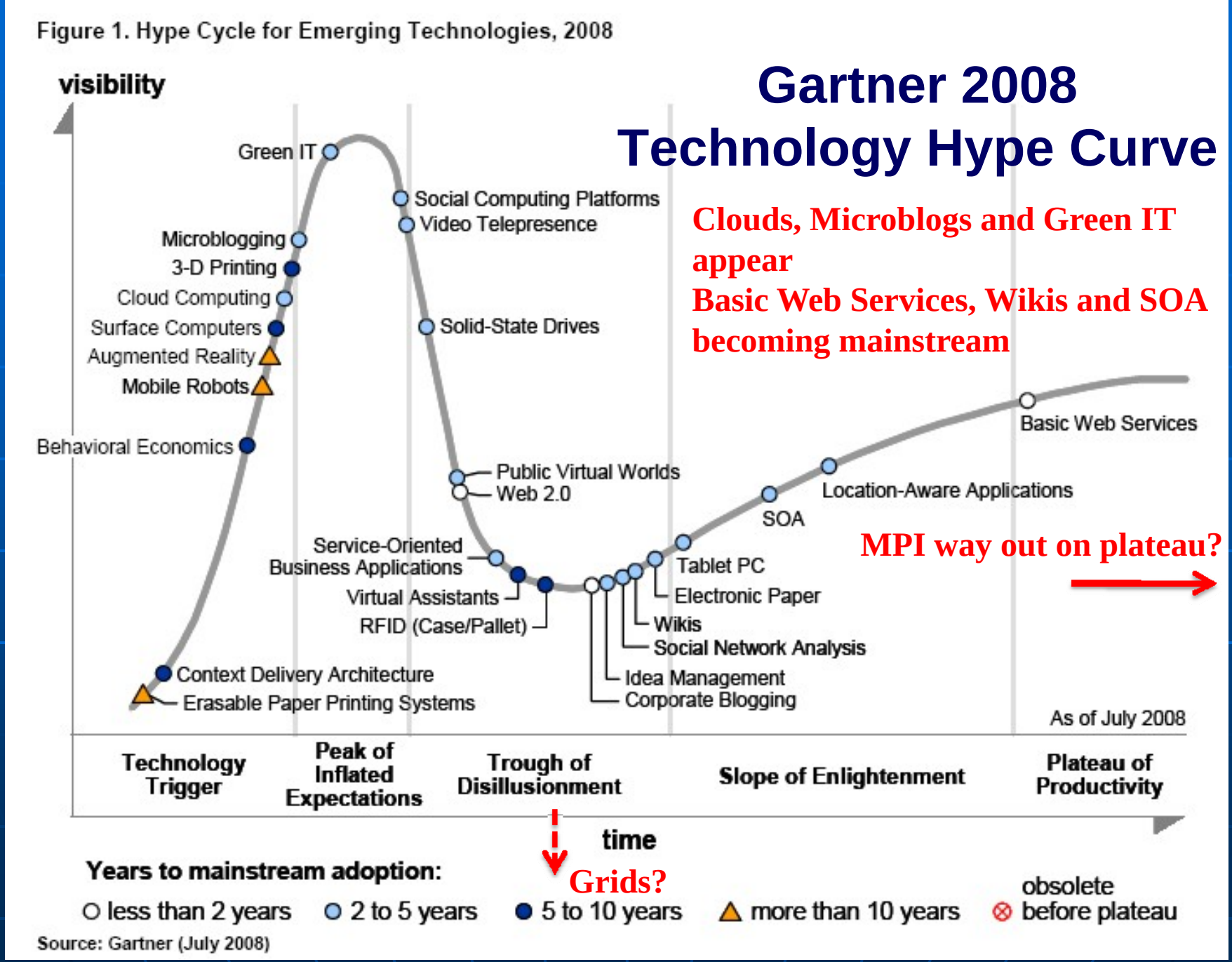
# Multicore and Cloud Futures

CCGSC
September 15 2008

Geoffrey Fox
Community Grids Laboratory, School of informatics
Indiana University

gcf@indiana.edu, http://www.infomall.org

# Gartner 2008 Technology Hype Curve



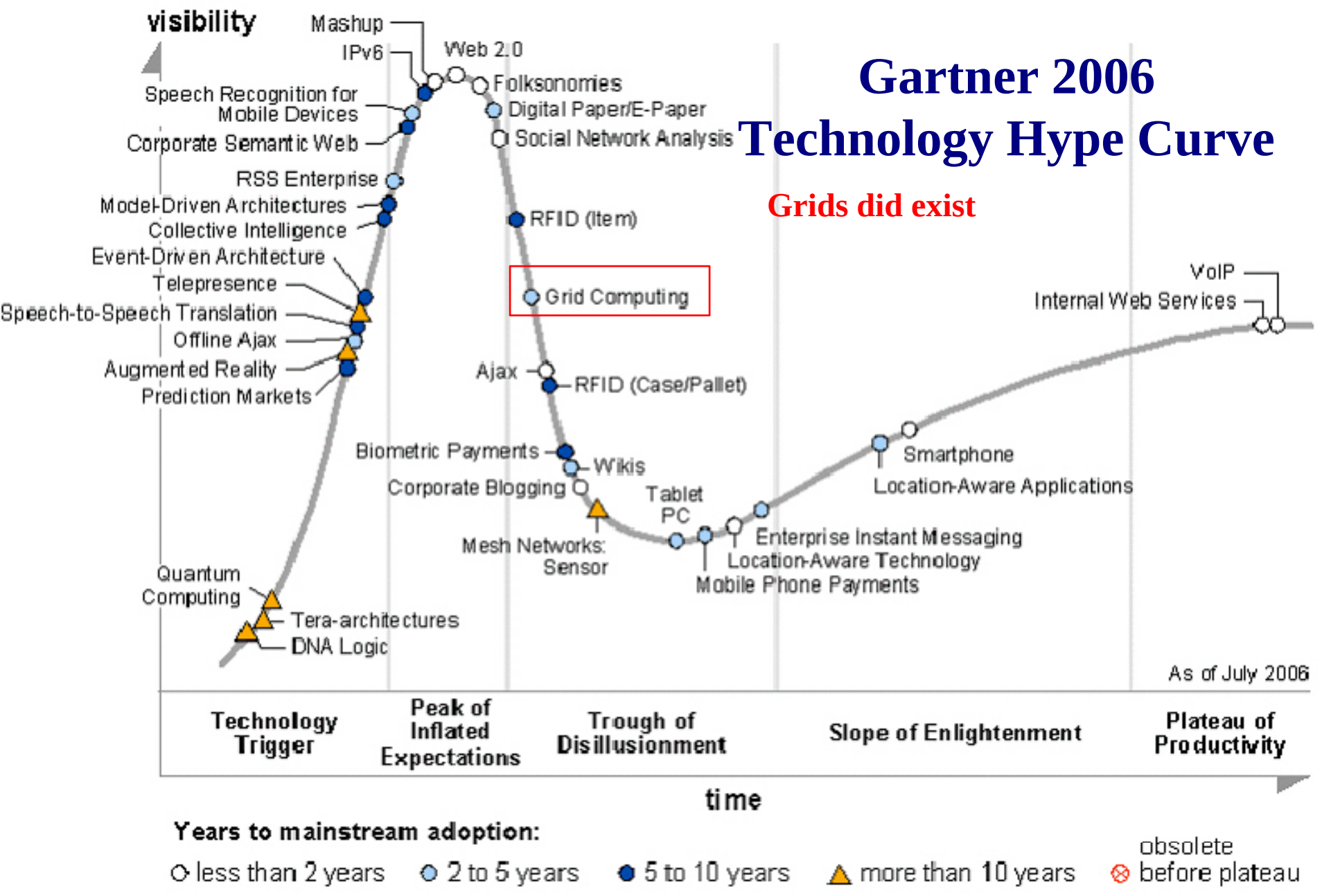

**Clouds, Microblogs and Green IT appear**

**Basic Web Services, Wikis and SOA becoming mainstream**

**MPI way out on plateau?**

# Gartner 2006 Technology Hype Curve

**Grids did exist**

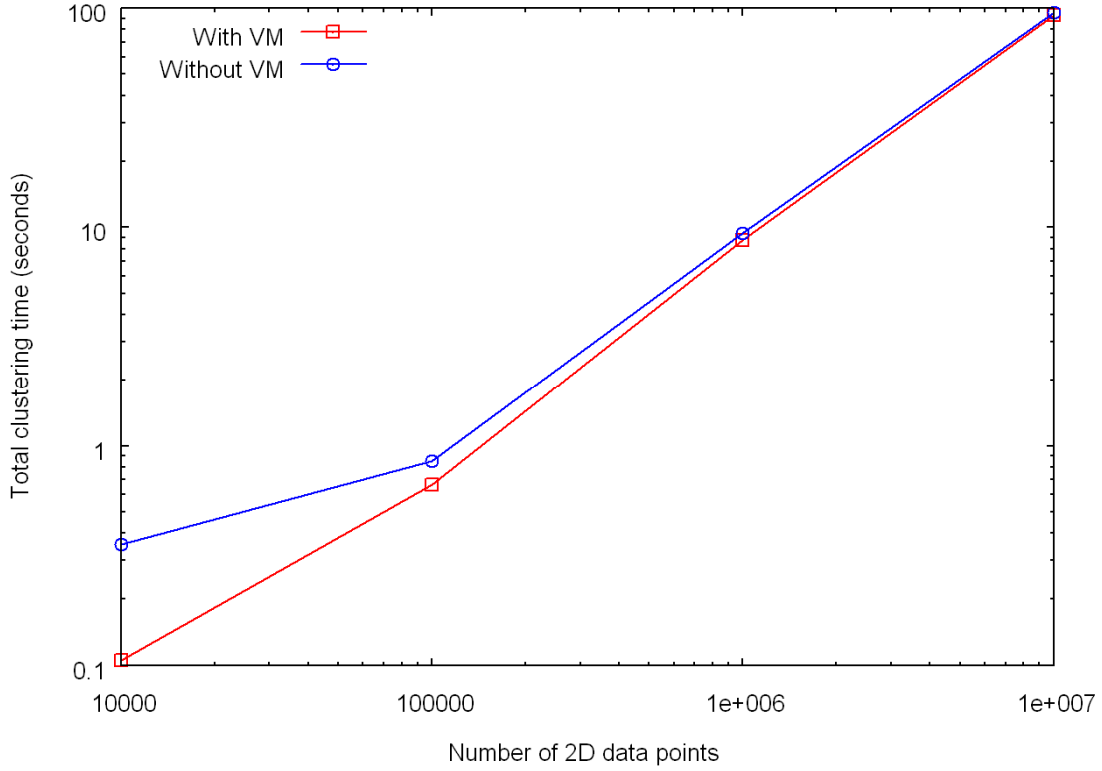


# Grids become Clouds

- Grids solve problem of too little computing: We need to harness all the world's computers to do Science
- Clouds solve the problem of too much computing: with multicore we have so much power that we need to make usage much easier
- Key technology: Virtual Machines (dynamic deployment) enable more dynamic flexible environments
  - Is Virtual Cluster or Virtual Machine correct primitive?
- Data Grids seem fine as data naturally distributed
- GGF/EGA false assumption: Web 2.0 not Enterprise defined commercial software stack
  - Some Web 2.0 applications (MapReduce) not so different from data-deluged eScience
- Citizen Science requires light weight friendly Cyberinfrastructure

# MPI on Nimbus for clustering

- Note fluctuations in runtime but performance OK for large enough problems
- 8 Nodes



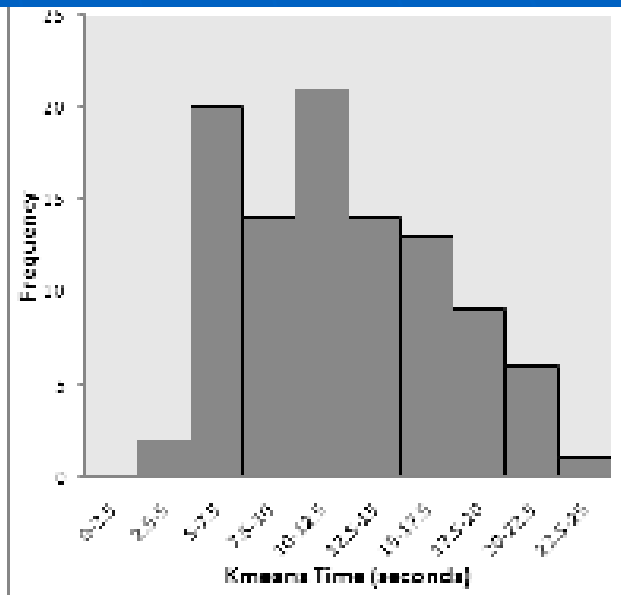




Figure 3. Histogram of Kmeans clustering time for 100 iterations (X=100) of the figure 2

Table 3. MPI Time for 100 iterations (X = 100) of the figure 2. Calculated using the following formula:

$$MPI\ Time = \frac{T(100) - T(1)}{99}$$

| Graph | MPI Time |
|---|---|
| VM MIN | 0.040 |
| VM Average | 0.234 |
| VM MAX | 0.112 |
| Direct Average | 0.014 |

# Plans for QuakeSpace

- QuakeSim supports Earthquake Scientists who want some features of their kid's (under 40) world
- Rebuild QuakeSim using Web 2.0 and Cloud Technology
- Applications, Sensors, Data Repositories as Services
- Computing via Clouds
- Portals as Gadgets
- Metadata by tagging
- Data sharing as in YouTube
- Alerts by RSS
- Virtual Organizations via Social Networking
- Workflow by Mashups
- Performance by multicore
- Interfaces via iPhone, Android etc.

Upload New Workflow | View All Workflows

## Top 50 tags for Workflows [See All Tags]

abstracts | affymetrix | AIDA | BioAID | **bioinformatics** | **biorange_nl** | BLAST | cel | clone | count | data-driven | ddbj | **disease** | entrez | file | gene | genotype | iteration | kegg | list | mesh | microarray | mining | mouse | pathway | pathway-driven | pathways | phenotype | protein | pubmed | qtl | rank | scaffold | SEG | sequence | shim | similarity | simplifier | synonyms | terms | text mining | text_mining | text_mining_network | ugi | uniprot | unitary matrix | up-and-down | utility | VL-e | weighting

Most Recent | Last Updated | Most Viewed | Most Downloaded

**Uploader:**

Marco Roos

### BioAID_DiseaseDiscovery_byHumanUniprot

View

Download (v1)

**Credits:** Marco Roos, Martijn Schuemie, AID

**License:** Creative Commons Attribution-Share Alike 3.0 License

This workflow finds disease relevant to the query string via the following steps: 1. A user query: a list of terms or boolean query - look at the Apache Lucene project for all details. E.g.: (EZH2 OR "Enhancer of Zeste" +(mutation chromatin) -clinical); consider adding 'ProteinSynonymsToQuery' in front of the input if your query is a protein. 2. Retrieve documents: finds 'maximumNumbe...

**Rating:** 0.00/5 (0 ratings) | **Versions:** 1 | **Viewed:** 21 times | **Downloaded:** 10 times

**Tags:**

AIDA | BioAID | biorange_nl | disease | protein | text_mining | text_mining_network | VL-e

**Uploader:**

### DataBiNS - Data Mining Workflow for Biological Pathways and Non-Synonymous SNPs

View

Download (v2)

## New/Upload

Workflow | GO

David De Roure

My Profile
My Inbox
**My Memberships (2)**
My History
My News

**1 new friendship request**

**Hiyashi Yoi**

**1 new group invite**

**From Hiyashi Yoi**

(for Group: Social Scientific Land)

**1 new group request**

**From Freekymayne**

(for Group: Music workflows)

## My Stuff

**18 friends** | **4 groups** | **1 file** | **1 workflow**

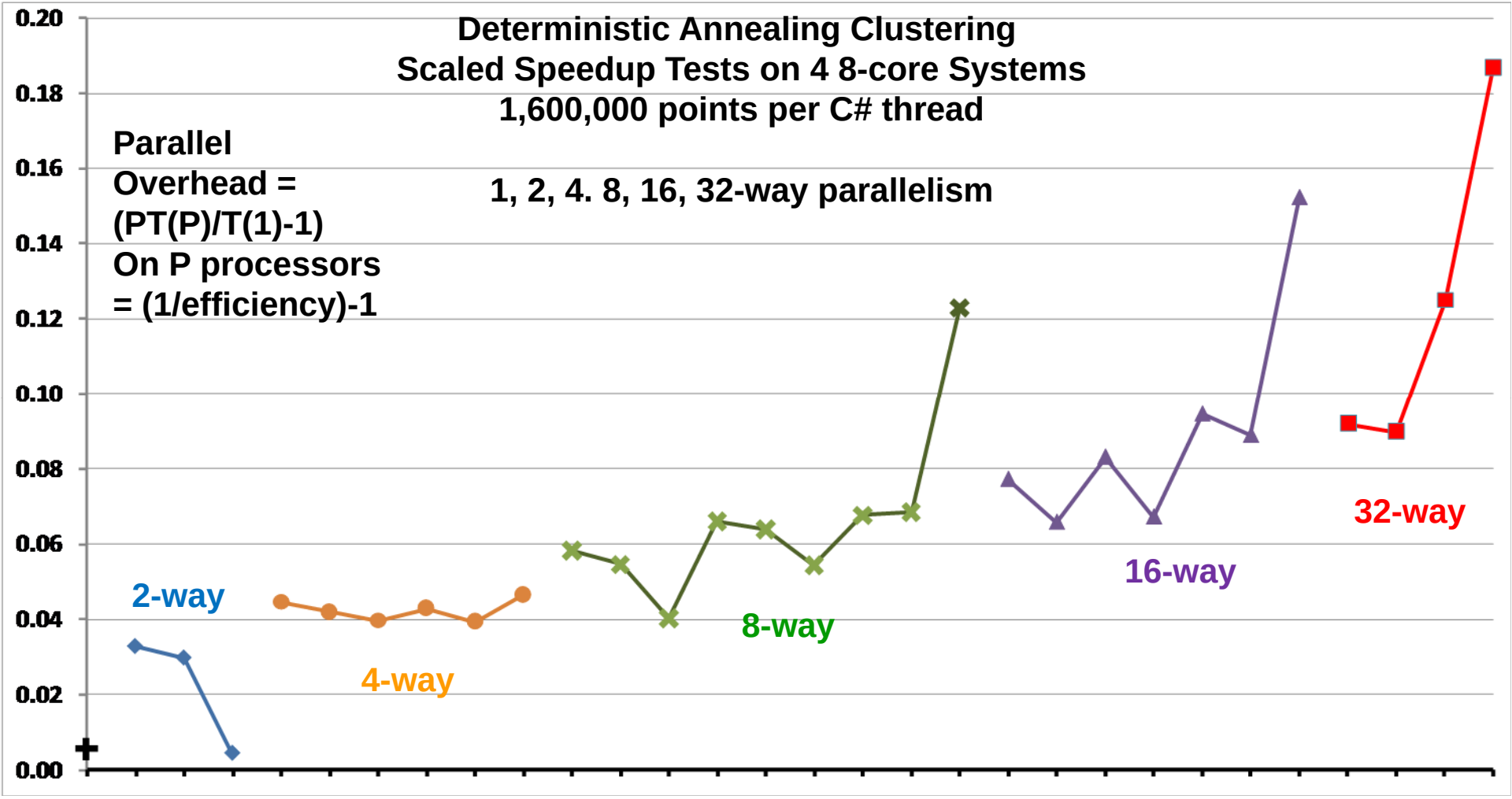
| Enterprise Approach | Web 2.0 Approach |
| --- | --- |
| JSR 168 Portlets | Gadgets, Widgets |
| Server-side integration and processing | AJAX, client-side integration and processing, JavaScript |
| SOAP | RSS, Atom, JSON |
| WSDL | REST (GET, PUT, DELETE, POST) |
| Portlet Containers | Open Social Containers (Orkut, LinkedIn, Shindig); Facebook; StartPages |
| User Centric Gateways | Social Networking Portals |
| Workflow managers (Taverna, Kepler, etc) | Mash-ups |
| Grid computing: Globus, Condor, etc | Cloud computing: Amazon WS Suite, Xen Virtualization, still Condor! |

# Different Programming Models

- (Web) services, "farm" computations, Workflow (including AVS, HeNCE from past), Mashups, MPI, MapReduce run **functionally or data decomposed execution units** with a wide variety of front ends
- **Front-end:** Language+communication library, Scripting, Visual, Functional, XML, PGAS, HPCS Parallel Languages, Templates, OpenMP
- Synchronize/Communicate with some variant of messaging (zero size for locks) with performance, flexibility, fault-tolerance, dynamism trade-offs
- **Synchronization:** Locks Threads Processes CCR CCI SOAP REST MPI Hadoop; not much difference for user?

# MPI becomes Ghetto MPI

- Multicore best practice not messaging will drive synchronization/communication primitives
- Party Line Programming Model: Workflow (parallel--distributed) controlling optimized library calls
  - Core parallel implementations no easier than before; deployment is easier
- MPI is wonderful; it will be ignored in real world unless simplified
- CCI notes MPI is HPCC Ghetto
- CCI is high performance distributed message passing ghetto?
- CCR from Microsoft – only ~7 primitives – is one possible commodity multicore driver
  - It is roughly active messages
  - Will run MPI style codes fine on multicore




Nodes

1 2 1 1 4 2 1 2 1 1 4 2 1 4 2 1 2 1 1 4 2 4 2 4 2 2 4 4 4 4

MPI Processes per Node

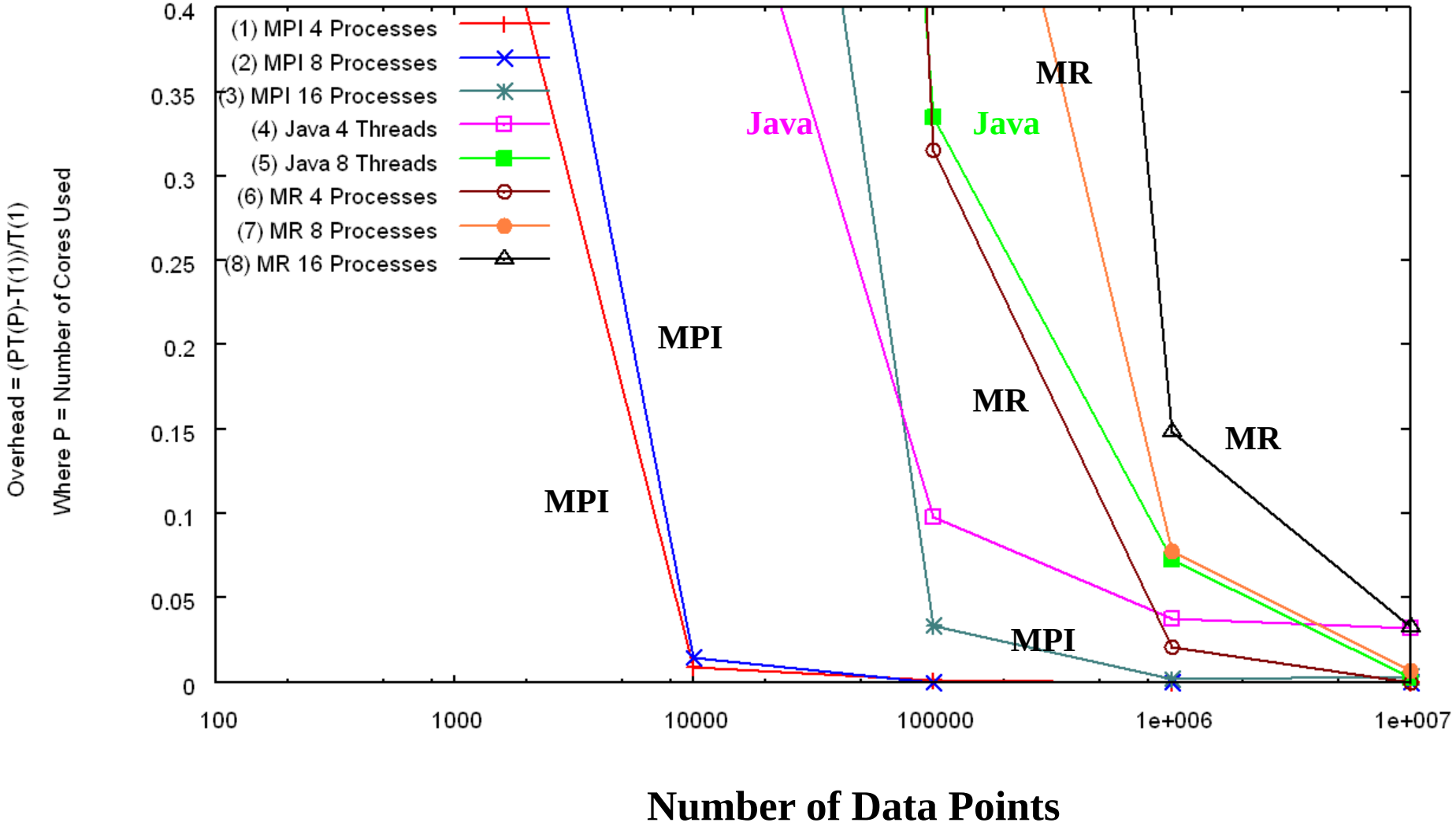
1 1 2 1 1 2 4 1 2 1 2 4 8 1 2 4 1 2 1 4 8 2 4 1 2 1 8 4 2 1

CCR Threads per Process

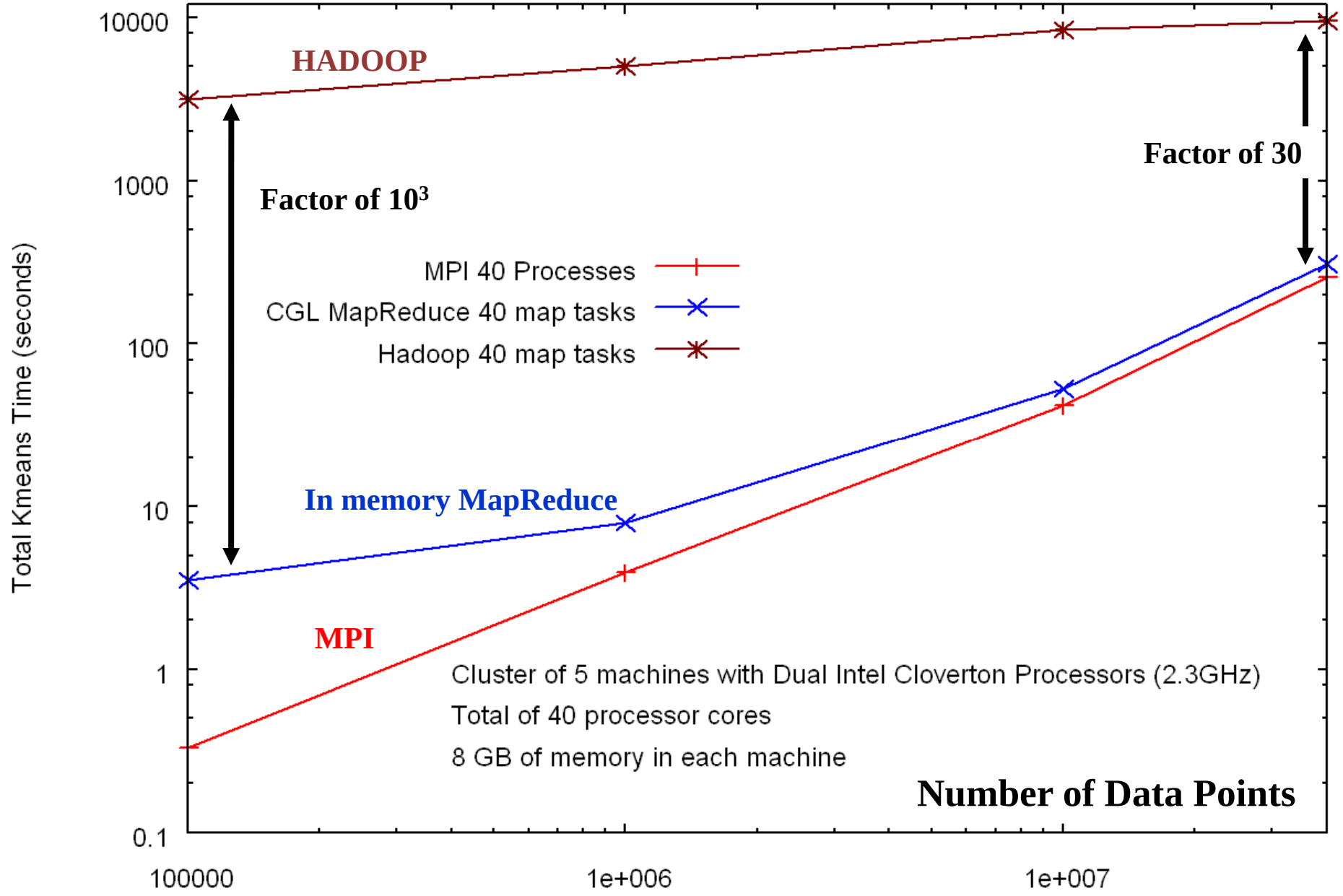
1 1 1 2 1 1 1 2 2 4 1 1 1 2 2 2 4 4 8 1 1 2 2 4 4 8 1 2 4 8

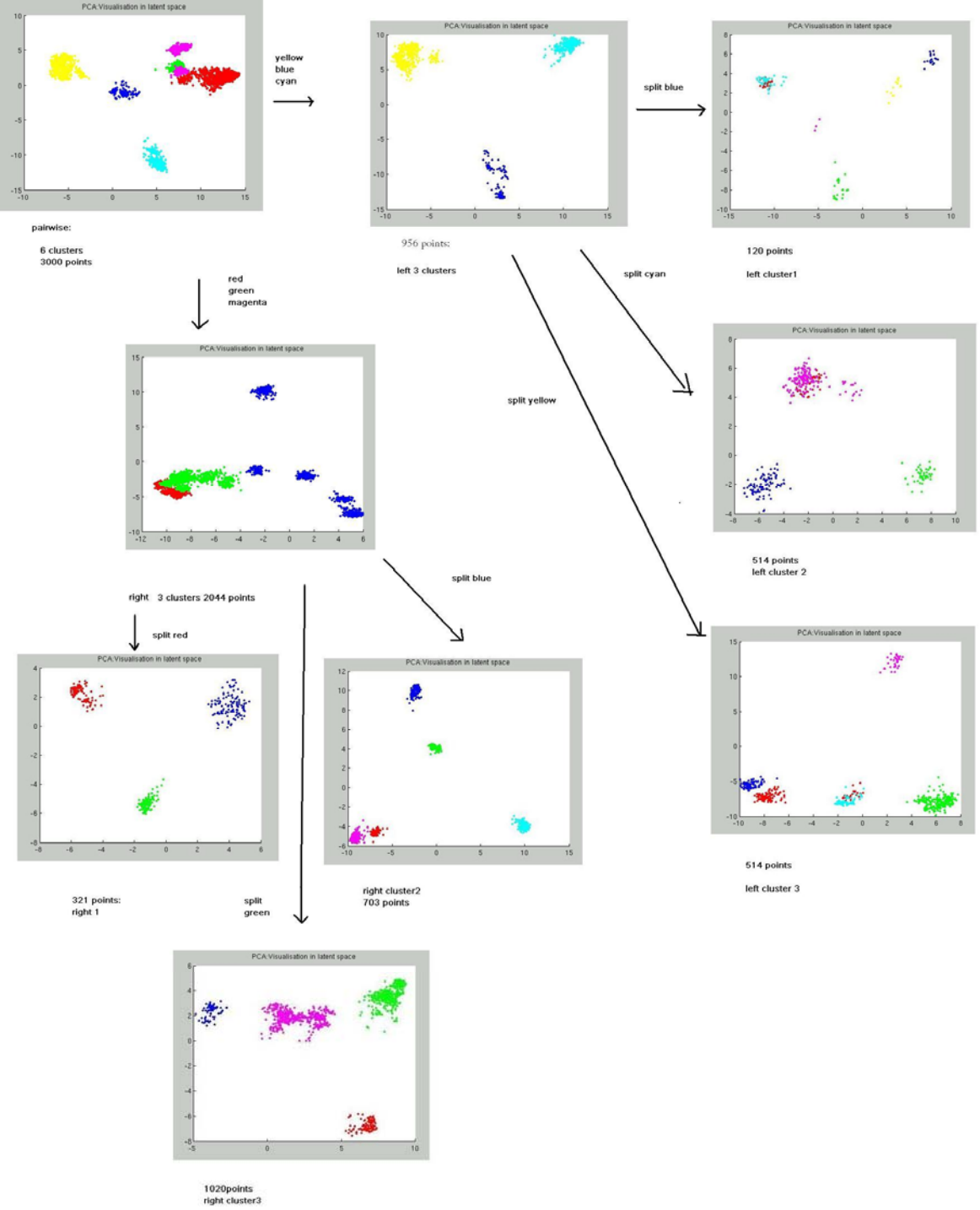
# MPI, MapReduce, Java Threads for Clustering

- Overhead PT(P)/T(1)-1 of the messaging runtime for the different data sizes
- All perform well for large enough datasets



# Hadoop v MPI and faster MapReduce for Clustering






N=3000 sequences each length ~1000 features

Only use pairwise distances

will repeat with 0.1 to `0.5 million sequences with a larger machine

C# with CCR and MPI