Big-Data: Theory, Analytics and Engineering Perspectives

Dr. Vijay Srinivas Agneeswaran

Director, Technology

Head, Big-Data R&D

Impetus InfoTech (India) Pvt. Ltd.

Big-data: Ming Boggling Numbers

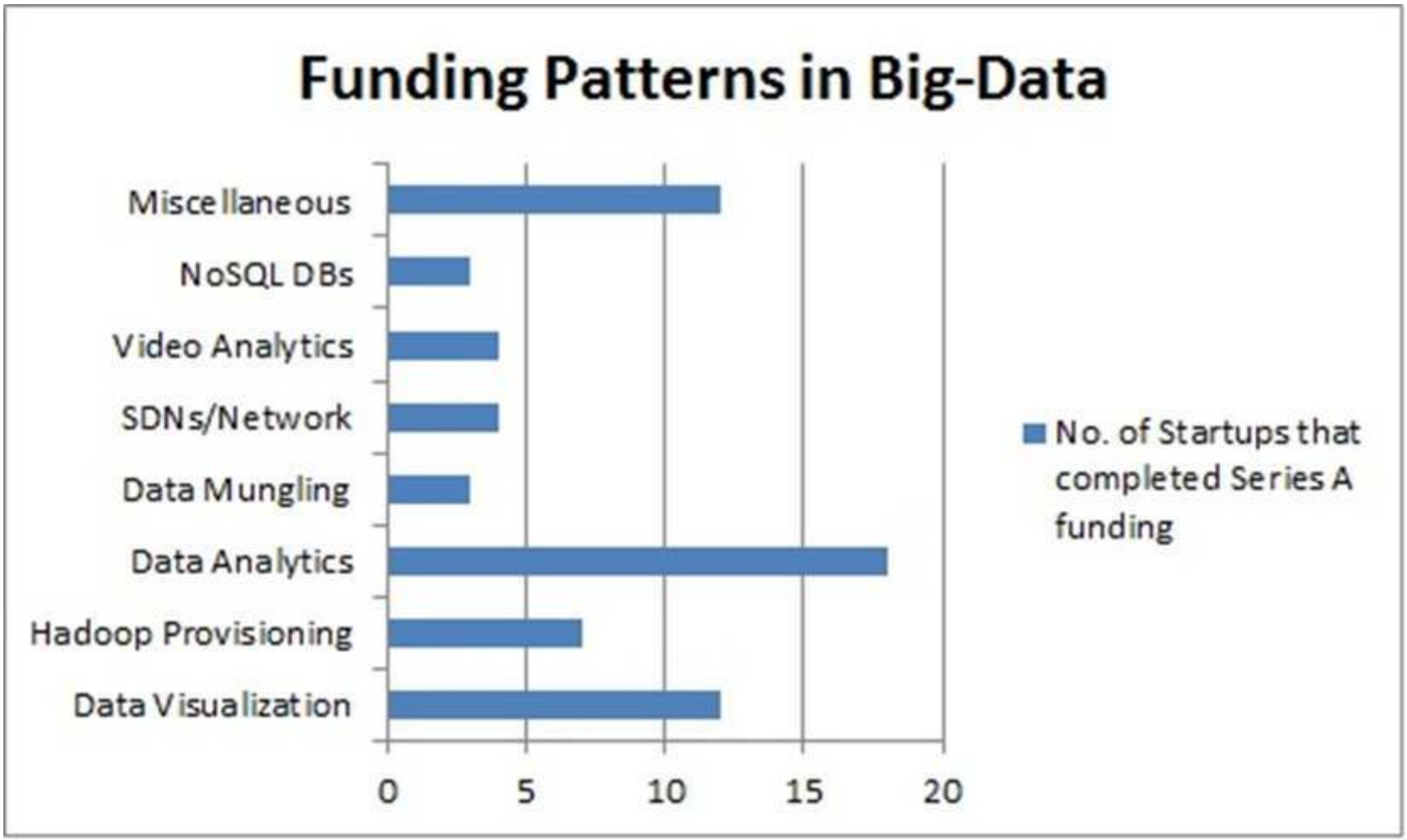
- Digital universe – 1.8 Zettabytes (1 billion terabytes or 10^{12} GB) of data in 2011 (EMC Report)
  - expected to be 2.7 ZB in 2012 and 8 ZB in 2015.
  - 10^{15} files
  - 75% of information generated by individual users.
- 5 billion mobile phones in 2011, 30 billion content pieces on Facebook every month (Mckinsey report).
- US Library of Congress has collected 235 TB of data (Infographic)
  - Data per company in 15/17 sectors in US is > than 235 TB.
- Important areas (Mckinsey report)
  - Healthcare – personalized medicine, clinical trial design, fraud detection etc.
  - Governments (Aadhar project) – increased tax collection, transparency.
  - Retail – consumer behaviour prediction, sentiment analysis, merchandizing
  - Manufacturing – digital factory, R&D design, supply chain management etc.
  - Telecom – Personal location data (GPS and other technologies) – smart routing (navigation), automotive telematics, mobile Location Based Services (LBS).

Top Big-data analyzers/processors

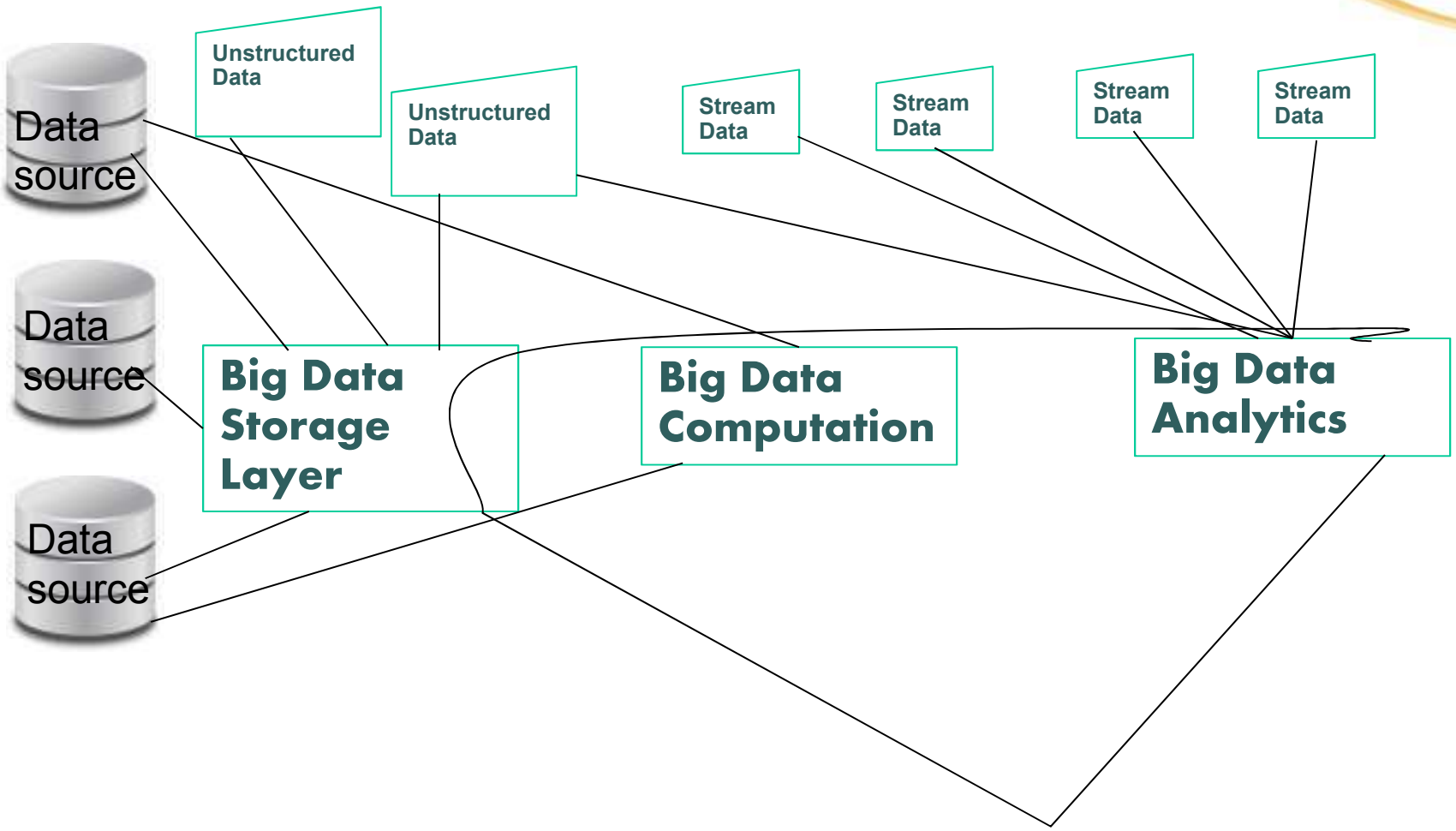
- LinkedIn – petabytes of social data represented as graphs
  - People You May Know feature
- Facebook – analyses petabytes of user generated data
- NY Times – processed 4 TB of raw images in less than a day.
- Amazon – retailer
  - Recommendation system – consumer behaviour analysis
    - 30% of books/products sold
- Akamai – analyzes 75 million events per day
  - Targeted advertising
- Twitter – 340 million tweets per day or about 4000 tweets per second on average.
  - Peak 15000 tweets/second for Spain's fourth goal in Euro 2012.
- Google – processes around 20000 terabytes (20 petabytes) per day.
- Flickr – 6 billion images (Flickr blog)

Big-data Funding Pattern

Big-data = volume + velocity + variety + (value)



Broad Focus



How to maximize efficiency, scalability of performing operations on Big-data – including storage, search, computation and analytics.

Hadoop Adoption Status: Sep 2012

- Enterprise level – not yet mainstream
  - Experimental – lot of big companies have their own Hadoop clusters including Sears, Walmart, Disney, AT&T etc.
  - Departmental production – not quite enterprise production yet?
- Business use case
  - Extract, Transform, Load ETL/ELT/data refinement
    - Pentaho, Datameer SMEs in this space.
    - Big-players – Informatica, Splunk (log analytics company) and IBM
- Industry-wise adoption
  - Financial investment/trading – quite high, just as for any new tech.
  - Banking Financial – slower.
  - Telecom, Retail – cautious.

Future of Hadoop Adoption

- Enterprises
  - ETL for production
  - Hindrance – single cluster – Hadoop YARN is the way forward.
- Analytics
  - May not replace data warehouses
  - Real-time analytics is certainly the way forward.
  - Hadoop can be alternative to scale-out analytical RDBMSs (Vertica/VoltDB/SAP-HANA)
- Appliance market for Hadoop?
- Map-Reduce for iterative computations
  - Hadoop not currently well suited
  - Alternatives include Twister, Spark, HaLoop.
- Beyond Map-Reduce
  - Pregel from Google, built on top of Bulk Synchronous Parallel (BSP).

Suitability of Map-Reduce

- Origin in functional programming languages (Lisp and ML)
- Built for embarrassingly parallel computations
  - The map function outputs key value pairs
    - Map: (k1, v1) -> list(k2, v2)
  - Reducer functions perform aggregate operations over the key
    - Reduce: list(k2, list(v2)) - > list(v2).
- Suitable for matrix multiplications, n-body problem and sorting problems – linear regression, batch gradient descent will work well – Mahout has these implementations.
  - Algorithms which can be expressed in Statistical Query Model in summation form – highly suitable for MR [1].
  - Linear regression, linear SVM, Naïve bayes etc. fall in this category.
  - Mahout has implemented only sequential version of logistic regression.
    - Very hard to do in MR – inherently iterative
    - Training is very fast and in parallel, but basic algorithm is sequential.

[1] Chu, C.-T., Kim, S. K., Lin, Y.-A., Yu, Y., Bradski, G. R., Ng, A. Y., and Olukotun, K. Map-reduce for machine learning on multicore. In *NIPS* (2006), pp. 281--288.

What about Iterative Algorithms?

- What are iterative algorithms?
  - Those that need communication among the computing entities
  - Examples – neural networks, PageRank algorithms, network traffic analysis
- Conjugate gradient descent
    - Commonly used to solve systems of linear equations
    - [1] tried implementing CG on dense matrices
    - DAXPY – Multiplies vector x by constant a and adds y.
    - DDOT – Dot product of 2 vectors
    - MatVec – Multiply matrix by vector, produce a vector.
  - 1 MR per primitive – 6 MRs per CG iteration, hundreds of MRs per CG computation, leading to 10 of GBs of communication even for small matrices.
  - Communication cost just overwhelms computation time – it takes unreasonable time to run CG on MR.
- Other iterative algorithms – fast fourier transform, block tridiagonal

[1] C. Bunch, B. Drawert, M. Norman, Mapscale: a cloud environment for scientific computing, Technical Report, University of California, Computer Science Department, 2009.

Further exploration: Iterative Algorithms

- [2] explores CG kind of iterative algorithms on MR
- Compare Hadoop MR with Twister MR (http://iterativemapreduce.org)
  - It took 220 seconds on a 16 node cluster to solve system with 24 unknowns, while for 8000 unknowns – took almost 2 hours.
  - MR tasks for each iteration – computation is too little, overhead of setup of MR tasks and communication is too high.
    - Data is reloaded from HDFS for each MR iteration.
    - Surprising that Hadoop does not have support for long running MR tasks
- Other alternative MR frameworks?
  - HaLoop [3] – extends MR with loop aware task scheduling and loop invariant caching.
  - Spark [4] – introduces resilient distributed datasets (RDD) – RDD can be cached in memory and reused across iterations.
- Beyond MR – Apache Hama (http://hama.apache.org) – BSP paradigm

[2] Satish Narayana Srirama, Pelle Jakovits, and Eero Vainikko. 2012. Adapting scientific computing problems to clouds using MapReduce. *Future Generation Computer Systems* 28, 1 (January 2012), 184-192, Elsevier Publications

[3] HaLoop: Efficient Iterative Data Processing on Large Clusters by Yingyi Bu, Bill Howe, Magdalena Balazinska, Michael D. Ernst. In*VLDB'10*: The 36th International Conference on Very Large Data Bases, Singapore, 24-30 September, 2010

[4] Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, and Ion Stoica. 2010. Spark: cluster computing with working sets. In *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing* (HotCloud'10). USENIX Association, Berkeley, CA, USA, 10-10

Data processing: Alternatives to Map-Reduce

- R language
  - Good for statistical algorithms
  - Does not scale well – single threaded, single node execution.
  - Inherently good for iterative computations – shared array architecture.
- Way forward
  - R-Hadoop integration – or R-Hive integration
  - R extensions to support distributed execution.
    - [1] is an effort to provide R runtime for scalable execution on cluster.
    - Revolution Analytics is an interesting startup in this area.
- Apache HAMA (http://hama.apache.org) is another alternative
  - Based on Bulk Synchronous Parallel (BSP) model – inherently good for iterative algorithms – can do Conjugate gradient, non-linear SVMs – hard in Hadoop MR.

[1] Shivaram Venkataraman, Indrajit Roy, Alvin AuYoung, and Robert S. Schreiber. 2012. Using R for iterative and incremental processing. In *Proceedings of the 4th USENIX conference on Hot Topics in Cloud Computing* (HotCloud'12). USENIX Association, Berkeley, CA, USA, 11-11.

Paradigms for Processing Large Graphs in Parallel

- Pregel [1] – Computation engine from Google for processing graphs
  - Implementation of Bulk Synchronous Parallel (BSP) – paradigm from traditional parallel programming
  - User defined compute() for each vertex at each superstep S.
  - Edges – messages between vertices.
  - Parallelism – Vertex compute functions run in parallel
  - Compute-communicate-barrier – each iteration.
  - Similar open source alternatives – Apache Giraph, Golden orb, Stanford GPS
  - Pregel is good at graph parallel abstraction, ensures deterministic computation, easy to reason with, but
    - user must architect movement of data
    - curse of slow job (barrier synchronization can be slowed by slow jobs – sequential dependencies in the graph).
    - Cannot prioritize/target computation where it is needed most – not adaptive

[1] Grzegorz Malewicz, Matthew H. Austern, Aart J.C Bik, James C. Dehnert, Ilan Horn, Naty Leiser, and Grzegorz Czajkowski. 2010. Pregel: A System for Large-scale Graph Processing. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*(SIGMOD '10). ACM, New York, NY, USA, 135-146.

Piccolo: Another Graph Processing Abstraction

- Piccolo [1] – provides asynchronous graph processing abstraction.
  - Application programs comprise
    - control functions – executed on a single machine (master)
      - Create kernels, shared tables, perform global synchronization.
    - Kernel functions – executed on slaves in parallel.
  - Table operations include get, put, update, flush, get_iterator.
  - User defined accumulation functions for concurrent access to table entries.
  - User defined table partition.
- Does not ensure serializable program execution.
  - May be required for some ML algorithms, including dynamic Alternating Least Squares (ALS) and Gibbs sampling.

[1] Russell Power and Jinyang Li. 2010. Piccolo: Building Fast, Distributed Programs with Partitioned Tables. In *Proceedings of the 9th USENIX conference on Operating systems design and implementation* (OSDI'10). USENIX Association, Berkeley, CA, USA, 1-14.

GraphLab: Ideal Engine for Processing Natural Graphs [1]

- Goals – targeted at machine learning.
  - Model graph dependencies, be asynchronous, iterative, dynamic.
- Data associated with edges (weights, for instance) and vertices (user profile data, current interests etc.).
- Update functions – lives on each vertex
  - Transforms data in scope of vertex.
  - Can choose to trigger neighbours (for example only if Rank changes drastically)
  - Run asynchronously till convergence – no global barrier.
- Consistency is important in ML algorithms (some do not even converge when there are inconsistent updates – collaborative filtering).
  - GraphLab – provides varying level of consistency. Parallelism VS consistency.
  - Implemented several algorithms, including ALS, K-means, SVM, Belief propagation, matrix factorization, Gibbs sampling, SVD, CoEM etc.
  - Co-EM (Expectation Maximization) algorithm 15x faster than Hadoop MR – on distributed GraphLab, only 0.3% of Hadoop execution time.

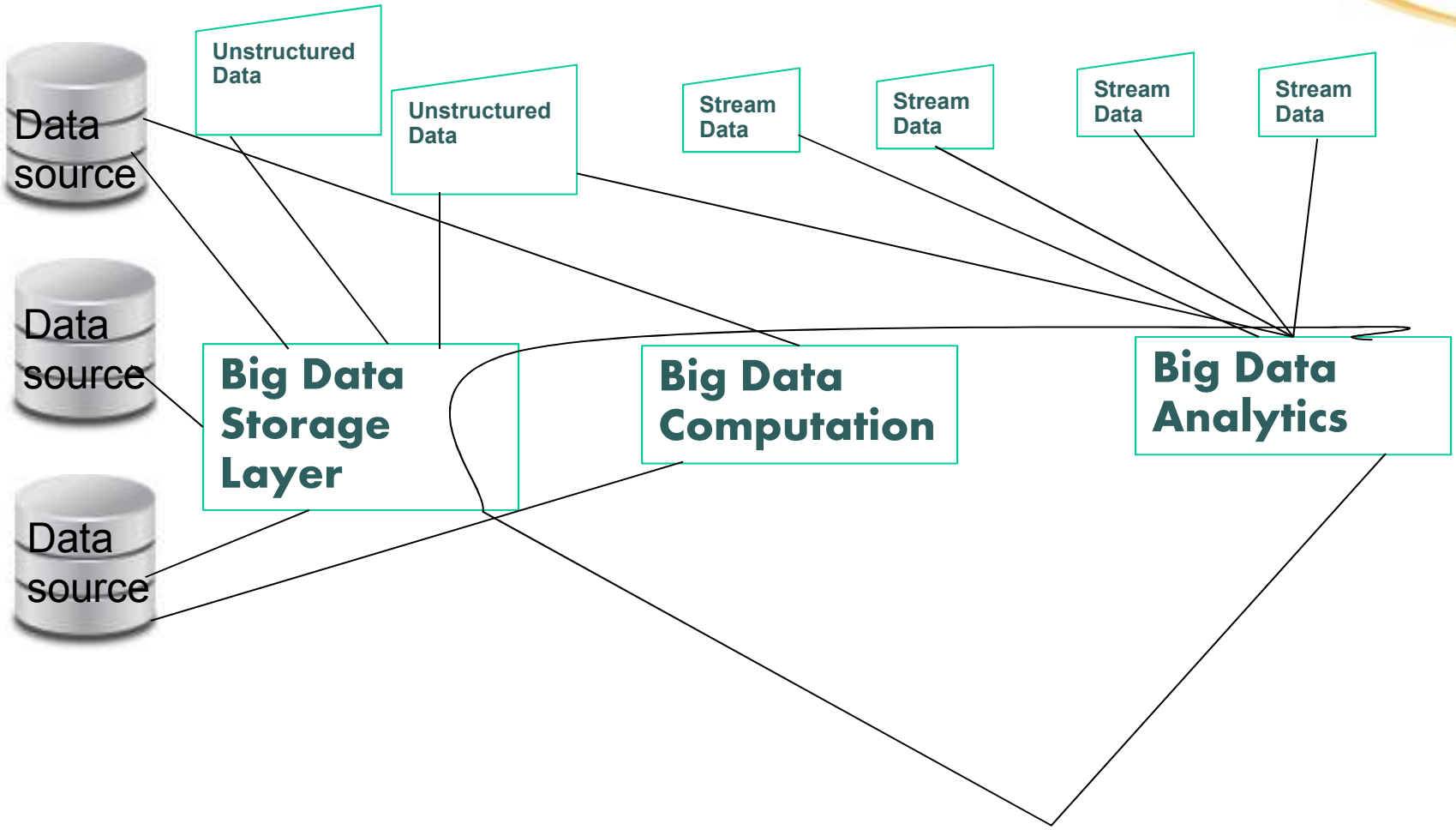
[1] Yucheng Low, Danny Bickson, Joseph Gonzalez, Carlos Guestrin, Aapo Kyrola, and Joseph M. Hellerstein. 2012. Distributed GraphLab: a framework for machine learning and data mining in the cloud. *Proceedings of the VLDB Endowment* 5, 8 (April 2012), 716-727.

GraphLab 2: PowerGraph – Modeling Natural Graphs [1]

- GraphLab could not scale to Altavista web graph 2002, 1.4B vertices, 6.7B edges.
  - Most graph parallel abstractions assume small neighbourhoods – low degree vertices
  - But natural graphs (LInkedIn, Facebook, Twitter) is not like that – power law graphs – small no. of highly connected people/vertices (popular) and large no. of low degree vertices.
  - Hard to partition power law graphs, high degree vertices limit parallelism.
- GraphLab provides new way of partitioning power law graphs
  - Edges are tied to machines, vertices (esp. high degree ones) span machines
  - Execution split into 3 phases:
    - Gather, apply and scatter.
  - Triangle counting on Twitter graph
    - Hadoop MR took 423 minutes on 1536 machines
    - GraphLab 2 took 1.5 minutes on 1024 cores (64 machines)

[1] Joseph E. Gonzalez, Yucheng Low, Haijie Gu, Danny Bickson, and Carlos Guestrin (2012). "**PowerGraph: Distributed Graph-Parallel Computation on Natural Graphs**." *Proceedings of the 10th USENIX Symposium on Operating Systems Design and Implementation (OSDI '12)*.

Broad Focus



How to maximize efficiency, scalability of performing operations on Big-data – including storage, search, computation and analytics.

Erasure Coding VS Replication [1]

	Fixed MTTF & Repair Epoch	Fixed Storage Overhead & Repair Epoch	Fixed Storage and MTTF (10 million machines, 10% down).
Erasure Coding	Much lower storage	MTTF ~ 10^{20} years	8 nines availability (with 32 fragments)
Replication	Much higher bandwidth	MTTF < 100 years	2 nines availability (with 2 replicas)

MTTF – mean time to failures
Repair epoch – protocol for repairing failed disks

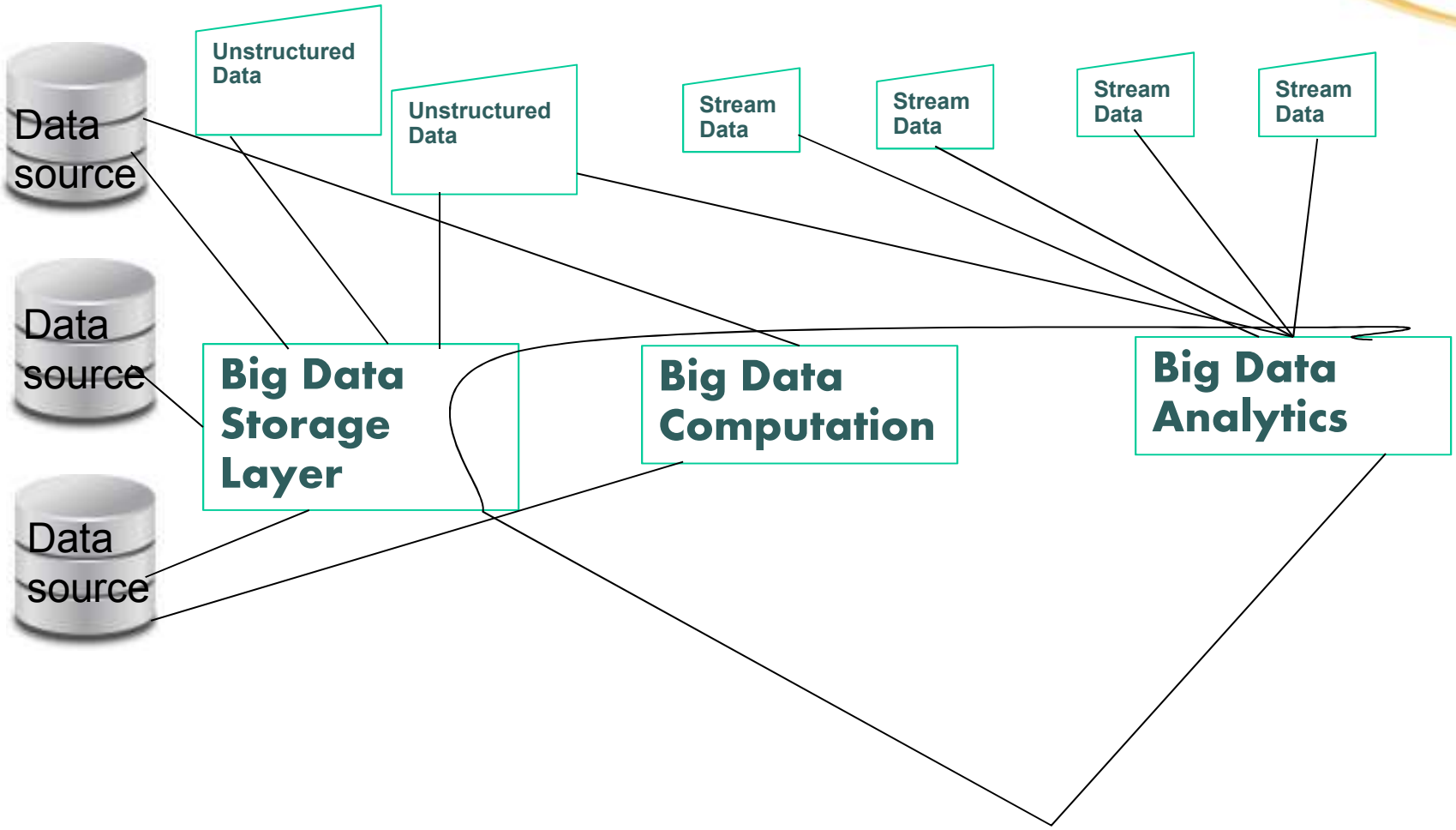
[1] Hakim Weatherspoon and John Kubiatowicz. 2002. Erasure Coding Vs. Replication: A Quantitative Comparison. In *Revised Papers from the First International Workshop on Peer-to-Peer Systems* (IPTPS '01), Peter Druschel, M. Frans Kaashoek, and Antony I. T. Rowstron (Eds.). Springer-Verlag, London, UK, 328-338.

Erasure Coding in Big-data Storage

- Microsoft Windows Azure File System (WAS)
  - Users can store infinite data forever.
  - Uses EC – local reconstruction codes
    - Lowers no. of EC fragments required for reconstruction.
    - Append only distributed file system
    - Active *extents* are replicated 3 times – once > 1 GB, ECed. Replicas deleted subsequently.
    - Performance trade-off between replication VS EC – fragments can be offline, network/node failures, reconstruction involves network bandwidth, computation time.
- HDFS RAID – uses 4/5 EC special case of general EC.
  - Hadoop 503 – incorporated into code, not a general EC mechanism.
- Rethinking EC for cloud [OK12] – proposes rotated Reed-Solomon codes.

[CH12] Cheng Huang, Huseyin Simitci, Yikang Xu, Aaron Ogus, Brad Calder, Parikshit Gopalan, Jin Li, and Sergey Yekhanin. 2012. Erasure coding in windows azure storage. In *Proceedings of the 2012 USENIX conference on Annual Technical Conference* (USENIX ATC'12). USENIX Association, Berkeley, CA, USA, 2-2.
[OK12] Osama Khan, Randal Burns, James Plank, William Pierce, and Cheng Huang. 2012. Rethinking erasure codes for cloud file systems: minimizing I/O for recovery and degraded reads. In*Proceedings of the 10th USENIX conference on File and Storage Technologies* (FAST'12). USENIX Association, Berkeley, CA, USA, 20-20.

Broad Focus



How to maximize efficiency, scalability of performing operations on Big-data – including storage, search, computation and analytics.

Real-time Analytics: Trends

- Analysis at the “speed of thought”
  - Qubole, DataDog, Boundary – startups in this space.
  - Space Time Insight – $14M funding for geospatial and visual analytics software in real-time Big-data space.
- Visualization + analytics at speed of thought
  - Self-service data science – no need of data scientist
  - Integration of visualization + big-data + Artificial intelligence + social + analytics
  - Interesting startups in this space – Tableau, Cliktech, Edgespring.

Thank You!

vijay.sa@impetus.co.in or

on LinkedIn at http://in.linkedin.com/in/vijaysrinivasagneeswaran

Or on Twitter @a_vijaysrinivas.

My company: Impetus Infotech Pvt. Ltd. (www.impetus.com)

I am looking for smart bees/ants to join my team!