Expressing and Running Big Data Analytics

Dr. Berthold Reinwald
IBM Almaden Research Center

3rd Workshop on Large-scale Data Mining: Theory and Applications (LDMTA 2011)
San Diego, Aug, 2011



© 2011 IBM Corporation

Disclaimer

© Copyright IBM Corporation 2010. All rights reserved.
U.S. Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. IT DOES NOT REFLECT ON IBM'S PRODUCT PLANS OR BUSINESS STRATEGY. IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION. NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, NOR SHALL HAVE THE EFFECT OF, CREATING ANY WARRANTIES OR REPRESENTATIONS FROM IBM (OR ITS SUPPLIERS OR LICENSORS), OR ALTERING THE TERMS AND CONDITIONS OF ANY AGREEMENT OR LICENSE GOVERNING THE USE OF IBM PRODUCTS AND/OR SOFTWARE.

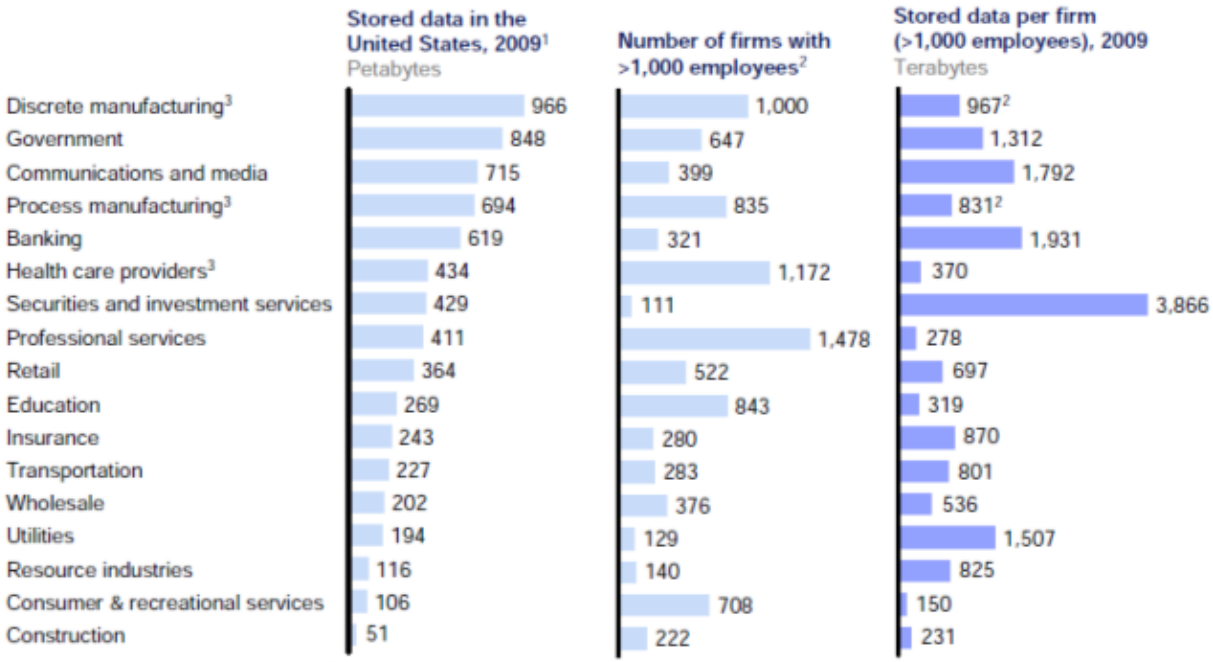
IBM, the IBM logo, ibm.com, DB2, and Websphere are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml

© 2011 IBM Corporation

Big Data

- What is “Big Data”?
  - Datasets whose size is beyond the capacity of current standard databases to capture, store, manage, and analyze.
- Who has “Big Data”?

Companies in all sectors have at least 100 terabytes of stored data in the United States; many have more than 1 petabyte



1 Storage data by sector derived from IDC.
2 Firm data split into sectors, when needed, using employment
3 The particularly large number of firms in manufacturing and health care provider sectors make the available storage per company much smaller.

SOURCE: IDC; US Bureau of Labor Statistics; McKinsey Global Institute analysis

© 2011 IBM Corporation

Big Data Analytics

Financial Services
- Improved risk decisions
- Customer sentiment analysis
- AML





Utilities
- Weather impact analysis on power generation (site planning)
- Smart meter data analysis

Transportation
- Weather and traffic impact on logistics and fuel consumption







IT
- Transition log analysis for multiple transactional systems

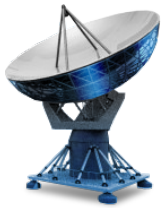
Call Centers
- Voice-to-text mining for customer behavior understanding





E Commerce
- Analyze internet behavior and buying patterns
- Digital Asset Piracy

Telecommunications
- Operations and failure analysis from device, sensor, and GPS inputs
- Customer behavior analysis





Multi-channel integration
- Integrated customer behavior modeling

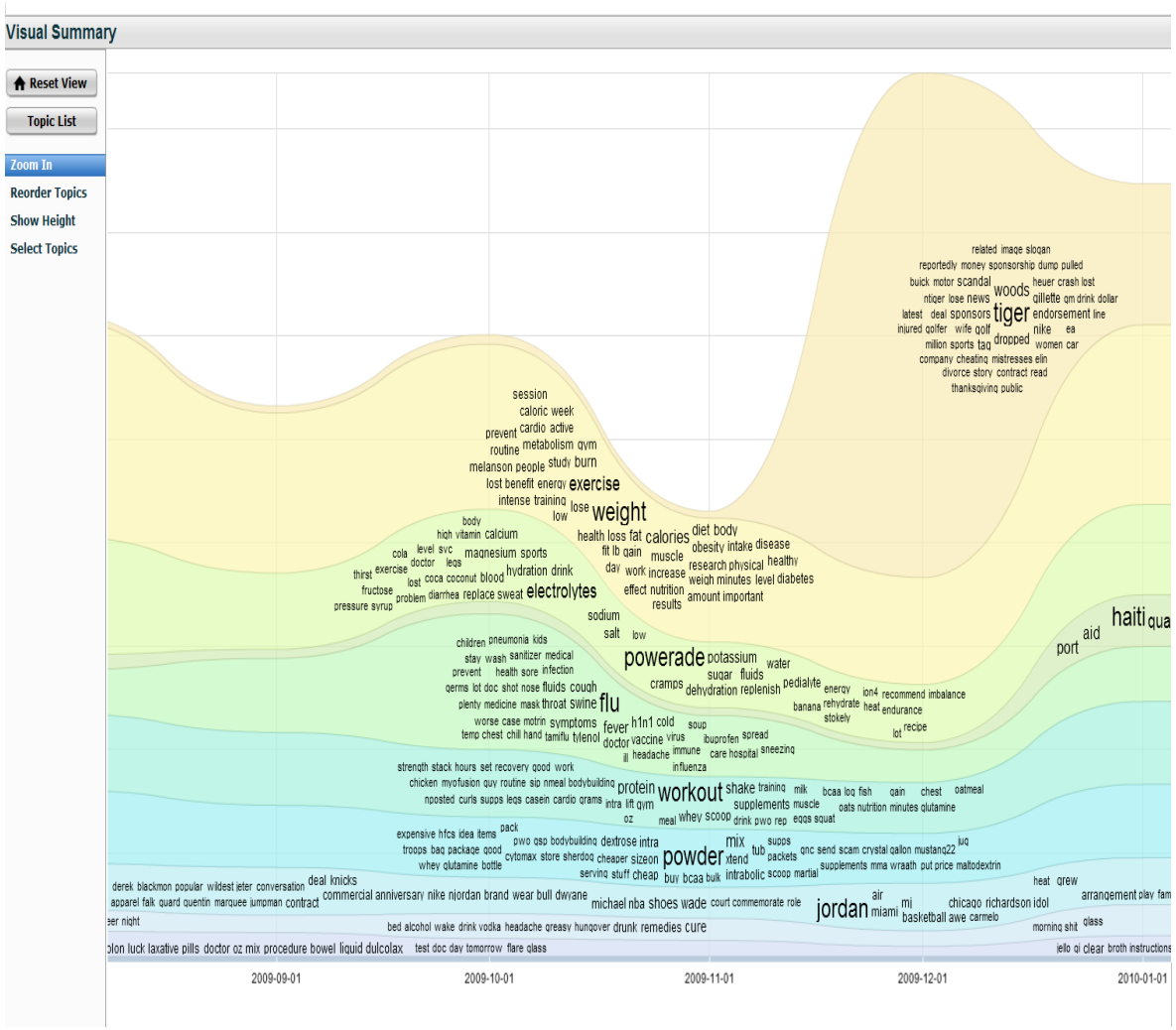
© 2011 IBM Corporation

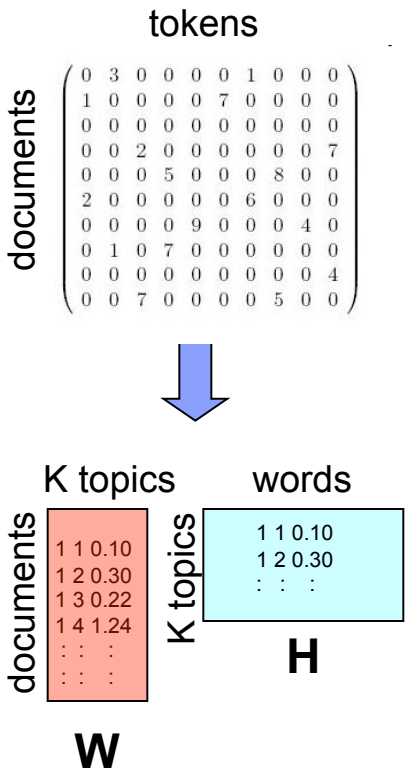
And more …

- Health Care
  - Predictive modeling
    - Earlier stage elimination of drugs
    - Improvement on drug pipeline workflow
  - Algorithms for improved clinical trials
    - Better recruitment
  - Analyzing clinical trials
    - Additional indications / uses
    - Side-effects
  - Personalized medicine
    - DNA/gene based approaches
  - Analyzing disease patterns
    - Predict demand, prepare response
- Public Sector
  - Culture of decision making needs to be based on data.
  - …

© 2011 IBM Corporation

Big data Clustering Application → Identify emerging topics in Social Media





© 2011 IBM Corporation

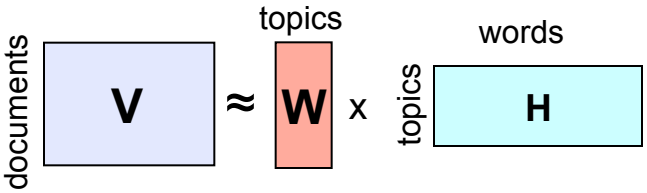
What are the bottlenecks in making such algorithms usable for large-scale problems ?

Learning the parts of objects by non-negative matrix factorization

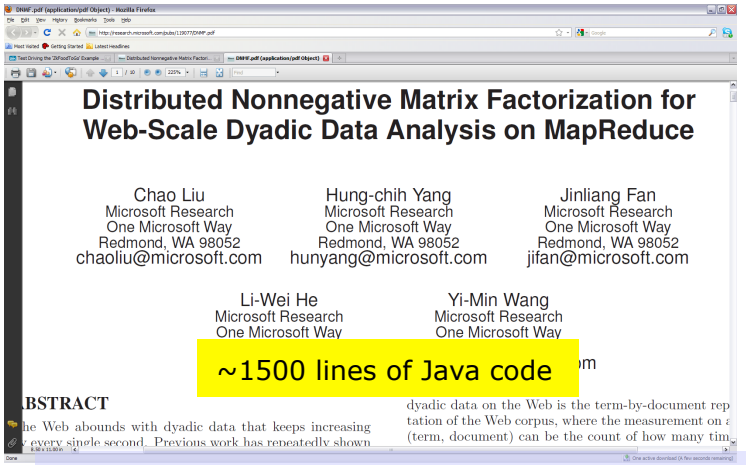
Daniel D. Lee* **& H. Sebastian Seung*†**

** Bell Laboratories, Lucent Technologies, Murray Hill, New Jersey 07974, USA*
† Department of Brain and Cognitive Sciences, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139, USA

```
while (i < max iteration) {
  H = H ∗ (W^T V / W^T WH);
  W = W ∗ (V H^T / WHH^T );
  i = i + 1;
}
```



WWW 2010 paper



Distributed Nonnegative Matrix Factorization for Web-Scale Dyadic Data Analysis on MapReduce

Chao Liu, Microsoft Research, One Microsoft Way, Redmond, WA 98052, chaoliu@microsoft.com
Hung-chih Yang, Microsoft Research, One Microsoft Way, Redmond, WA 98052, hunyang@microsoft.com
Jinliang Fan, Microsoft Research, One Microsoft Way, Redmond, WA 98052, jifan@microsoft.com
Li-Wei He, Microsoft Research, One Microsoft Way
Yi-Min Wang, Microsoft Research, One Microsoft Way

~1500 lines of Java code

- Careful partitioning of data
- Maximize data locality and parallelism
- Billions of non-zeros within tens of hours

Parallel implementation is half the story !
Typical application requires experimenting with multiple variants

```
% initialize W, H
while (~converged)
  W = W*(V%*%t(H))/(W%*%H%*%t(H))
  H  = H*(t(W)%*%V)/(t(W)%*%W%*%H)
end
```

```
W = W*max(V%*%t(H) – alphaW JW, 0)/(W%*%H%*%t(H))
H  = H*max(t(W)%*%V – alphaH JH, 0)/(t(W)%*%W%*%H)
```

Regularizers JW,JH

```
W = W*((S*V)%*%t(H))/((S*(W%*%H))%*%t(H))
H  = H*(t(W)%*%(S*V))/(t(W)%*%(S*(W%*%H)))
```

Weighted Sq Loss/ Matrix Completion Setting

```
W = W*(V/(W%*%H) %*% t(H))/(E*%t(H))
H  = H*(t(W)%*%(V/(W%*%H)))/(t(W)%*%E)
```

Different Loss function KL-divergence

© 2011 IBM Corporation

Challenges in Expressing and Running Big Data Analytics

- Many “small” systems for in-memory/in-core datasets
  - Mbytes and GBytes in R, Matlab

➧ ***Big Data: TBytes and PBytes***

- Single machine implementations
  - multi-core exploitation through kernel libraries
  - R packages for big memory and big analytics

➧ ***Scalability: clusters with 1,000s of nodes with parallel data access & compute***

- Data analysts explicitly code parallelism
  - extensions in Parallel Matlab, R/SNOW
  - hand-crafted for specific platforms, e.g. MPI

➧ ***Productivity: Higher-level language***

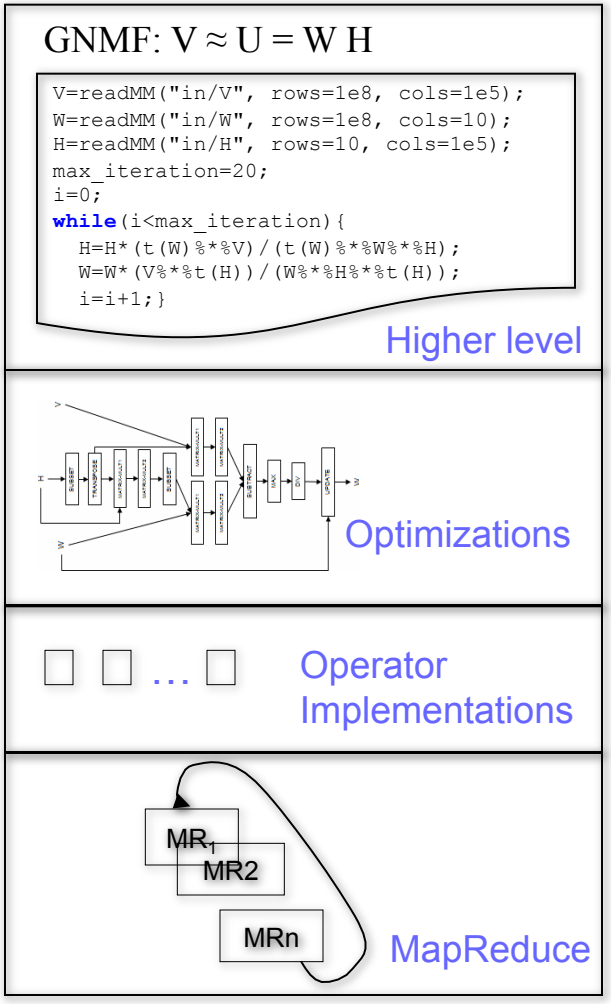
- Procedural implementation of algorithms
  - with fixed execution strategy and runtime
  - expensive to maintain for different datasets

➧ ***Optimization: Declarative language***

© 2011 IBM Corporation

Expressing and Running Big Data Analytics

- Provide language to implement ML algorithms
- Support specific ML constructs such as cross validation and ensemble learning as first class citizens
- Optimizations based on data and system characteristics
- Scalable operator implementations
- Run in MapReduce



© 2011 IBM Corporation

Why Did we take this Approach?

- MapReduce is natural for parallelizing a large class of ML problems.
  - e.g. Mappers compute local stats (e.g. gradients over data subsets) which Reducers aggregate into global stats (full gradient) need for learning

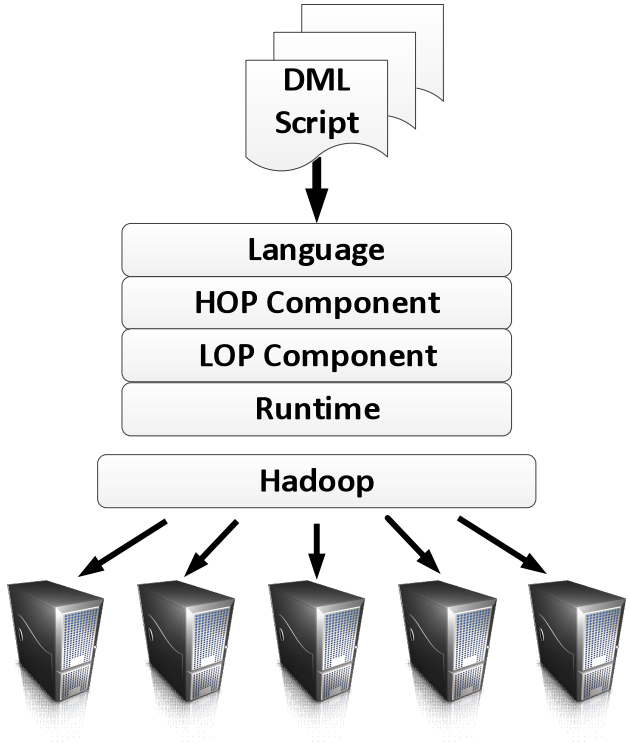
- Directly implementing ML algorithms on MapReduce is burdensome.
  - Mathematical operators need to be re-expressed in terms of key-value pairs, map and reduce functions.
  - Data characteristics dictate the optimal MapReduce implementation, so user bears responsibility for efficient hand-coding.

- *Approach*
  - *Machine learning algorithms are expressed in declarative high-level language and executed in a MapReduce environment*
    - *Productivity of data analyst*
    - *Optimizations in generating low-level execution plans for MapReduce*

- **SystemML is a project at IBM Research implementing the approach**
  - **http://www.almaden.ibm.com/cs/projects/systemml/**

© 2011 IBM Corporation

SystemML Architecture



- **Language**
  - **DML: D**eclarative **M**achine learning **L**anguage
    - Constructs such as *while* and *for* loops
    - Specific ML constructs such as cross validation, ensemble learning
  - Program Analysis
- **High-Level Operator (HOP) Component**
  - Represent dataflow in DAGs of operations on matrices, scalars
  - Choosing from alternative execution plans, such as algebraic rewrites
- **Low-Level Operator (LOP) Component**
  - Low-level physical execution plan (represented as LOPDags) over key-value pairs
  - "Piggybacking" operations into MapReduce jobs
- **Runtime**
  - Key-value pair representation of matrices
  - Generic MapReduce job
  - Control module to orchestrate of MR jobs

© 2011 IBM Corporation

DML

- Syntax borrowed from R
- DML: **D**eclarative **M**achine learning **L**anguage
  - Data Constructs: matrix, scalar
  - Input/Output
  - Assignment
  - Control Structures
  - DML Expressions
    - Operators: arithm., mmult, power, relational, boolean
    - Functions: linear algebra, matrix (partial) aggr., comparison, mathematical
- Program Analysis
  - Type assignment
    - V, W, H are typed as matrices
  - Divide a script into smaller units to be analyzed and optimized separately
  - Statement Block identification
    - While loops and functions are natural boundaries
    - Statements in each statement block are compiled and optimized.
  - Live variable analysis
    - Connect variable use with the preceding write(s)
    - Decide what variables need to be passed cross statement blocks

Statement Block SB_1

Live Variables In : None

```
1. V = readMM ("in/V", rows = 1e8, cols =1e5, nnzs =1e10);
2. W = readMM ("in/W", rows = 1e8, cols = 10);
3. H = readMM ("in/H", rows = 10, cols = 1e5);
4. max_iteration = 20;
5. i = 0;
```

Live Variables Out
Matrix : W, H, V
Scalar : i, max_iteration

Live Variables In
Matrix : W, H, V
Scalar : i, max_iteration

W refers to output of Step 2 or Step 8 of previous iteration and is a 10^8 x 10 matrix

Statement Block SB_2

```
6.  while (i < max_iteration) {
7.    H = H * ( t(W) %*% V ) /   ( t(W) %*% W %*% H );
8.    W = W *  ( V %*% t(H)  ) / ( W %*% H %*% t(H) );
9.     i = i + 1 ; }
```

Live Variables Out
Matrix : W, H, V
Scalar : i, max_iteration

Live Variables In
Matrix : W, H

Statement Block SB_3

```
10. writeMM (W, "result/W");
11. writeMM (H, "result/H");
```

H refers to output of Step 7 and is a 10 x 10^5 matrix

Live Variables Out : None

© 2011 IBM Corporation

Additional Features

- Functions
  - DML Packages
    - sharing of algorithms written in DML as reusable libraries
  - External Packages
    - existing single-node implementations in external languages
  - External Parallel Packages
    - Java-based API and programming framework
- Meta-learning constructs
  - Cross validation
  - Ensemble learning
- Built-in functions to support univariate and bivariate statistics, e.g.
  - contingency table
  - order statistics (exploits sort capabilities of underlying platform)

© 2011 IBM Corporation

Applications

Customer Targeting Churn/Up sell/Appetency Analysis

Revenue Estimation Predictive Modeling

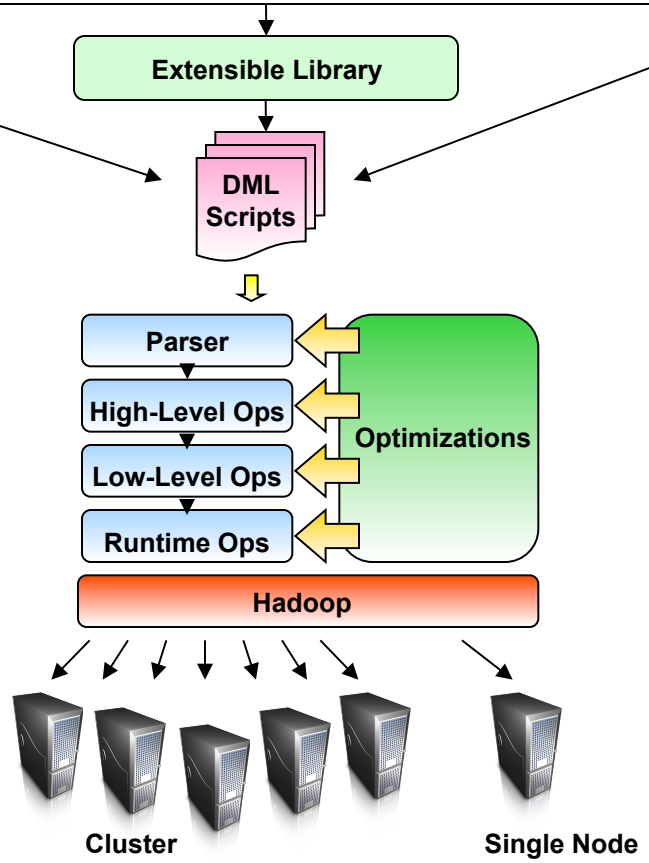
Recommender Systems Topic Modeling

Customer Segmentation

Search Counterparty Analysis

ML Algorithms

Classification	*Regression*	*Matrix Factorizations*	*Clustering*	*Ranking*
Linear SVMs, Logistic Reg	**Linear Regression**	**SVD, NMF**	**Spectral Clustering**	**PageRank, HITS**



© 2011 IBM Corporation

Algorithms in SystemML

DML

- Sparse Linear Solvers based on Conjugate Gradient
  - Solve A x = b where A is symmetric, positive semi-definite, large and sparse
  - Only access A via (sparse) matrix-vector products
  - Matlab/R allow A to be implicitly specified by a function that consumes v, returns A %*% v
- Linear Regression for large, sparse datasets
  - X : data matrix (large n x large p but sparse), y: n x 1 label vector
  - Adapts CG for (regularized) normal equations: A = X'X + c I, b = X' y
- Non-negative Matrix Factorizations
  - minimize ||X – WH||^2 subject to W>=0, H>=0
- PageRank of a directed graph
  - Recursive definition: Pagerank(i) = sum of Pageranks of nodes pointing to i.
  - Computes leading eigenvector of the adjacency matrix G via power method.
  - Application: ranking web pages, or people.
- HITS Hubs and Authorities
  - Hub_score(i) = sum of authority scores of nodes it points to.
  - Authority_score(i) = sum of hub scores of nodes that point to it.
  - Leading eigenvectors of GG' and G'G computed via power method.
- Descriptive Statistics (univariate and bivariate)
- PCA (large n, small p)
- Linear Logistic Regression
  - Lin,Weng,Keerthi, Trust Region Method for Large-scale Logistic Regression, Journal of Machine Learning Research (JMLR) 9 (2008), 627-650)
- Linear SVMs
  - Keerthi and Decoste, Modified Finite Newton Method for Fast Solution of Large-scale Linear SVMs, Journal of Machine Learning Research, 2005
- Parallel Optimization for sparse linear models

External parallel Packages

- Apriori for frequent itemset mining
- Random decision trees for classification & regression
- Pattern growth for frequent itemset mining
- Recursive binning and reprojection for distance based outlier detection
- Stochastic gradient descent for Matrix Factorization

© 2011 IBM Corporation



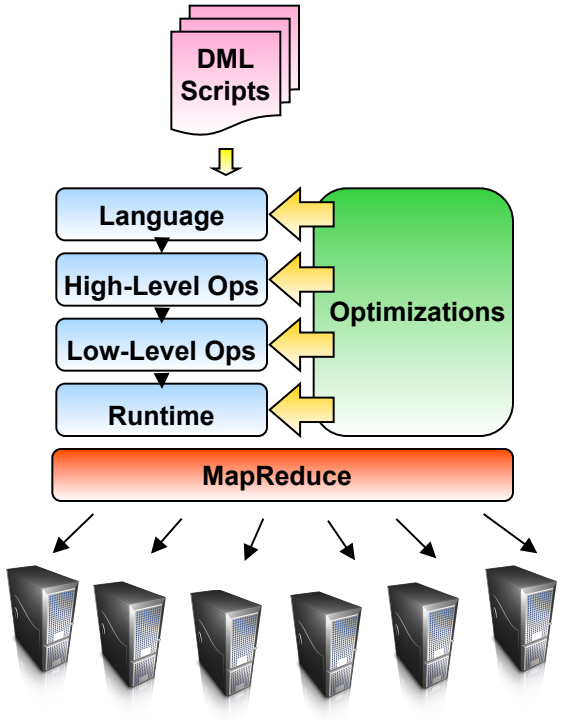
Does it work

… and Why?

- **Optimizations**
- **Distributed Platform**

© 2011 IBM Corporation

Technical Approach



SystemML Architecture

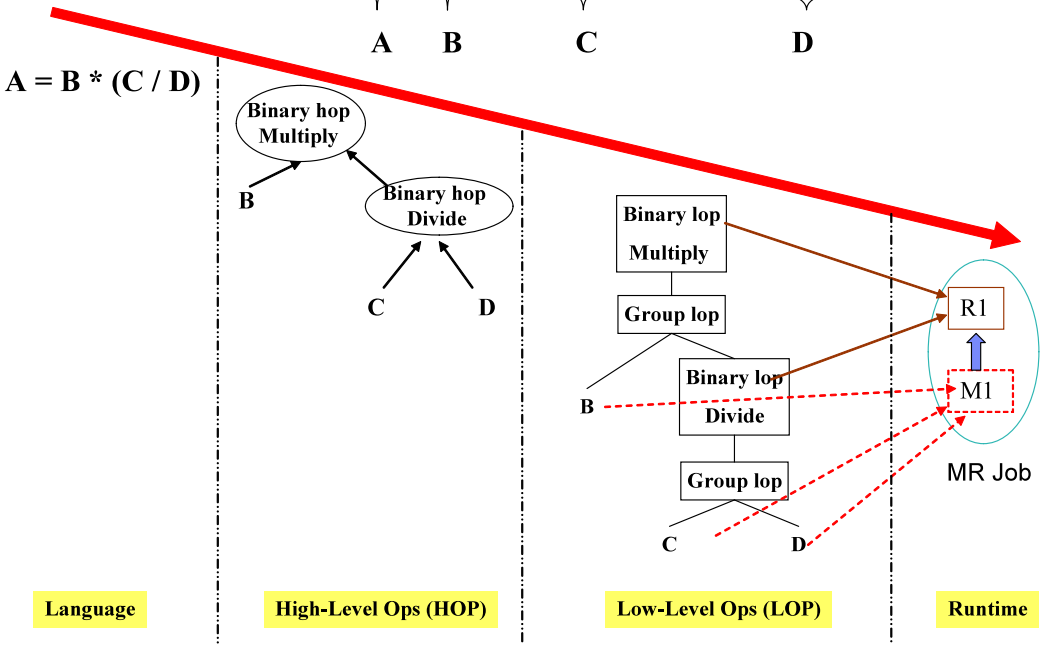
NMF

```
V=readMM("in/V", rows=1e8, cols=1e5);
W=readMM("in/W", rows=1e8, cols=10);
H=readMM("in/H", rows=10, cols=1e5);
max_iteration=20;
i=0;
while(i<max_iteration){
  H=H*(t(W)%*%V)/(t(W)%*%W%*%H);
  W=W*(V%*%t(H))/(W%*%H%*%t(H));
  i=i+1;}
writeMM(W, "out/W");
writeMM(H, "out/H");
```

H = H * (t(W)%*%V / t(W)%*%W%*%H)

A B C D

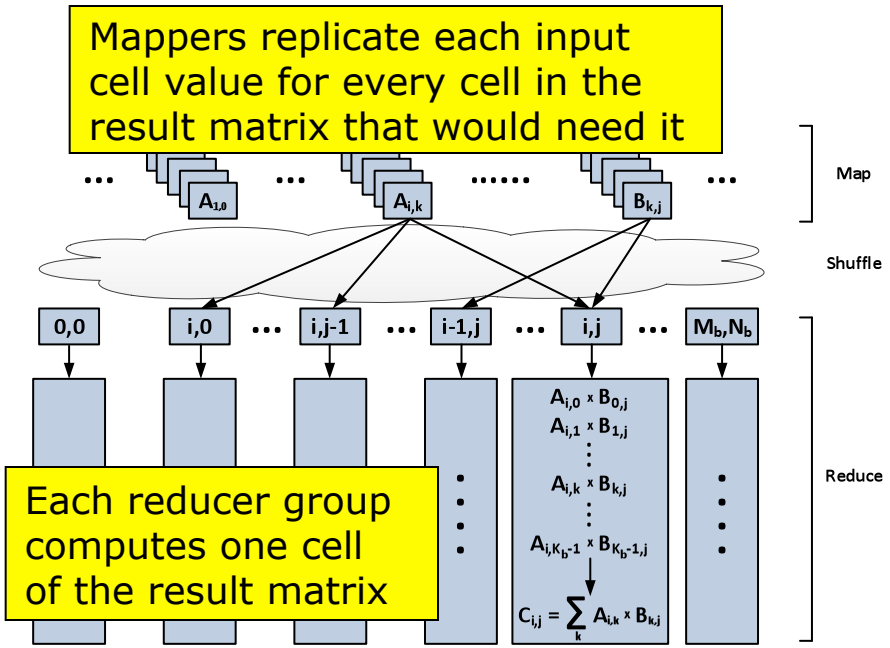
A = B * (C / D)



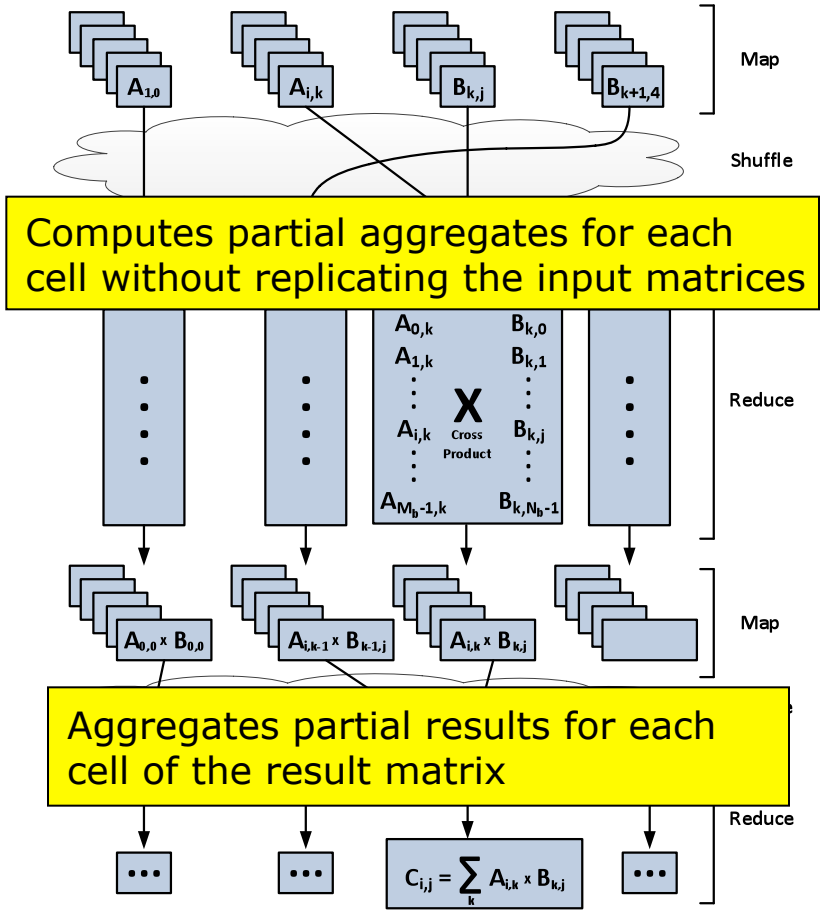
© 2011 IBM Corporation

HOP Optimization: Cost based optimization for matrix multiplication

RMM: Replication Based Matrix Multiplication



CPMM: Cross Product Based Matrix Multiplication



In practice processing is at the level of blocks of matrices as opposed to individual cells

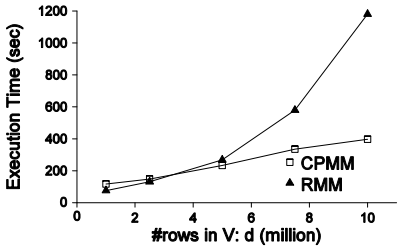
© 2011 IBM Corporation

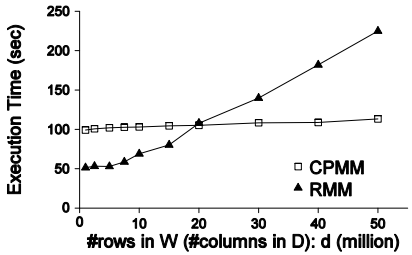
Do these different matrix multiplication plans matter ?

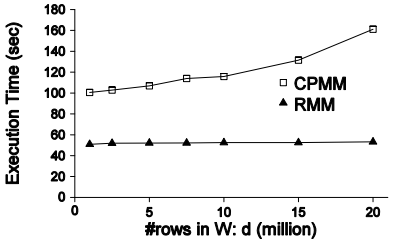
With large matrices replication can become expensive compared to materialization

Cross-over point depends on data characteristics

When one matrix is small, replication is inexpensive
Even if other matrix is large, common dimension remains small







V %*% H'

V: d x 100000, sparsity=0.001

H': 100000 x 10, dense

Scenario: Fixed large matrix, varying size of second large matrix

W' %*% W

W': 10 x d, dense

W: d x 10, dense

Scenario: Varying both matrices

W %*% S

W: d x 10, dense

S: 10 x 10, dense

Scenario: Fixed small matrix, varying size of second large matrix

© 2011 IBM Corporation

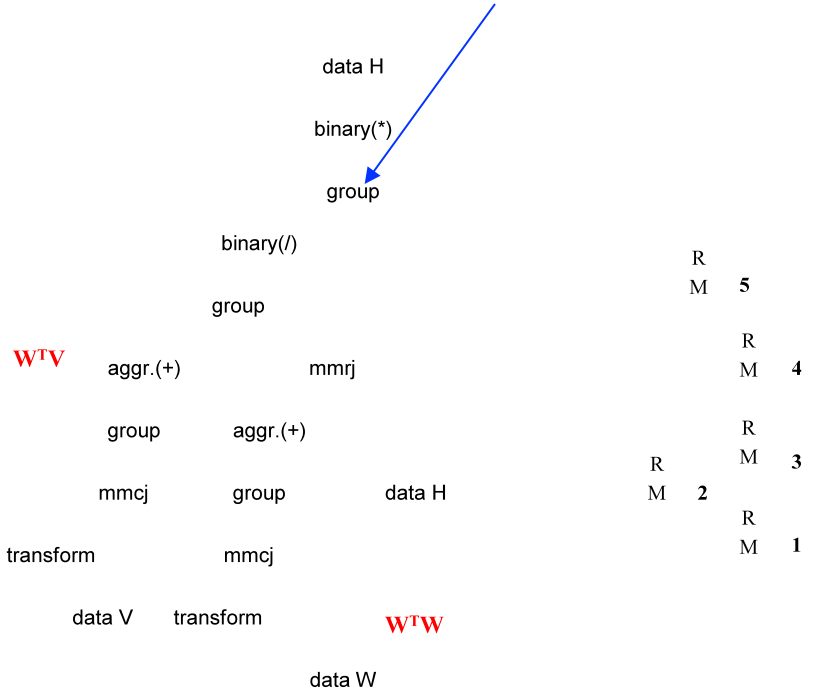
“Piggybacking” Optimization to result in small number of MR jobs

$$H = H * ((t(W) \%*\% V) / (t(W) \%*\% W \%*\% H))$$

LOP	Execution Location	Input Requirements/Output Characteristics
aggregate	Reduce	Output keys aligned with input
binary	Reduce	Output keys aligned with input
unary	Map or Reduce	Output keys aligned with input
group	Map and Reduce	None
transform	Map or Reduce	Output keys NOT aligned with input
data	Map or Reduce	none
mmcj	Map and Reduce	none
mmrj	Map and Reduce	none

Group Elimination:

Group LOP does not need a separate MapReduce job as the input need not be re-sorted *(binary(/) does not alter the key alignment*)



© 2011 IBM Corporation

Runtime Representation and Optimization

Key-value representation of matrices and vectors

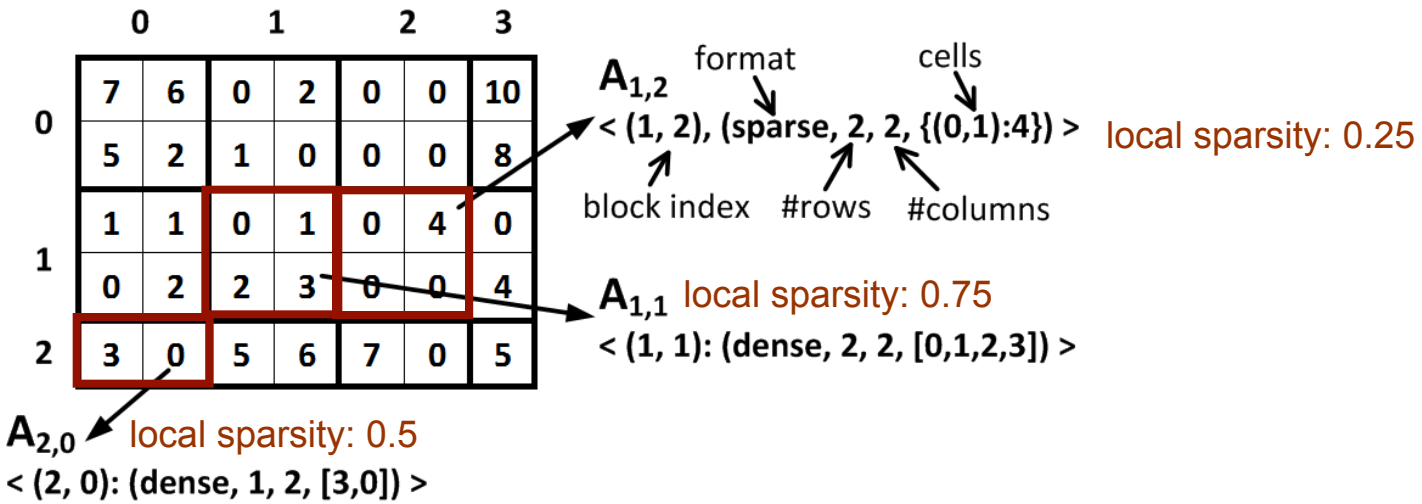
- Cell Representation
  - E.g. <(0,0), 7>, <(0,1), 6> …
- Block Representation
  - Local sparsity: fraction of non-zeros in a block
  - Different physical representation based on local sparsity
  - **Optimization: Dynamic block-level runtime based on local sparsity**

    e.g. C=A%*%B

    $C_{ij} = \Sigma_k (A_{ik} \%*\% B_{kj})$, A_{ik}, B_{kj} and C_{ij} are blocks

    if A_{ik} and B_{kj} are dense, all cells have to be accessed

    if A_{ik} is sparse, only need to go through non-zeros in A_{ik} and find the corresponding cells in B_k



© 2011 IBM Corporation

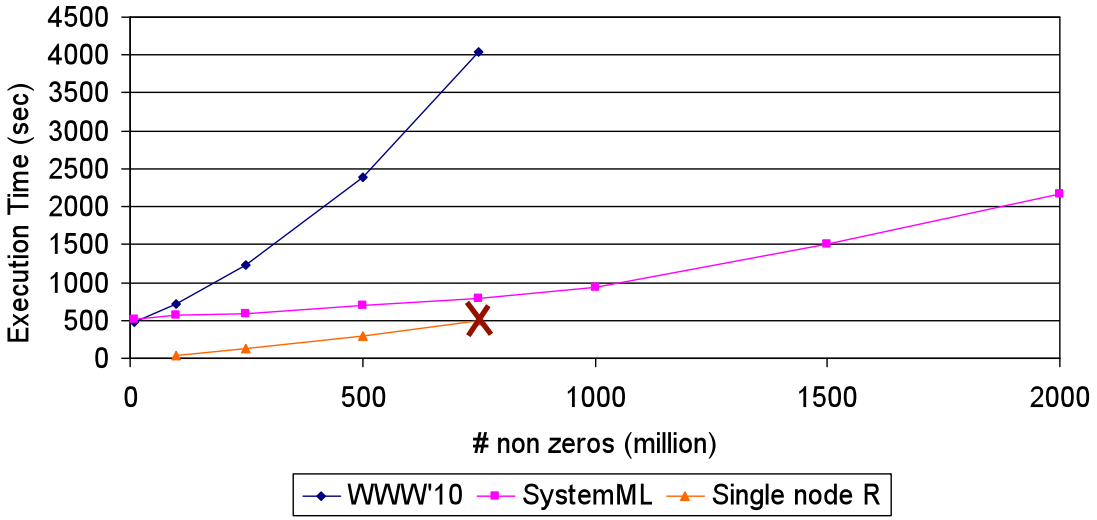
Performance Numbers

Gaussian NMF:

```
V = readMM ("example.GNMF.V", rows= 1000, cols=100, nnzs= 2000, format="text");
W = readMM ("example.GNMF.W", rows= 1000, cols=20, nnzs= 20000, format="text");
H = readMM ("example.GNMF.H", rows= 20, cols=100, nnzs= 2000, format="text");
max_iteration = 10
i = 0

while (i < max_iteration) {
  H = H * ((t(W) %*% V) /  ( (t(W) %*% W) %*% H))
  W = W * ((V %*% t(H)) / ( W %*% (H %*% t(H))))
  i = i + 1
}

writeMM (W,"example.GNMF.W.result", format="text");
writeMM (H,"example.GNMF.H.result", format="text");
```



5 billion non zeros

(50m X 100k, sparsity $1x10^{-3}$)

WWW'10	> 1500 lines of java code
SystemML	11 lines of DML script
Single Node R	64 bit R running with 64GB RAM

© 2011 IBM Corporation

Numeric Stability

- Big Data → Big Imprecision due to
  - finite precision floating point arithmetic
- Two strategies
  - Sort
  - Compensate (running compensation for accumulated small errors)
    - e.g. Kahan sum
    - Distributed implementation that computes partials w/ compensation
- High-order statistics, e.g. variance, skewness, kurtosis
  - Dilemma choosing between
    - expensive 2-pass algorithms or
    - unstable 1-pass algorithms
  - Single-pass, numerically stable computation of arbitrary-order central moment (Bennett ’09)
    - Uses updates rule to combine partial results
  - Distributed MapReduce implementation

© 2011 IBM Corporation

Integration w/ rest of the enterprise

- Integration with rest of IT
  - warehouses, security, archiving, ...
- Administrative tools
  - configuration management
- Processing pipeline
  - data preparation, transformation, post processing, etc.
- UX
  - end user, developer

© 2011 IBM Corporation

Summary

- Big Data Analytics are ubiquitous
- Declarative high-level language
  - System-provided optimizations
  - Productivity
- Platforms for large clusters
- Numeric stability
- Integration into an enterprise platform for end-to-end solutions

© 2011 IBM Corporation



© 2011 IBM Corporation