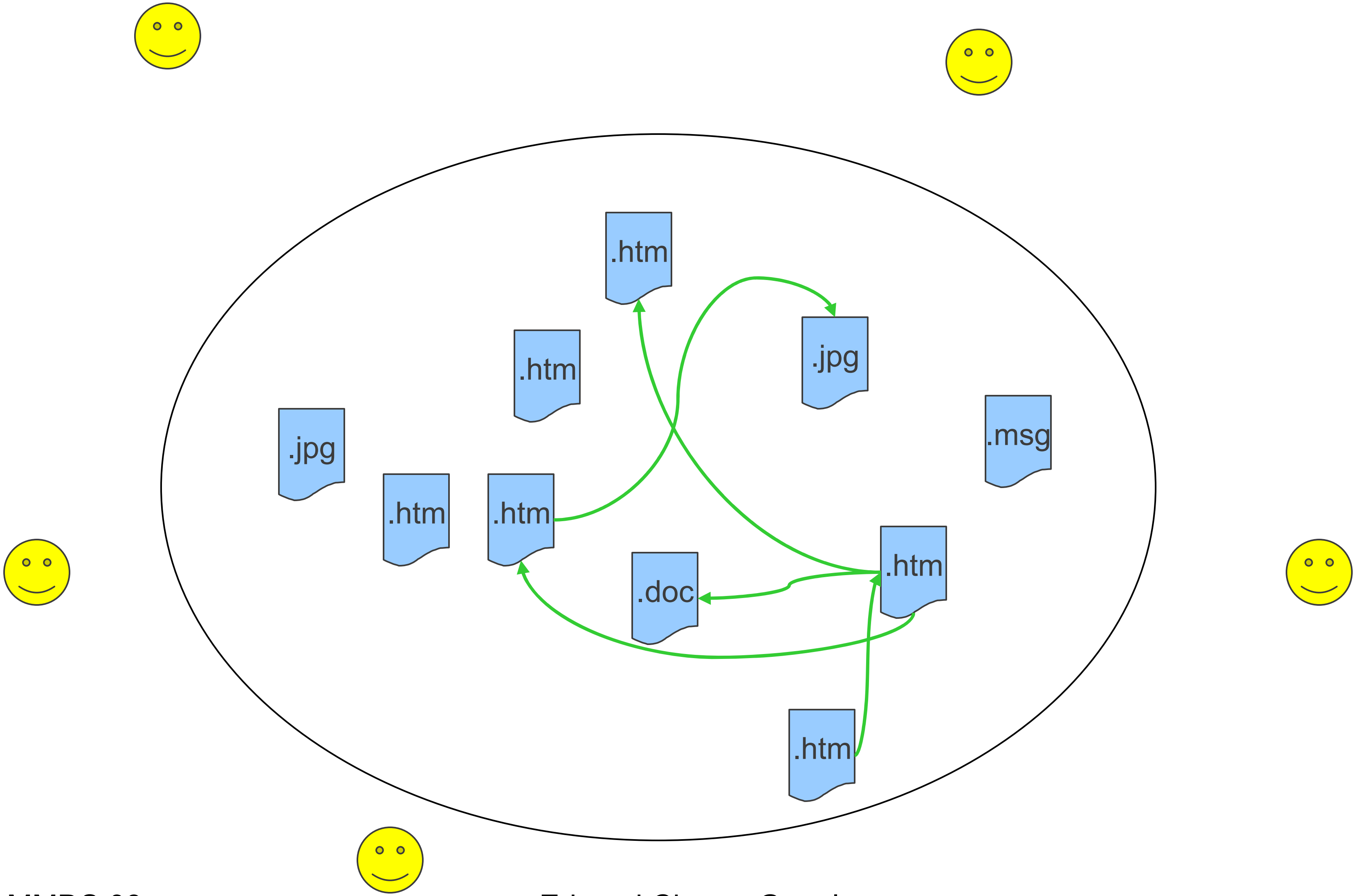
Mining Large-scale Social Networks
Challenges & Scalable Solutions

Edward Chang
Google Research

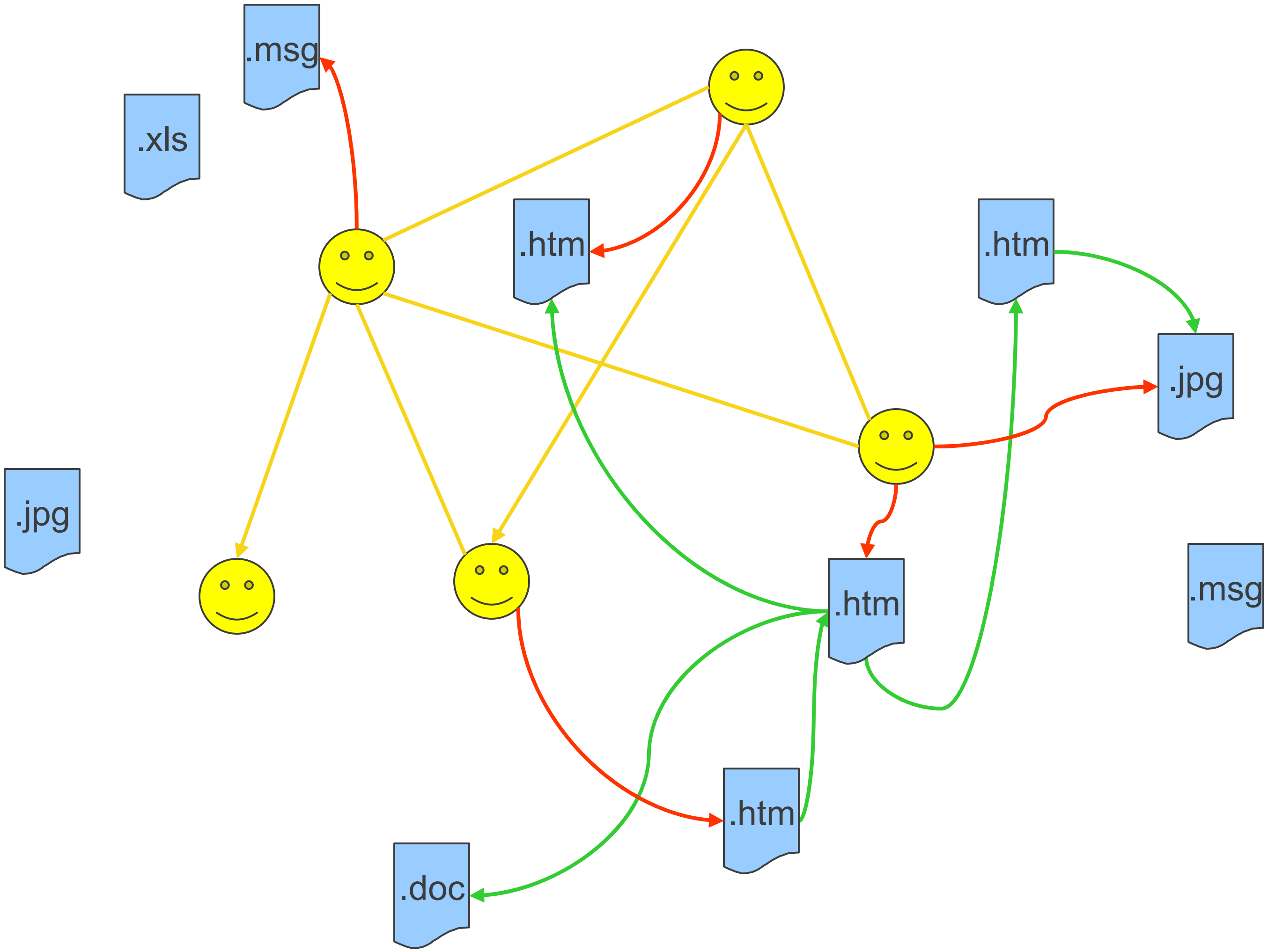
Collaborators

- Prof. Chih-Jen Lin (NTU)
- Hongjie Bai (Google)
- Wen-Yen Chen (UCSB)
- Haoyuan Li (PKU)
- Yangqiu Song (Tsinghua)
- Matt Stanton (Google)
- Yi Wang (Google)
- Dong Zhang (Google)
- Kaihua Zhu (Google)

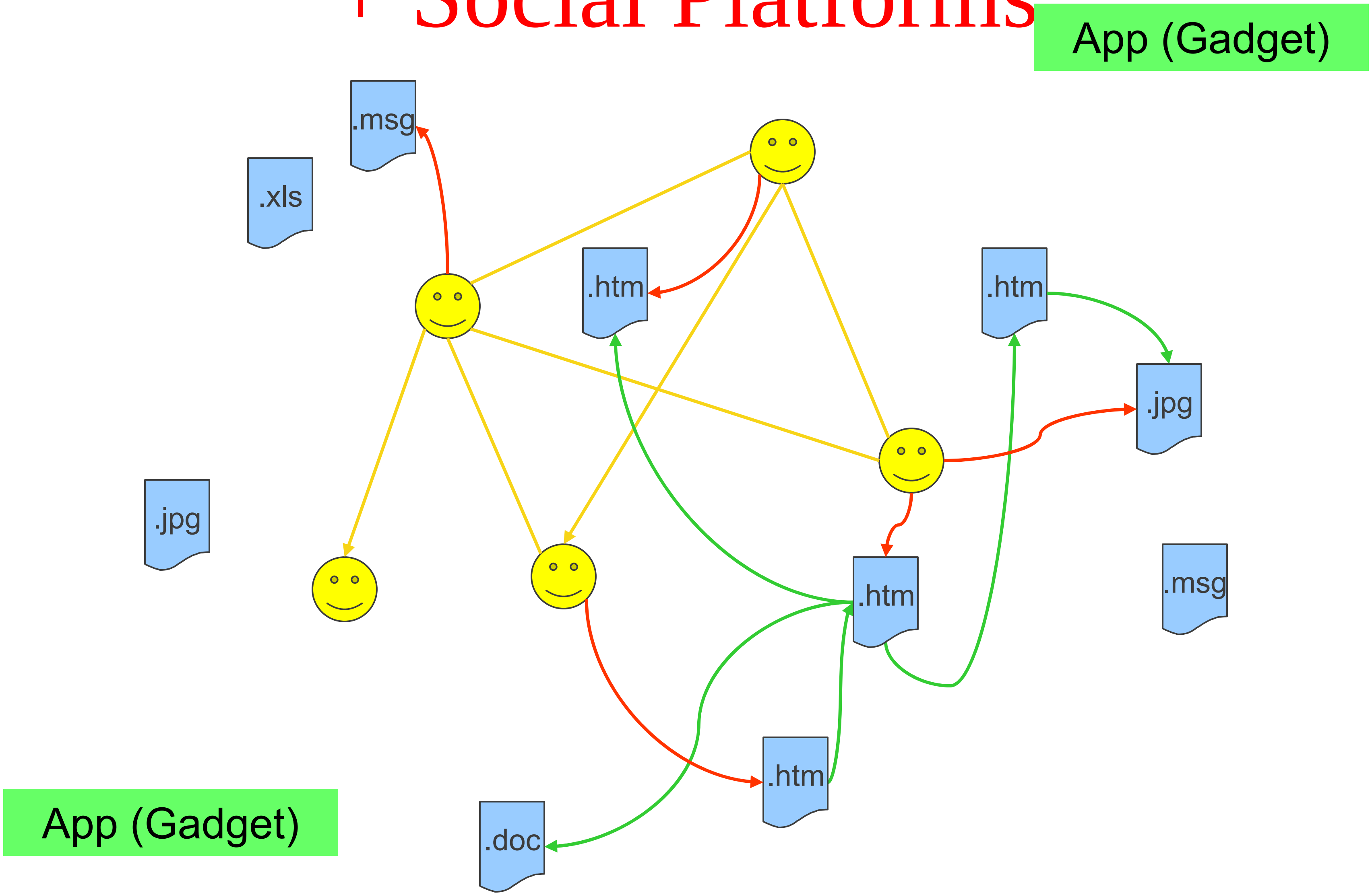
Web 1.0



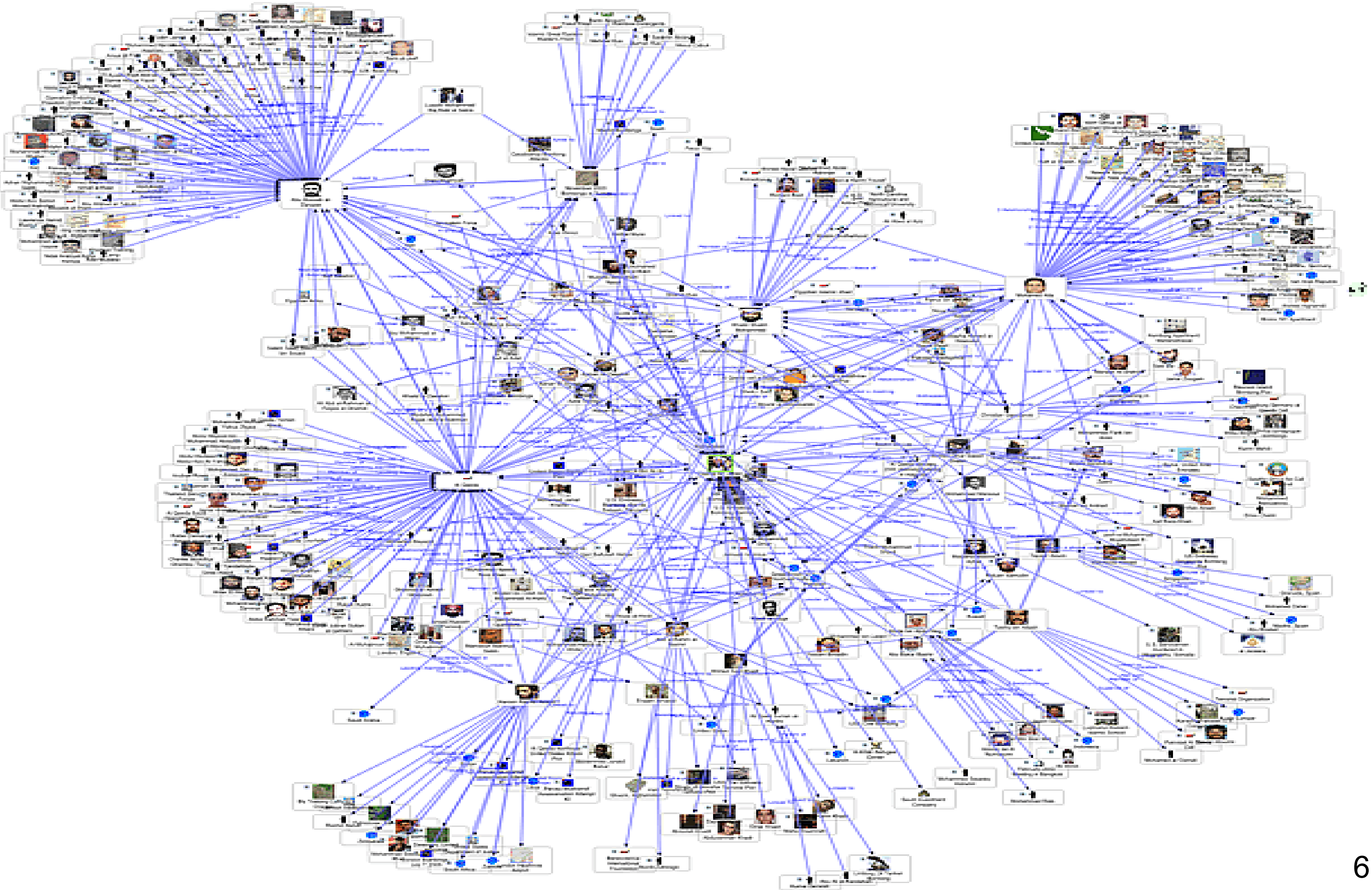
Web 2.0 --- Web with People



+ Social Platforms



People Centric Web



天涯来吧 - 我的朋友圈 - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://laiba.tianya.cn/laiba/FlashFriends
Google
Google.com - In...
google gadget -...
OpenSocial Plat...
Fensi Dashboard
天涯来吧 - 我...
Darren Huang - ...
Google.com - C...
我的主页
资料
朋友
来吧
帖子
影集
日记
礼物
New!
评价
留言簿
我的朋友圈 | 我的朋友
上一步 |
回到自己
易绍
永远来吧
机械舞
汤汤汤圆
开复
暗香浮月
沉网
武汉牛牛
水星cici
梦侠人
无聊最寂寞
Technorati
Google Webmaster Tools
Transferring data from laiba.tianya.cn...
start
3 Firefox
C:\Docum...
Document...
2 Micros...
Google R...
EN
10:04 PM

Recommendation Systems

- Friend Recommendation
- Application Recommendation
- Community/Forum Recommendation
- Ads Matching

- Performance Requirements
  - Scalability, scalability, scalability

Google Data Centers

Outline

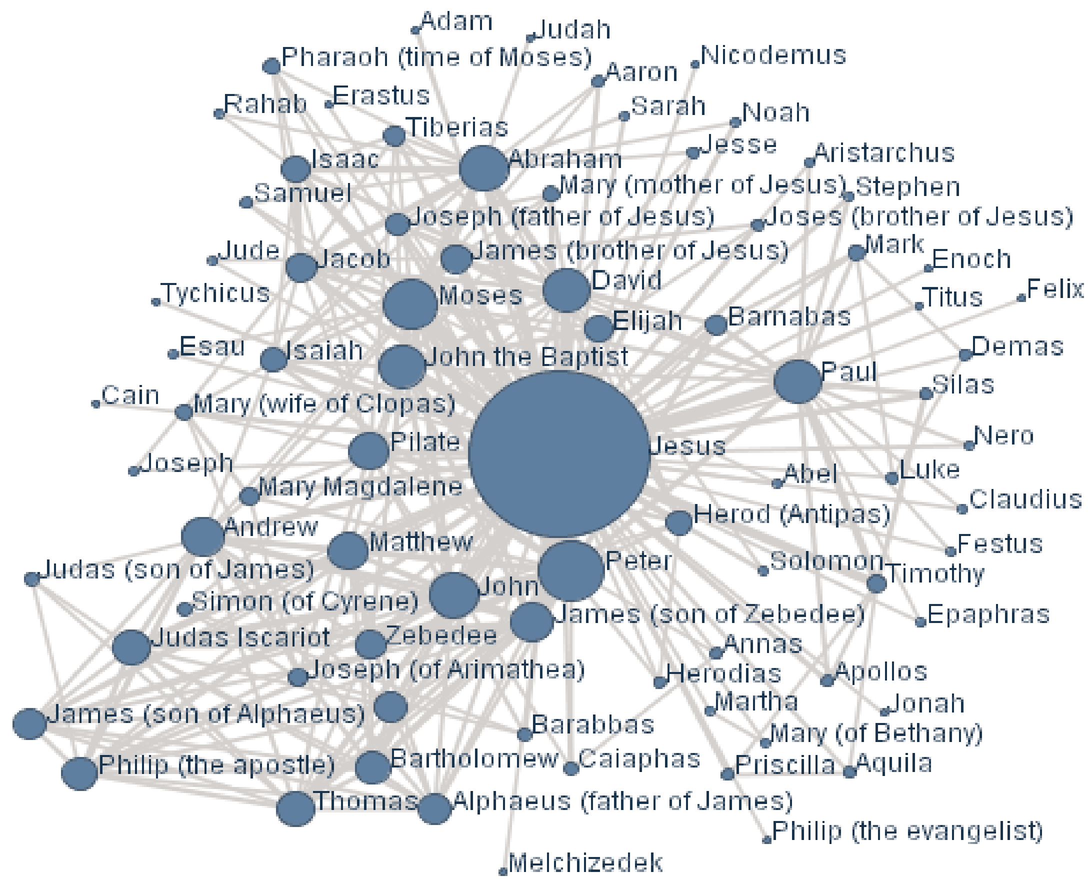
- Emerging Applications
  - Social networks
  - Personalized Information retrieval
- Key Subroutines for Mining Massive SNS
  - Clustering [ECML 08]
  - Frequent Itemset Mining [Google Tech Report 08]
  - Combinational Collaborative Filtering [KDD 08]
    - with PLSA
    - with LDA
  - Support Vector Machines [NIPS 07]

Outline

- Emerging Applications
  - Social networks
  - Personalized Information retrieval
- Key Subroutines
  - Clustering
  - Frequent Itemset Mining (FIM)
  - Combinational Collaborative Filtering
    - with PLSA
    - with LDA
  - Support Vector Machines

Clustering for SNS Analysis

- Centrality
- Degree Centrality
- Closeness
- Betweenness

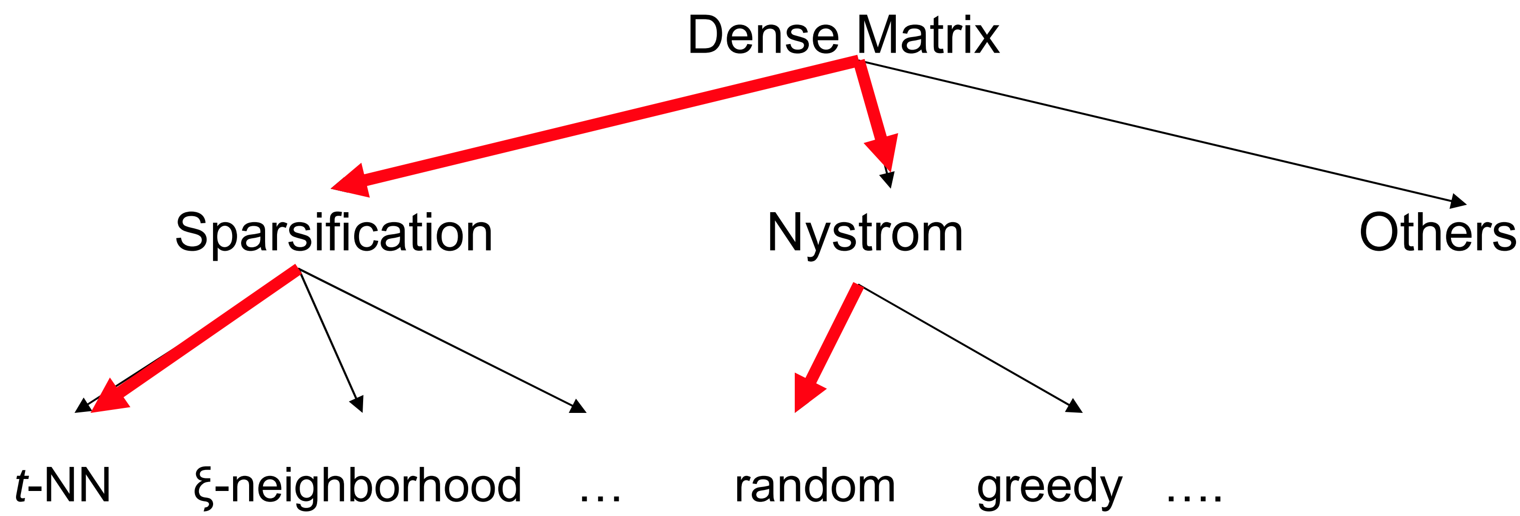


Spectral Clustering [A. Ng, M. Jordan]

- Important subroutine in tasks of machine learning and data mining
  - Exploit *pairwise similarity* of data instances
  - More effective than traditional methods e.g., k-means
- Key steps
  - Construct pairwise similarity matrix
  - Compute the Laplacian matrix
  - Apply eigendecomposition
  - Perform *k*-means

Scalability Problem

- Quadratic computation of nxn matrix
- Approximation methods



Sparsification vs. Sampling

- Construct the dense similarity matrix S
- Sparsify S
- Compute Laplacian matrix L

$$L = I - D^{-1/2} S D^{-1/2}, \quad D_{ii} = \sum_{j=1}^{n} S_{ij}$$

- Apply *ARPACLK* on L
- Use *k*-means to cluster rows of V into *k* groups

- Randomly sample *l* points, where *l* << *n*
- Construct dense similarity matrix [A B] between *l* and *n* points
- Normalize A and B to be Laplacian form
- $R = A + A^{-1/2}BB^{T}A^{-1/2}$; $R = U\Sigma U^{T}$
- *k*-means

Single Machine

- Nystrom approximation by random sampling
  - Random sampling is least costly
  - Trade clustering quality for speed
- Sparsification with t-NN
  - Keep only t-NN of each instance
  - O(n^2) computation and storage

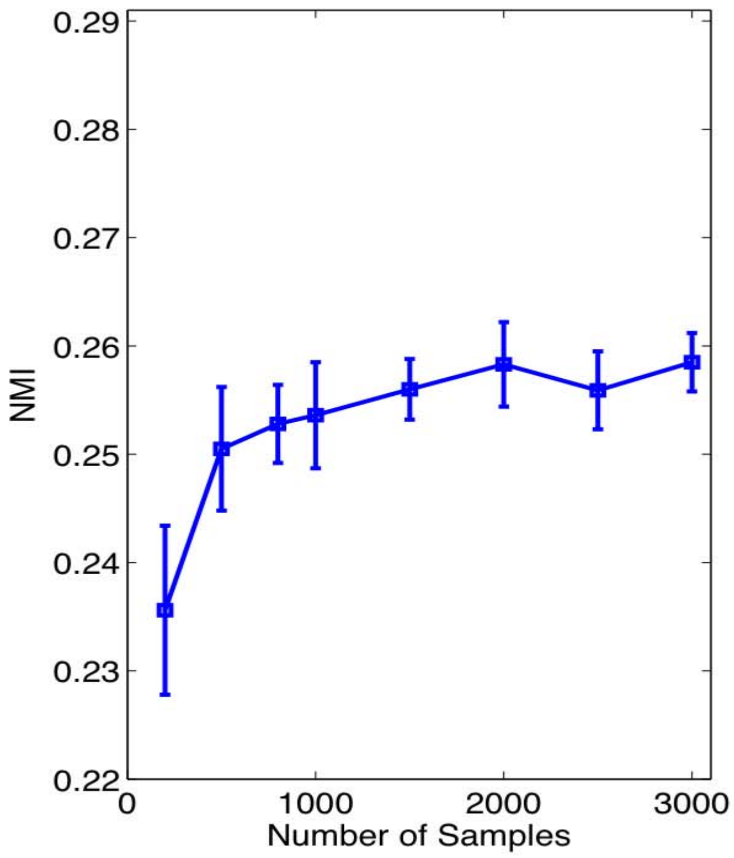
Empirical Study

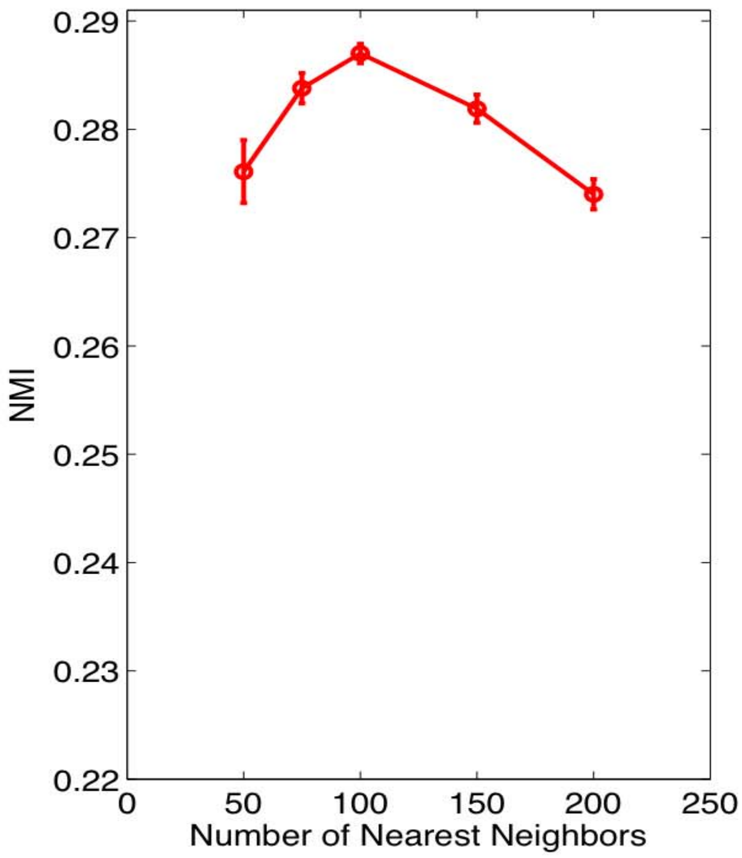
- Dataset: RCV1 (Reuters Corpus Volume I)
  - A filtered collection of *193,944* documents in*103* categories
- Photo set: PicasaWeb
  - *637,137* photos
- Experiments
  - Clustering quality vs. computational time
    - Measure the similarity between CAT and CLS
    - Normalized Mutual Information (NMI)

$$NMI(CAT;CLS) = \frac{I(CAT;CLS)}{\sqrt{H(CAT)H(CLS)}}$$

  - Scalability

NMI Comparison (on RCV1)





Nystrom method **Sparse matrix approximation**

Speedup Test on RCV1

	Eigensolver		k-means	
Machines	Time (sec.)	Speedup	Time (sec.)	Speedup
1	1.870×10^3	1.00	1.557×10^2	1.00
2	8.529×10^2	2.19	1.433×10^2	1.09
4	4.765×10^2	3.92	8.565×10^1	1.82
8	3.094×10^2	6.04	3.235×10^1	4.81
16	2.352×10^2	7.95	4.579×10^1	3.40
32	2.150×10^2	8.70	4.318×10^1	3.61
64	2.563×10^2	7.30	5.008×10^1	3.11

Speedup Test on *637,137* Photos

- K = 1000 clusters

	Eigensolver		k-means	
Machines	Time (sec.)	Speedup	Time (sec.)	Speedup
1	—	—	—	—
2	8.074×10^4	2.00	3.609×10^4	2.00
4	4.427×10^4	3.65	1.806×10^4	4.00
8	2.184×10^4	7.39	8.469×10^3	8.52
16	9.867×10^3	16.37	4.620×10^3	15.62
32	4.886×10^3	33.05	2.021×10^3	35.72
64	4.067×10^3	39.71	1.433×10^3	50.37
128	3.471×10^3	46.52	1.090×10^3	66.22
256	4.021×10^3	40.16	1.077×10^3	67.02

- Achiever linear speedup when using 32 machines, after that, sub-linear speedup because of increasing communication and sync time

Sparsification vs. Sampling

	Sparsification	Nystrom, random sampling
Information	Full n x n similarity scores	None
Pre-processing Complexity (bottleneck)	$O(n^2)$	$O(nl)$, $l \ll n$
Effectiveness	Good	Not bad (Jitendra M., PAMI)

Outline

- Emerging Applications
  - Social networks
  - Personalized Information retrieval
- Key Subroutines
  - Clustering
  - Frequent Itemset Mining (FIM)
  - Combinational Collaborative Filtering
    - with PLSA
    - with LDA
  - Support Vector Machines

Collaborative Filtering

Given a matrix that "*encodes*" data

Many applications (collaborative filtering):

- User – Community
- User – User
- Ads – User
- Ads – Community
- etc.

Communities

Users

	2		1			4				5	
	5		4				?		1		3
		3		5			2				
4			?			5		3		?	
		4		1	3				5		
			2				1	?			4
	1					5		5		4	
		2		?	5		?		4		
	3		3		1		5		2		1
	3				1			2		3	
	4			5	1			3			
		3				3	?			5	
2	?		1		1						
		5			2	?		4		4	
	1		3		1	5		4		5	
1		2			4				5	?	

FIM-based Recommendation

"Growing" the knowledge-base

- $\{\text{BMW}, \text{Volkswagen}\} \rightarrow \{\text{BMW}, \text{Volkswagen}, \text{Volvo}, \text{Benz}, \text{QQ}, \ldots\}$
- $\{\text{T72}, \text{iPhone}\} \rightarrow \{\text{Nokia}, \text{T72}, \text{Apple}, \text{iPhone}, \ldots\}$
- $\{\text{Java}, \text{C++}\} \rightarrow \{\text{Java}, \text{C++}, \text{Python}, \text{Perl}, \text{Javascript}, \ldots\}$

To grow the base, we need association rules

- An association rule: $a, b, c \longrightarrow d$
- A Bayesian interpretation: $P(d \mid a, b, c) = \frac{N(a,b,c,d)}{N(a,b,c)}$
- The key is to count the occurrences (*support*) of itemsets $N(\ldots)$

FIM Preliminaries

- Observation 1: If an item A is not frequent, any pattern contains A won't be frequent [R. Agrawal]

  ➔ use a threshold to eliminate infrequent items

  ~~{a}~~ → ~~{a,b}~~
- Observation 2: Patterns containing A are subsets of (or found from) transactions containing A [J. Han]

  ➔ divide-and-conquer: select transactions containing A to form a conditional database (CDB), and find patterns containing A from that conditional database

  {a, b}, {a, c}, {a} → {a, b, c}
- Observation 3: To prevent the same pattern from being found in multiple CDBs, all itemsets are sorted by the same manner (e.g., by descending support)

Preprocessing

	f: 4	
	c: 4	
	a: 3	
f a c d g i m p	b: 3	f c a m p
	m: 3	
a b c f l m o	p: 3	f c a b m
b f h j o	o: 2	f b
	d: 1	
b c k s p	e: 1	c b p
	g: 1	
a f c e l p m n	h: 1	f c a m p
	i: 1	
	k: 1	
	l : 1	
	n: 1	

- According to Observation 1, we count the support of each item by scanning the database, and eliminate those infrequent items from the transactions.
- According to Observation 3, we sort items in each transaction by the order of descending support value.

Parallel Projection

- According to Observation 2, we construct CDB of item A; then from this CDB, we find those patterns containing A
- How to construct the CDB of A?
  - If a transaction contains A, this transaction should appear in the CDB of A
  - Given a transaction {B, A, C}, it should appear in the CDB of A, the CDB of B, and the CDB of C
- However, this leads to *duplicates*
  - Suppose {B,A,C} is a frequent pattern, it will be found three times --- from the CDBs of A, B and C respectively
- Solution: using the order of items:
  - sort {B,A,C} by the order of items → <A,B,C>
  - Put <> into the CDB of A
  - Put <A> into the CDB of B
  - Put <A,B> into the CDB of C

Example of Projection

f c a m p	p: { f c a m / f c a m / c b }
f c a b m	m: { f c a / f c a / f c a b }
f b	b: { f c a / f / c }
c b p	a: { f c / f c / f c }
f c a m p	c: { f / f / f }

Example of Projection of a database into CDBs.
Left: sorted transactions;
Right: conditional databases of frequent items

Example of Projection

f c a m p	p: { f c a m / f c a m / c b }
f c a b m	m: { f c a / f c a / f c a b }
f b	b: { f c a / f / c }
c b p	a: { f c / f c / f c }
f c a m p	c: { f / f / f }

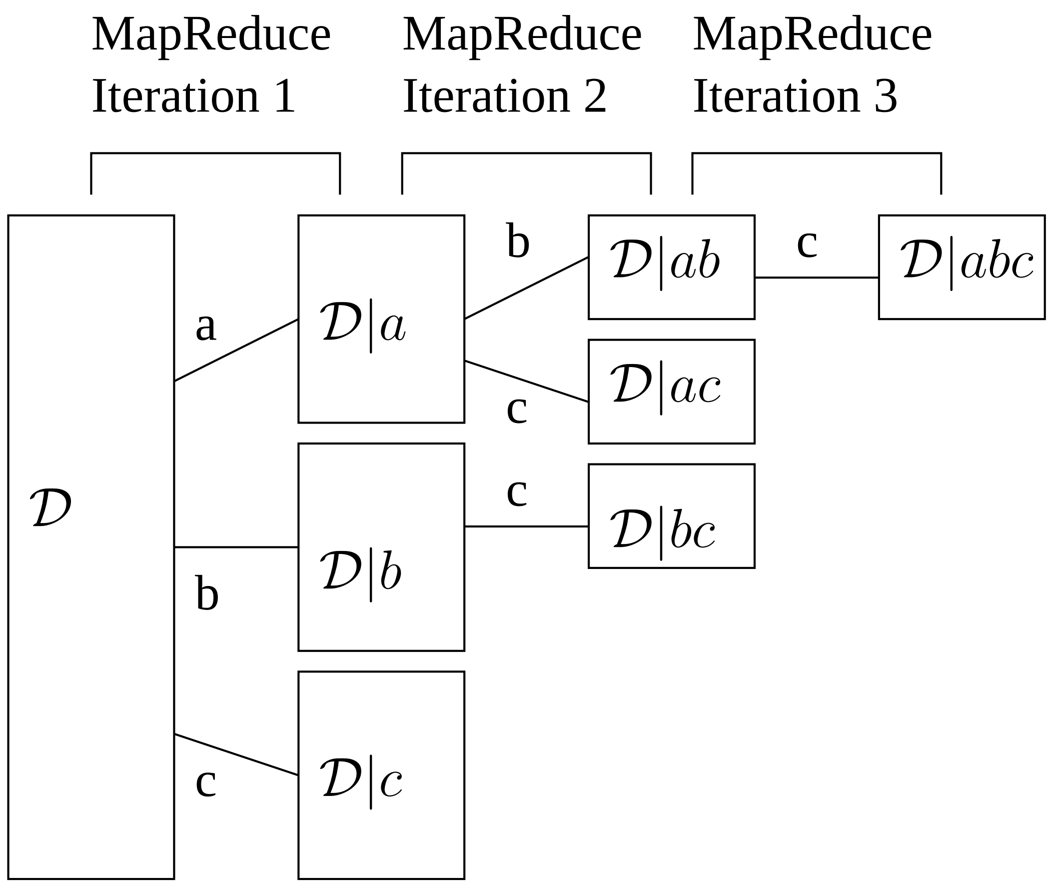
Example of Projection of a database into CDBs.
Left: sorted transactions;
Right: conditional databases of frequent items

Example of Projection

f c a m p	p: { f c a m / f c a m / c b }
f c a b m	m: { f c a / f c a / f c a b }
f b	b: { f c a / f / c }
c b p	a: { f c / f c / f c }
f c a m p	c: { f / f / f }

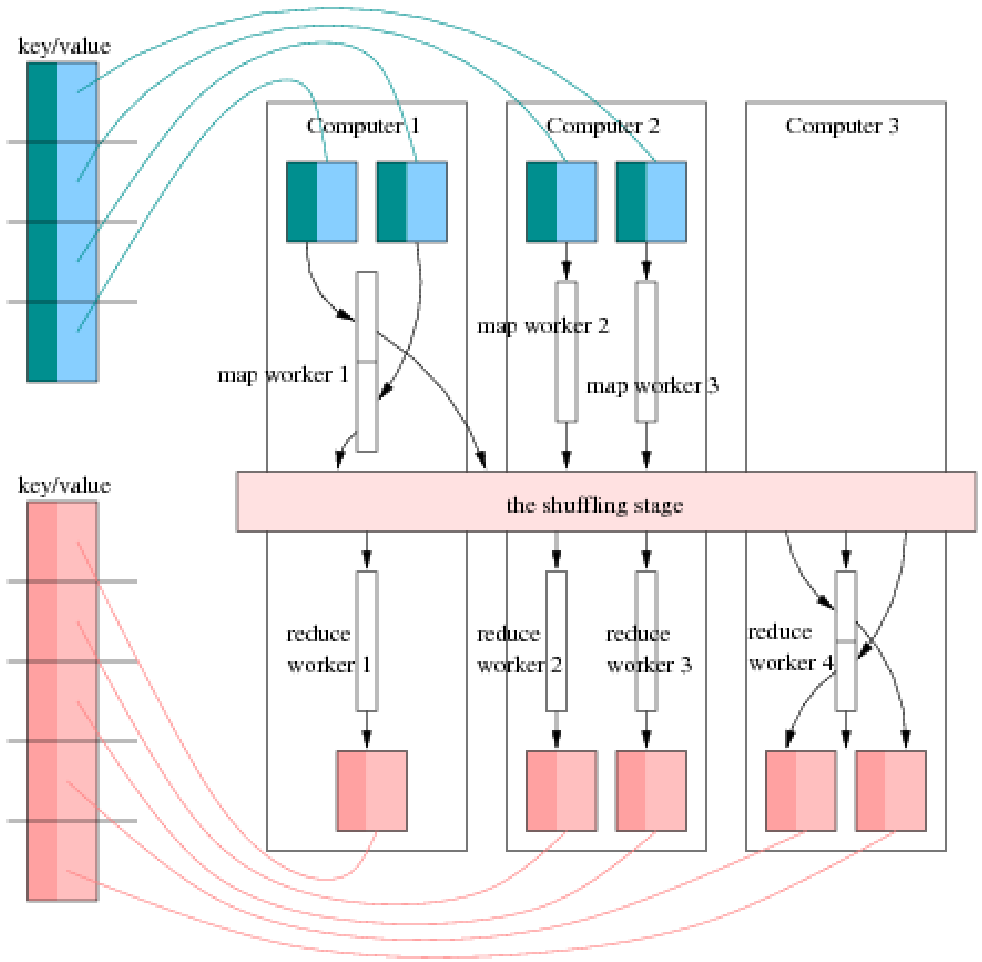
Example of Projection of a database into CDBs.
Left: sorted transactions;
Right: conditional databases of frequent items

Recursive Projections



- Recursive projection form a search tree
- Each node is a CDB
- Using the order of items to prevent duplicated CDBs.
- Size(D|ac)=supp(ac)
- Each level of breath-first search of the tree can be done by a MapReduce iteration.
- Once a CDB is small enough to fit in memory, we can invoke FP-growth to mine this CDB, and no more growth of the sub-tree.

MapReduce



- Logically, the input transaction database is in one file, but physically distributed across many computers.
- Logically, the output pattern database is in one file, but physically distributed across many computers.

Projection using MapReduce

Map inputs (transactions) key="": value	Sorted transactions (with infrequent items eliminated)	Map outputs (conditional transactions) key: value
f a c d g i m p	f c a m p	p: f c a m m: f c a a: fc c: f
a b c f l m o	f c a b m	m: f c a b b: f c a a: f c c: f
b f h j o	f b	b: f
b c k s p	c b p	p: c b
a f c e l p m n	f c a m p	b: c p: f c a m m: f c a a: f c c: f

Reduce inputs (conditional databases) key: value	Reduce outputs (patterns and supports) key: value
p: { f c a m / f c a m / c b }	p : 3 p c : 3
m: { f c a / f c a / f c a b }	m f : 3 m c : 3 m a : 3 m f c : 3 m f a : 3 m c a : 3 m f c a : 3
b: { f c a / f / c }	b : 3
a: { f c / f c / f c }	a : 3 a f : 3 a c : 3 a f c : 3
c: { f / f / f }	c : 3 c f : 3

Outline

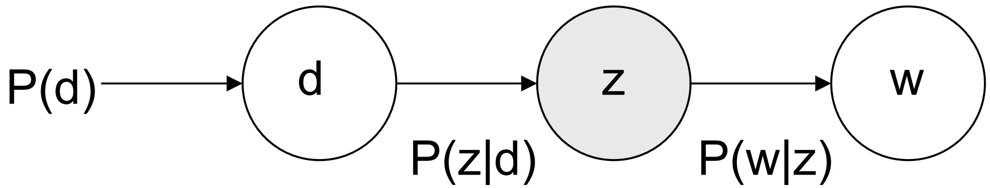
- Emerging Applications
  - Social networks
  - Personalized Information retrieval
- Key Subroutines
  - Clustering
  - Frequent Itemset Mining (FIM)
  - Combinational Collaborative Filtering
    - with PLSA
    - with LDA
  - Support Vector Machines

Notations

- Given a collection of co-occurrence data
  - Community: $C = \{c_1, c_2, \ldots, c_N\}$
  - User: $U = \{u_1, u_2, \ldots, u_M\}$
  - Description: $D = \{d_1, d_2, \ldots, d_V\}$
  - Latent aspect: $Z = \{z_1, z_2, \ldots, z_K\}$
- Models
  - Baseline models
    - Community-User (C-U) model
    - Community-Description (C-D) model
  - CCF: Combinational Collaborative Filtering
    - *Combines* both baseline models

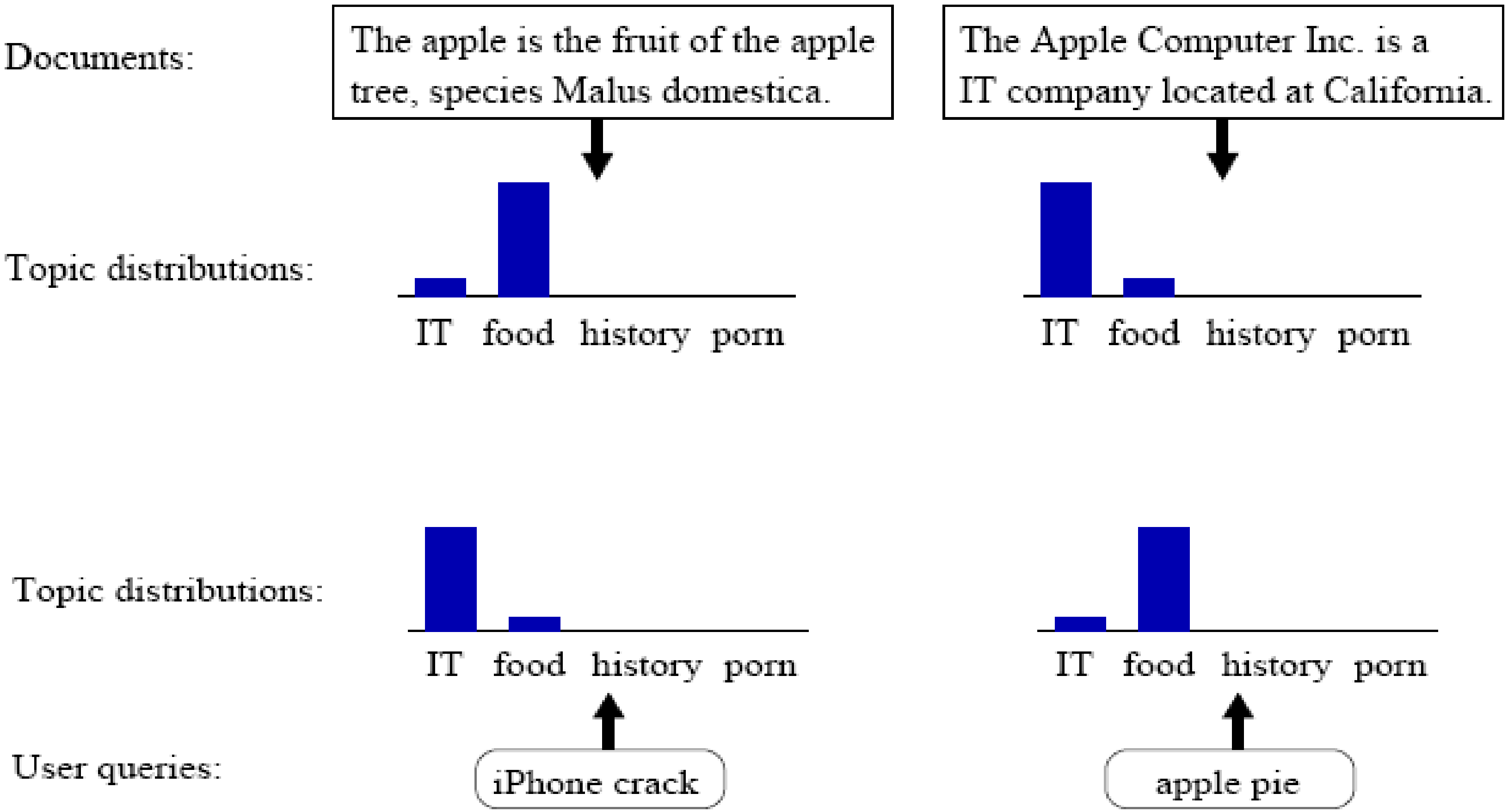
Probabilistic Latent Semantic Analysis (PLSA) [Hoffman 1999; Hoffman 2004]

- Document is viewed as a bag of words
- A *latent semantic layer* is constructed in between documents and words
- $P(w, d) = P(d)\,P(w|d) = P(d)\sum_z P(w|z)P(z|d)$



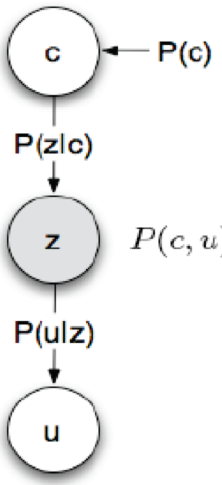
- Probability delivers explicit meaning
  - $P(w|w)$, $P(d|d)$, $P(d, w)$
- Model learning via EM or Gibbs sampling

Example of Latent Analysis



Baseline Models

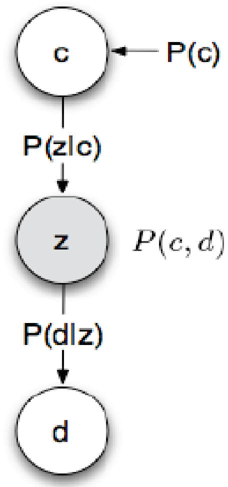
Community-User (C-U) model



$$P(c,u) = P(c)\sum_z P(u|z)P(z|c)$$

- Community is viewed as a bag of users
- ***c*** and ***u*** are rendered conditionally independent by introducing ***z***
- Generative process, for each user *u*
  1. A community *c* is chosen uniformly
  2. A topic *z* is selected from P(z|c)
  3. A user *u* is generated from P(u|z)

Community-Description (C-D) model

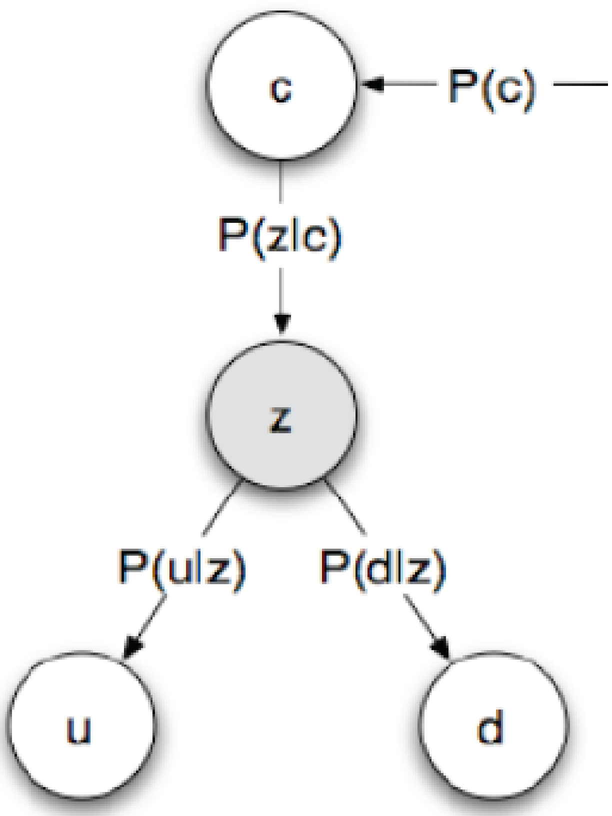


$$P(c,d) = P(c)\sum_z P(d|z)P(z|c)$$

- Community is viewed as a bag of words
- ***c*** and ***d*** are rendered conditionally independent by introducing ***z***
- Generative process, for each word *d*
  1. A community *c* is chosen uniformly
  2. A topic *z* is selected from P(z|c)
  3. A word *d* is generated from P(d|z)

CCF Model [Chen, et. al. KDD 08]

Combinational Collaborative Filtering (CCF) model



- CCF *combines* both baseline models
- A community is viewed as
  - a bag of users AND a bag of words
- By adding C-U, CCF can perform personalized recommendation which C-D alone *cannot*
- By adding C-D, CCF can perform better personalized recommendation than C-U alone, which may suffer from sparsity
- Things CCF can do that C-U and C-D cannot
  - P(d|u), relate user to word
  - Useful for user targeting ads

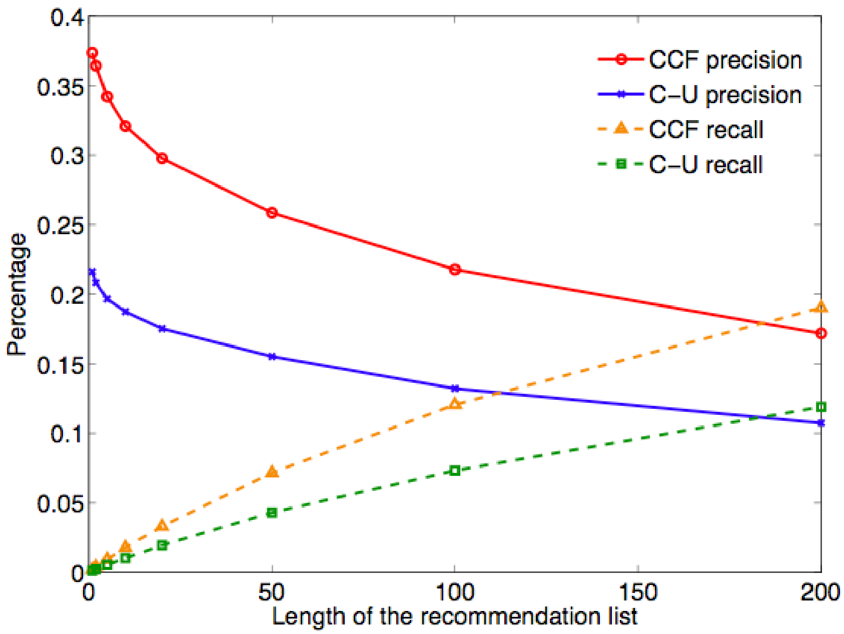
Empirical Study

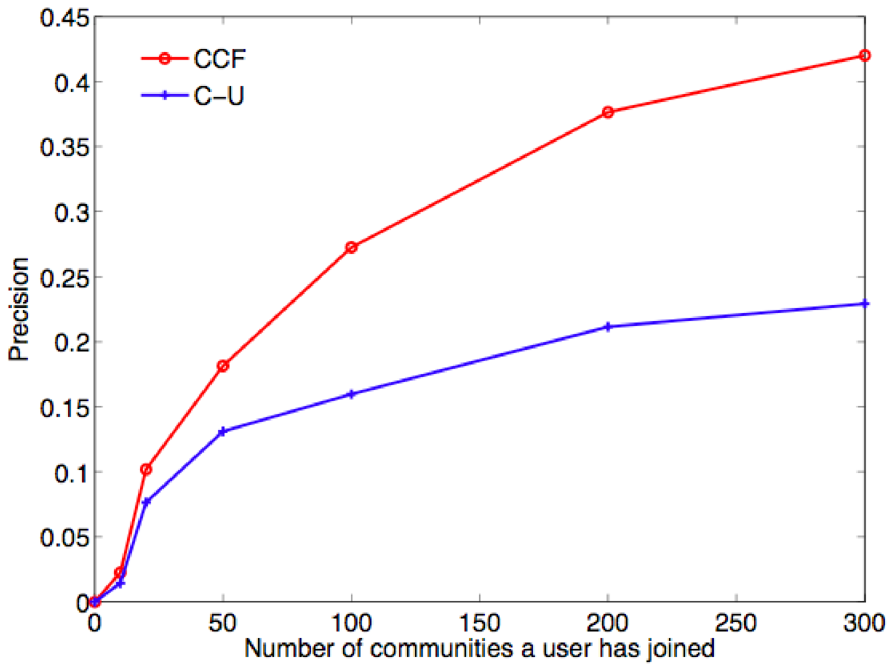
- Orkut Dataset
  - Collected in July, 2007
  - Two types of data were extracted
    - Community-user, community-description
  - 312,385 users
  - 109,987 communities
- Machine farm
  - Up to 200 machines in Google datacenters
  - Each machine is configured with:
    - A CPU faster than 2GHz
    - Memory larger than 4GBytes
- Evaluations
  - Community recommendation
  - Speedup

Community Recommendation

- Evaluation Method
  - Leave-one-out: randomly delete one community for each user
  - Whether a removed community can be recovered
- Evaluation metric
  - Precision and Recall

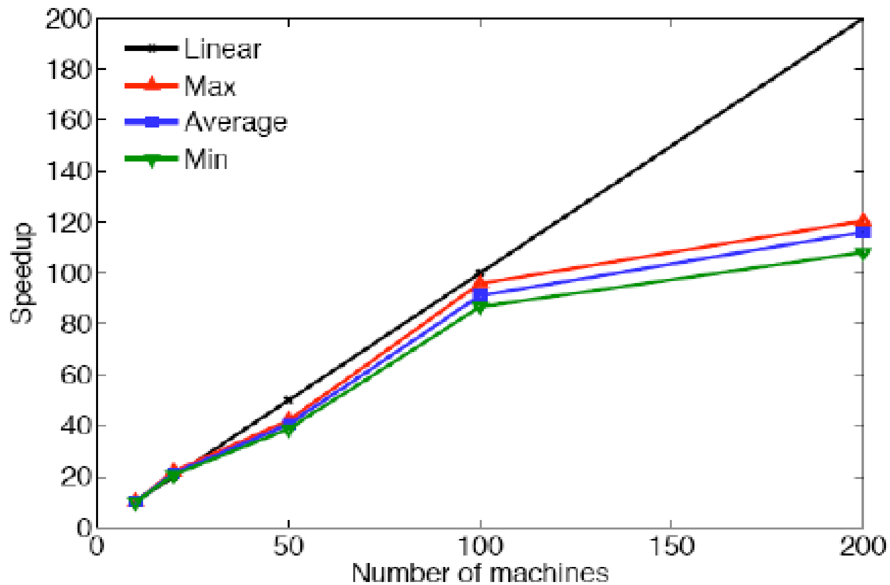
Results





- CCF outperforms C-U
- The more information, the higher accuracy

Gibbs Sampling MapRedue Speedup

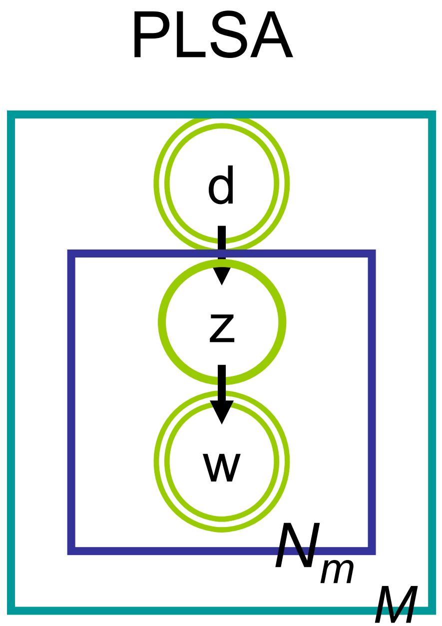


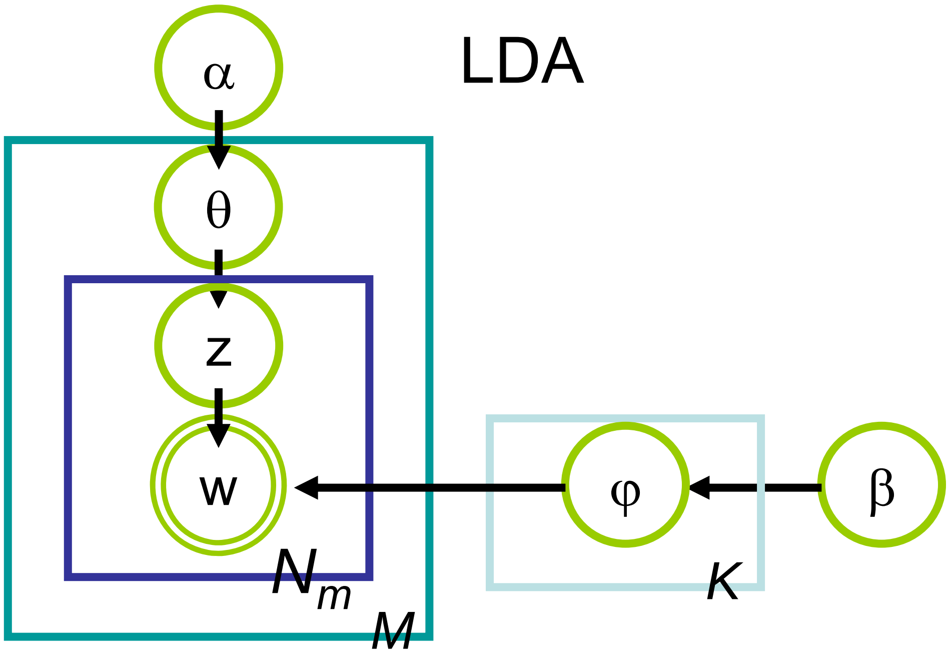
Machines	Time (sec.)	Speedup
10	9,233	10
20	4,326	21.3
50	2,280	40.5
100	1,014	91.1
200	796	116

- The Orkut dataset enjoys a linear speedup when the number of machines is up to 100
- Reduces the training time from one day to less than 14 minutes
- But, what makes the speedup slow down after 100 machines?

Extensions

- Expand CCF to incorporate more types of information
- Replace PLSA with LDA





…Extensions

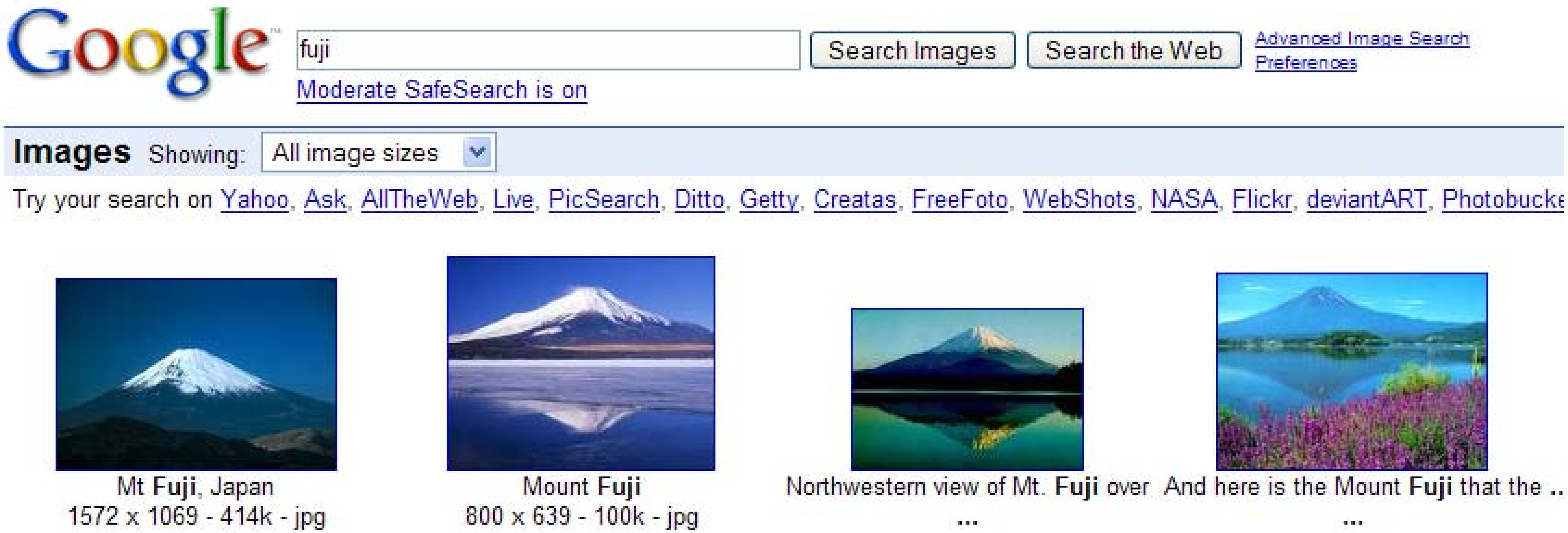
- Fusing more information sources
- Considering time dimension
- Incremental learning
- Topic hierarchy
- Etc.

Outline

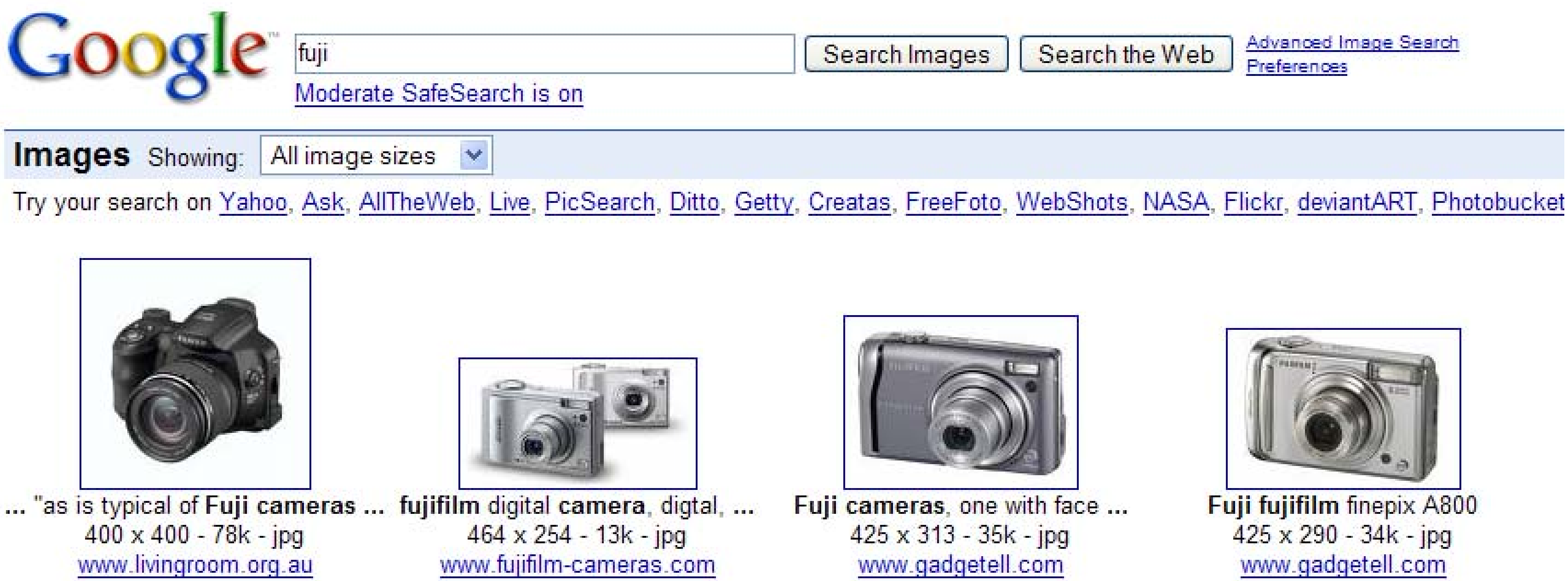
- Emerging Applications
  - Social networks
  - Personalized Information retrieval
- Key Subroutines
  - Clustering
  - Frequent Itemset Mining (FIM)
  - Combinational Collaborative Filtering
    - with PLSA
    - with LDA
  - Support Vector Machines

Personalized Search Example

- Infer relevance through social networks
- Query “fuji” can return
  - Fuji mountain
  - Fuji apples
  - Fuji cameras


Google
fuji
Search Images
Search the Web
Advanced Image Search
Preferences
Moderate SafeSearch is on
Images Showing: All image sizes
Try your search on Yahoo, Ask, AllTheWeb, Live, PicSearch, Ditto, Getty, Creatas, FreeFoto, WebShots, NASA, Flickr, deviantART, Photobucke
Mt Fuji, Japan
1572 x 1069 - 414k - jpg
Mount Fuji
800 x 639 - 100k - jpg
Northwestern view of Mt. Fuji over ...
And here is the Mount Fuji that the .. ...

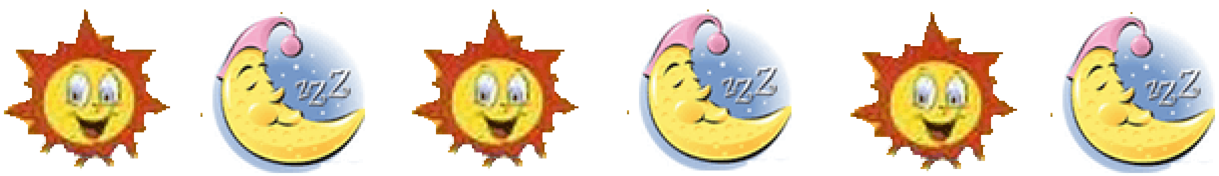

Google
fuji
Search Images
Search the Web
Advanced Image Search
Preferences
Moderate SafeSearch is on
Images Showing: All image sizes
Try your search on Yahoo, Ask, AllTheWeb, Live, PicSearch, Ditto, Getty, Creatas, FreeFoto, WebShots, NASA, Flickr, deviantART, Photobuck
(Apples, Fuji) Fuji apples are an ...
765 x 792 - 37k - jpg
www.all-creatures.org
fuji apple
300 x 294 - 17k - jpg
www.wisegeek.com
Organic - Apples, Fuji
375 x 375 - 67k - jpg
www.cleanfoodconnection.com
fuji apple Manufacturer
800 x 600 - 81k - jpg
www.supplierlist.com


Google
fuji
Search Images
Search the Web
Advanced Image Search
Preferences
Moderate SafeSearch is on
Images Showing: All image sizes
Try your search on Yahoo, Ask, AllTheWeb, Live, PicSearch, Ditto, Getty, Creatas, FreeFoto, WebShots, NASA, Flickr, deviantART, Photobucket
... "as is typical of Fuji cameras ...
400 x 400 - 78k - jpg
www.livingroom.org.au
fujifilm digital camera, digtal, ...
464 x 254 - 13k - jpg
www.fujifilm-cameras.com
Fuji cameras, one with face ...
425 x 313 - 35k - jpg
www.gadgetell.com
Fuji fujifilm finepix A800
425 x 290 - 34k - jpg
www.gadgetell.com

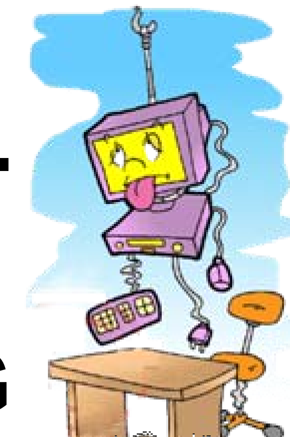

Mozilla Firefox
File Edit View History Bookmarks Tools Help
我的主页 资料 朋友 来吧 帖子 影集 日记 礼物 New! 评价 留言簿
我的朋友圈 | 我的朋友
上一步
回到自己
jifilm digital camera, digtal, ...
464 x 254 - 13k - jpg
www.fujifilm-cameras.com
Fuji cameras, one with face ...
425 x 313 - 35k - jpg
www.gadgetell.com
Fuji Camera
450 x 333 - 440k - bmp
emergencygadget.com
cheap fuji digital camera
400 x 318 - 14k - jpg
cheap-digital-camera.com.au
Google
邀请朋友加入来吧
邮件
姓名
发送邀请
我要群发邀请»
看看我的朋友在哪
Transferring data from laiba.tianya.cn...
start
2 Windows Explorer
Microsoft PowerPoi...
http://laiba.tianya....
9:30 PM

SVM Bottlenecks

Time consuming – 1M dataset, 8 days



... ...



Memory consuming – 1M dataset, 10G

Matrix Factorization Alternatives

Factorization	Cost
QR	$O(\frac{4}{3}n^3)$
LU	$O(\frac{2}{3}n^3)$
Cholesky	$O(\frac{1}{3}n^3 + 2n^2)$
LDL^T	$O(\frac{1}{3}n^3)$
Incomplete Cholesky	$O(p^2 n)$
Kronecker	$O(2n^2)$

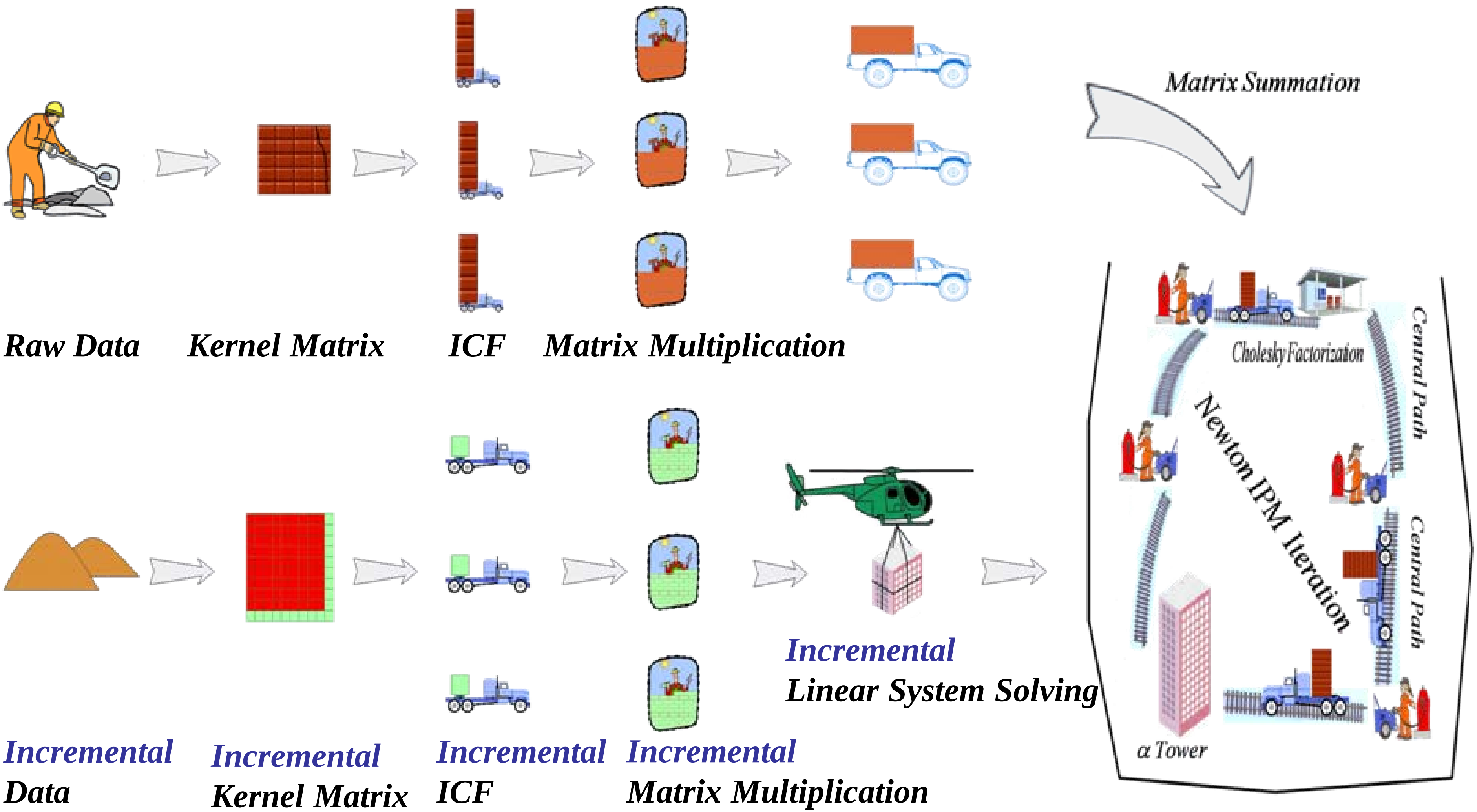
exact

approximate

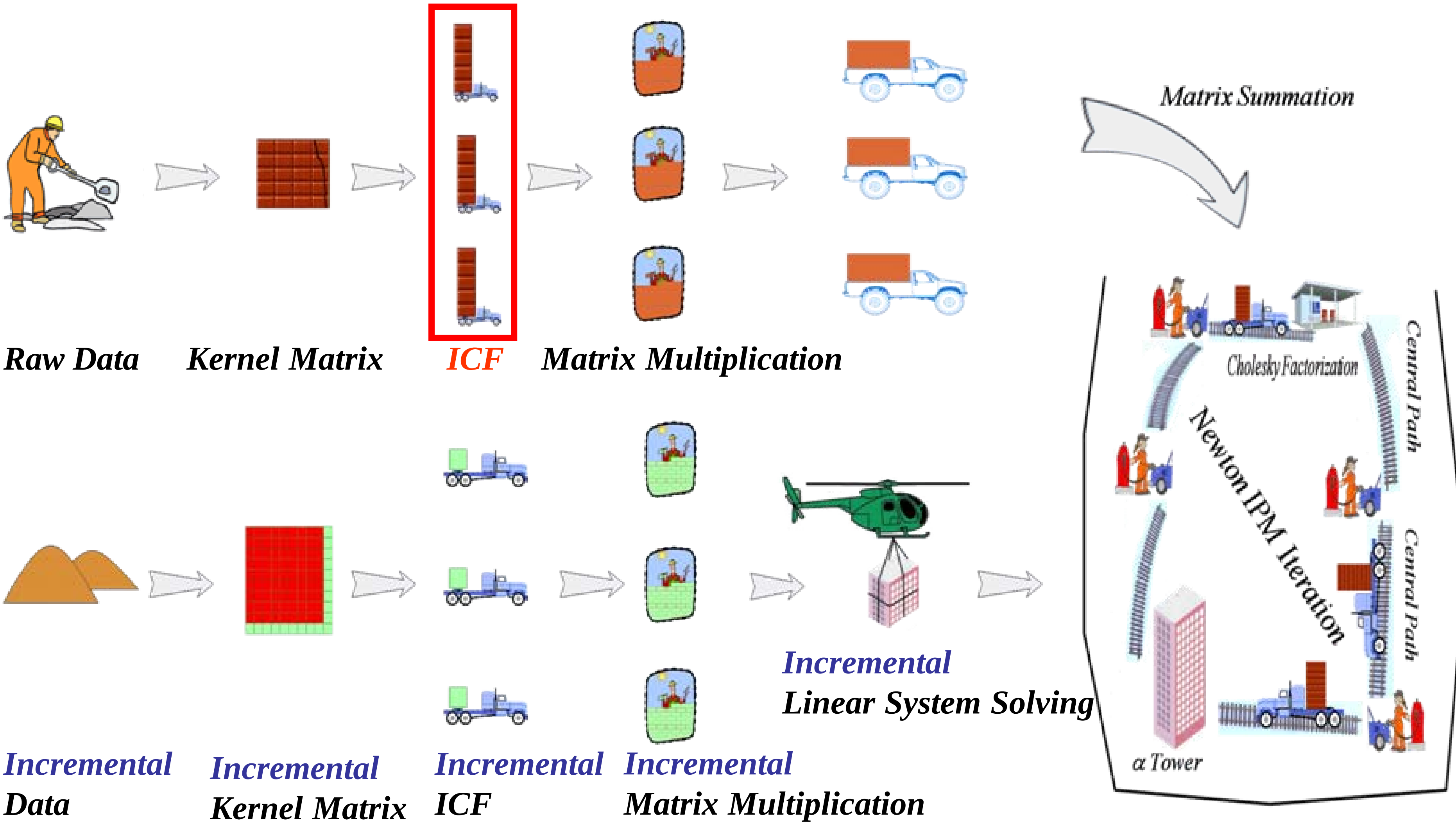
PSVM [E. Chang, et al, NIPS 07]

- Column-based ICF
  - Slower than row-based on single machine
  - Parallelizable on multiple machines
- Changing IPM computation order to achieve parallelization

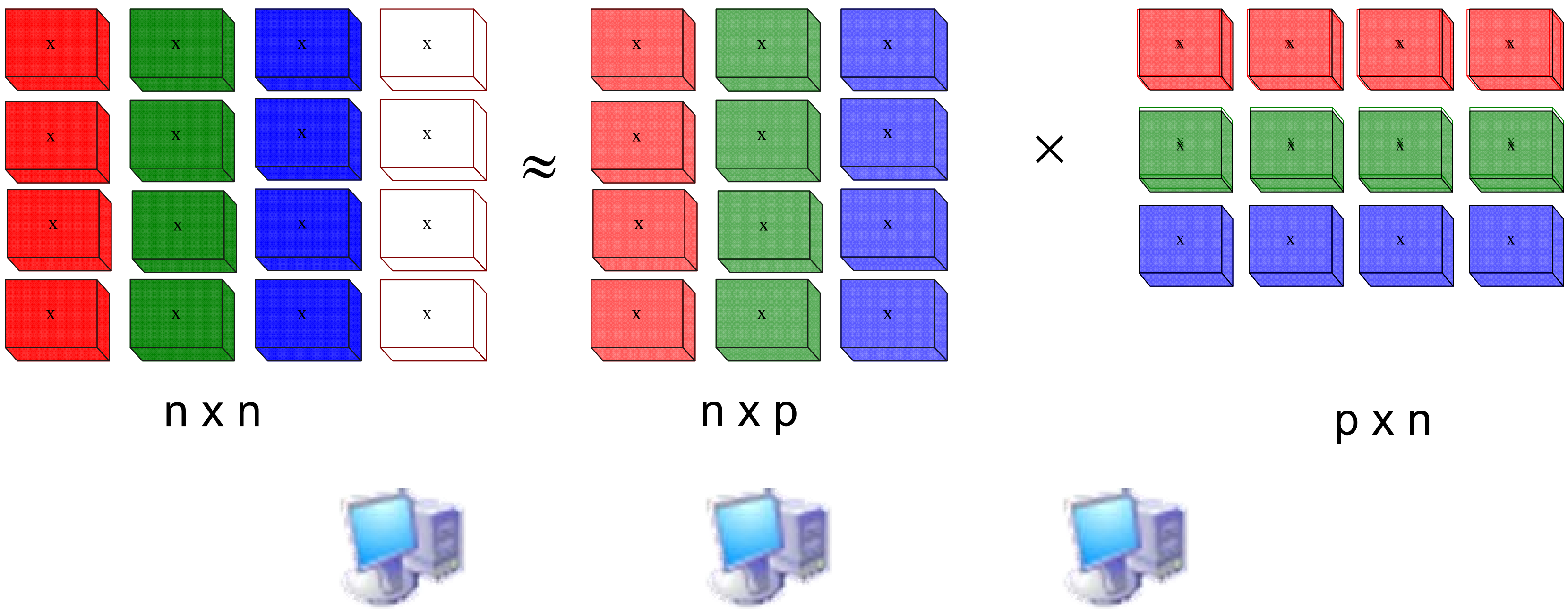
Parallelized and Incremental SVM



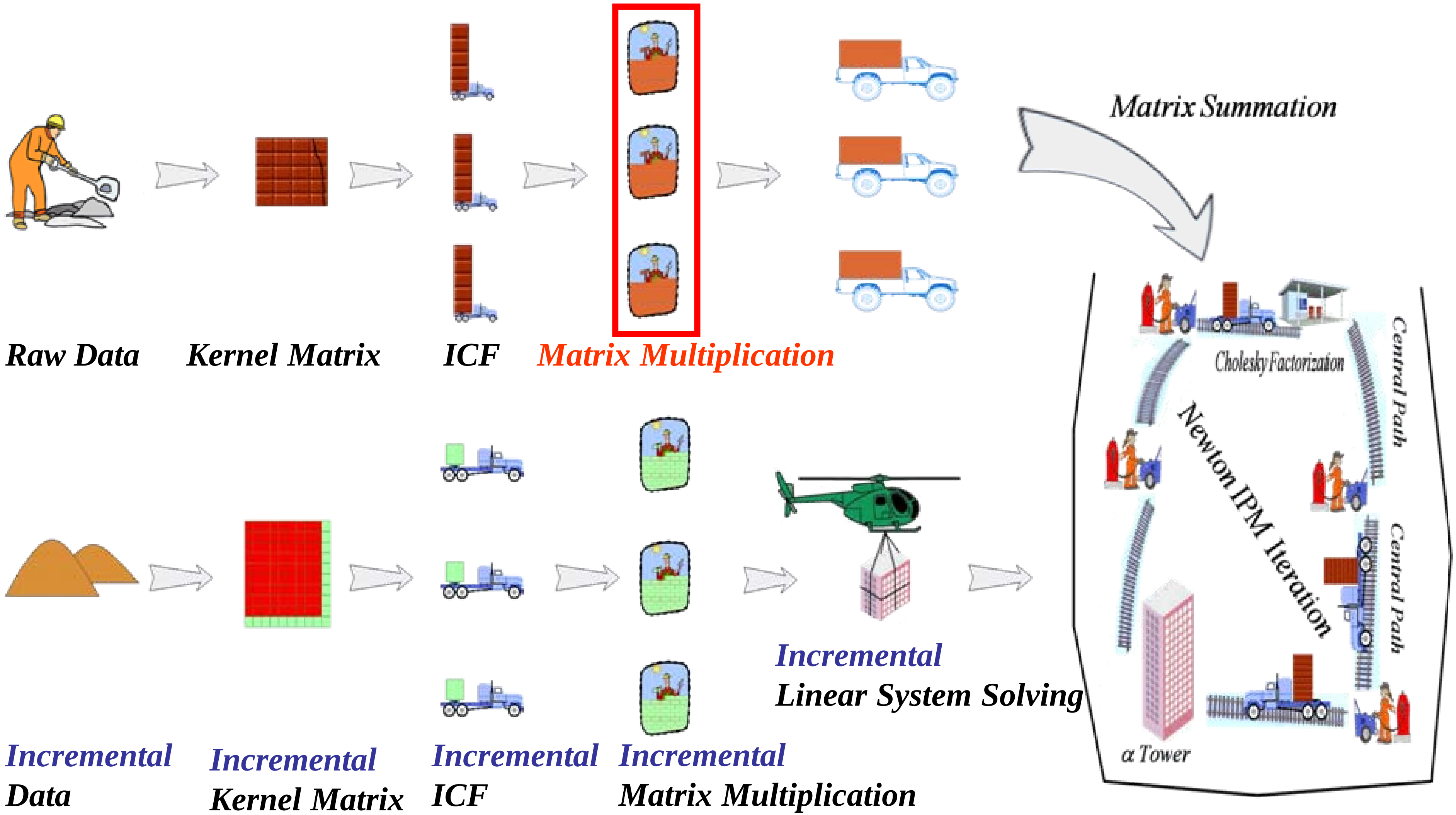
Parallelized and Incremental SVM



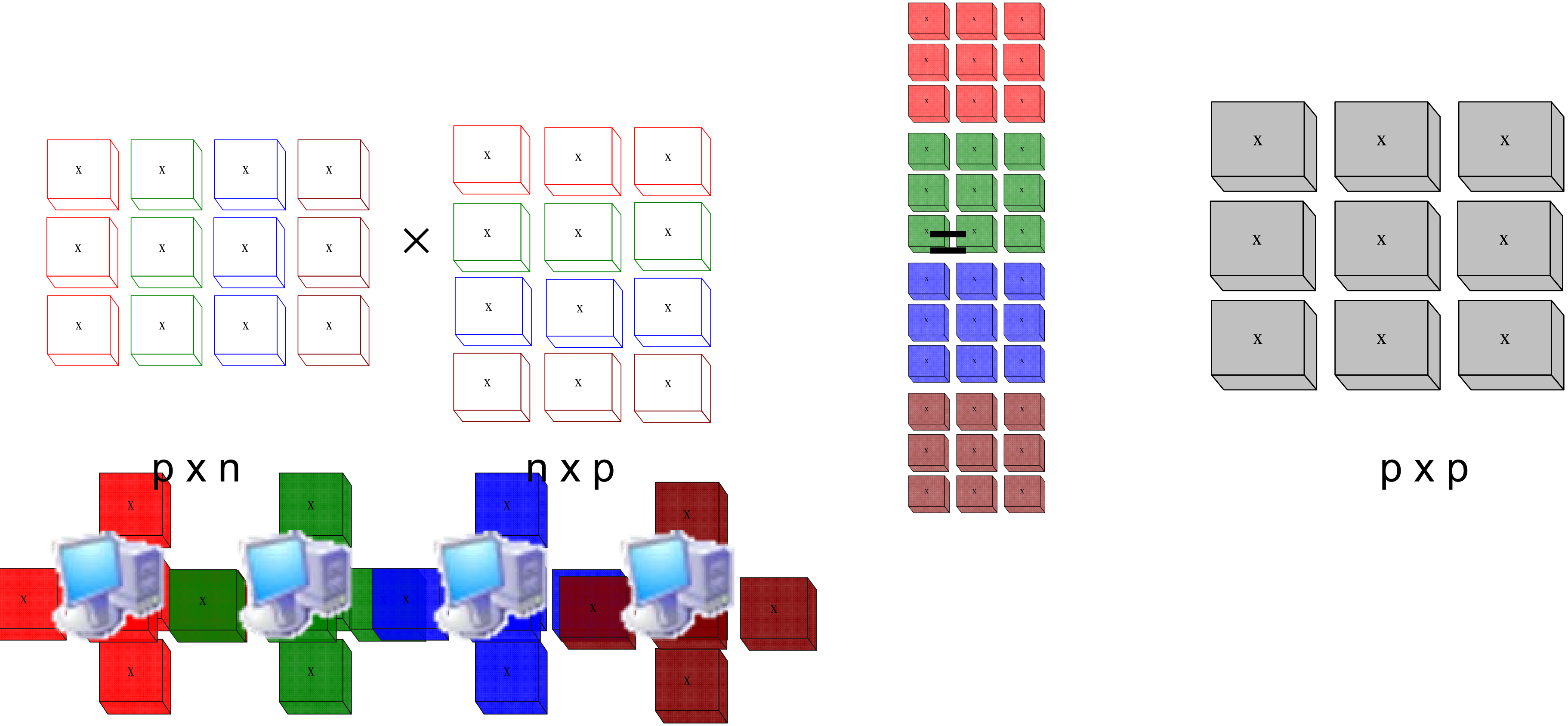
Incomplete Cholesky Factorization (ICF)



Parallelized and Incremental SVM



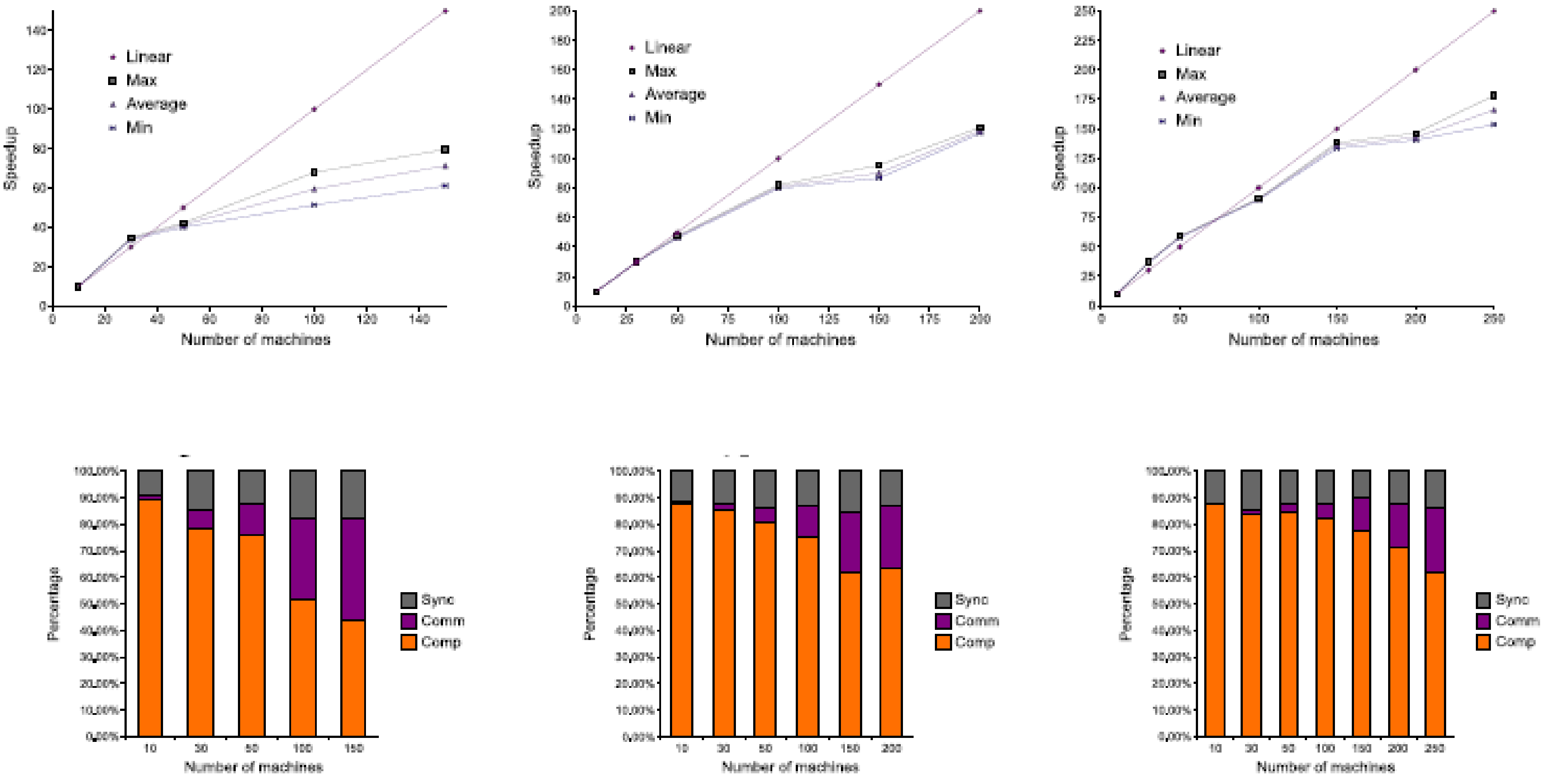
Matrix Product



Speedup

	Image (200k)			CoverType (500k)			RCV (800k)		
Machines	Time (s)		Speedup	Time (s)		Speedup	Time (s)		Speedup
10	1,958	(9)	10*	16,818	(442)	10*	45,135	(1373)	10*
30	572	(8)	34.2	5,591	(10)	30.1	12,289	(98)	36.7
50	473	(14)	41.4	3,598	(60)	46.8	7,695	(92)	58.7
100	330	(47)	59.4	2,082	(29)	80.8	4,992	(34)	90.4
150	274	(40)	71.4	1,865	(93)	90.2	3,313	(59)	136.3
200	294	(41)	66.7	1,416	(24)	118.7	3,163	(69)	142.7
250	397	(78)	49.4	1,405	(115)	119.7	2,719	(203)	166.0
500	814	(123)	24.1	1,655	(34)	101.6	2,671	(193)	169.0
LIBSVM	4,334	NA	NA	28,149	NA	NA	184,199	NA	NA

Overheads


Linear
Max
Average
Min
Speedup
Number of machines
Linear
Max
Average
Min
Speedup
Number of machines
Linear
Max
Average
Min
Speedup
Number of machines
Sync
Comm
Comp
Percentage
Number of machines
Sync
Comm
Comp
Percentage
Number of machines
Sync
Comm
Comp
Percentage
Number of machines

Summary

- Have parallelized key subroutines for mining massive data sets
  - Spectral Clustering
  - Frequent Itemset Mining
  - Combinational Collaborative Filtering
    - with PLSA
    - with LDA
  - Support Vector Machines
- Relevant papers
  - http://infolab.stanford.edu/~echang/
- Open Source PSVM
  - http://code.google.com/p/psvm/

Concluding Remarks

- Google distributed computing infrastructure is cost effective
- Timeliness can be as good as real-time
  - E.g., timely recommendation
- An expensive and parallelizable algorithm can be a better choice than a fast but non-parallelizable one
  - Column-based ICF over row-based in PSVM
  - t-NN over Nystrom in Spectral Clustering
- Relevant Information critical
  - Information fusion of CCF
  - Sparsification of Spectral Clustering

References

[1] Alexa internet. http://www.alexa.com/.
[2] D. M. Blei and M. I. Jordan. Variational methods for the dirichlet process. In Proc. of the 21st international conference on Machine learning, pages 373-380, 2004.
[3] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. Journal of Machine Learning Research, 3:993-1022, 2003.
[4] D. Cohn and H. Chang. Learning to probabilistically identify authoritative documents. In Proc. of the Seventeenth International Conference on Machine Learning, pages 167-174, 2000.
[5] D. Cohn and T. Hofmann. The missing link - a probabilistic model of document content and hypertext connectivity. In Advances in Neural Information Processing Systems 13, pages 430-436, 2001.
[6] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. Journal of the American Society of Information Science, 41(6):391-407, 1990.
[7] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society. Series B (Methodological), 39(1):1-38, 1977.
[8] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. IEEE Transactions on Pattern recognition and Machine Intelligence, 6:721-741, 1984.
[9] T. Hofmann. Probabilistic latent semantic indexing. In Proc. of Uncertainty in Arti□cial Intelligence, pages 289-296, 1999.
[10] T. Hofmann. Latent semantic models for collaborative filtering. ACM Transactions on Information System, 22(1):89-115, 2004.
[11] A. McCallum, A. Corrada-Emmanuel, and X. Wang. The author-recipient-topic model for topic and role discovery in social networks: Experiments with enron and academic email. Technical report, Computer Science, University of Massachusetts Amherst, 2004.
[12] D. Newman, A. Asuncion, P. Smyth, and M. Welling. Distributed inference for latent dirichlet allocation. In Advances in Neural Information Processing Systems 20, 2007.
[13] M. Ramoni, P. Sebastiani, and P. Cohen. Bayesian clustering by dynamics. Machine Learning, 47(1):91-121, 2002.

References (cont.)

[14] R. Salakhutdinov, A. Mnih, and G. Hinton. Restricted boltzmann machines for collaborative filtering. In Proc. Of the 24th international conference on Machine learning, pages 791-798, 2007.

[15] E. Spertus, M. Sahami, and O. Buyukkokten. Evaluating similarity measures: a large-scale study in the orkut social network. In Proc. of the 11th ACM SIGKDD international conference on Knowledge discovery in data mining, pages 678-684, 2005.

[16] M. Steyvers, P. Smyth, M. Rosen-Zvi, and T. Griffiths. Probabilistic author-topic models for information discovery. In Proc. of the 10th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 306-315, 2004.

[17] A. Strehl and J. Ghosh. Cluster ensembles - a knowledge reuse framework for combining multiple partitions. Journal on Machine Learning Research (JMLR), 3:583-617, 2002.

[18] T. Zhang and V. S. Iyengar. Recommender systems using linear classifiers. Journal of Machine Learning Research, 2:313-334, 2002.

[19] S. Zhong and J. Ghosh. Generative model-based clustering of documents: a comparative study. Knowledge and Information Systems (KAIS), 8:374-384, 2005.

[20] L. Admic and E. Adar. How to search a social network. 2004

[21] T.L. Griffiths and M. Steyvers. Finding scientific topics. Proceedings of the National Academy of Sciences, pages 5228-5235, 2004.

[22] H. Kautz, B. Selman, and M. Shah. Referral Web: Combining social networks and collaborative filtering. Communitcations of the ACM, 3:63-65, 1997.

[23] R. Agrawal, T. Imielnski, A. Swami. Mining association rules between sets of items in large databses. SIGMOD Rec., 22:207-116, 1993.

[24] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In Proceedings of the Fourteenth Conference on Uncertainty in Artifical Intelligence, 1998.

[25] M.Deshpande and G. Karypis. Item-based top-n recommendation algorithms. ACM Trans. Inf. Syst., 22(1):143-177, 2004.

References (cont.)

[26] B.M. Sarwar, G. Karypis, J.A. Konstan, and J. Reidl. Item-based collaborative filtering recommendation algorithms. In Proceedings of the 10th International World Wide Web Conference, pages 285-295, 2001.
[27] M.Deshpande and G. Karypis. Item-based top-n recommendation algorithms. ACM Trans. Inf. Syst., 22(1):143-177, 2004.
[28] B.M. Sarwar, G. Karypis, J.A. Konstan, and J. Reidl. Item-based collaborative filtering recommendation algorithms. In Proceedings of the 10th International World Wide Web Conference, pages 285-295, 2001.
[29] M. Brand. Fast online svd revisions for lightweight recommender systems. In Proceedings of the 3rd SIAM International Conference on Data Mining, 2003.
[30] D. Goldbberg, D. Nichols, B. Oki and D. Terry. Using collaborative filtering to weave an information tapestry. Communication of ACM 35, 12:61-70, 1992.
[31] P. Resnik, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: An open architecture for aollaborative filtering of netnews. In Proceedings of the ACM, Conference on Computer Supported Cooperative Work. Pages 175-186, 1994.
[32] J. Konstan, et al. Grouplens: Applying collaborative filtering to usenet news. Communication ofACM 40, 3:77-87, 1997.
[33] U. Shardanand and P. Maes. Social information filtering: Algorithms for automating “word of mouth”. In Proceedings of ACM CHI, 1:210-217, 1995.
[34] G. Kinden, B. Smith and J. York. Amazon.com recommendations: item-to-item collaborative filtering. IEEE Internet Computing, 7:76-80, 2003.
[35] T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. Machine Learning Journal 42, 1:177-196, 2001.
[36] T. Hofmann and J. Puzicha. Latent class models for collaborative filtering. In Proceedings of International Joint Conference in Artificial Intelligence, 1999.
[37] http://www.cs.carleton.edu/cs_comps/0607/recommend/recommender/collaborativefiltering.html
[38] E. Y. Chang, et. al., Parallelizing Support Vector Machines on Distributed Machines, NIPS, 2007.
[39] Wen-Yen Chen, Dong Zhang, and E. Y. Chang, Combinational Collaborative Filtering for personalized community recommendation, ACM KDD 2008.
[40] Y. Sun, W.-Y. Chen, H. Bai, C.-j. Lin, and E. Y. Chang, Parallel Spectral Clustering, ECML 2008.