Mining at scale with latent factor models for matrix completion

Fabio Petroni

SAPIENZA
UNIVERSITÀ DI ROMA

Recommender systems

Marco wants to watch a movie.

DVD rental store

A B C D E F G H I J K ... Z

A (1998)
A-ge-man (1990)
A.K.G. (2007)
A Nous Amours (1983)
... many pages later ...
Azumi (2003)

Recommender systems

Marco wants to watch a movie.

DVD rental store

A B C D E F G H I J K ... Z

A (1998)
A-ge-man (1990)
A.K.G. (2007)
A Nous Amours (1983)
... many pages later ...
Azumi (2003)

But there are so many movies!
Which ones will he like?

Collaborative filtering

- problem
  - set of users
  - set of items (movies, books, songs, ...)
  - feedback
    - explicit (ratings, ...)
    - implicit (purchase, click-through, ...)
- predict the preference of each user for each item
  - assumption: similar feedback $\leftrightarrow$ similar taste
- example (explicit feedback):

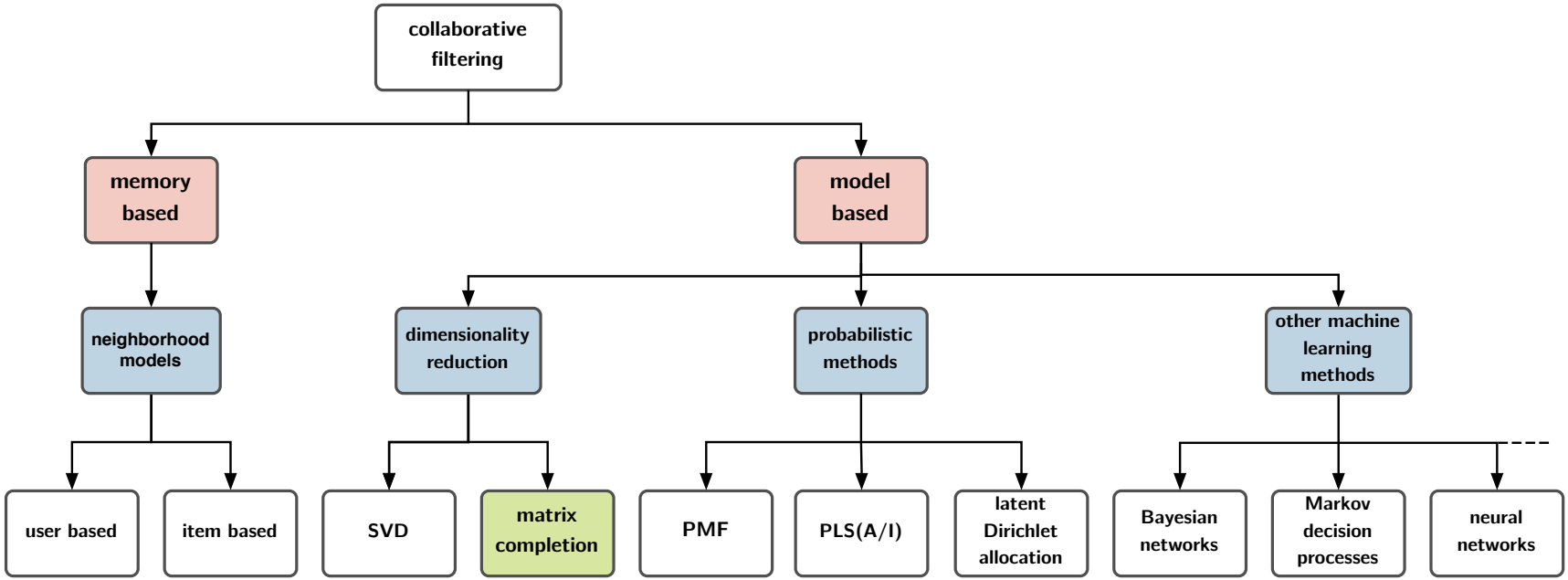
	Avatar	The Matrix	Up
Marco		4	2
Luca	3	2	
Anna	5		3

Collaborative filtering

- problem
  - set of users
  - set of items (movies, books, songs, ...)
  - feedback
    - explicit (ratings, ...)
    - implicit (purchase, click-through, ...)
- predict the preference of each user for each item
  - assumption: similar feedback $\leftrightarrow$ similar taste
- example (explicit feedback):

	Avatar	The Matrix	Up
Marco	?	4	2
Luca	3	2	?
Anna	5	?	3

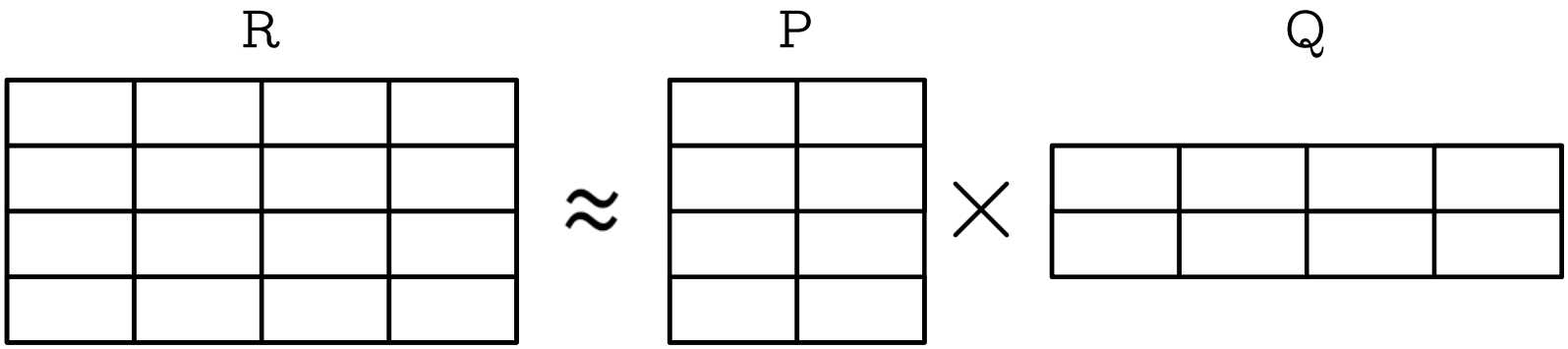
Collaborative filtering taxonomy



- matrix completion is currently considered the best approach
- advantages with respect to both scalability and accuracy

Matrix completion for collaborative filtering

- the completion is driven by a factorization

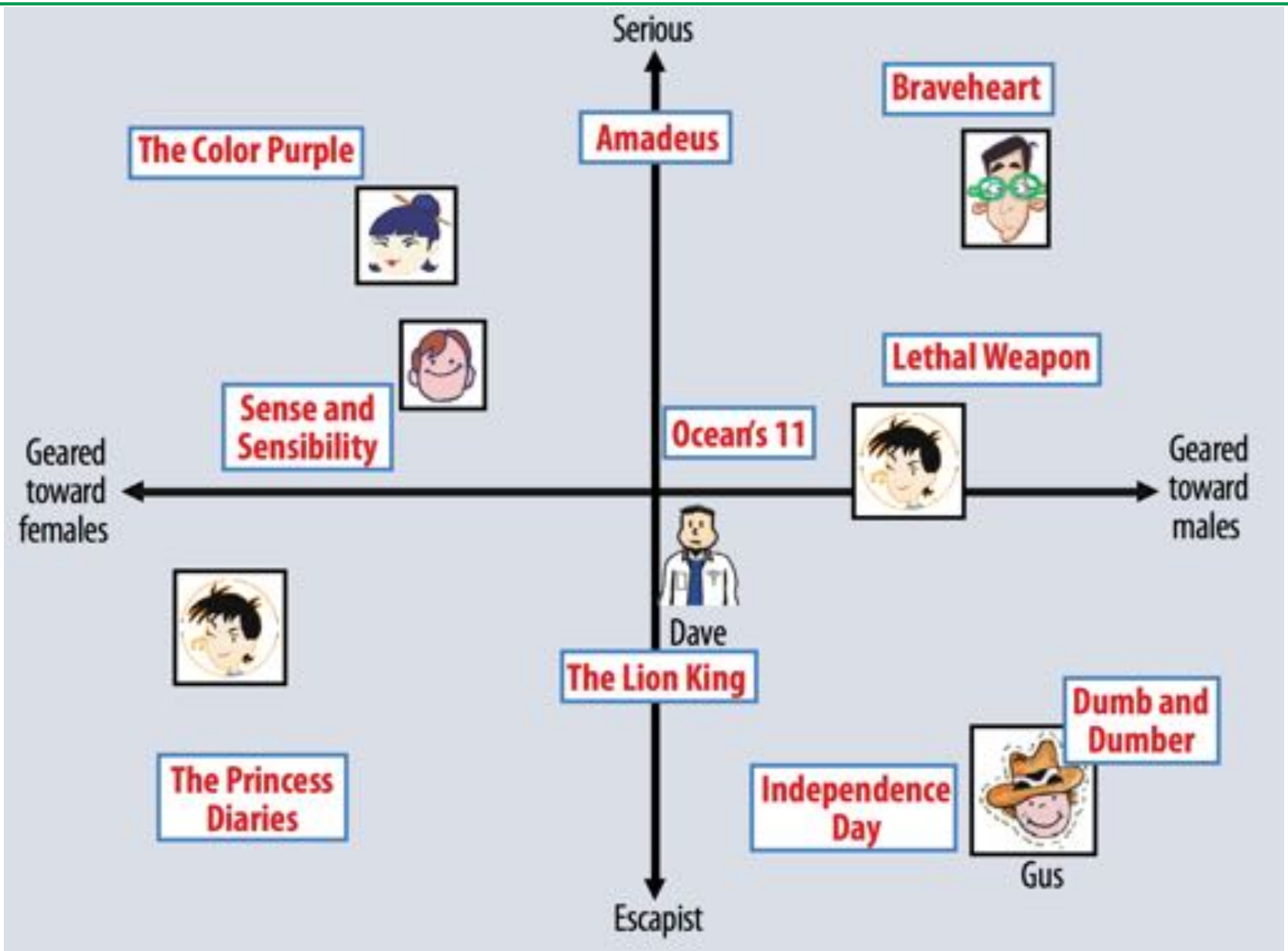


- associate a latent factor vector with each user and each item
- missing entries are estimated through the dot product

$$r_{ij} \approx p_i q_j$$

Latent factor models

(Koren et al., 2009)



Latent factor models - explicit feedback

- discover latent factors (d = 1)

	Avatar (2.24)	The Matrix (1.92)	Up (1.18)
Marco (1.98)	?	4 (3.8)	2 (2.3)
Luca (1.21)	3 (2.7)	2 (2.3)	?
Anna (2.30)	5 (5.2)	?	3 (2.7)

- optimization criterion: <u>minimize squared loss</u>

$$\underset{P,Q}{\text{minimize}} \sum_{(i,j)\in O} (r_{ij} - p_i q_j)^2$$

Latent factor models - explicit feedback

- discover latent factors (d = 1)

	Avatar (2.24)	The Matrix (1.92)	Up (1.18)
Marco (1.98)	**4.4**	4 (3.8)	2 (2.3)
Luca (1.21)	3 (2.7)	2 (2.3)	**1.4**
Anna (2.30)	5 (5.2)	**4.4**	3 (2.7)

- optimization criterion: <u>minimize squared loss</u>

$$\underset{P,Q}{\text{minimize}} \sum_{(i,j)\in O} (r_{ij} - p_i q_j)^2$$

Latent factor models - explicit feedback

- discover latent factors (d = 1)

	Avatar (2.24)	The Matrix (1.92)	Up (1.18)
Marco (1.98)	**4.4**	4 (3.8)	2 (2.3)
Luca (1.21)	3 (2.7)	2 (2.3)	**1.4**
Anna (2.30)	5 (5.2)	**4.4**	3 (2.7)

- optimization criterion: minimize squared loss

$$\underset{P,Q}{\text{minimize}} \sum_{(i,j)\in O} (r_{ij} - b_\mu - b_{u_i} - b_{x_j} - p_i q_j)^2 + \lambda(||P||_F + ||Q||_F)$$

Latent factor models - implicit feedback

- discover latent factors (d = 1)

	Avatar	The Matrix	Up
Marco	?	1	1
Luca	1	1	?
Anna	1	?	1

- the squared loss minimization criterion **is not effective**!

Latent factor models - implicit feedback

- discover latent factors (d = 1)

	Avatar (1)	The Matrix (1)	Up (1)
Marco (1)	1	1 (1)	1 (1)
Luca (1)	1 (1)	1 (1)	1
Anna (1)	1 (1)	1	1 (1)

- the squared loss minimization criterion **is not effective**!
- the system simply complete the matrix with all 1s

Bayesian personalized ranking

- problem related with ranking more than prediction
  - e.g., ranked list of items that the user might like the most
- the BPR criterion adopts a pairwise approach
  - predict whether item x_j is more probable to be liked than x_k
  - assumption: the user prefers items for which a feedback exists

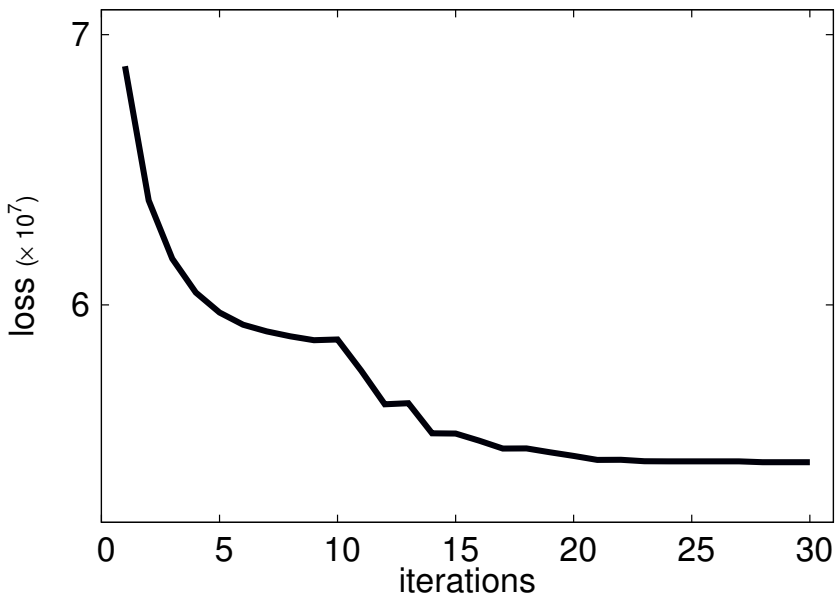
$$D_T := \{(u_i, x_j, x_k) | u_i \in U \wedge x_j, x_k \in I, j \neq k \wedge r_{ij} = 1 \wedge r_{ik} = ?\}$$

- user u_i is assumed to prefer item x_j over x_k

$$\underset{P,Q}{\text{maximize}} \sum_{(u_i,x_j,x_k) \in D_T} \left[\ln \sigma(p_i q_j - p_i q_k) \right]$$

Stochastic gradient descent

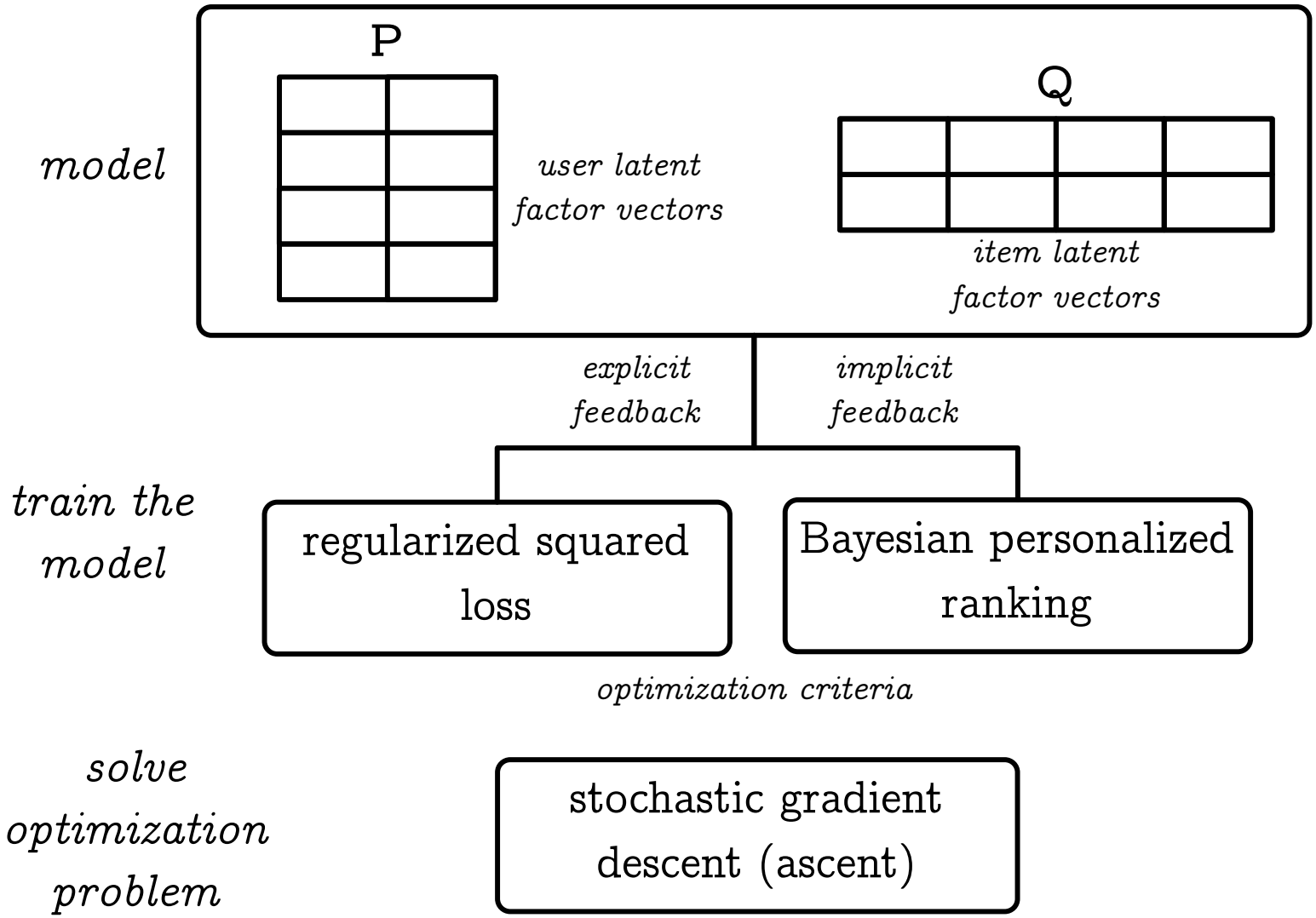
- parameters $\Theta = \{P, Q\}$
- find minimum Θ^* of loss function L, or maximum for BPR (ascent)
- pick a starting point Θ^0
- iteratively update current estimations for Θ



$$\Theta_{n+1} \leftarrow \Theta_n - \eta \frac{\partial L}{\partial \Theta}$$

- learning rate η
- an update for each given training point

Overview matrix completion



Challenges of matrix completion

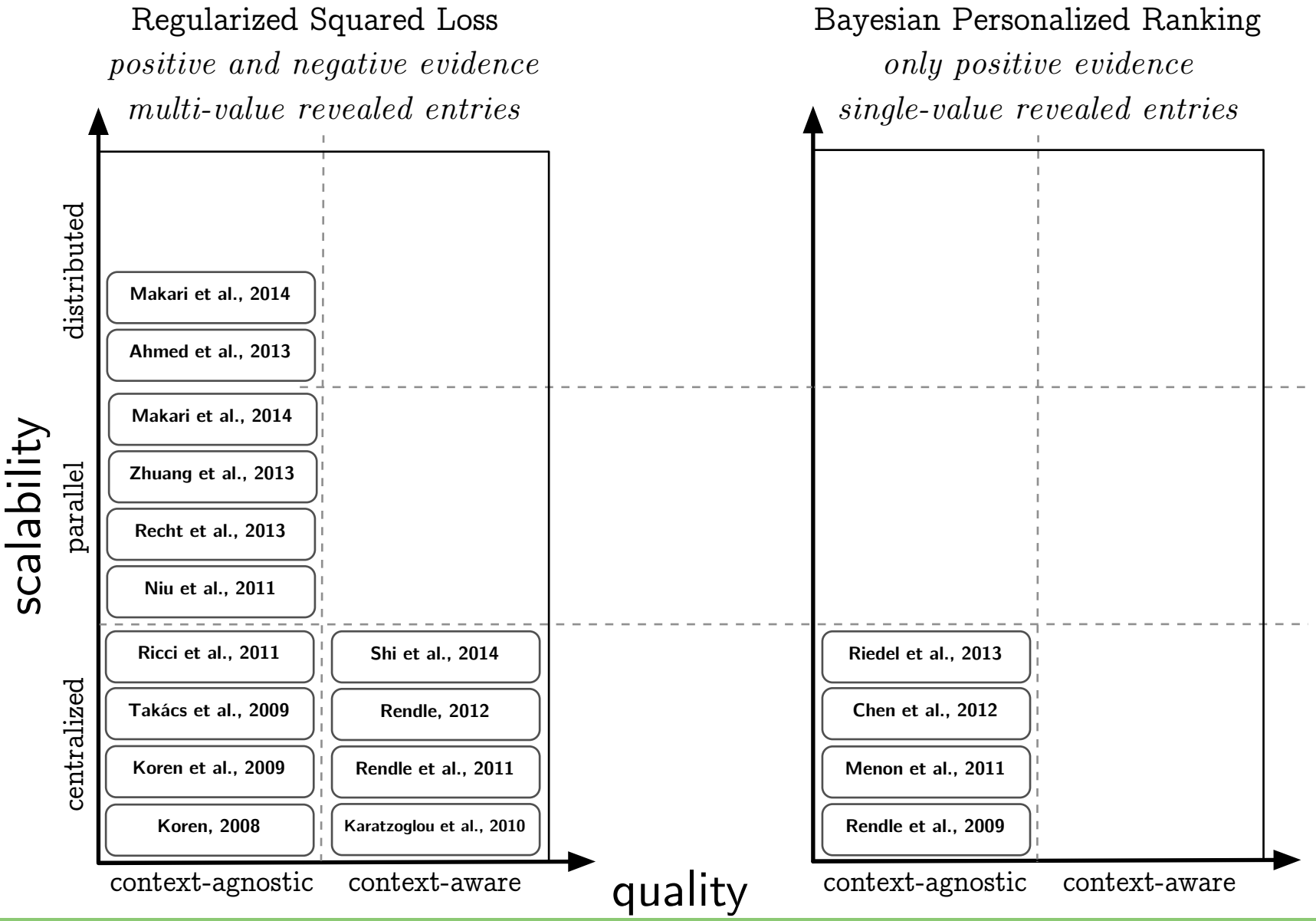
(1) **scalability**: handle large scale data

- 2B purchases on Amazon.com by 200M customers (2014)
- parallel and distributed approaches are essential!

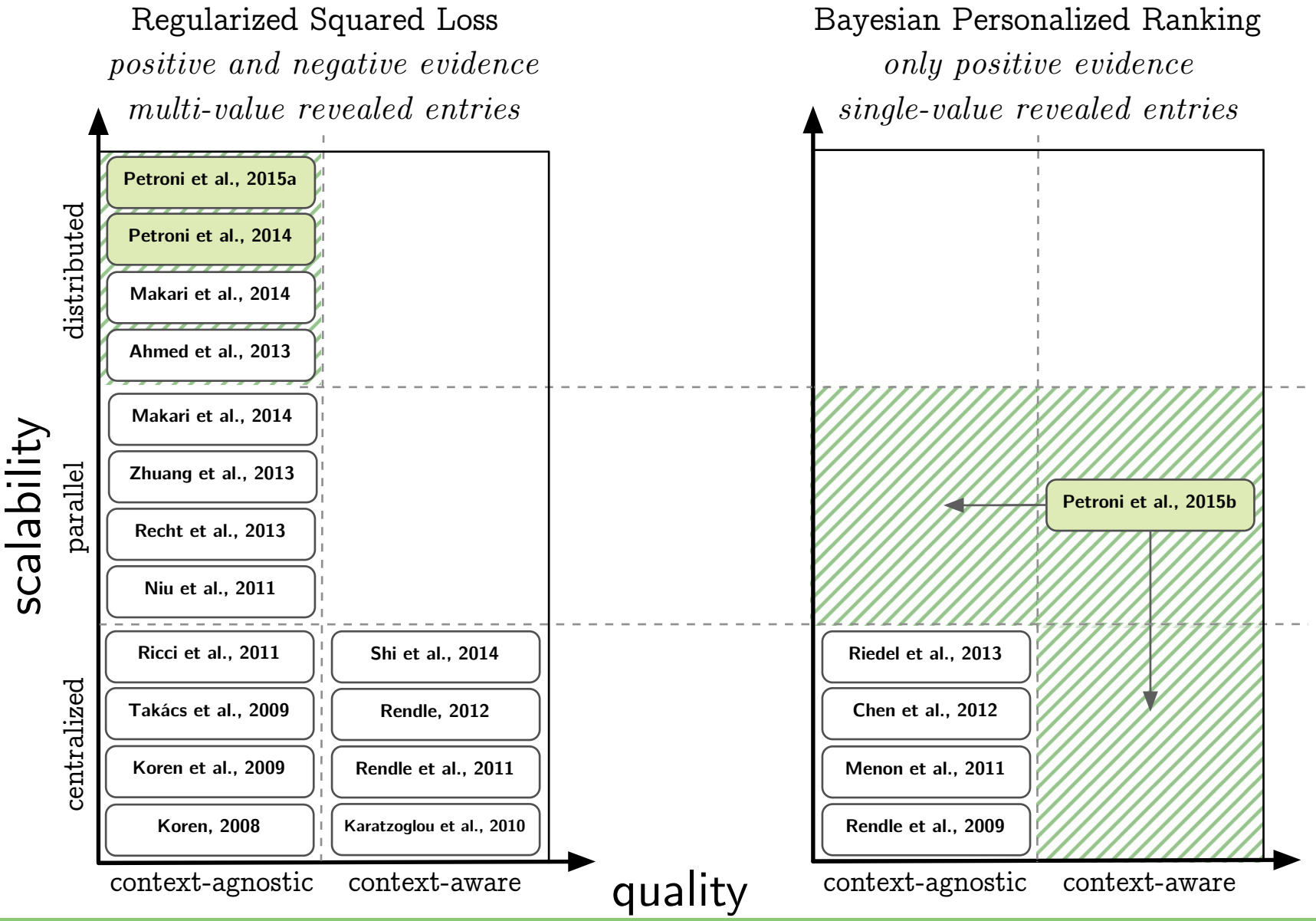
(2) **quality**: improve prediction performance

- Netflix awarded a 10% improvement with $1M (2006)
- the performance of matrix completion can be boosted by:
  - feeding the system with more data
  - integrating contextual information in the model

Overview - state-of-the-art



Overview - contributions

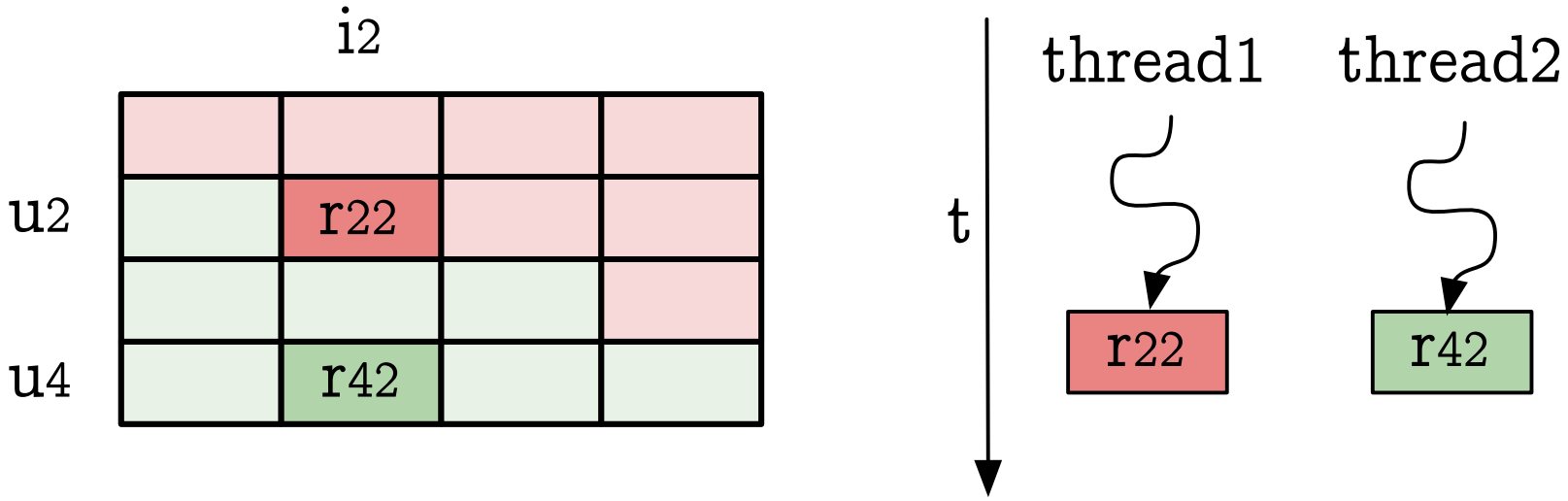


Contents

Contents

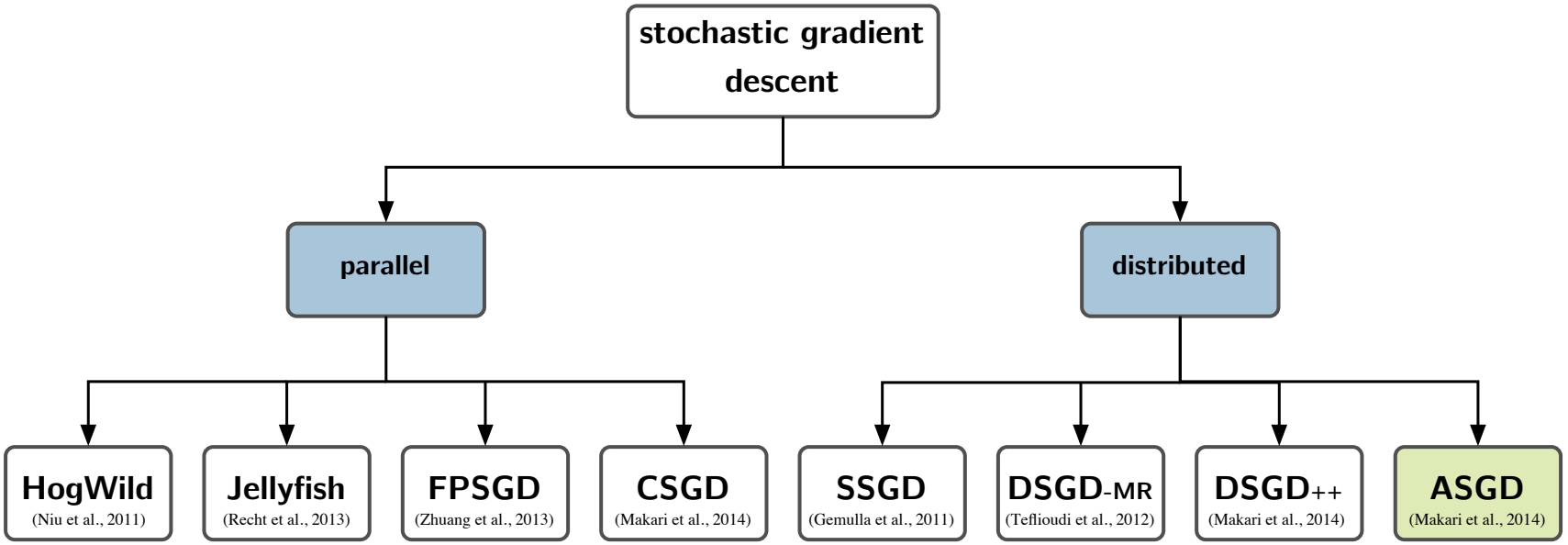
Problems of parallel and distributed SGD

- divide the training points (SGD updates) among threads
- SGD updates might depend on each other!



- both threads concurrently update the same latent vector
- lock-based approaches adversely affect concurrency

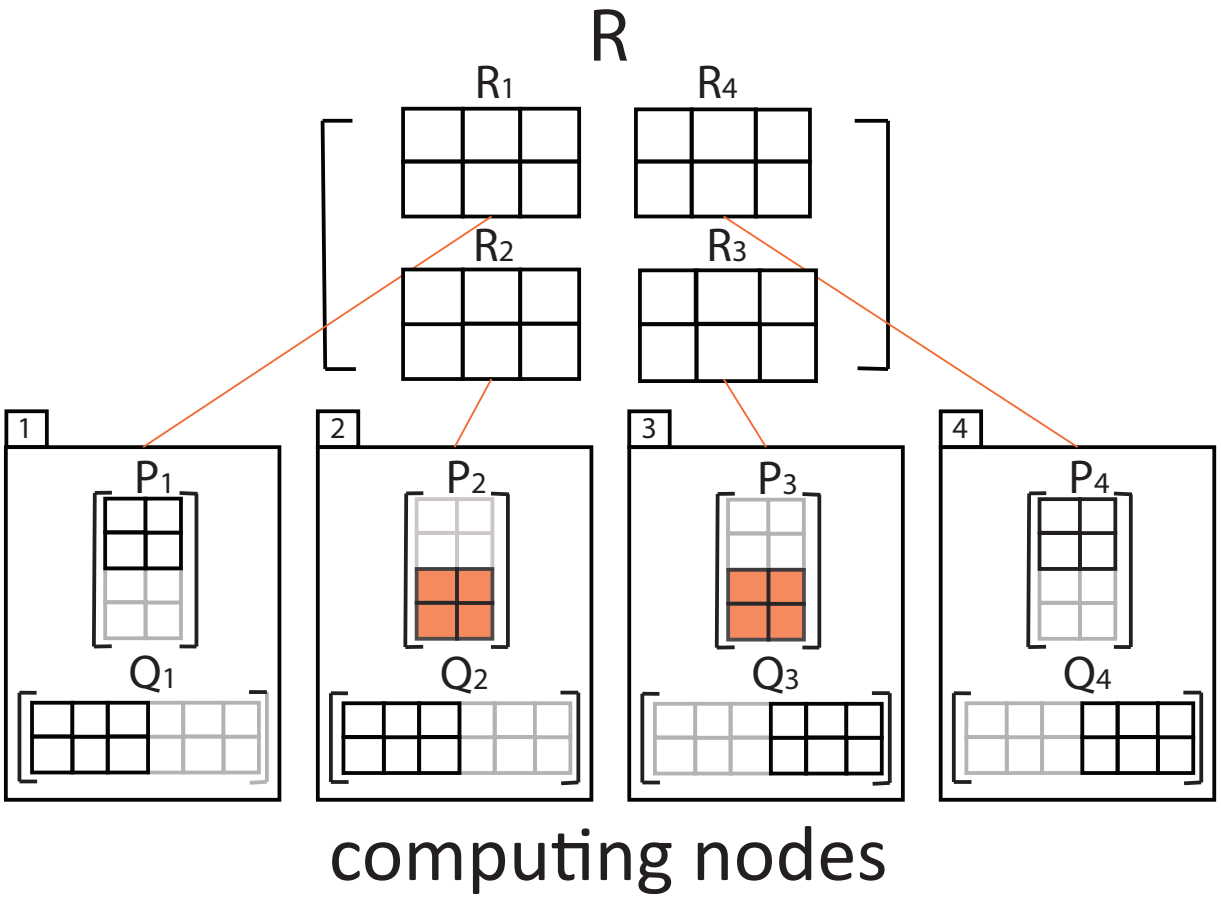
SGD Taxonomy



- parallel SGD is hardly applicable to very large datasets
  - the time-to-convergence may be too slow
  - the input data may not fit into the main memory
- ASGD has advantages in scalability and efficiency

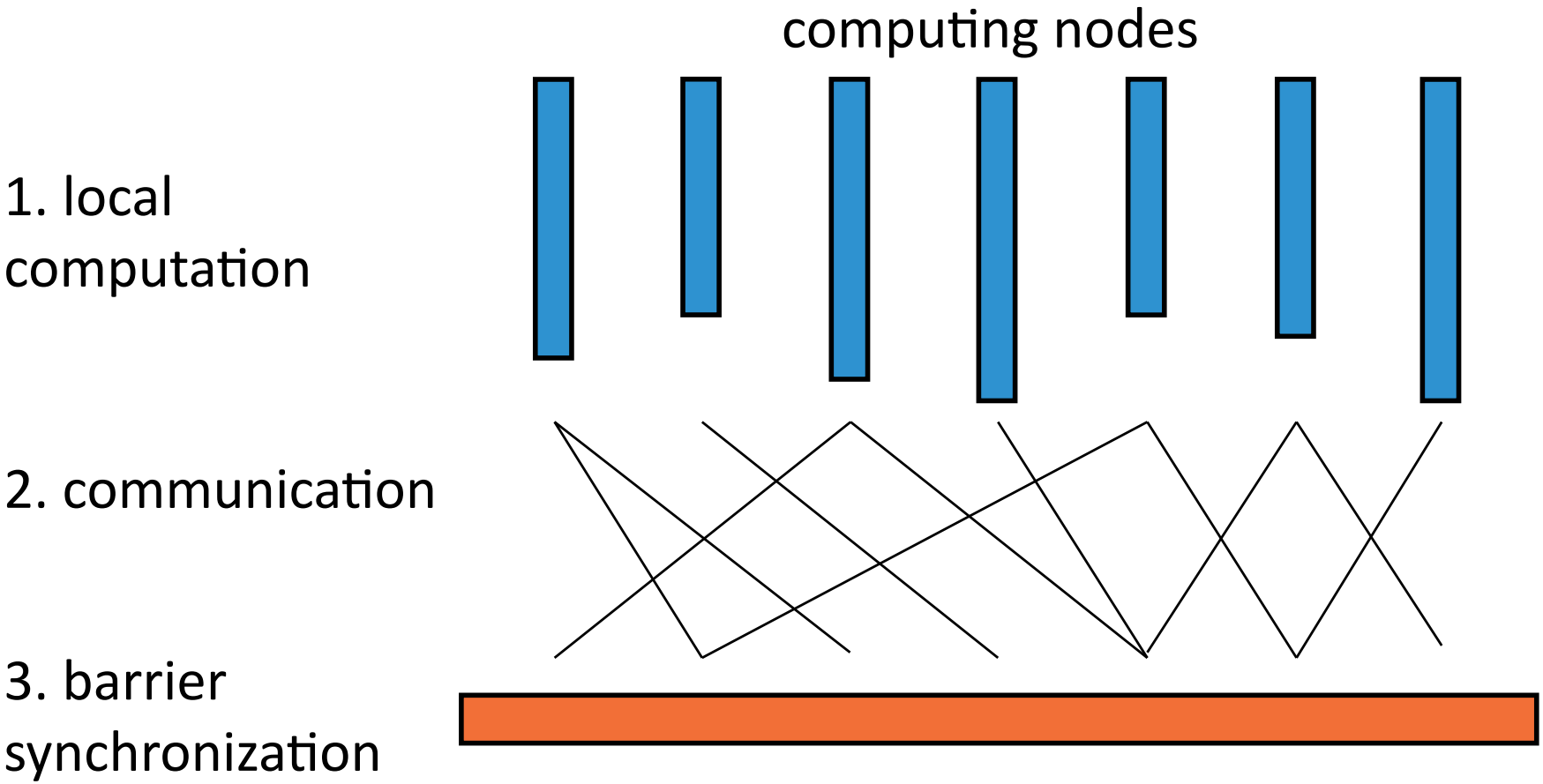
Asynchronous stochastic gradient descent

- distributed shared-nothing environment (cluster of machines)



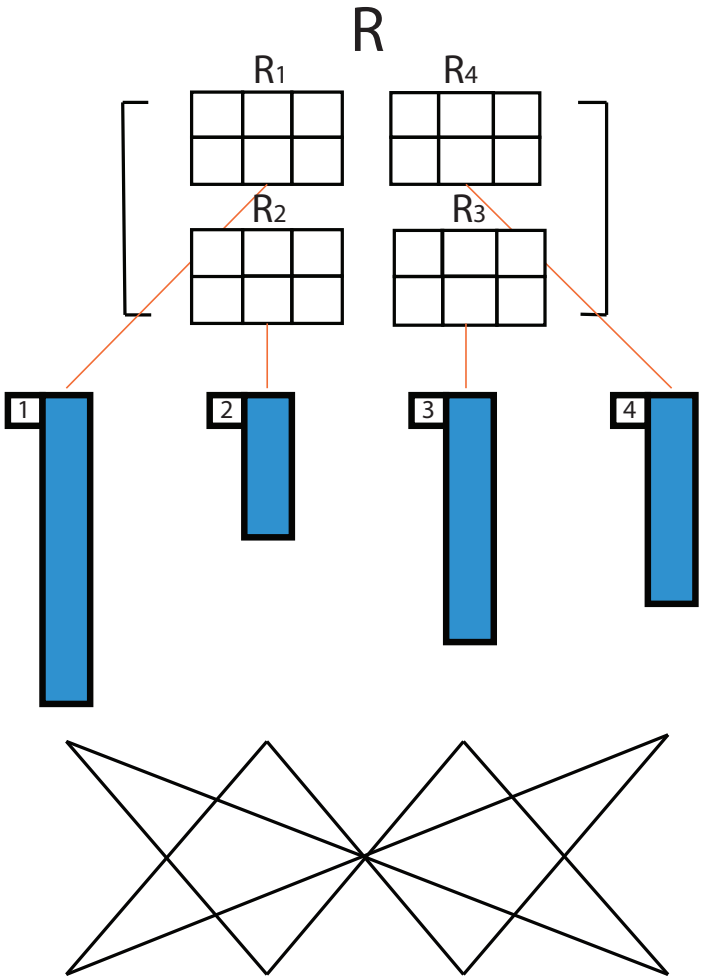
- R is splitted
- vectors are replicated
- replicas concurrently updated
- replicas deviate inconsistently
- synchronization

Bulk Synchronous Processing Model



- currently used by most of the ASGD implementations

Challenges

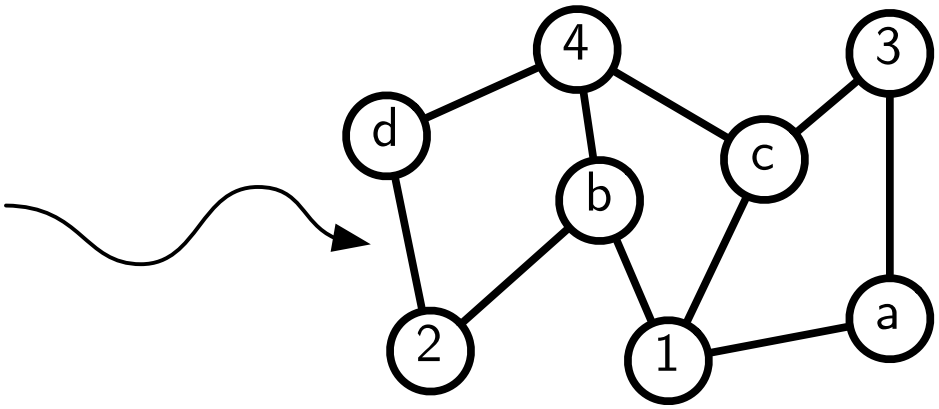


- **1. workload balance**
  - ensure that computing nodes are fed with the same load
  - improve efficiency
- **2. minimize communication**
  - minimize vector replicas
  - improve scalability

Graph representation

- the rating matrix describes a graph

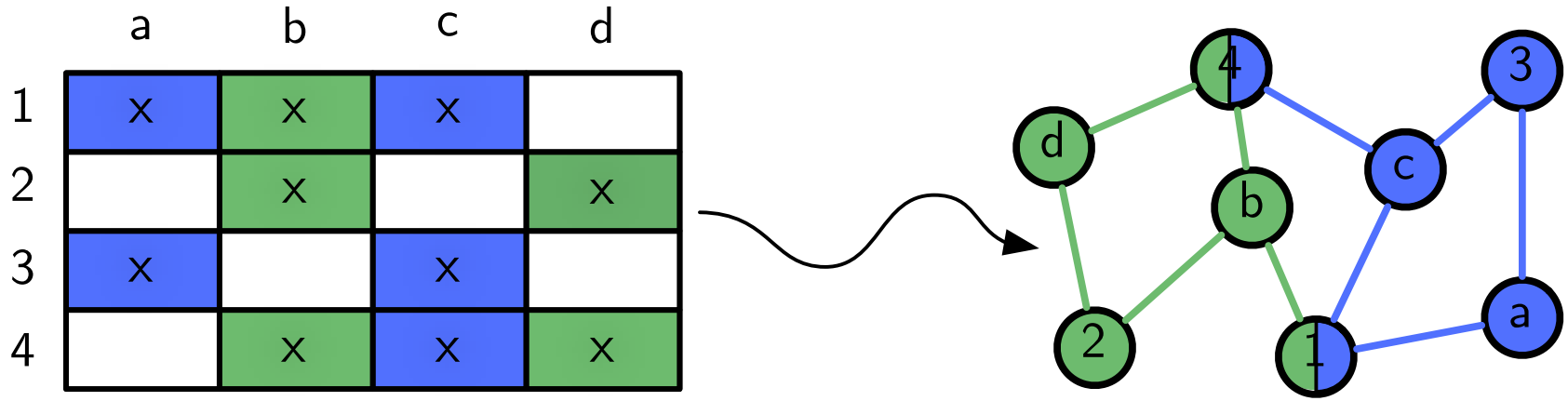
	a	b	c	d
1	x	x	x	
2		x		x
3	x		x	
4		x	x	x



- vertices represent users and items
- edges represent training points (e.g., ratings)

Graph representation

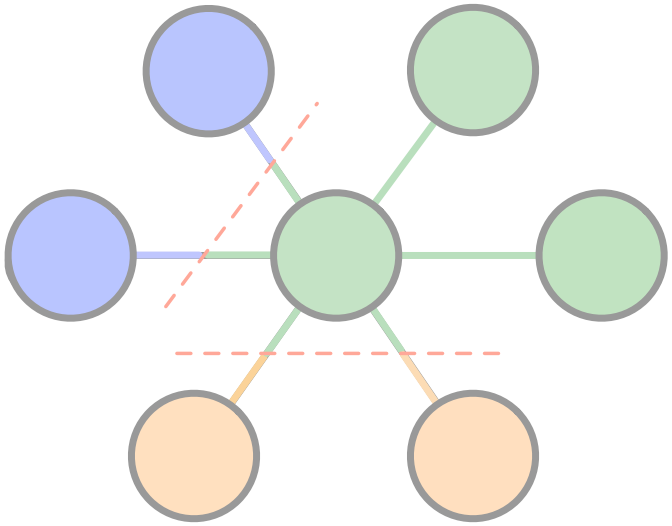
- the rating matrix describes a graph



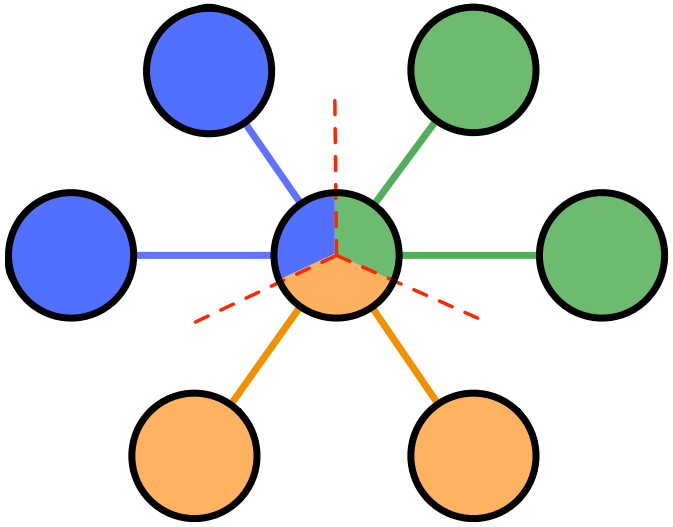
- find a partitioning of the graph
- assign each part to a different machine

Balanced graph partitioning

- partition G into smaller parts of (ideally) equal size



edge-cut

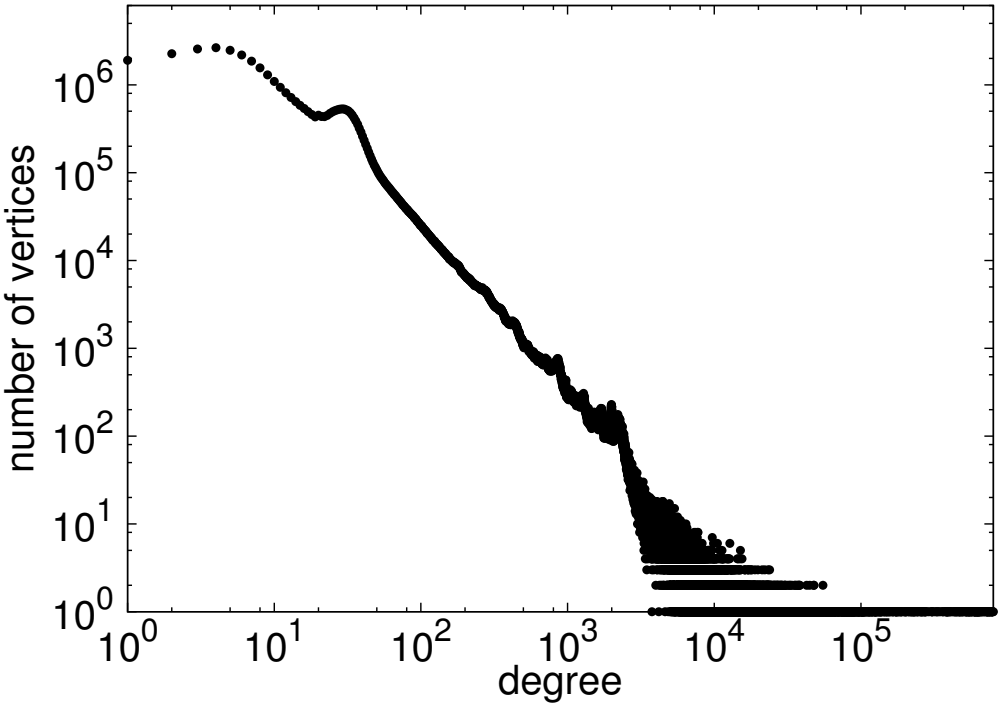


vertex-cut

- a vertex can be cut in multiple ways and span several parts while a cut edge connects only two parts
- computation steps are associated with edges
- v-cut better on **power-law graphs** *(Gonzalez et al, 2012)*

Power-law graphs

- characteristic of real graphs: power-law degree distribution
  - most vertices have few connections while a few have many



crawl Twitter connection 2010

log-log scale

- the probability that a vertex has degree d is $\boxed{P(d) \propto d^{-\alpha}}$
- α controls the "skewness" of the degree distribution

Balanced Vertex-Cut Graph Partitioning

- $v \in V$ vertex; $e \in E$ edge; $p \in P$ part
- $A(v)$ set of parts where vertex v is replicated
- $\sigma \geq 1$ tolerance to load imbalance
- the size $|p|$ of part p is its edge cardinality

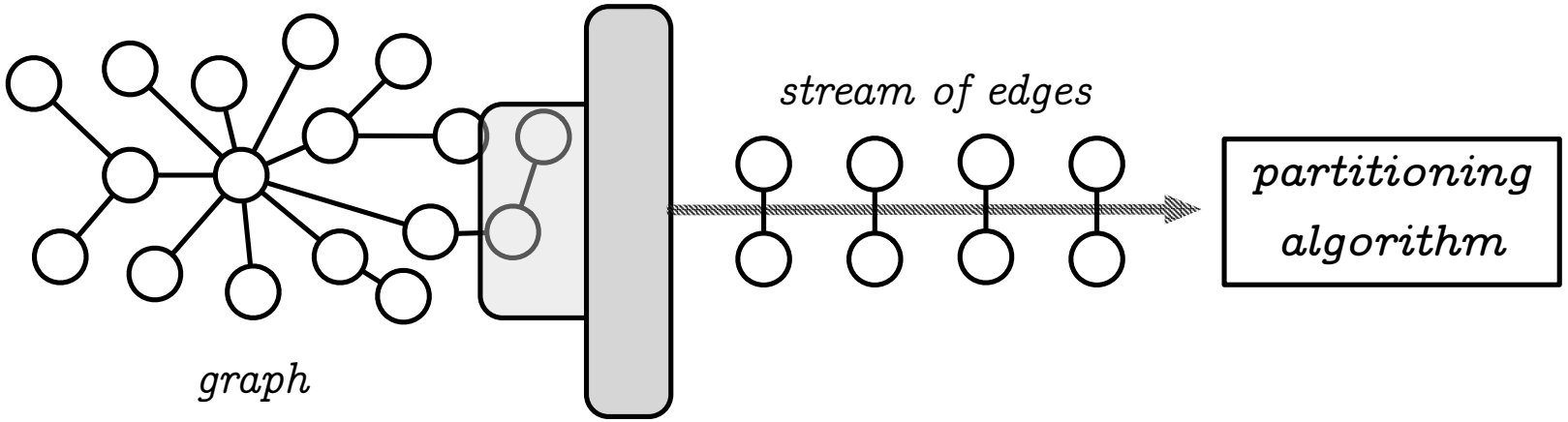
minimize replicas	***balance the load***
reduce (1) bandwidth, (2) memory usage and (3) synchronization	*efficient usage of available computing resources*

$$\min \boxed{\frac{1}{|V|} \sum_{v \in V} |A(v)|} \quad s.t. \quad \max_{p \in P} |p| < \sigma \frac{|E|}{|P|}$$

- the objective function is the **replication factor** (RF)
  - average number of replicas per vertex

Streaming Setting

- input data is a list of edges, consumed in *streaming fashion*, requiring only a **single pass**



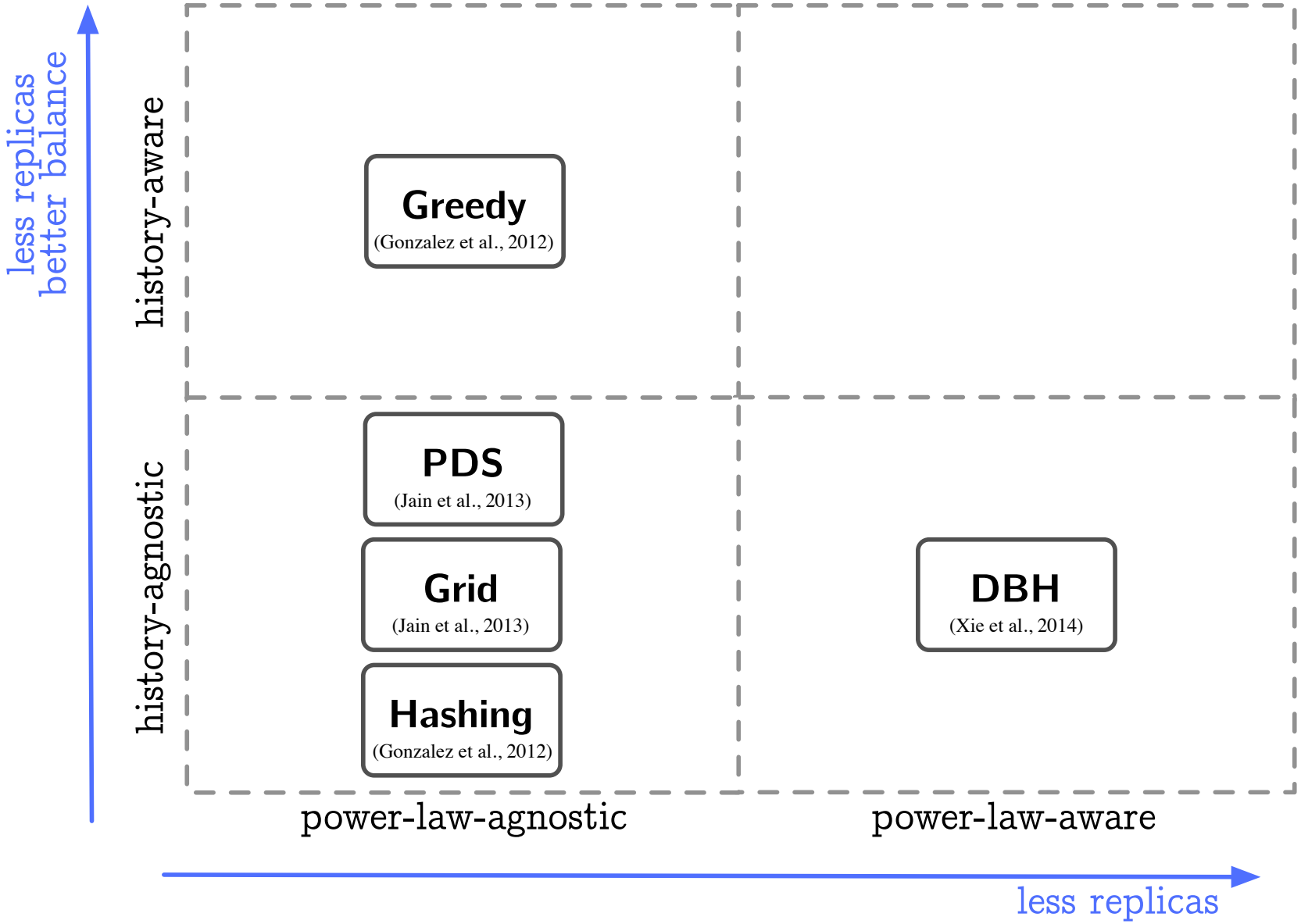
✓ handle graphs that don't fit in the main memory

✓ impose minimum overhead in time

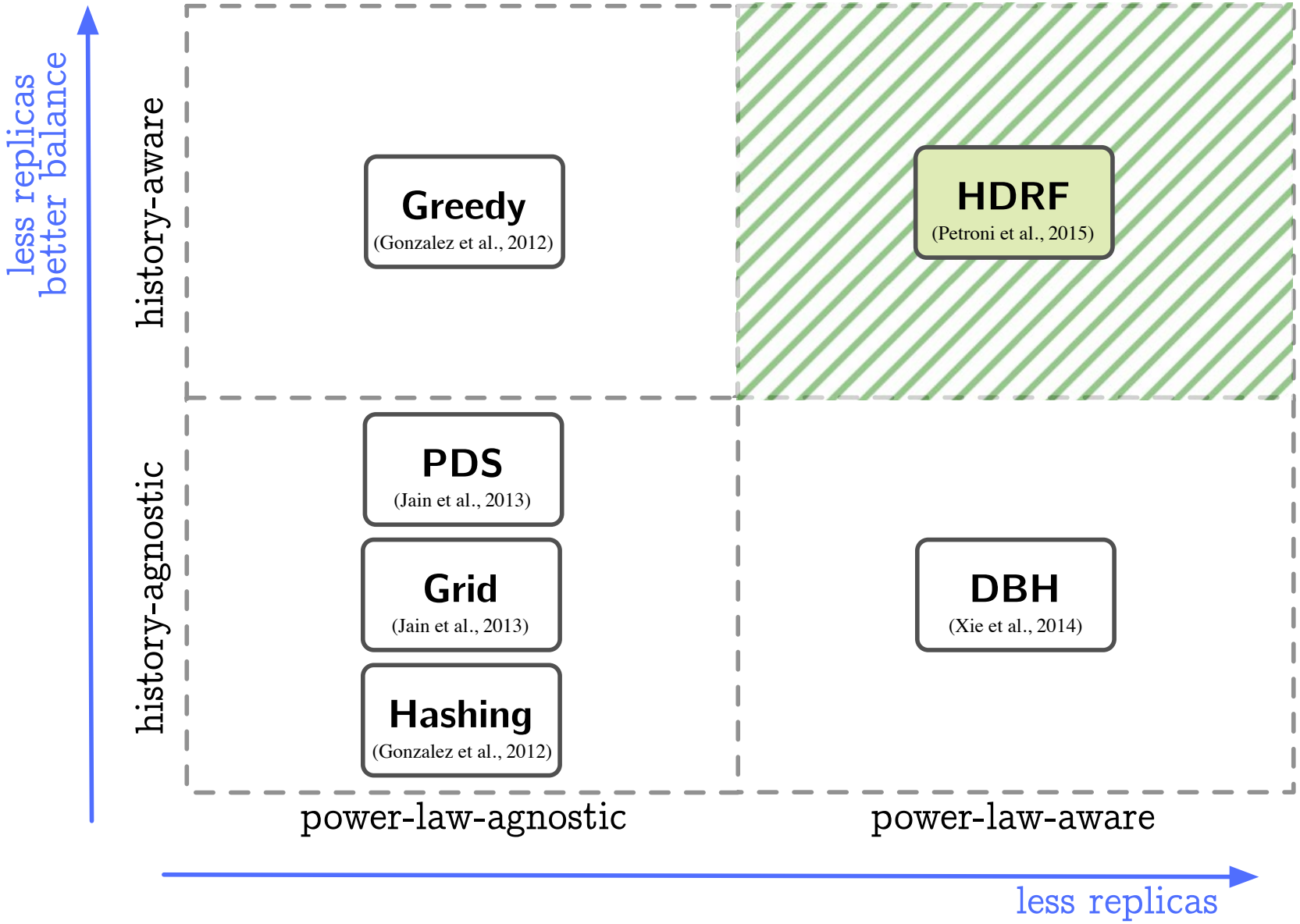
✓ scalable, easy parallel implementations

✗ assignment decision taken cannot be later changed

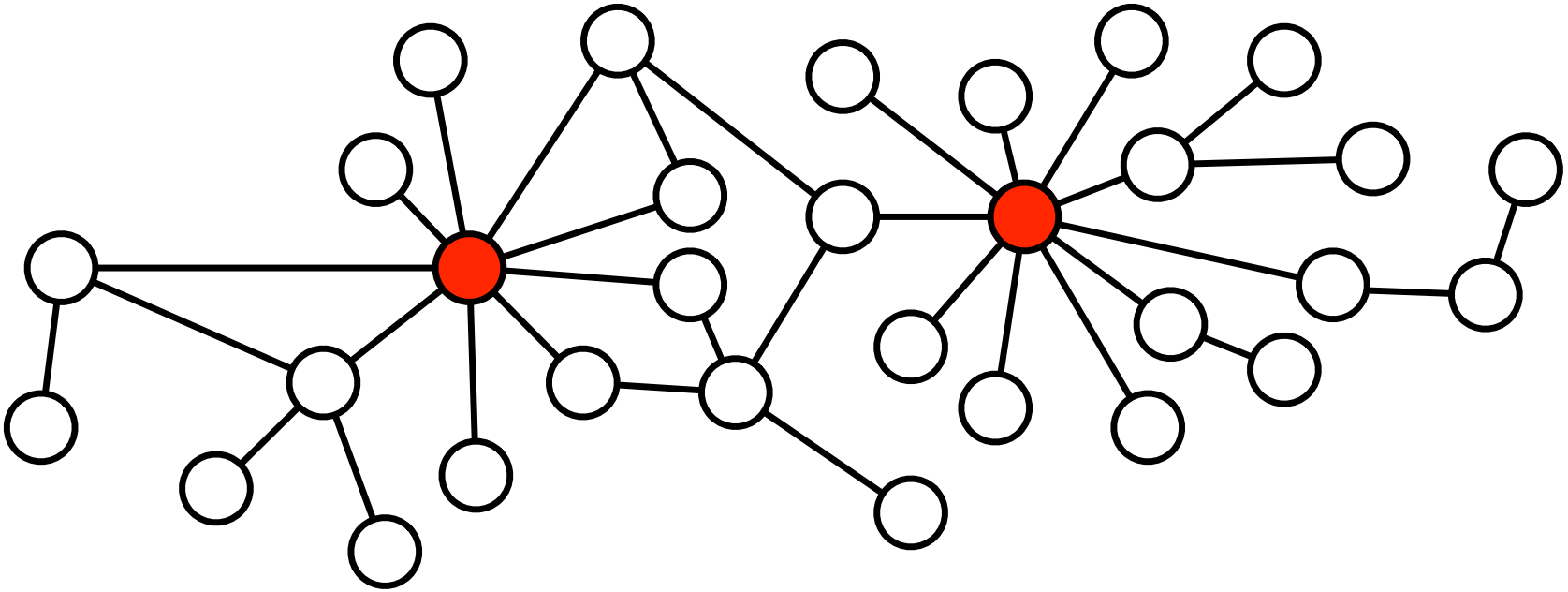
Streaming algorithms



Streaming algorithms - contributions

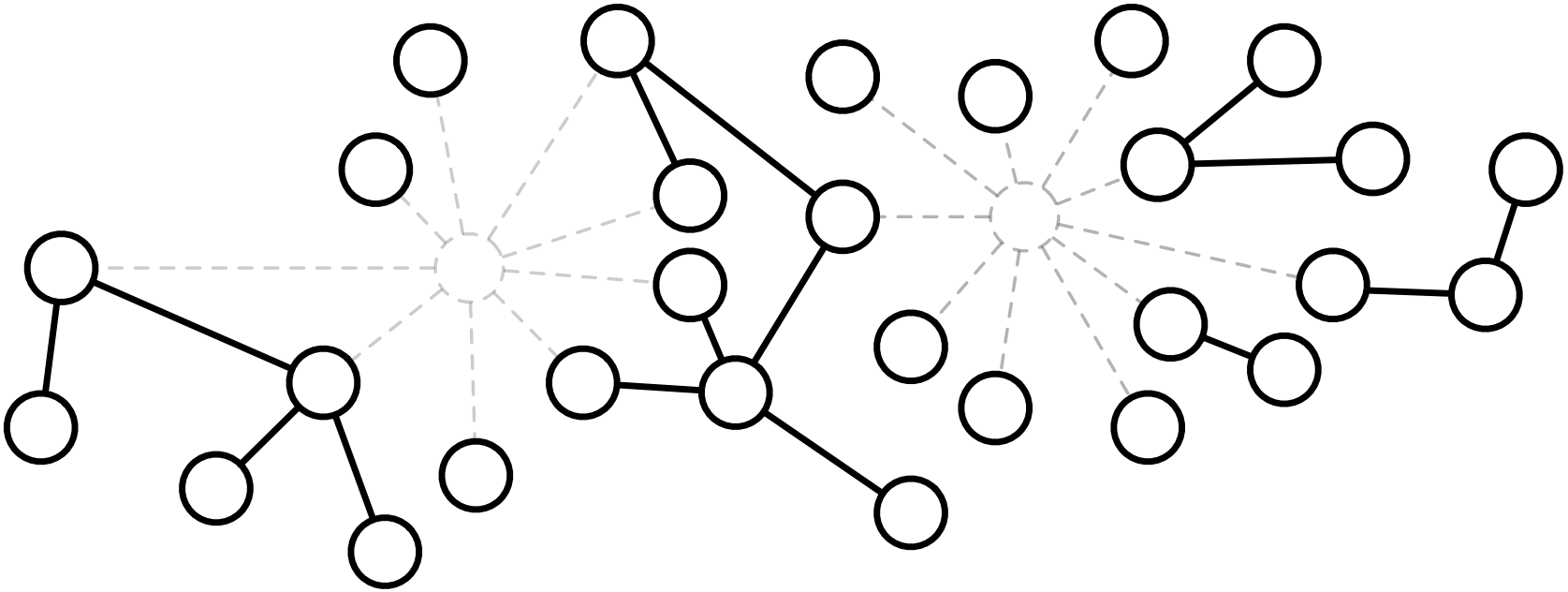


HDRF: High Degree are Replicated First



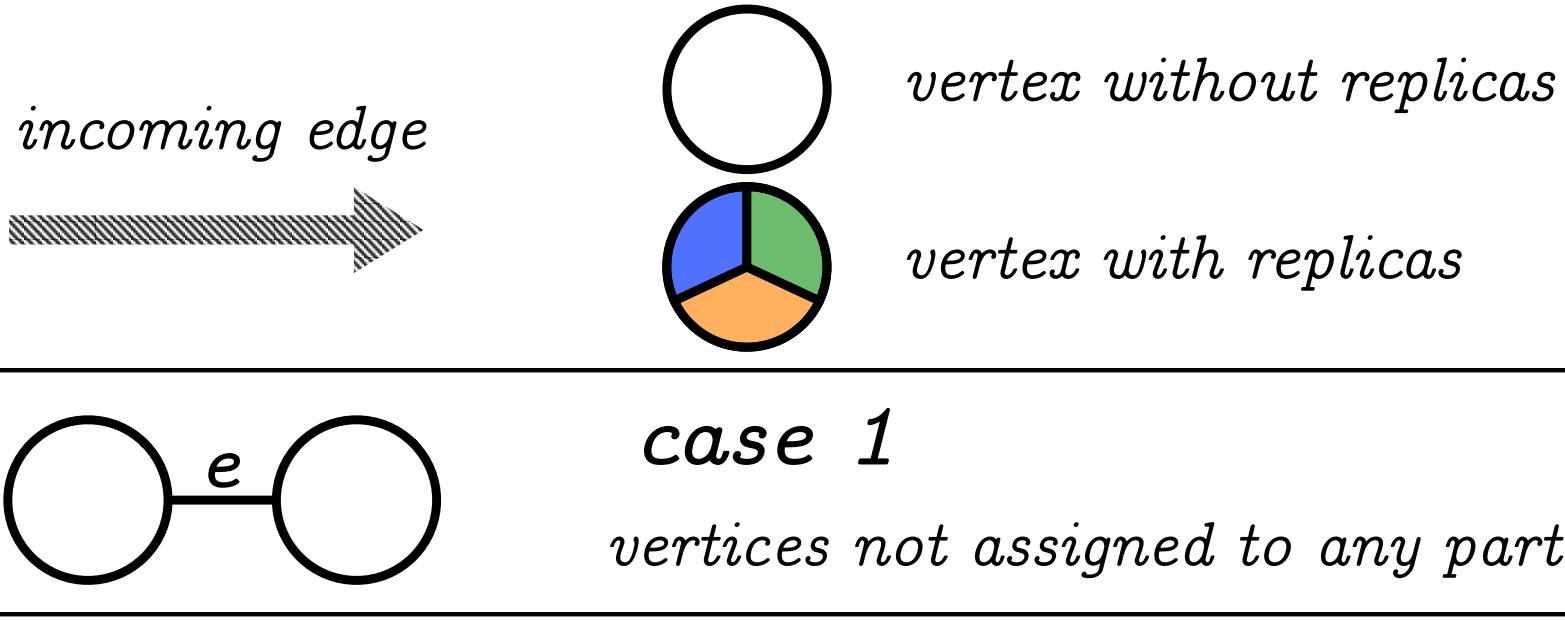
- favor the replication of high-degree vertices
- the number of high-degree vertices in power-law graphs is very small
- overall reduction of the replication factor

HDRF: High Degree are Replicated First

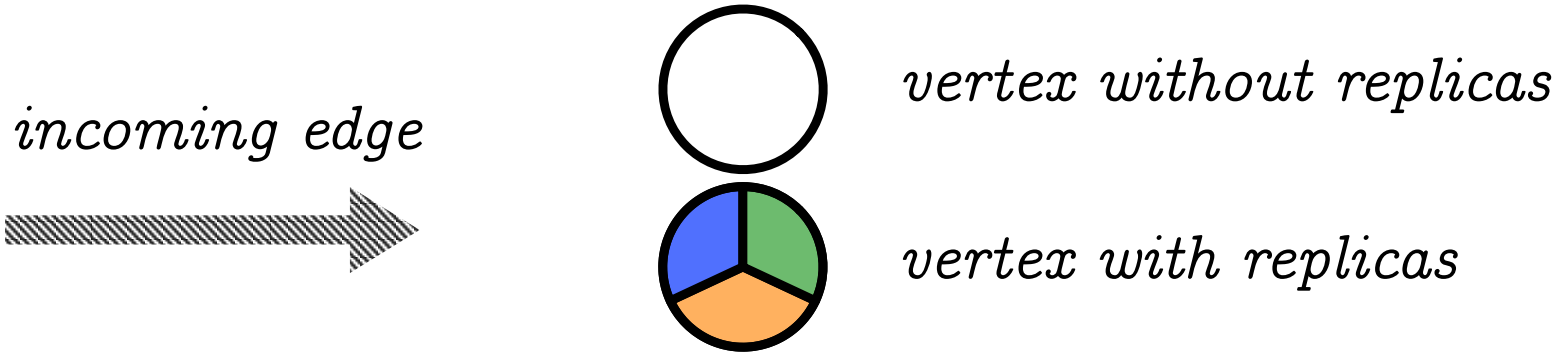


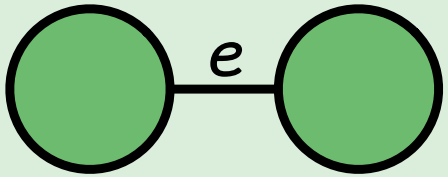
- in the context of robustness to network failure
- if few high-degree vertices are removed from a power-law graph then it is turned into a set of isolated clusters
- focus on the locality of low-degree vertices

The HDRF Algorithm



The HDRF Algorithm

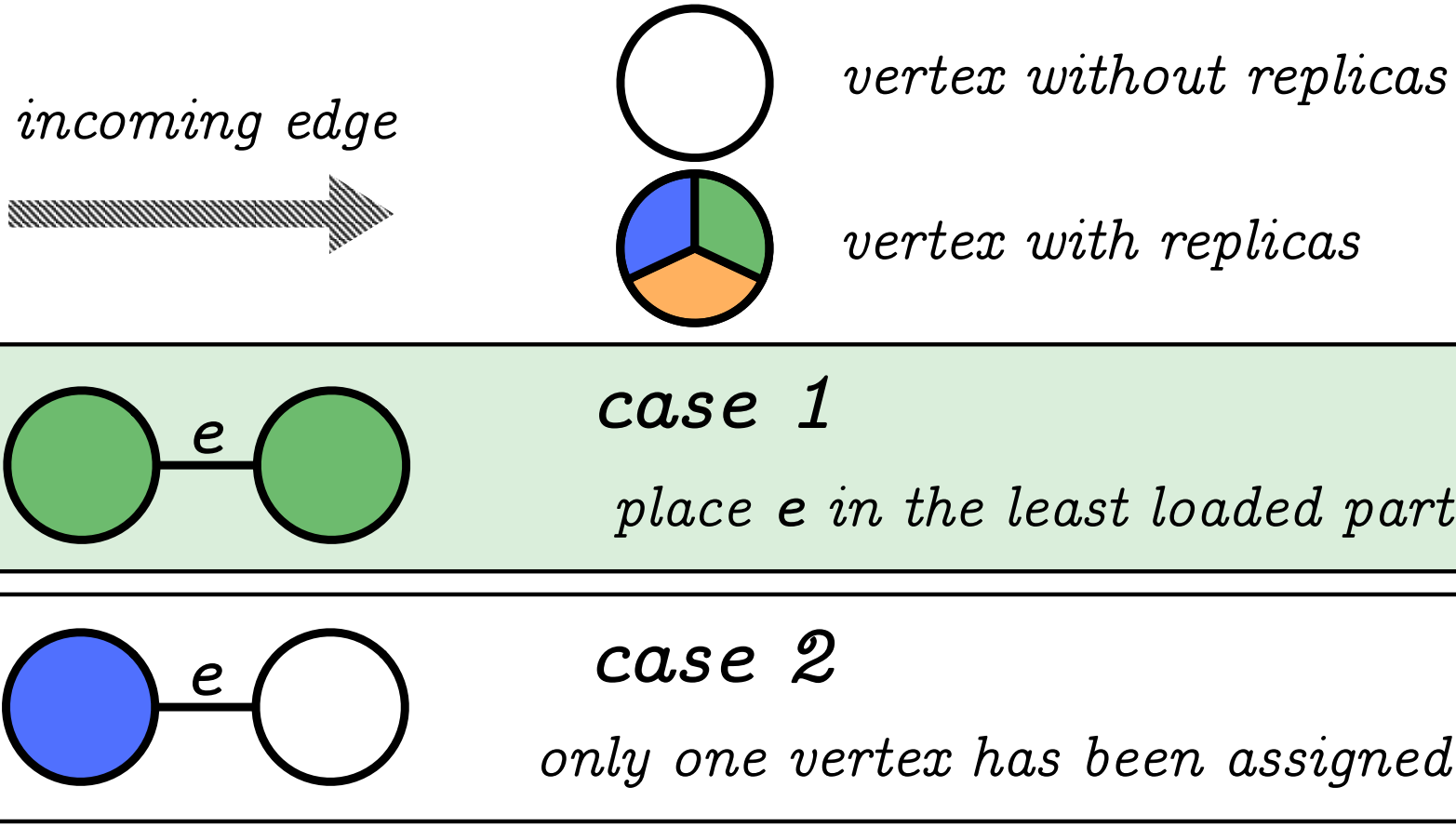




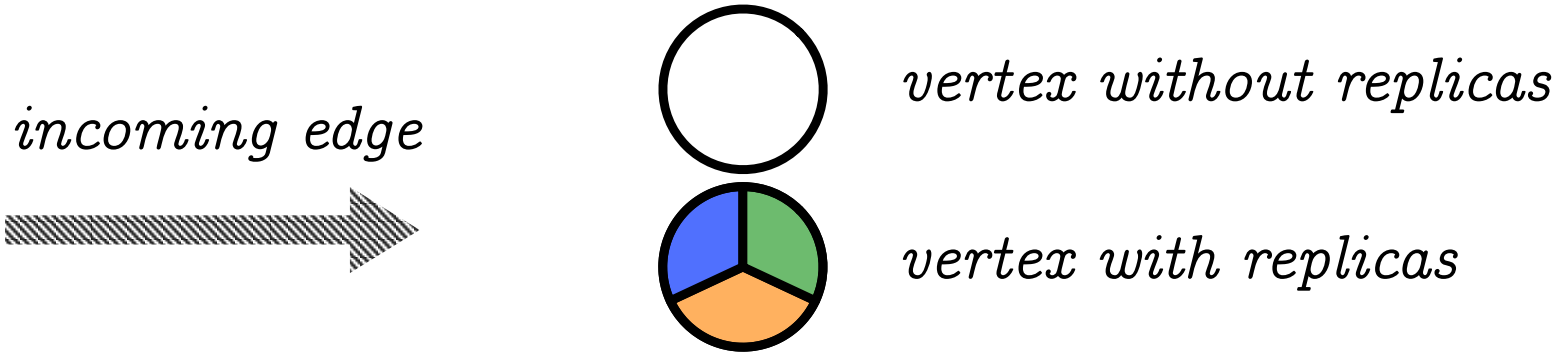
case 1

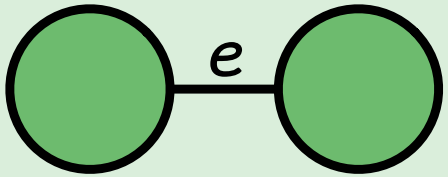
place **e** *in the least loaded part*

The HDRF Algorithm



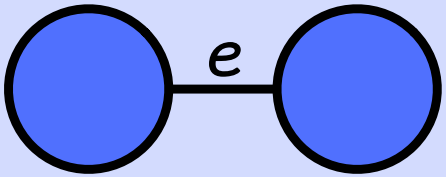
The HDRF Algorithm





case 1

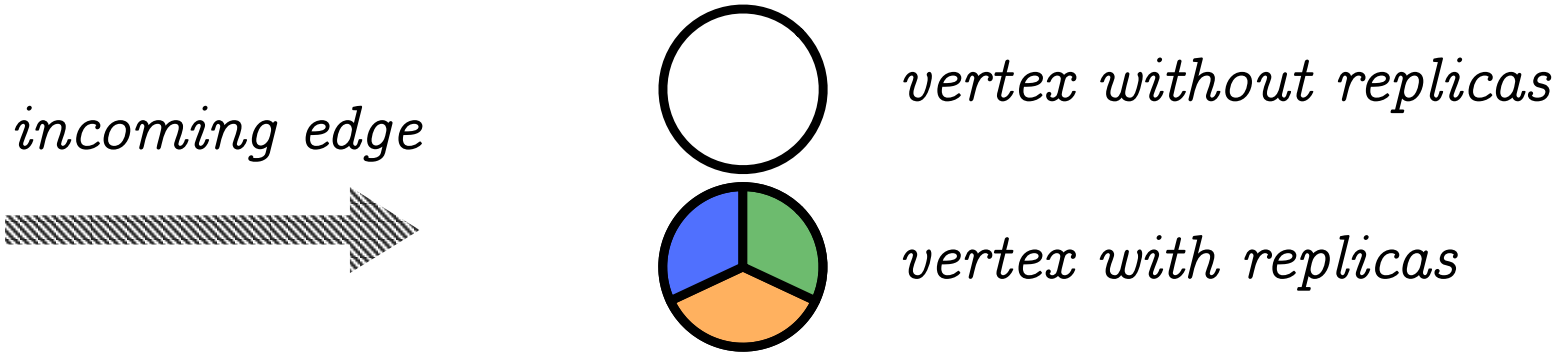
place **e** *in the least loaded part*

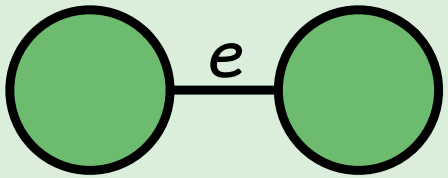


case 2

place **e** *in the part*

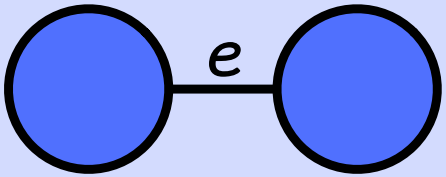
The HDRF Algorithm





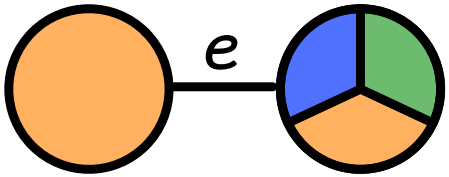
case 1

place **e** *in the least loaded part*



case 2

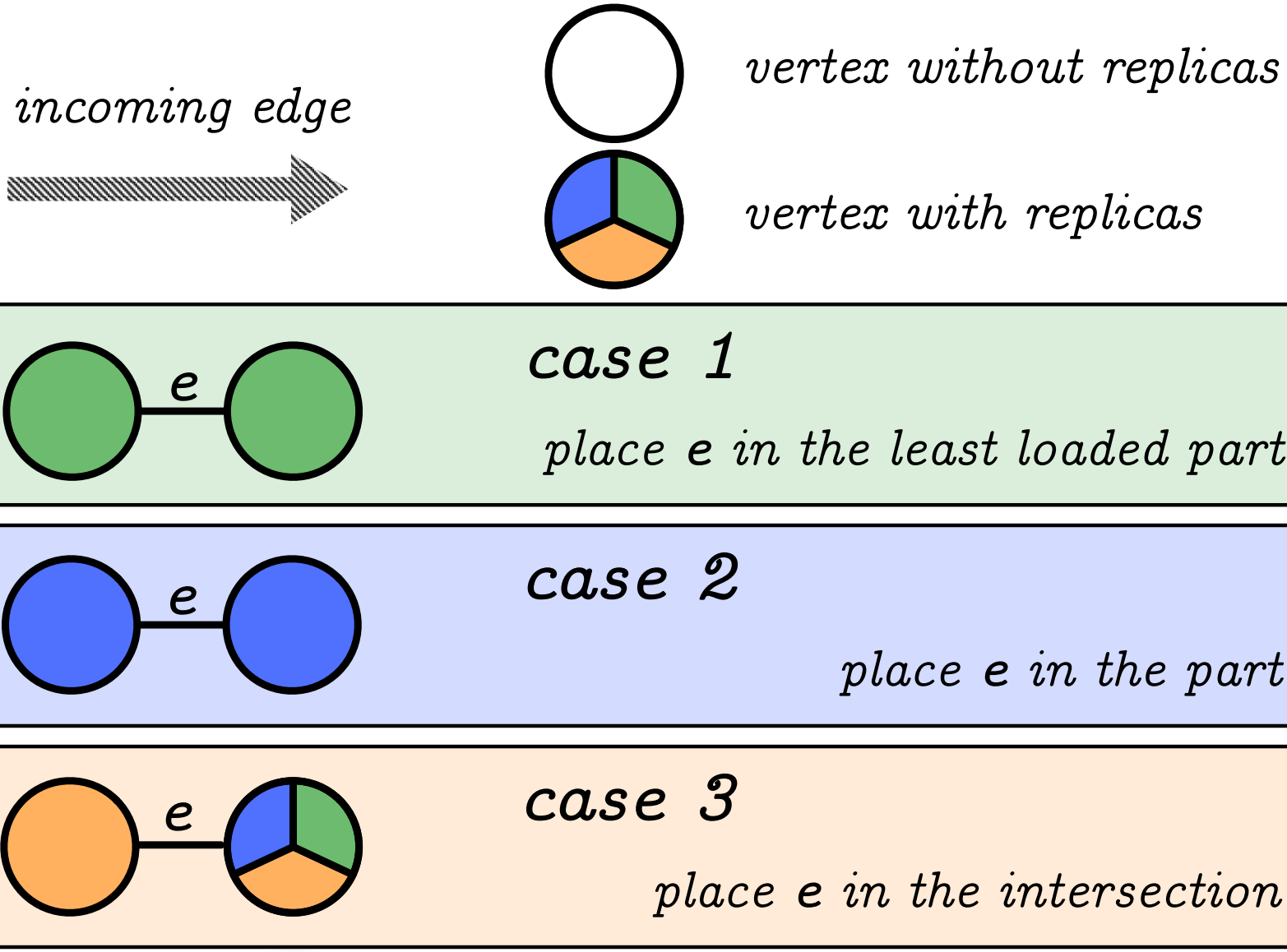
place **e** *in the part*



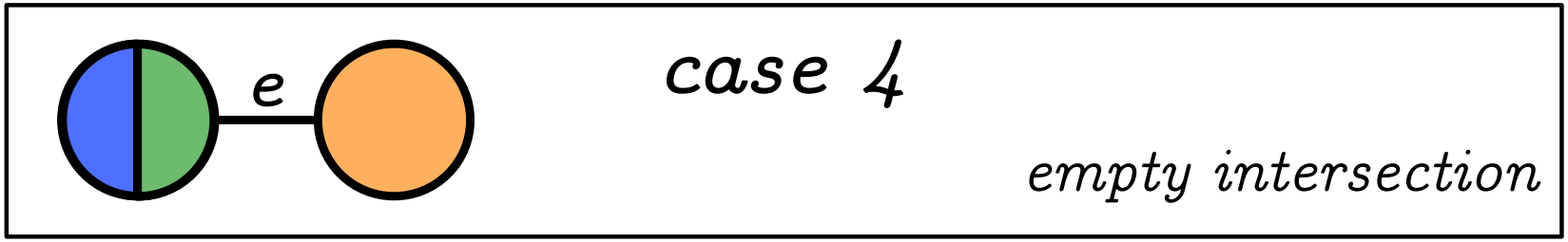
case 3

vertices assigned, common part

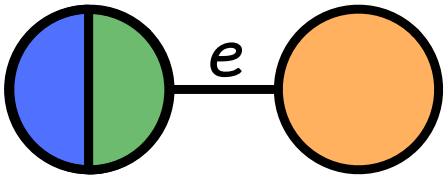
The HDRF Algorithm



Create Replicas



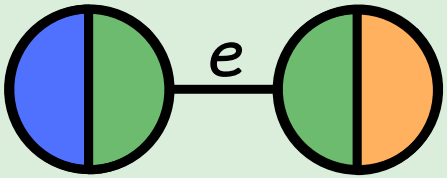
Create Replicas



case 4

empty intersection

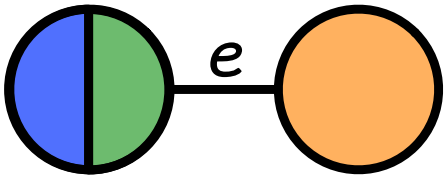
standard Greedy solution



case 4

least loaded part in the union

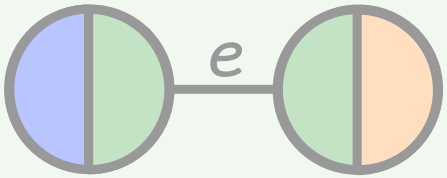
Create Replicas



case 4

empty intersection

standard Greedy solution

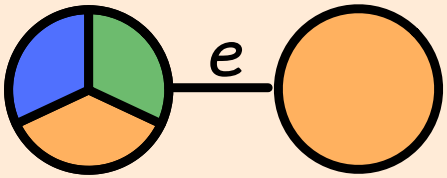


case 4

least loaded part in the union

HDRF

$\delta(v1) > \delta(v2)$



case 4

replicate vertex with highest degree

Experiments - Settings

- standalone partitioner
  - VGP, a software package for one-pass vertex-cut balanced graph partitioning
  - measure the performance: replication and balancing
- GraphLab
  - HDRF has been integrated in GraphLab PowerGraph 2.2
  - measure the impact on the execution time of graph computation in a distributed graph computing frameworks
- stream of edges in random order

Experiments - Datasets

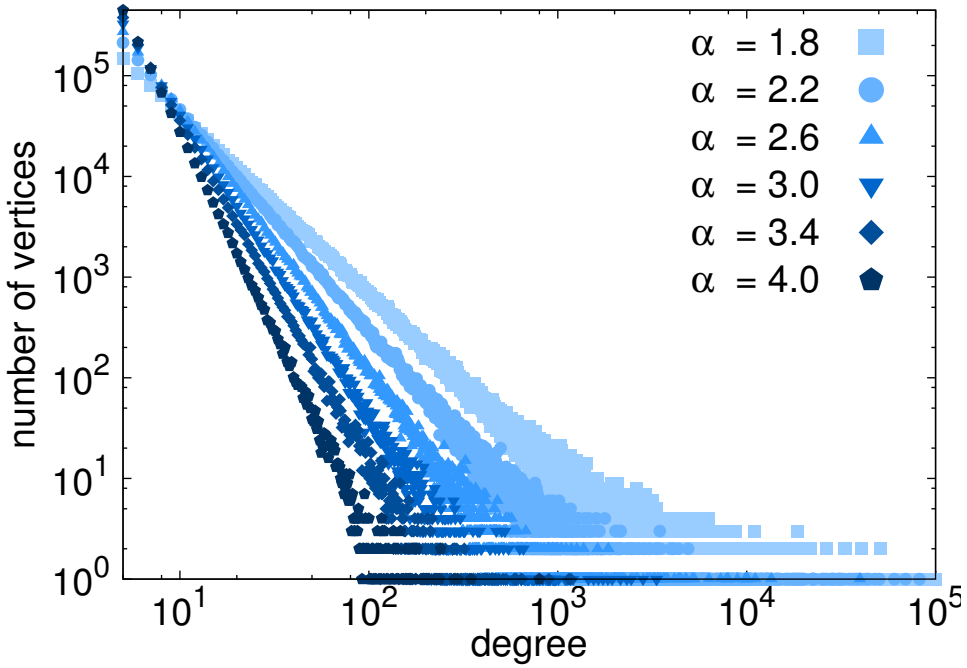
- real-word graphs

Dataset	$\|V\|$	$\|E\|$
MovieLens 10M	80.6K	10M
Netflix	497.9K	100.4M
Tencent Weibo	1.4M	140M
twitter-2010	41.7M	1.47B

- synthetic graphs

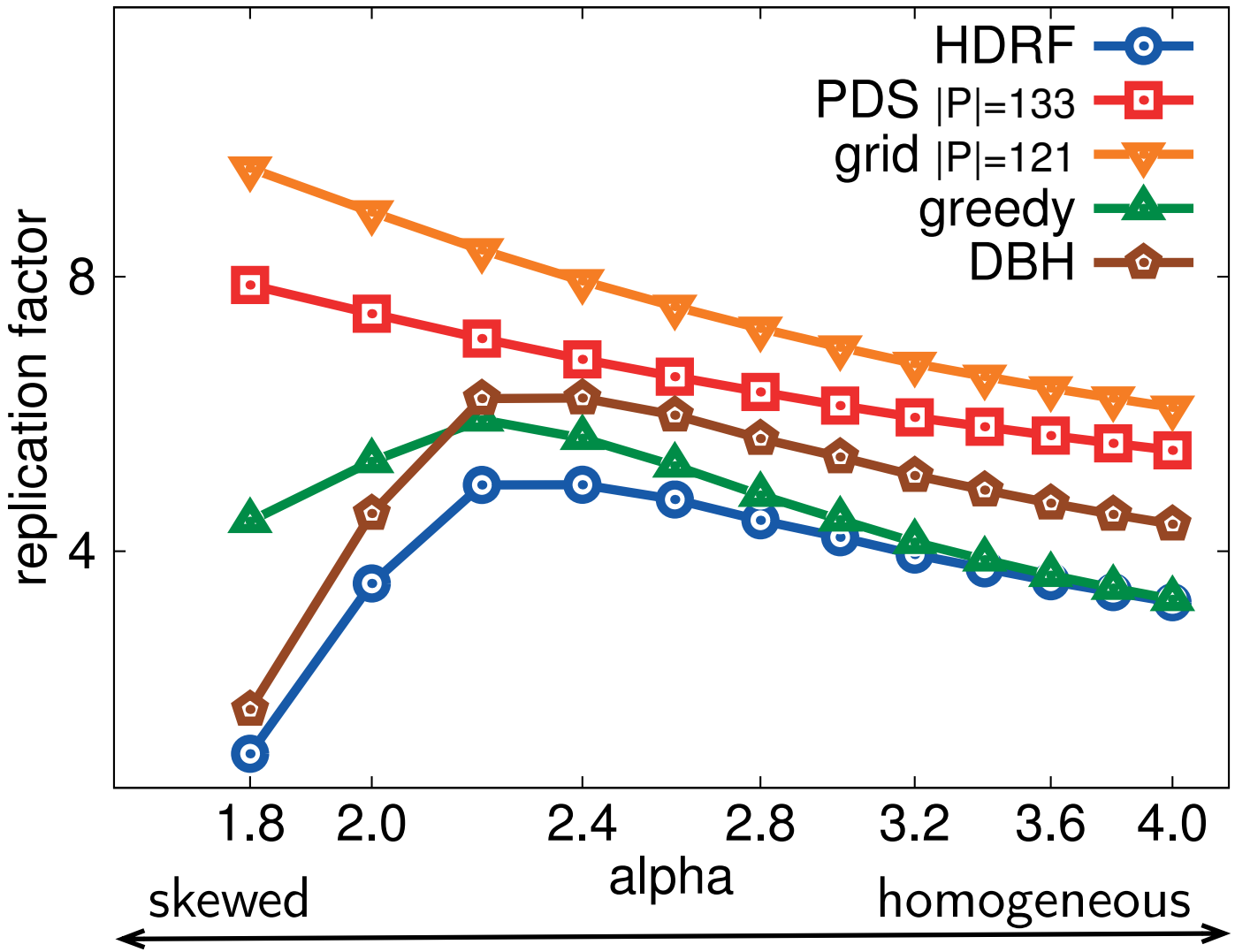
1M vertices

60M to 3M edges



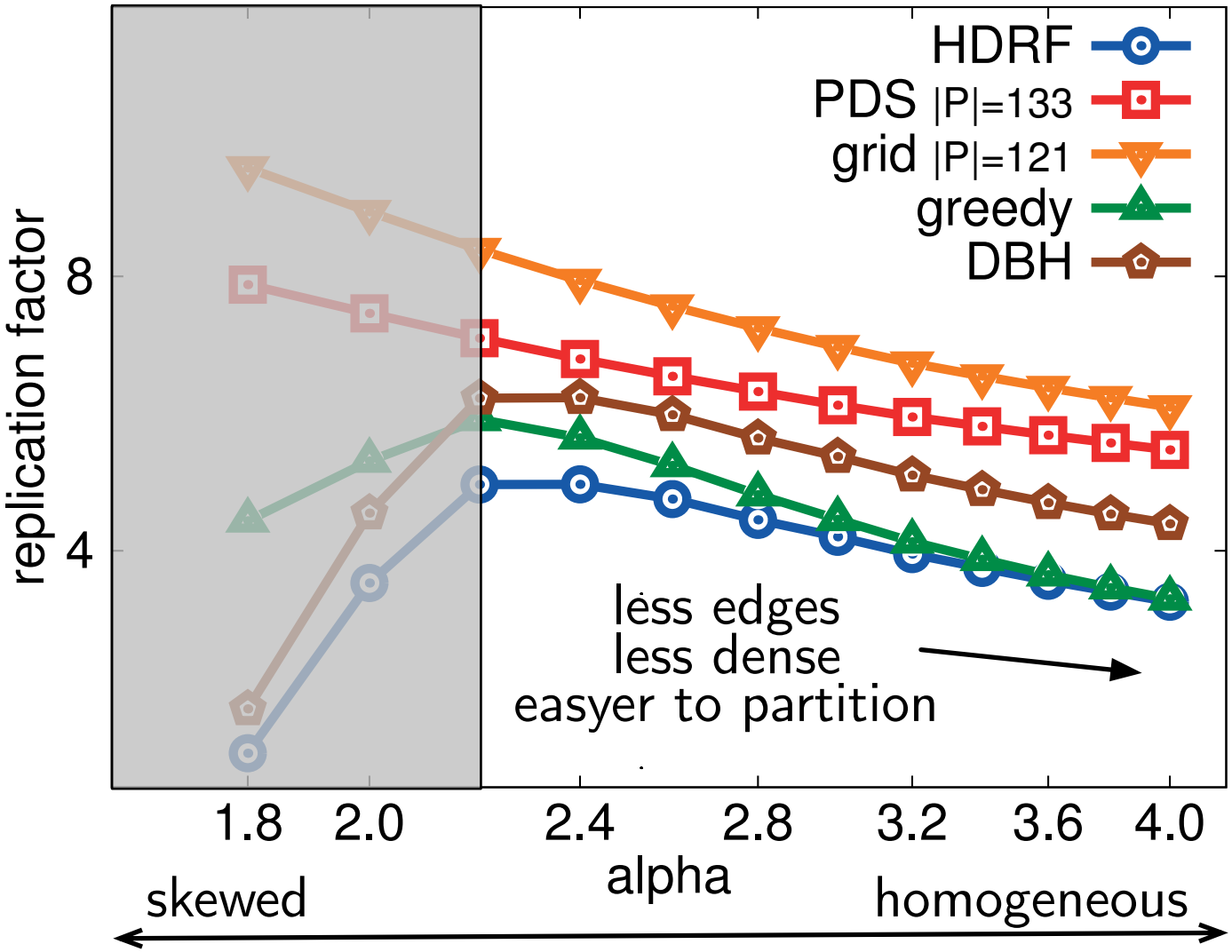
Results - Synthetic Graphs Replication Factor

- 128 parts



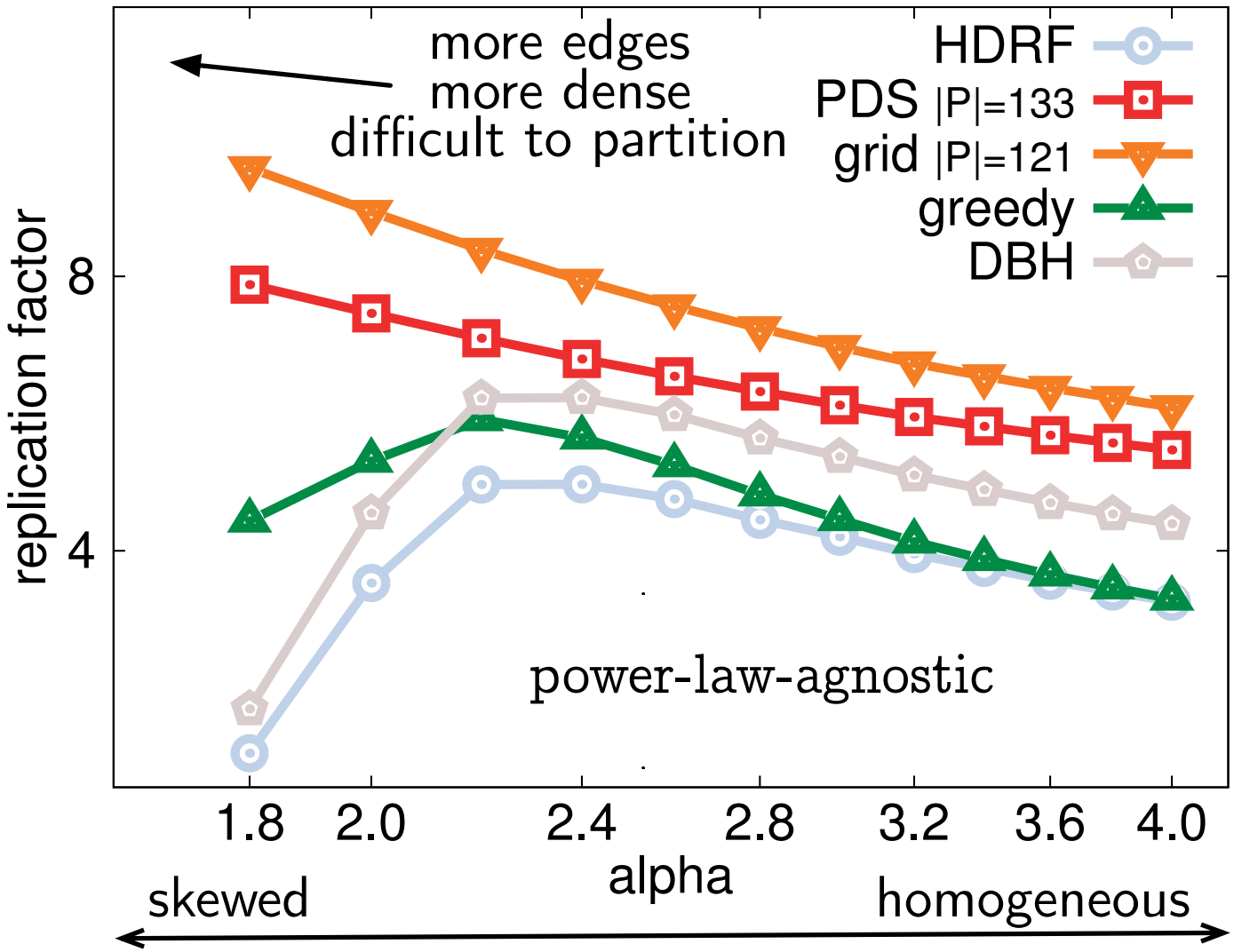
Results - Synthetic Graphs Replication Factor

- 128 parts



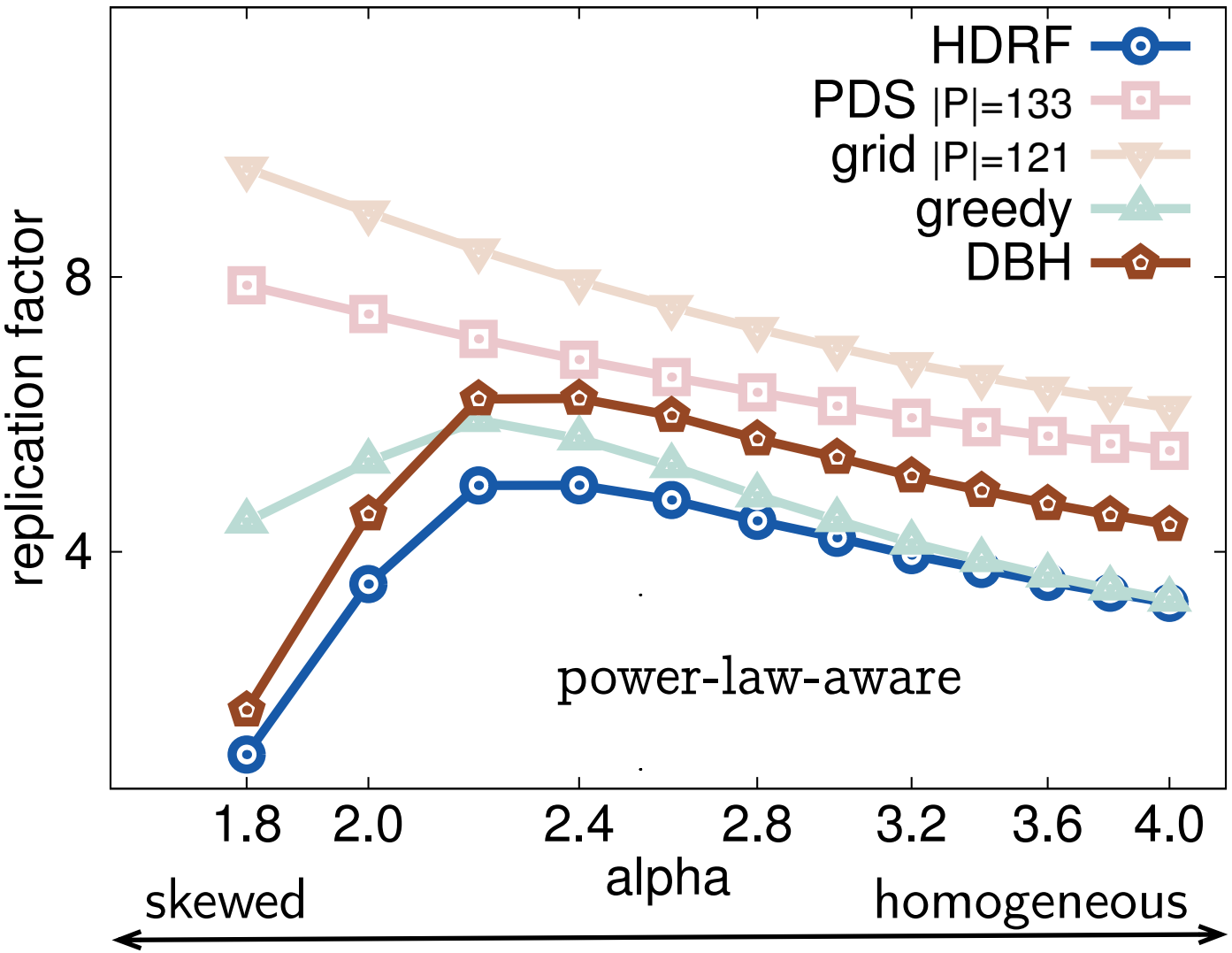
Results - Synthetic Graphs Replication Factor

- 128 parts



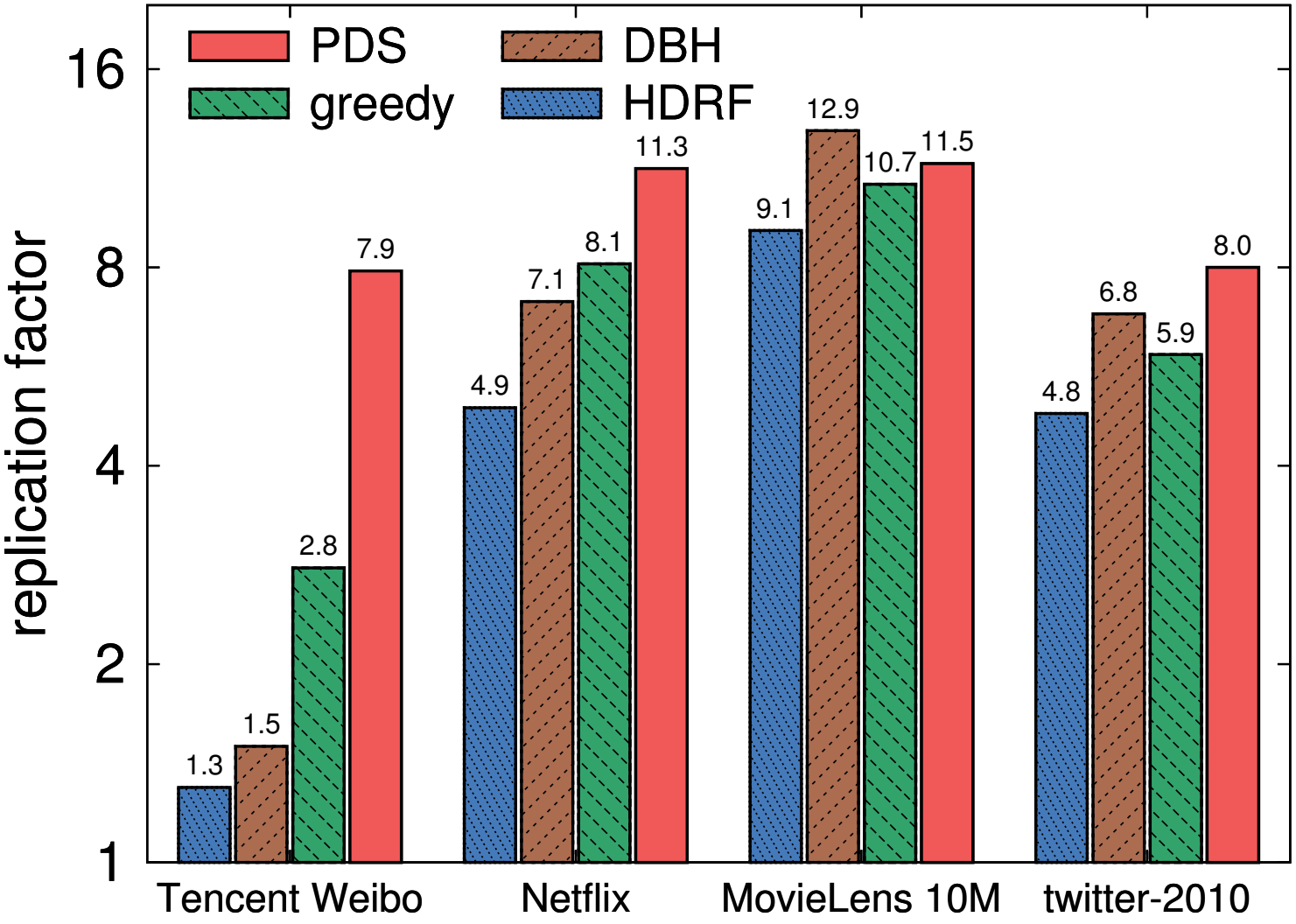
Results - Synthetic Graphs Replication Factor

- 128 parts



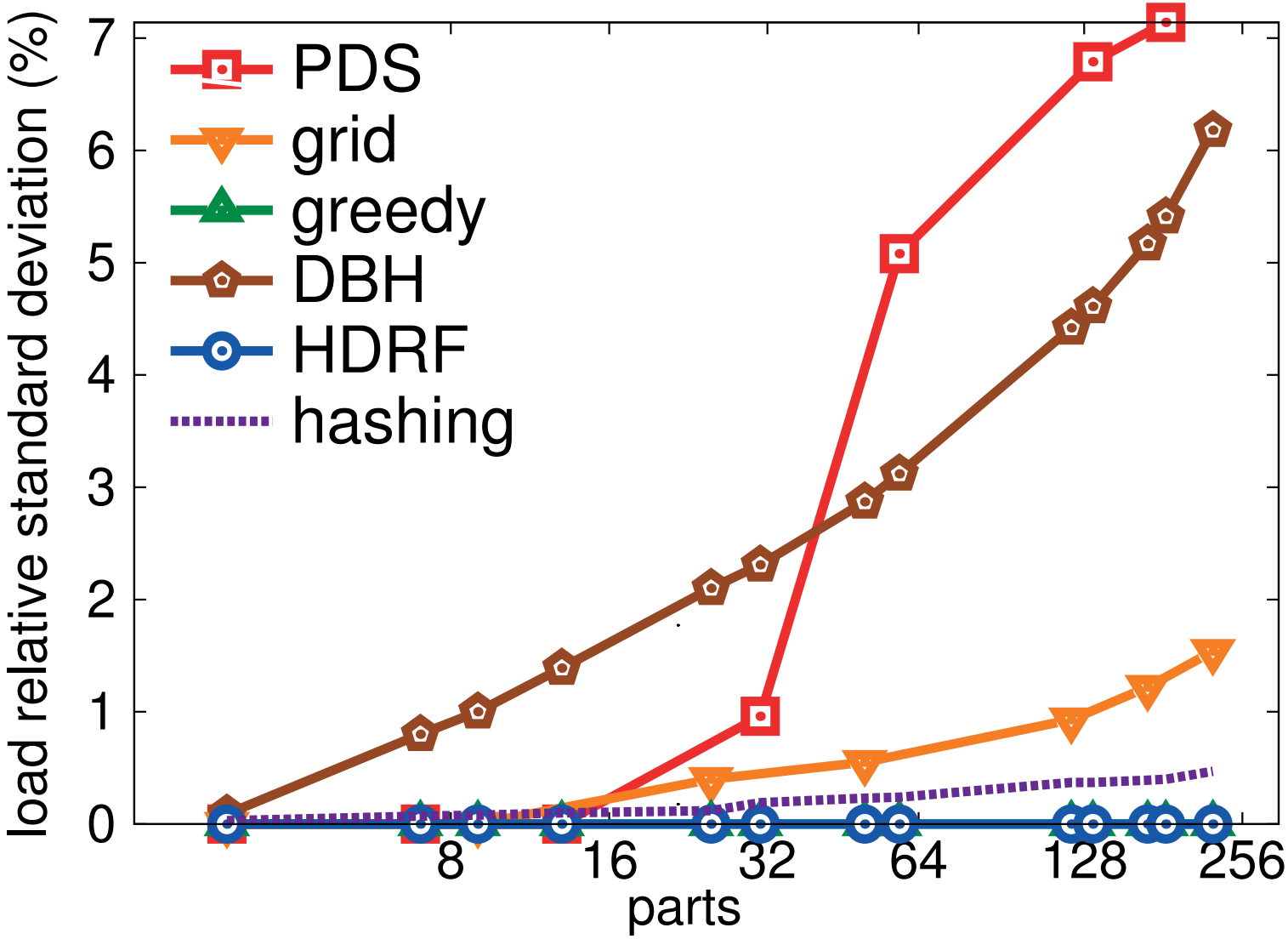
Results - Real-Word Graphs Replication Factor

- 133 parts



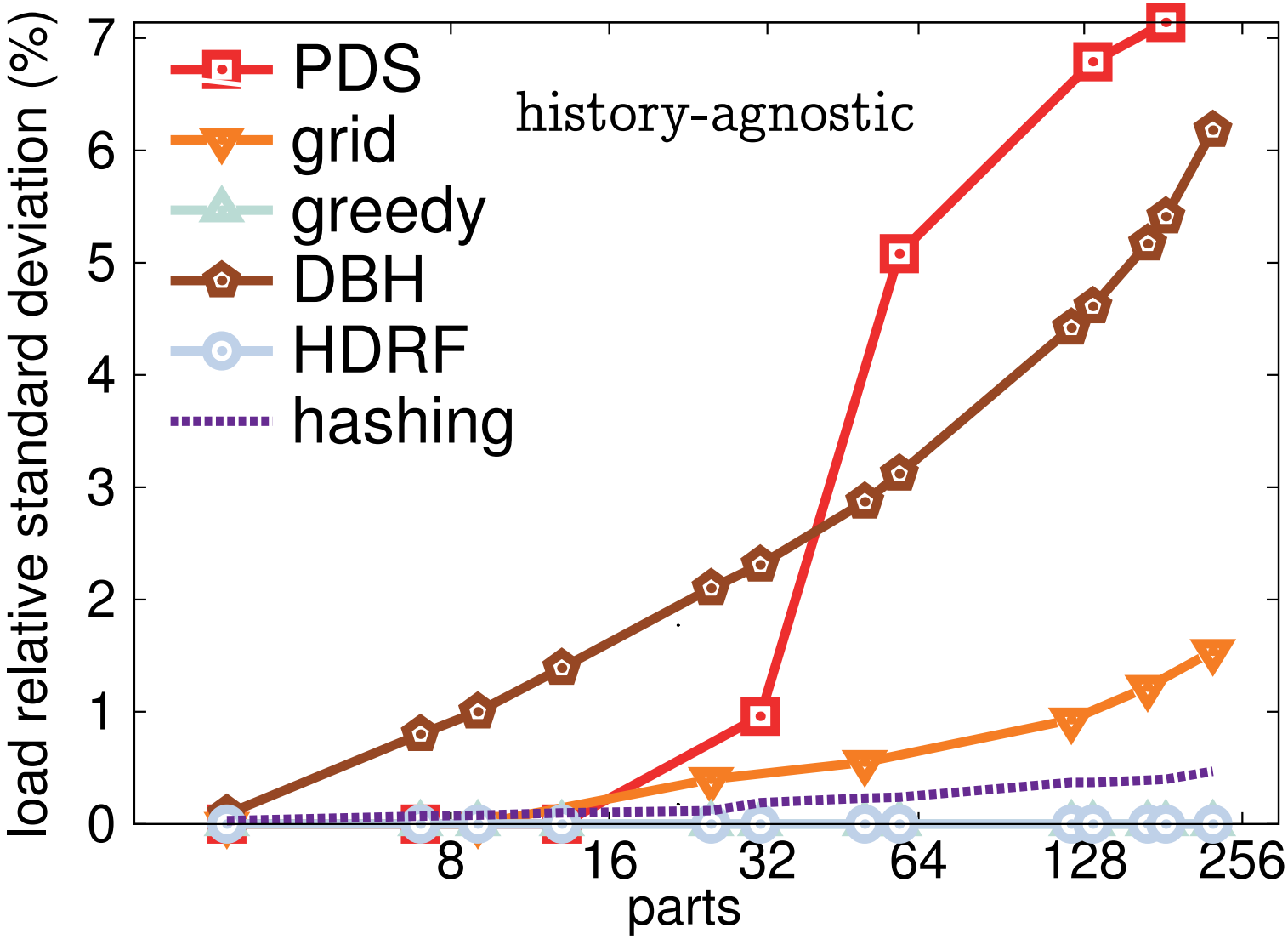
Results - Load Relative Standard Deviation

- MovieLens 10M



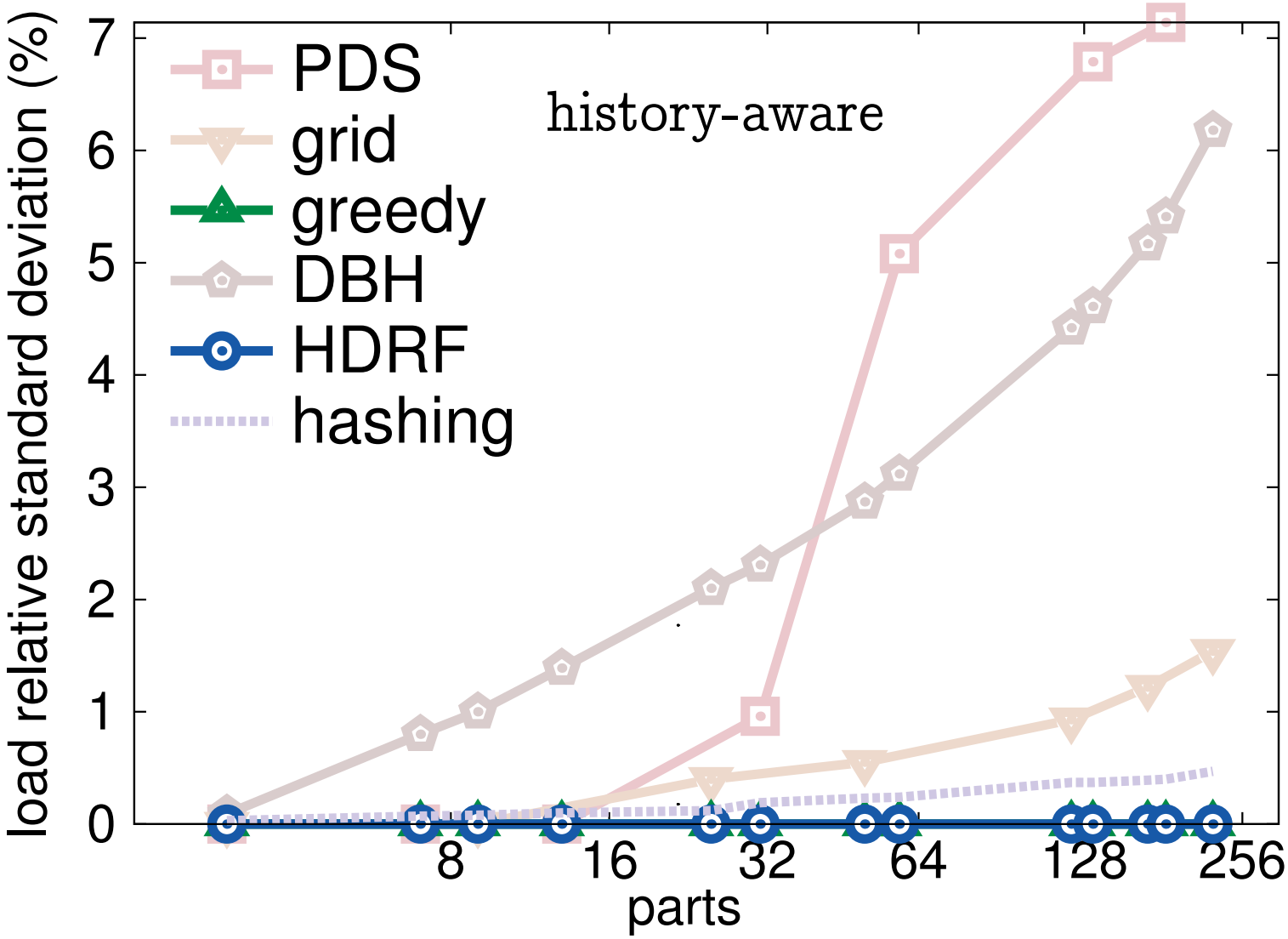
Results - Load Relative Standard Deviation

- MovieLens 10M



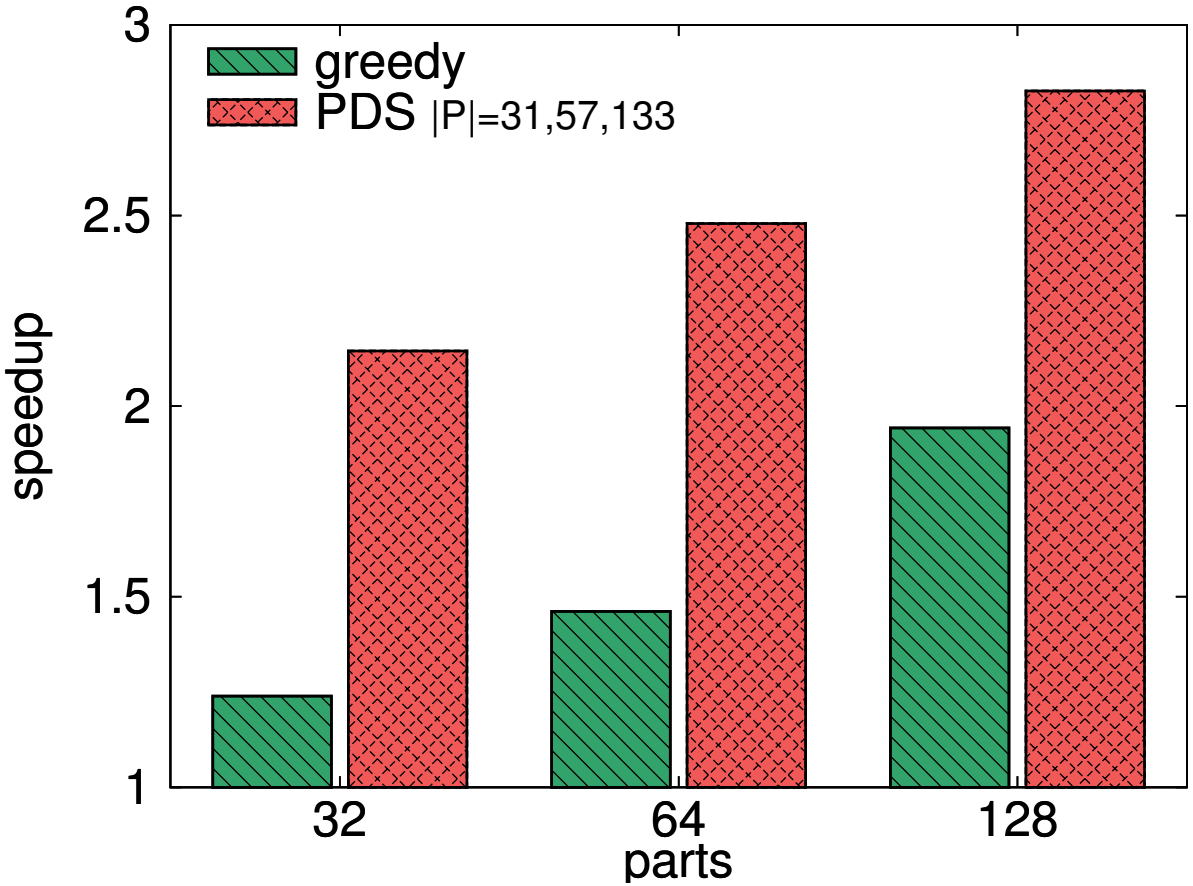
Results - Load Relative Standard Deviation

- MovieLens 10M



Results - Graph Algorithm Runtime Speedup

- *ASGD* algorithm for collaborative filtering on *Tencent Weibo*



- the speedup is proportional to both:
  - the advantage in replication factor
  - the actual network usage of the algorithm

Summary

- HDRF is a simple and remarkably effective one-pass vertex-cut graph partitioning algorithm
  - achieves on average a replication factor
    - about 40% smaller than DBH
    - more than 50% smaller than greedy
    - almost 3× smaller than PDS
    - more than 4× smaller than grid
    - almost 14× smaller than hashing
- close to optimal load balance
- ASGD execution time is up to 2× faster when using HDRF
- HDRF has been included in GraphLab!

Contents

Knowledge bases

- New generation algorithms for web information retrieval make use of a **knowledge base (KB)** to increase their accuracy

- match the words in a query to real world entities (e.g., a person, a location, etc)

- use real world connections among these entities

- improve the task of providing the user with the proper content he was looking for

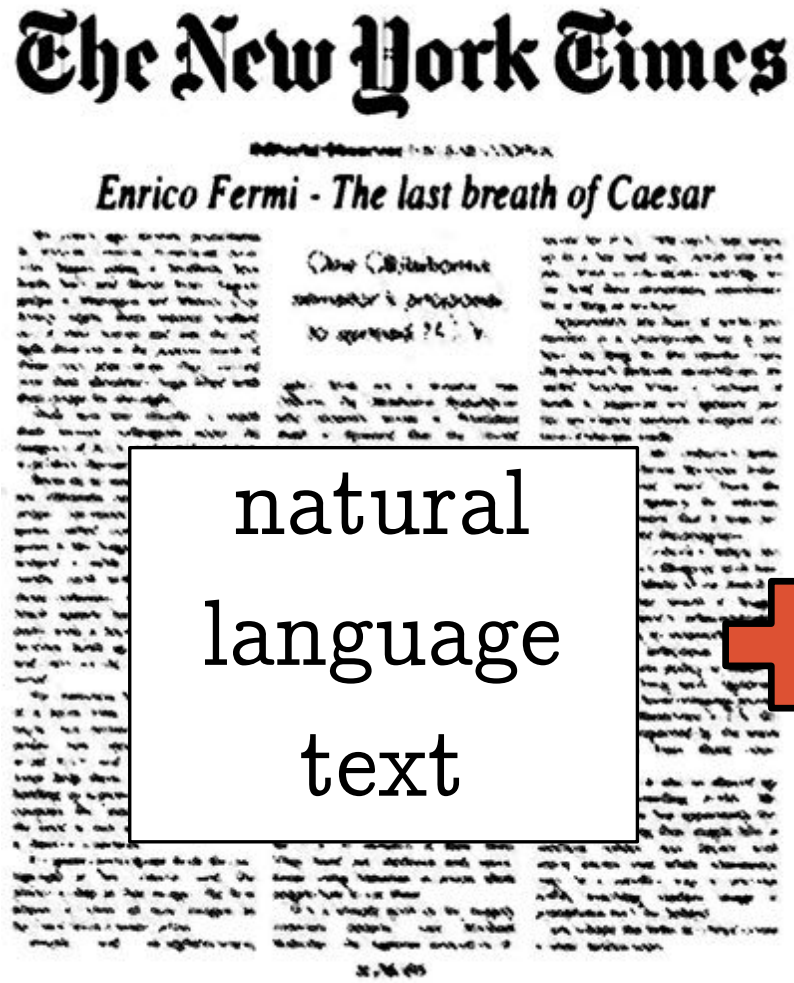
- KB represented as a graph

Knowledge bases challenges

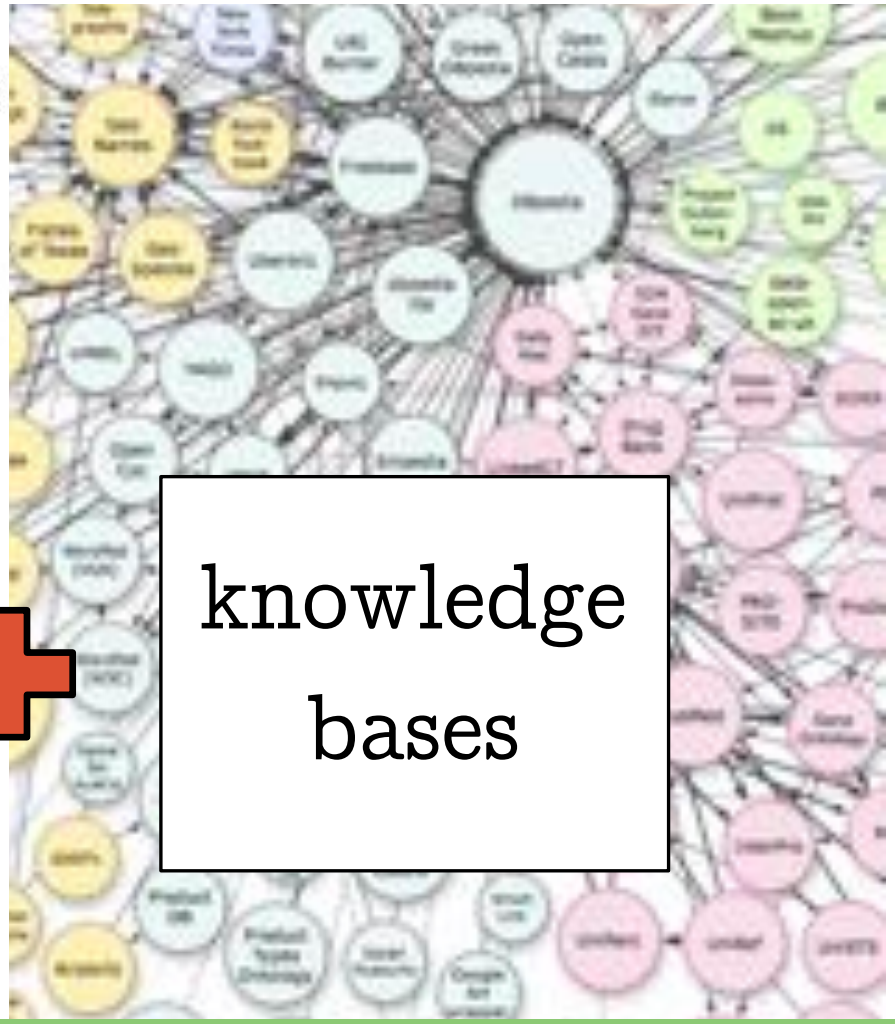
- large-scale example: LOD contains over 60B facts (sub-rel-obj)
- despite the size, KBs are <u>far from being complete</u>
  - 75% people have unknown nationality in *Freebase*
  - 71% people with place of birth attribute missing in *Freebase*
- data not only incomplete but also uncertain, noisy or false
- challenges:
  - fill missing information
  - remove incorrect facts
- idea: scan the web extracting new information

Open relation extraction

- open relation extraction is the task of extracting new facts for a potentially unbounded set of relations from various sources

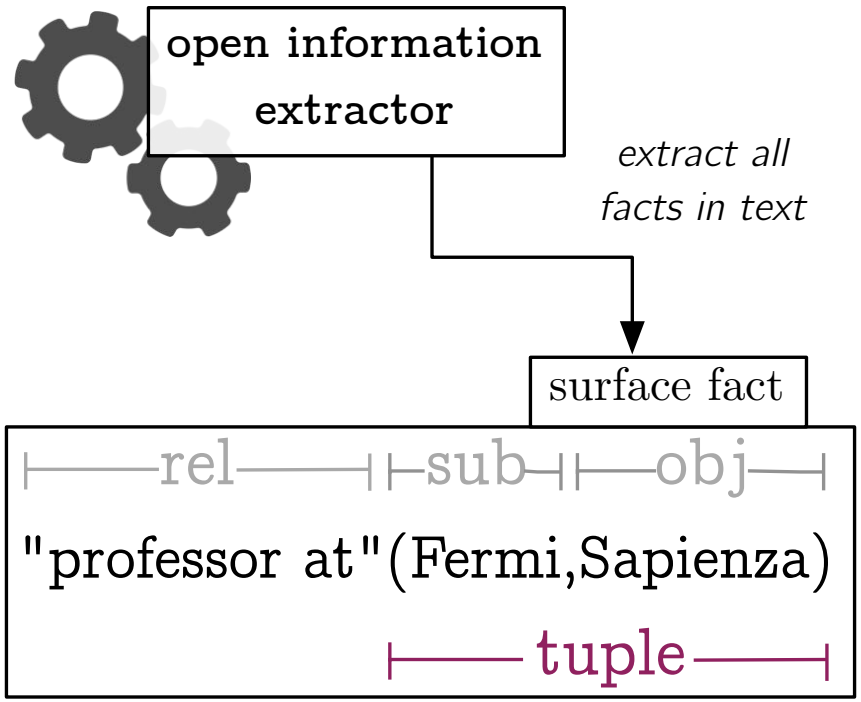


natural language text



knowledge bases

Input data: facts from natural language text



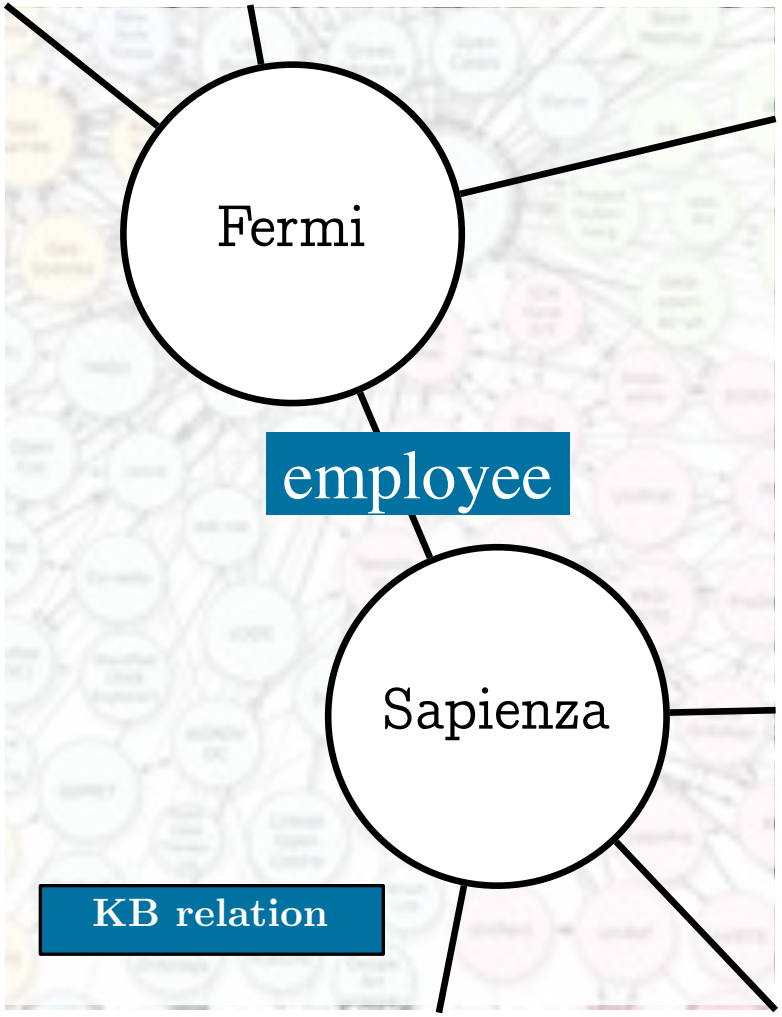
Input data: facts from knowledge bases

KB fact

employee(Fermi,Sapienza)

"professor at"(Fermi,Sapienza)

surface fact



Matrix completion for open relation extraction

	born in	professor at	mayor of	employee
(Caesar,Rome)	1			
(Fermi,Rome)	1			
(Fermi,Sapienza)		1		1
(de Blasio,NY)			1	

tuples x relations

surface relation **KB relation**

Matrix completion for open relation extraction

	born in	professor at	mayor of	employee
(Caesar,Rome)	1	?	?	?
(Fermi,Rome)	1	?	?	?
(Fermi,Sapienza)	?	1	?	1
(de Blasio,NY)	?	?	1	?

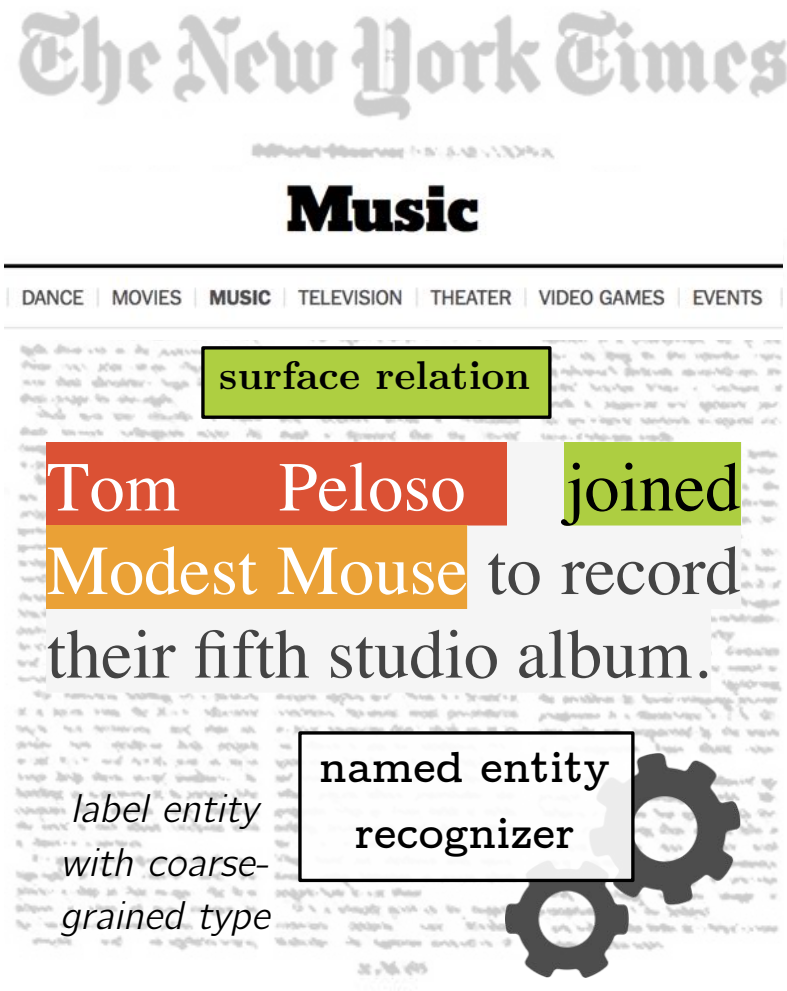
tuples x relations

surface relation **KB relation**

Contributions

we propose CORE (context-aware open relation extraction) that integrates contextual information into such models to improve prediction performance

Contextual information



unspecific relation

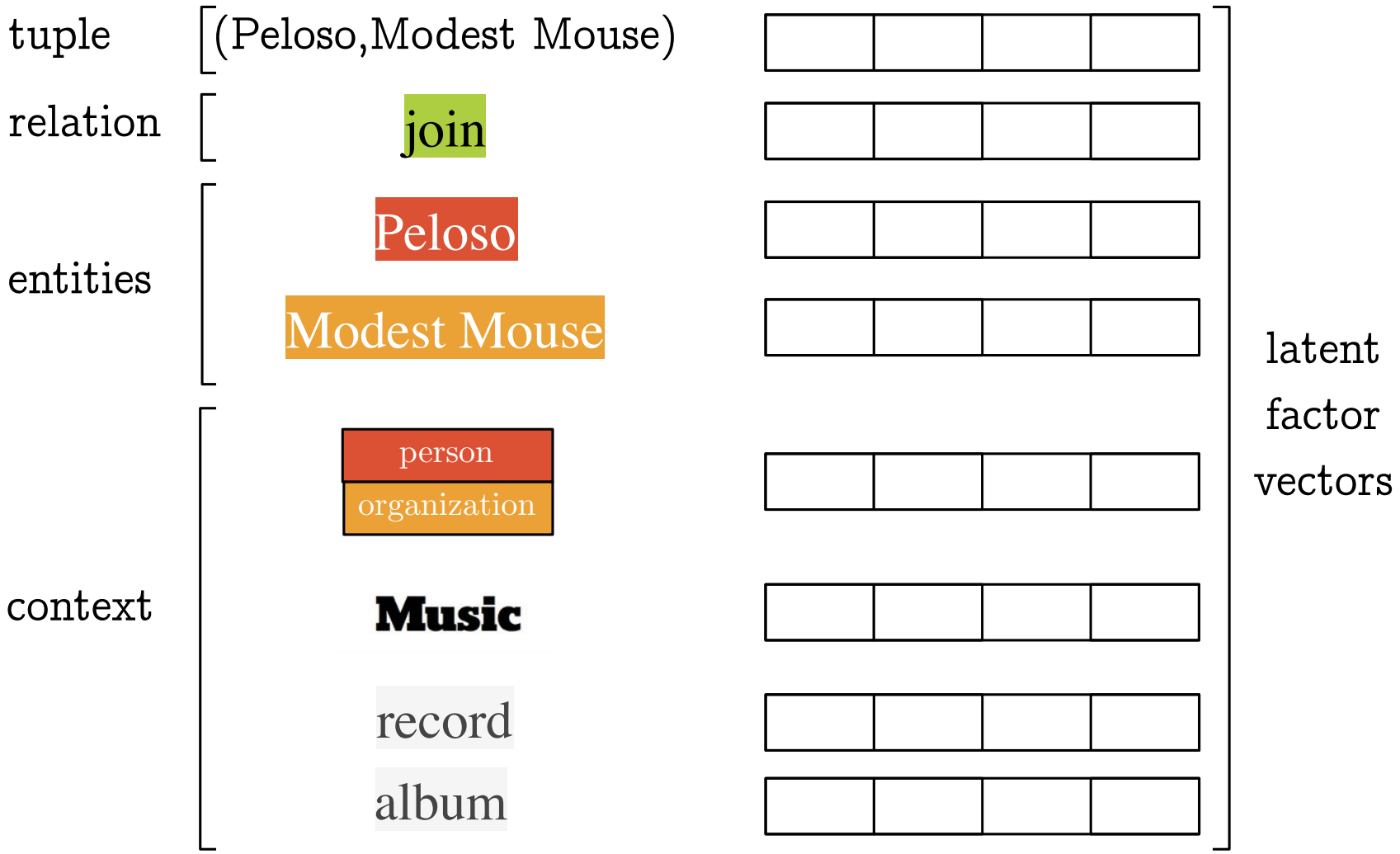
"join"(Peloso,Modest Mouse)

Contextual information

entity types	person, organization
article topic	Music
words	record, album

CORE - latent representation of variables

- associates latent representations $\boldsymbol{f}_v$ with each variable $v \in V$



CORE - modeling facts

	Surface		KB								Context				
	"born in"(x,y)	"professor at"(x,y)	employee(x,y)	Caesar,Rome	Fermi,Rome	Fermi,Sapienza	Caesar	Rome	Fermi	Sapienza	person, organization	person, location	physics	history	...
x_1	1	0	0	1	0	0	0.5	0.5	0	0	0	1	0	1	
x_2	0	1	0	0	0	1	0	0	0.5	0.5	1	0	1	0	
x_3	1	0	0	0	1	0	0	0.5	0.5	0	0	1	0.6	0.4	
x_4	0	0	1	0	0	1	0	0	0.5	0.5	1	0	1	0	
	relations			tuples			entities				tuple types		tuple topics		

- models the input data in terms of a matrix in which each row corresponds to a fact x and each column to a variable v
- groups columns according to the type of the variables
- in each row the values of each column group sum up to unity

CORE - modeling context

	Surface		KB								Context				
	"born in"(x,y)	"professor at"(x,y)	employee(x,y)	Caesar,Rome	Fermi,Rome	Fermi,Sapienza	Caesar	Rome	Fermi	Sapienza	person, organization	person, location	physics	history	...
x_1	1	0	0	1	0	0	0.5	0.5	0	0	0	1	0	1	
x_2	0	1	0	0	0	1	0	0	0.5	0.5	1	0	1	0	
x_3	1	0	0	0	1	0	0	0.5	0.5	0	0	1	0.6	0.4	
x_4	0	0	1	0	0	1	0	0	0.5	0.5	1	0	1	0	
	relations			tuples			entities				tuple types		tuple topics		

- aggregates and normalizes contextual information by tuple
  - a fact can be observed multiple times with different context
  - there is no context for new facts (never observed in input)
- this approach allows us to provide comprehensive contextual information for both observed and unobserved facts

CORE - factorization model

	Surface		KB								Context				
	"born in"(x,y)	"professor at"(x,y)	employee(x,y)	Caesar,Rome	Fermi,Rome	Fermi,Sapienza	Caesar	Rome	Fermi	Sapienza	person, organization	person, location	physics	history	...
x_1	1	0	0	1	0	0	0.5	0.5	0	0	0	1	0	1	
x_2	0	1	0	0	0	1	0	0	0.5	0.5	1	0	1	0	
x_3	1	0	0	0	1	0	0	0.5	0.5	0	0	1	0.6	0.4	
x_4	0	0	1	0	0	1	0	0	0.5	0.5	1	0	1	0	
	relations			tuples			entities				tuple types		tuple topics		

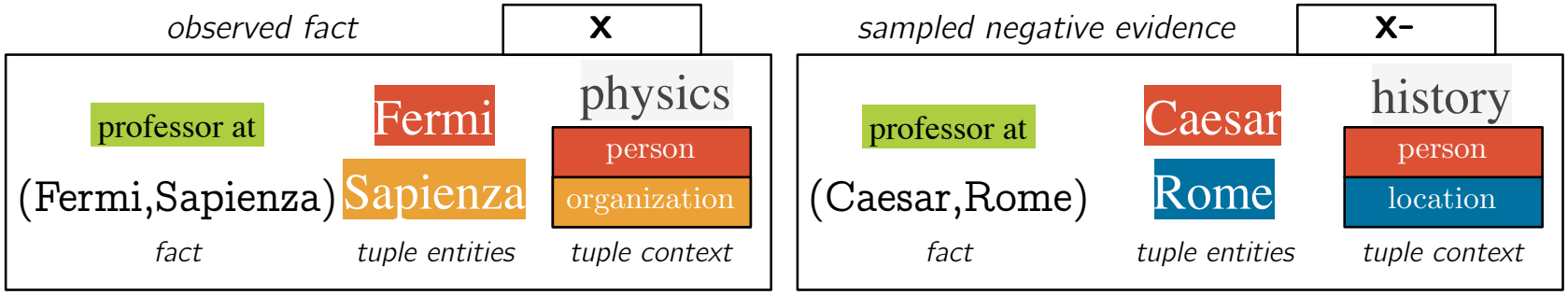
- uses factorization machines as underlying framework
- associates a score $s(x)$ with each fact x

$$s(x) = \sum_{v_1 \in V} \sum_{v_2 \in V \setminus \{v_1\}} x_{v_1} x_{v_2} \boldsymbol{f}_{v_1}^T \boldsymbol{f}_{v_2}$$

- weighted pairwise interactions of latent factor vectors

CORE - parameter estimation

- parameters: $\Theta = \{ \boldsymbol{f}_v \mid v \in V \}$
- Bayesian personalized ranking, all observations are positive
- goal: produce a ranked list of tuples for each relation



- pairwise approach, **x** is more likely to be true than **x-**

$$\text{maximize} \quad \boxed{\sum_{x} f(s(\mathbf{x}) - s(\mathbf{x}\text{-}))}$$

- stochastic gradient ascent

$$\Theta \leftarrow \Theta + \eta \nabla_{\Theta} \left(\boxed{} \right)$$

Experiments - dataset

440k facts extracted from The New York Times corpus

entity mentions linked using string matching

15k facts from Freebase

- Contextual information

article metadata

news desk (e.g., foreign desk)
descriptors (e.g., finances)
online section (e.g., sports)
section (e.g., a, d)
publication year

m

entity type

person
organization
location
miscellaneous

t

bag-of-word

sentences where the fact has been extracted

w

- letters to indicate contextual information considered

Experiments - methodology

- we consider (to keep experiments feasible):

10k tuples	**19** Freebase relations	**10** surface relations

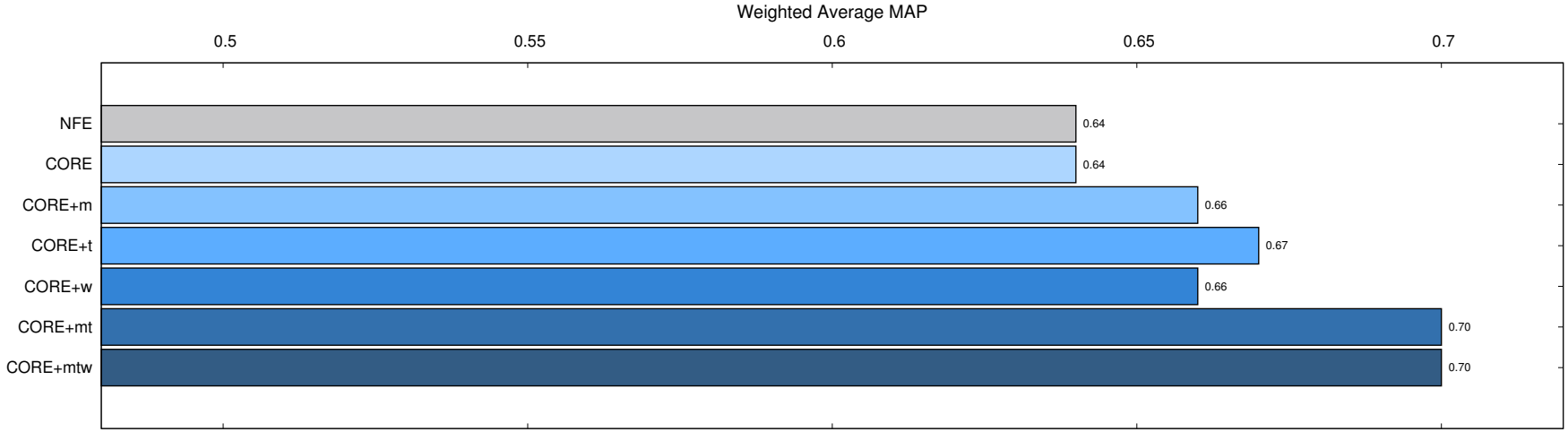
- for each relation and method:
  - we rank the tuples subsample
  - we consider the top-100 predictions and label them manually
- evaluation metrics:

number of true facts	**MAP** (quality of the ranking)

- methods:
  - **PITF**, tensor factorization method (designed to work in-KB)
  - **NFE**, matrix completion method (best context-agnostic)
  - **CORE**, uses relations, tuples and entities as variables
  - **CORE**+m, +t, +w, +mt, +mtw

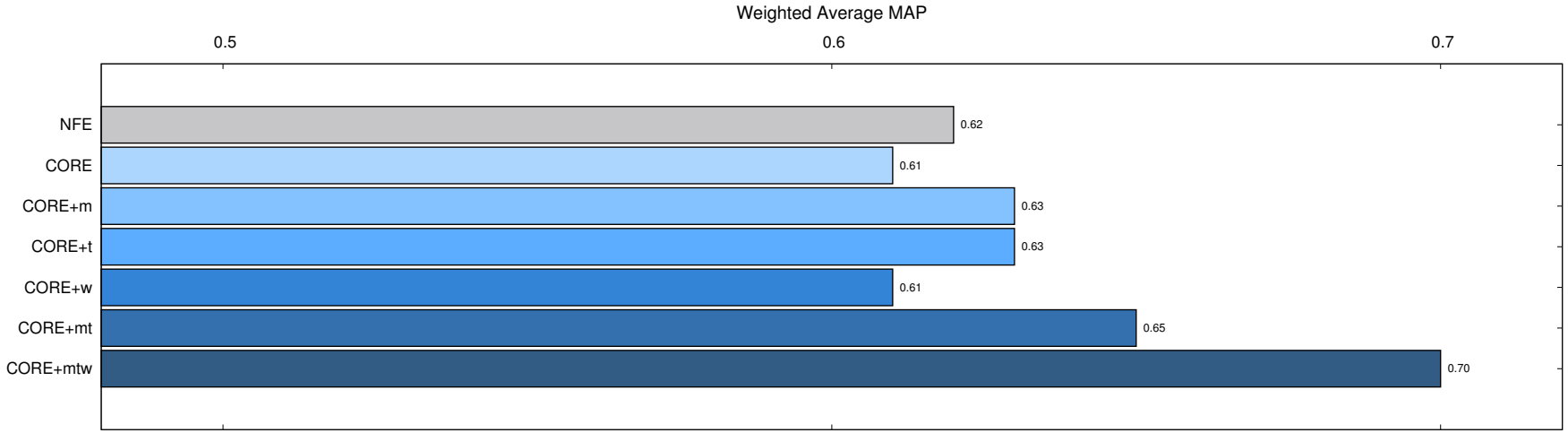
Results - Freebase relations

Relation	#	PITF	NFE	CORE	CORE+m	CORE+t	CORE+w	CORE+mt	CORE+mtw
person/company	208	70 (0.47)	92 (0.81)	91 (0.83)	90 (0.84)	91 (0.87)	92 (0.87)	95 (0.93)	**96** (**0.94**)
person/place_of_birth	117	1 (0.0)	92 (*0.9*)	90 (0.88)	92 (*0.9*)	92 (*0.9*)	89 (0.87)	**93** (*0.9*)	92 (*0.9*)
location/containedby	102	7 (0.0)	63 (0.47)	62 (0.47)	63 (0.46)	61 (0.47)	61 (0.44)	62 (0.49)	**68** (**0.55**)
parent/child	88	9 (0.01)	64 (0.6)	64 (0.56)	64 (0.59)	64 (0.62)	64 (0.57)	67 (**0.67**)	**68** (0.63)
person/place_of_death	71	1 (0.0)	67 (0.93)	67 (0.92)	*69* (*0.94*)	67 (0.93)	67 (0.92)	*69* (*0.94*)	67 (0.92)
person/parents	67	20 (0.1)	51 (0.64)	52 (0.62)	51 (0.61)	49 (0.64)	47 (0.6)	*53* (**0.67**)	*53* (0.65)
author/works_written	65	24 (0.08)	45 (0.59)	49 (0.62)	51 (0.69)	50 (0.68)	50 (0.68)	51 (**0.7**)	**52** (0.67)
person/nationality	61	21 (0.08)	25 (0.19)	27 (0.17)	28 (0.2)	26 (0.2)	**29** (0.19)	27 (0.18)	27 (**0.21**)
neighbor./neighborhood_of	39	3 (0.0)	24 (0.44)	23 (0.45)	26 (0.5)	27 (0.47)	27 (0.49)	*30* (0.51)	*30* (**0.52**)
...									
Average $MAP_{\#}^{100}$		0.09	0.46	0.47	0.49	0.47	0.49	0.49	**0.51**
Weighted Average $MAP_{\#}^{100}$		0.14	0.64	0.64	0.66	0.67	0.66	*0.70*	*0.70*



Results - surface relations

Relation	#	PITF	NFE	CORE	CORE+m	CORE+t	CORE+w	CORE+mt	CORE+mtw
head	162	34 (0.18)	80 (0.66)	83 (0.66)	82 (0.63)	76 (0.57)	77 (0.57)	83 (0.69)	**88** (**0.73**)
scientist	144	44 (0.17)	76 (0.6)	74 (0.55)	73 (0.56)	74 (0.6)	73 (0.59)	*78* (0.66)	*78* (**0.69**)
base	133	10 (0.01)	85 (0.71)	86 (0.71)	86 (0.78)	88 (0.79)	85 (0.75)	83 (0.76)	**89** (**0.8**)
visit	118	4 (0.0)	73 (0.6)	75 (0.61)	76 (0.64)	80 (0.68)	74 (0.64)	75 (0.66)	**82** (**0.74**)
attend	92	11 (0.02)	65 (0.58)	64 (0.59)	65 (0.63)	62 (0.6)	66 (0.63)	62 (0.58)	**69** (**0.64**)
adviser	56	2 (0.0)	42 (0.56)	**47** (0.58)	44 (0.58)	43 (0.59)	45 (*0.63*)	43 (0.53)	44 (*0.63*)
criticize	40	5 (0.0)	31 (0.66)	33 (0.62)	33 (**0.7**)	33 (0.67)	33 (0.61)	35 (0.69)	**37** (0.69)
support	33	3 (0.0)	19 (0.27)	22 (*0.28*)	18 (0.21)	19 (*0.28*)	22 (0.27)	**23** (0.27)	21 (0.27)
praise	5	0 (0.0)	2 (0.0)	2 (0.01)	4 (*0.03*)	3 (0.01)	3 (0.02)	**5** (*0.03*)	2 (0.01)
vote	3	2 (0.01)	*3* (0.63)	*3* (0.63)	*3* (0.32)	*3* (0.49)	*3* (0.51)	*3* (0.59)	*3* (**0.64**)
Average $MAP_{\#}^{100}$		0.04	0.53	0.53	0.51	0.53	0.53	0.55	**0.59**
Weighted Average $MAP_{\#}^{100}$		0.08	0.62	0.61	0.63	0.63	0.61	0.65	**0.70**



Anecdotal results

author(x,y)

ranked list of tuples	similar relations
1 (Winston Groom, Forrest Gump)	0.98 "reviews x by y"(x,y)
2 (D. M. Thomas, White Hotel)	0.97 "book by"(x,y)
3 (Roger Rosenblatt, Life Itself)	0.95 "author of"(x,y)
4 (Edmund White, Skinned Alive)	0.95 " 's novel"(x,y)
5 (Peter Manso, Brando: The Biography)	0.95 " 's book"(x,y)

"scientist at"(x,y)

ranked list of tuples	similar relations
1 (Riordan Roett, Johns Hopkins University)	0.87 "scientist"(x,y)
2 (Dr. R. M. Roberts, University of Missouri)	0.84 "scientist with"(x,y)
3 (Linda Mayes, Yale University)	0.80 "professor at"(x,y)
4 (Daniel T. Jones, Cardiff Business School)	0.79 "scientist for"(x,y)
5 (Russell Ross, University of Iowa)	0.78 "neuroscientist at"(x,y)

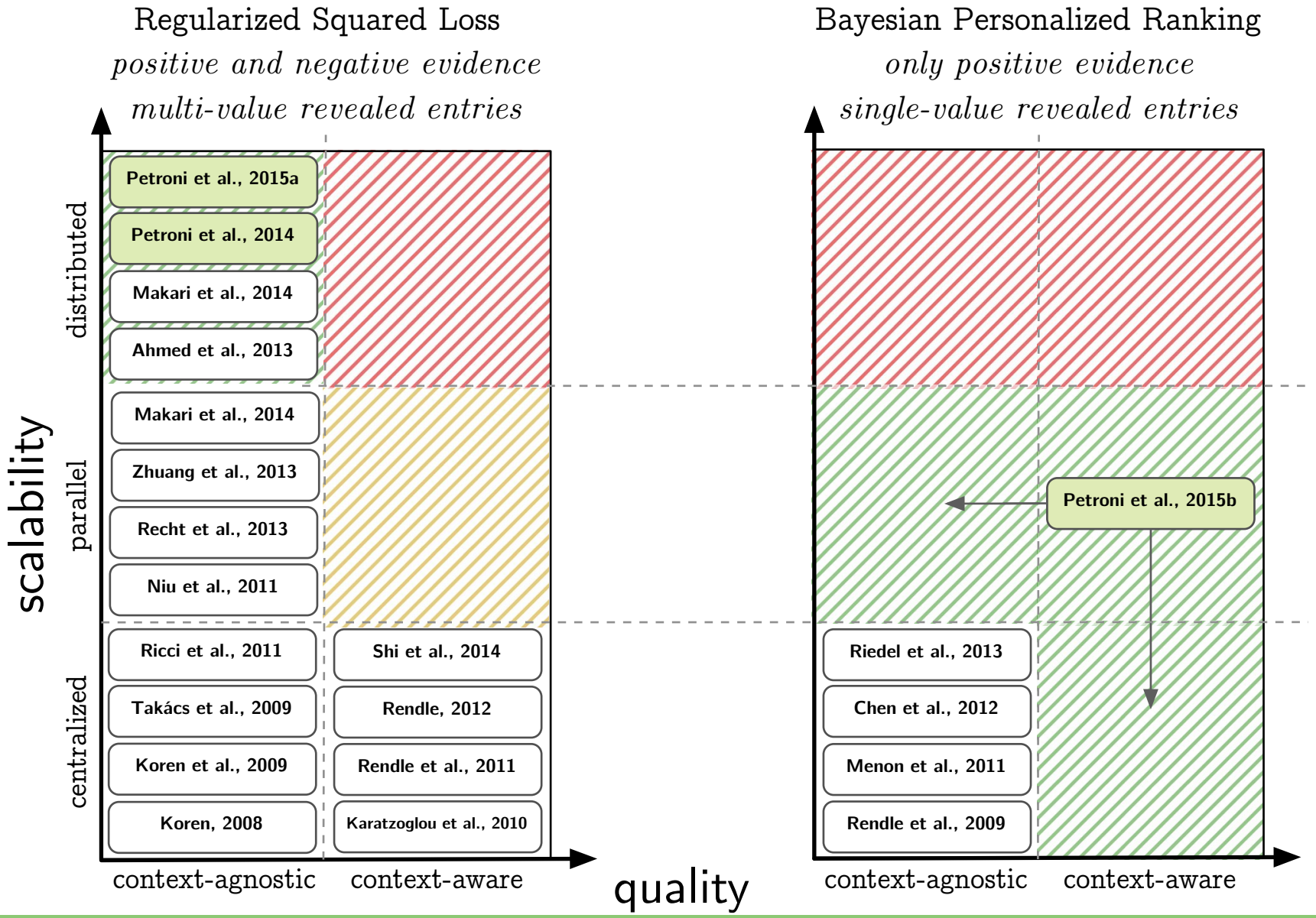
- semantic similarity of relations is one aspect of our model
- similar relations treated differently in different contexts

Contents

Conclusion

- we tackle the **scalability** and the **quality** of MC
- we investigate graph partitioning techniques for ASGD
- we propose HDRF, one-pass v-cut graph partitioning alg
  - exploit power-law nature of real-word graphs
  - provides minimum replicas with close to optimal load balance
  - significantly reduces the time needed to perform computation
- we propose CORE, a matrix completion model for open relation extraction that incorporates contextual information
  - based on factorization machines and BPR
  - extensible model, additional information can be integrated
  - exploiting context significantly improve prediction quality
- all code released `https://github.com/fabiopetroni`

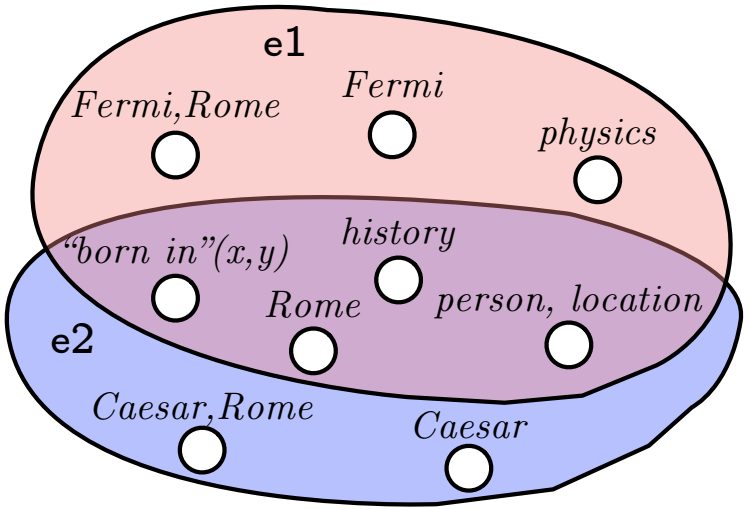
Overview - Future work



Future Directions

- distribute training for context-aware matrix completion
  - asynchronous approach: local vector copies, synchronization
  - challenging input placement, the input describes an hypergraph

- adaptation of the HDRF algorithm to hypergraphs
  - do high degree vertices play a crucial role?



- distributed Bayesian personalized ranking
  - sampling of a negative counterpart for each training point
  - sampling from the local portion of the dataset in current node?

Thank you!

Questions?

Fabio Petroni
Sapienza University of Rome, Italy