Interactions Between Communication and Computation in Emerging Systems

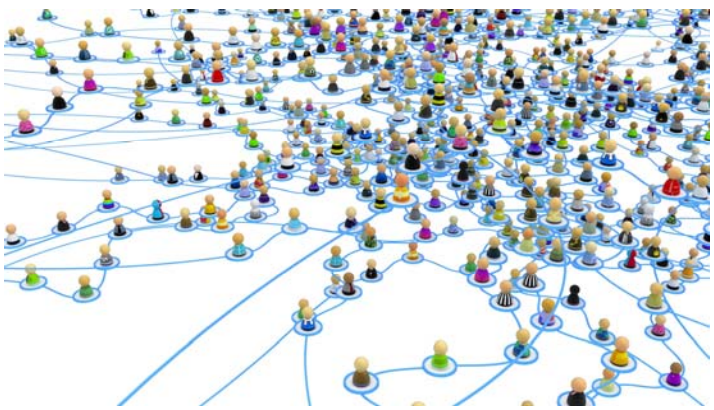
Linqi Song
Computer Science Department
City University of Hong Kong
Joint work with Christina Fragouli @ UCLA

Chinese University of Hong Kong
July 20, 2018

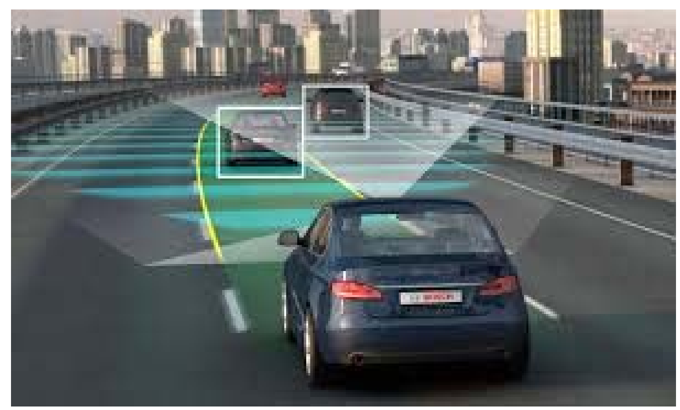
Motivation: promising big data applications

E-commerce

Medical & healthcare



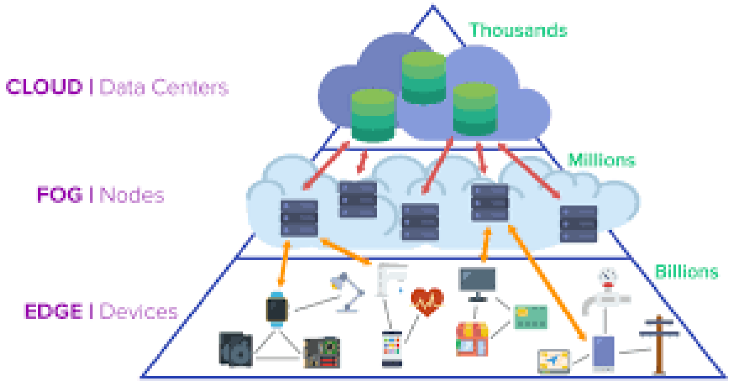
Social networks



Transportation

Motivation: innovative technologies for big data

IoT



Cloud, fog, and edge computing

Distributed computing systems

Cyber-physical systems

Challenges and opportunities

New challenges
comm. - compt. interactions

Heavy traffic

Towards distributed

Ubiquitous

Security & privacy

Single domain knowledge & methods insufficient

Joint design framework

Overview of research: interdisciplinary area

Applications

Smart grid | Medical, healthcare | Distributed system

Communication | Big data | Recommender system

Theories

Information theory | Combinatorics | Statistics

Algorithms & complexity | Graph theory

Machine learning | Game theory | Optimization

Theories: understanding of comm. - compt. interactions

Applications: scheme design

Overview of recent research

- Novel communication paradigms
  - Content-type coding [TIT’18, NetCod’15, ISIT’16], to increase communication efficiency for big data traffic
  - Privacy [ISIT’17C, ISIT’18, ITW’17B], to protect privacy of users in same broadcast domain
- Learning and communications for recommendations
  - Online learning algorithms for recommender systems [TSC’16]
  - Communication and user preference trade-off [ISIT’17A, TIT’18]
- Data shuffling for distributed machine learning
  - Communication and computational performance trade-off [ISIT’17B, ITW’17A, *arXiv*’17, submitted to TIT]

Learning and communications in recommender systems

Recommender systems

- Conventional recommender systems recommend items to users based on their preferences.

- Challenges

Personalization & contextualization

Scalability

Cold start

- Unknown preferences: to learn preference
  Collaborative filtering [Adomavicius2005], Reinforcement learning [Ricci2011]
- Known preferences: to group of users
  Rank aggregation (rank based, score based, etc.) [Borda1781, Dwork2001]

Recommender systems

- Conventional recommender systems recommend items to users based on their preferences.

- Challenges

Personalization & contextualization

Scalability

Cold start

Not addressing all these challenges!

Considered contextual learning framework

- Contextual recommendations in a multi-armed bandit framework

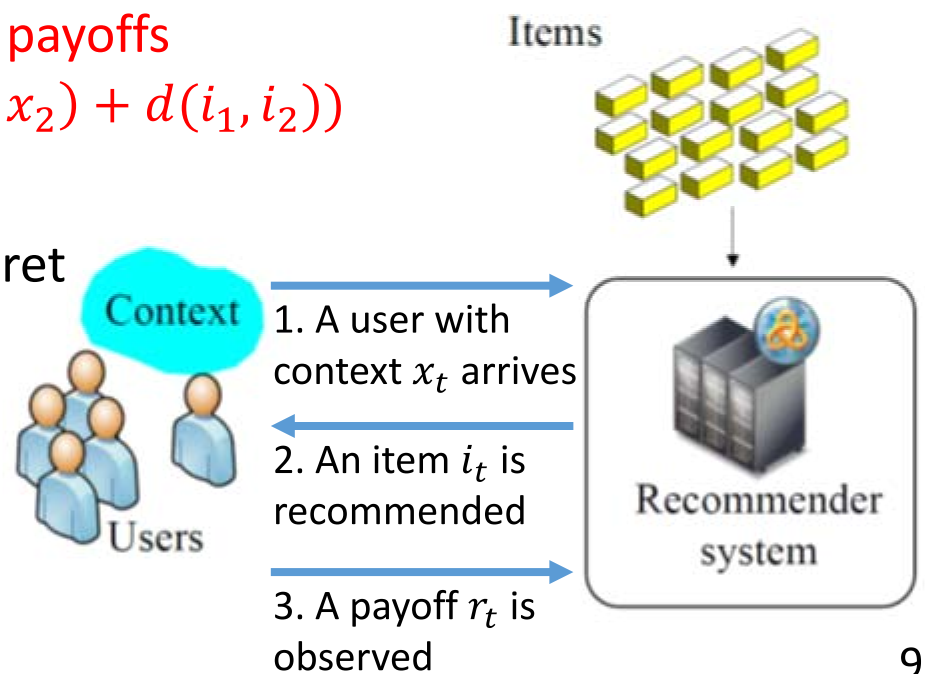
  for time $t = 1,2,\ldots$
  - Context arrival (unknown process) & observation
  - Item recommendation
  - Payoff observation
- Basic assumptions for recommendations
  - $r_t = r_t(x_t, i_t)$ i.i.d. distributed with mean $\mu(x_t, i_t)$
  - Similar contexts/items have similar payoffs

    $|\mu(x_1, i_1) - \mu(x_2, i_2)| \leq L(d(x_1, x_2) + d(i_1, i_2))$
- Learning goal
  - Learning algorithm to minimize regret

$$R(T) = \mathbb{E} \sum_{t=1}^{T} [\mu(x_t, i^*(x_t)) - r_t(x_t, i_t)]$$

Best possible action

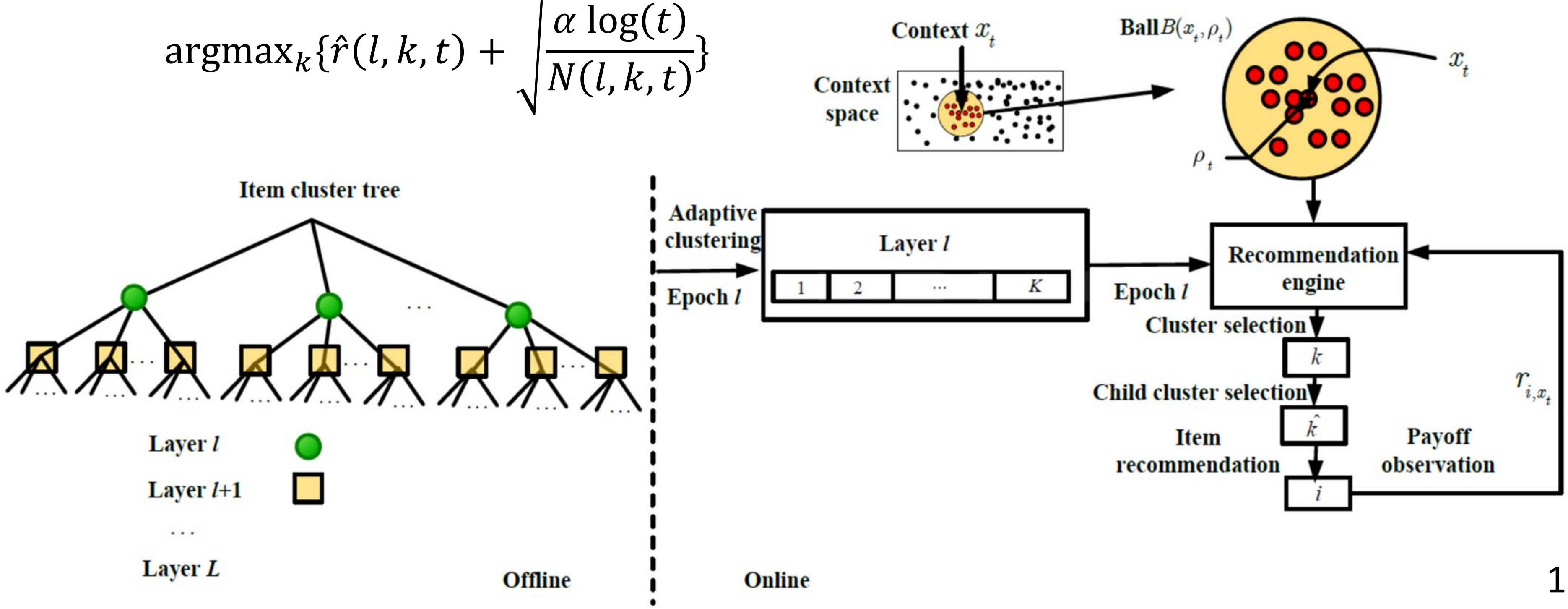
Alg’s action



Proposed online contextual learning algorithm

- Item-cluster tree
  - Offline
  - $d(x_1, x_2) < d(x_1, x_3)$, if x_1, x_2 belong to a smaller cluster than x_1, x_3
- Adaptive context neighborhood - Finer over time
- Cluster recommendation
  - Index based: exploitation + exploration

$$\mathrm{argmax}_k\{\hat{r}(l,k,t) + \sqrt{\frac{\alpha \log(t)}{N(l,k,t)}}\}$$



Performance of algorithm

- Address the challenges
  - Contextual framework -> personalization
  - Clustering of items, neighborhood of contexts -> scalability
  - Exploration-exploitation balance -> cold start

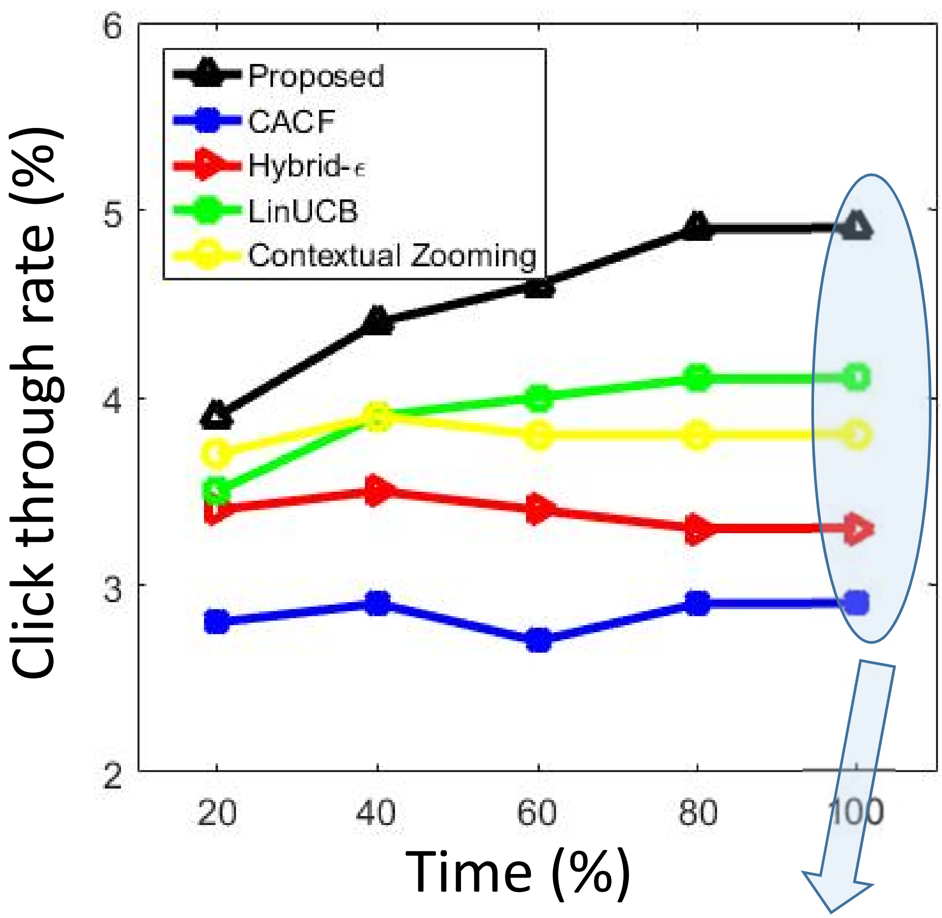
- Regret (matches the upper bound in literature [Lu’10][Slivkins’14])

$$R(T) = O(T^{\frac{d_X+d_I+1}{d_X+d_I+2}} \log(T))$$

d_X, d_I are the covering dimensions of the context and item spaces

Experimental result

Yahoo! Today Module (news) dataset



Proposed learning algorithm outperforms existing algorithms by 20% !

Does bandwidth matter?

Video 1
Quality: low (480p)
Required bandwidth: low

Video 2
Quality: high (1080p)
Required bandwidth: high

Given
limited bandwidth

Recommender systems in fog computing

- Shopping mall, coupon recommendation example
- User preference + limited bandwidth

Coupons: b_1, b_2, b_3, b_4

Wi-Fi hotspot

Wireless user 1
Has b_1, b_3
Prefer b_2 to b_4

Wireless user 2
Has b_1, b_2, b_4
Prefer b_3

Wireless user 3
Has b_3, b_4
Prefer b_1 to b_2

e.g.,
Case 1: bandwidth=3
transmit b_1 & b_2 & b_3

Case 2: bandwidth=2
transmit $b_2 + b_3$ & b_1

Case 3: bandwidth=1
transmit $b_2 + b_3$

Recommender systems in fog computing

- Shopping mall, coupon recommendation example
- User preference + limited bandwidth

e.g.,

Case 1: bandwidth=3
transmit b_1 & b_2 & b_3

Case 2: bandwidth=2
transmit $b_2 + b_3$ & b_1

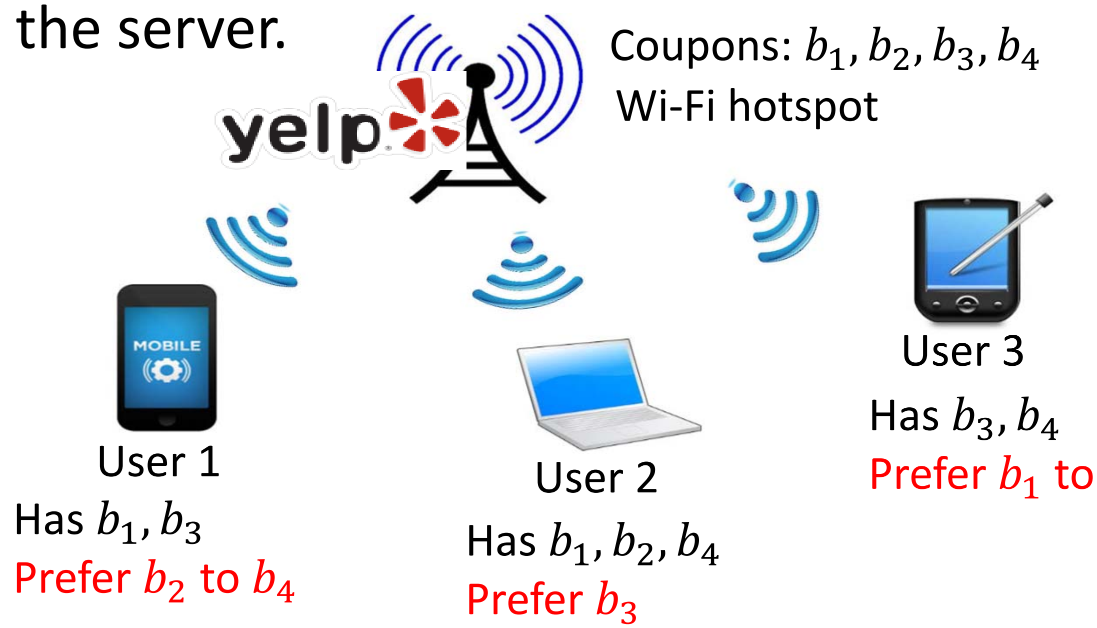
Case 3: bandwidth=1
transmit $b_2 + b_3$

Coding gain

Benefit-bandwidth trade-off

System model

- A server and n users with different contexts.
- m messages (i.e., coupons) to be recommended to the users.
- The server can broadcast encoded messages to users.
  - Bandwidth constraint K = allowed # broadcastings
- Each user has pre-downloaded some messages (side information).
- Each user has a preference over un-downloaded messages, depending on the preference model.
- All information known to the server.



Preference model

- Preference matrix $n \times m$

Messages

Users

$$\begin{array}{cc} & \begin{array}{cccc} b_1 & b_2 & b_3 & b_4 \end{array} \\ \begin{array}{c} c_1 \\ c_2 \\ c_3 \end{array} & \begin{bmatrix} X & 2 & X & 1 \\ X & X & 2 & X \\ 2 & 1 & X & X \end{bmatrix} \end{array}$$

Individual preference $s(i,j)$

Side information

- User i's individual preference for message j: $s(i,j)$.
  - Direct score $s(i,j) \geq 0$.
  - Borda score model: a user has scores of a permutation of $[1\!:\!r]$ for r undownloaded messages

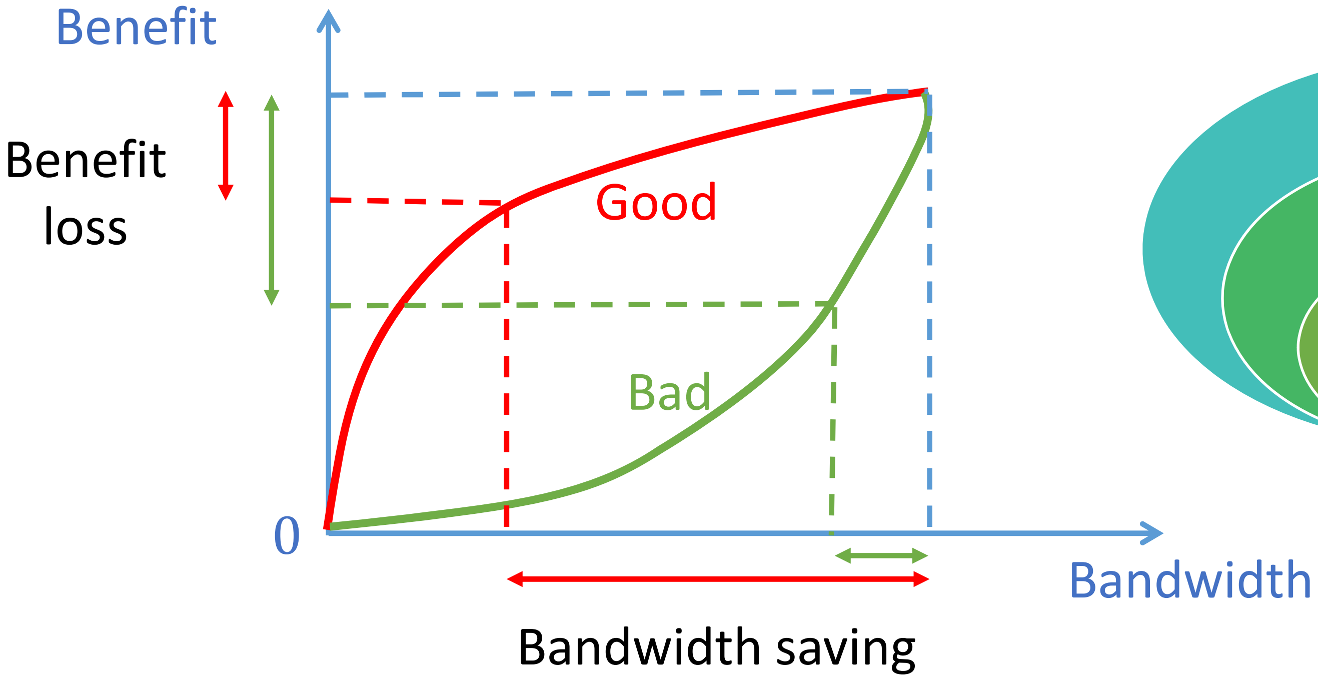
Messages:	b_1	b_2	b_3	b_4
Borda score:	3	1	X	2

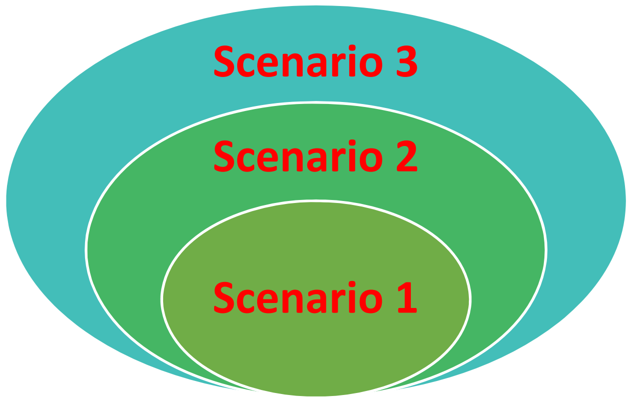
- Benefits collected after transmissions
  - User i receives benefit $s_i = \max\{s(i,j)\}$ among the decoded messages.
  - Total benefit B is the aggregate of users' benefits.

$$\boldsymbol{B = \Sigma_i s_i}$$

Problem formulation: benefit vs. bandwidth

Design broadcast transmission schemes
Maximize benefit B, given bandwidth constraint K





Consistent benefit-bandwidth trade-off diminishing return!

Scenarios 1 and 2

Scenario 1

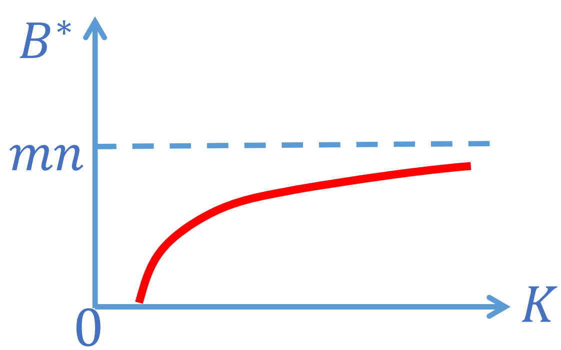
- No side info.
- Borda score model

$$\begin{array}{c c} & \begin{array}{cccc} b_1 & b_2 & b_3 & b_4 \end{array} \\ \begin{array}{c} c_1 \\ c_2 \\ c_3 \end{array} & \begin{bmatrix} 2 & 4 & 1 & 3 \\ 3 & 2 & 4 & 1 \\ 4 & 1 & 3 & 2 \end{bmatrix} \end{array}$$

- Uncoded transmission
- Optimal benefit

$$B^* = \Theta(mn(1 - 1/K))$$

- NP-hard
- Greedy algorithm -> $B^*/1.58$



Scenario 2

- Equal-size side info.
- Borda score model

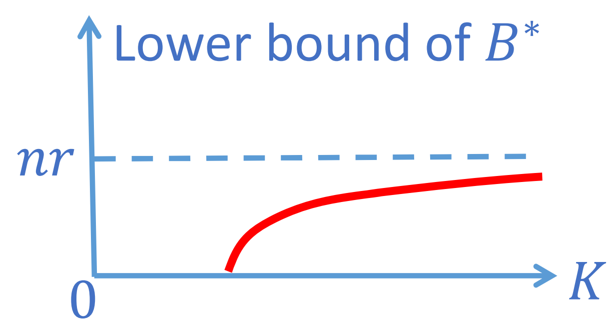
$$\begin{array}{c c} & \begin{array}{cccc} b_1 & b_2 & b_3 & b_4 \end{array} \\ \begin{array}{c} c_1 \\ c_2 \\ c_3 \end{array} & \begin{bmatrix} X & 2 & X & 1 \\ 1 & X & 2 & X \\ 2 & X & X & 1 \end{bmatrix} \end{array}$$

- DP-based coded transmission
- Optimal benefit

$$B^* \geq nr\left(1 - \frac{4e}{K} + \frac{12e}{K^2}\right), 5 \leq K \leq r$$

$$B^* = nr, K \geq r, \ B^* \geq C_K nr, K \leq 4$$

$(C_1 = 0.25/e, C_2 = 0.462/e, C_3 = 0.666/e, C_4 = 0.798/e)$

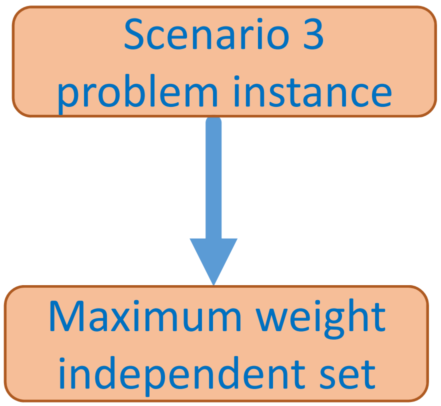


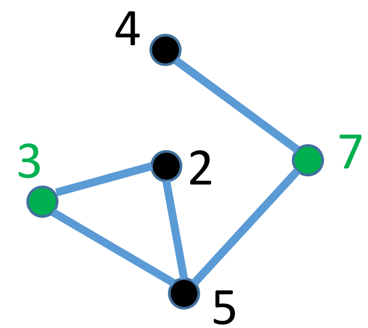
Scenario 3

Scenario 3

- Arbitrary-size side info.
- Arbitrary score model

$$\begin{array}{c c} & \begin{array}{cccc} b_1 & b_2 & b_3 & b_4 \end{array} \\ \begin{array}{c} c_1 \\ c_2 \\ c_3 \end{array} & \begin{bmatrix} X & 2 & X & 1 \\ X & X & 2 & X \\ 3 & 1 & X & 1 \end{bmatrix} \end{array}$$

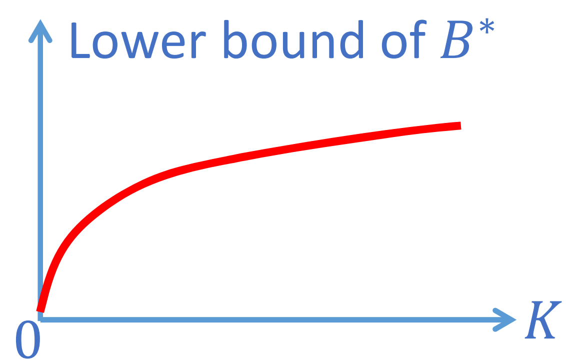




- Heuristic coded transmission
- Optimal benefit

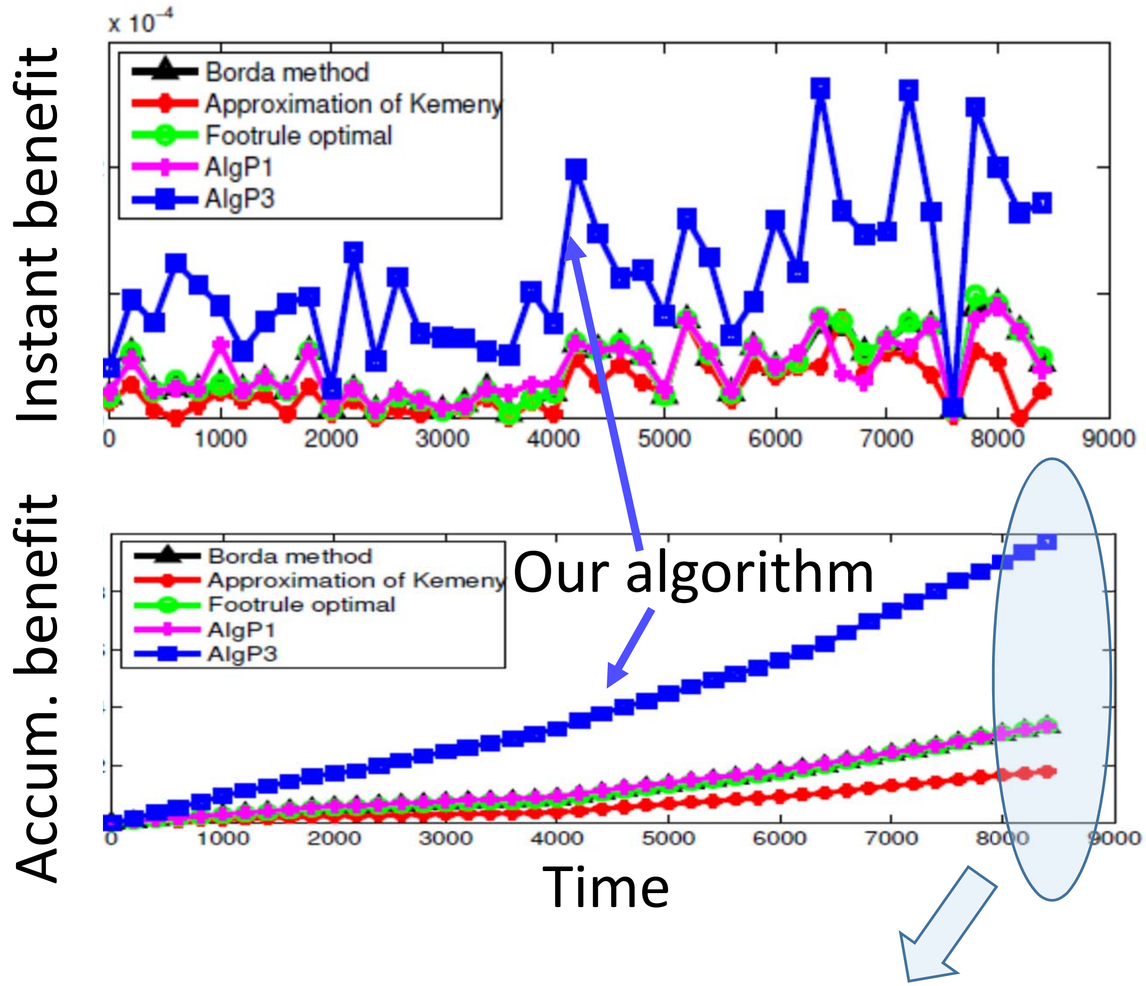
$$B^* \geq \sum_{k=1}^{K} MWIS(G_k)$$

G_k are sequentially constructed graphs,
$MWIS(G_k) \geq MWIS(G_{k+1})$



Experimental result

Yahoo! advertising dataset

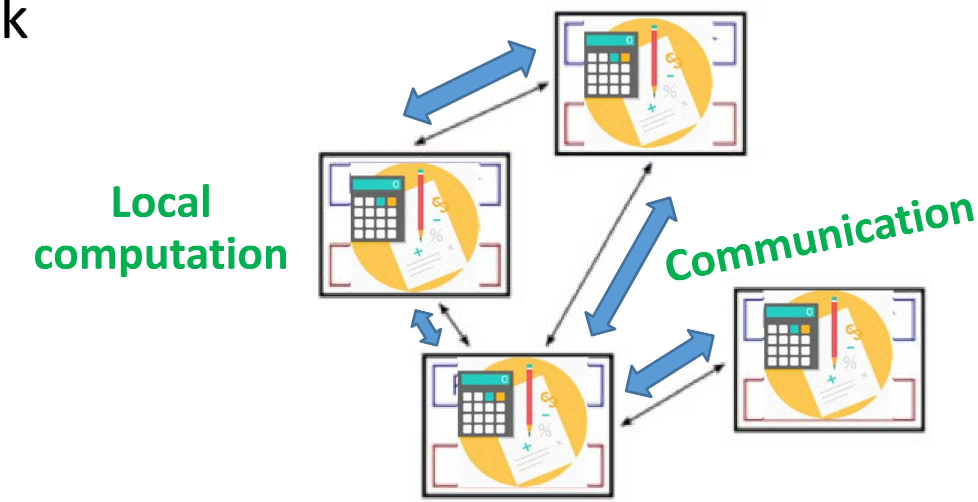


Proposed coding algorithm more than doubles the benefits (over uncoded ones)!

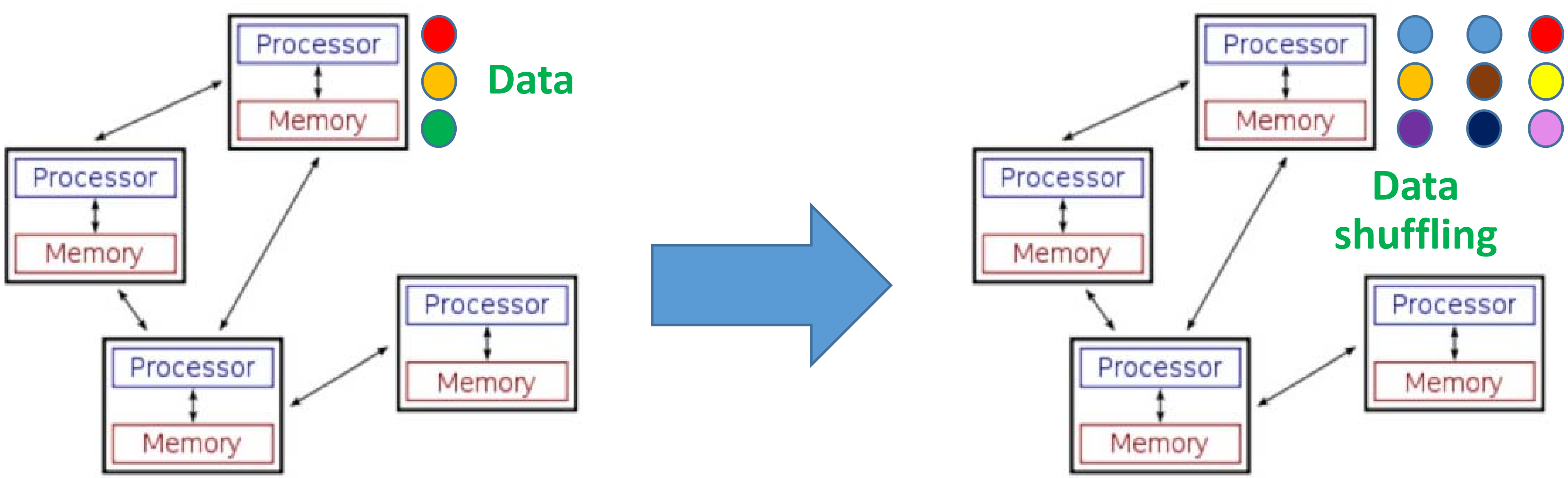
Data shuffling
for distributed machine learning

Data shuffling for distributed machine learning

- Massive data -> distributed machine learning
- Communication -> bottleneck
  - More than 30% runtime for Facebook [Chowdhury2011]

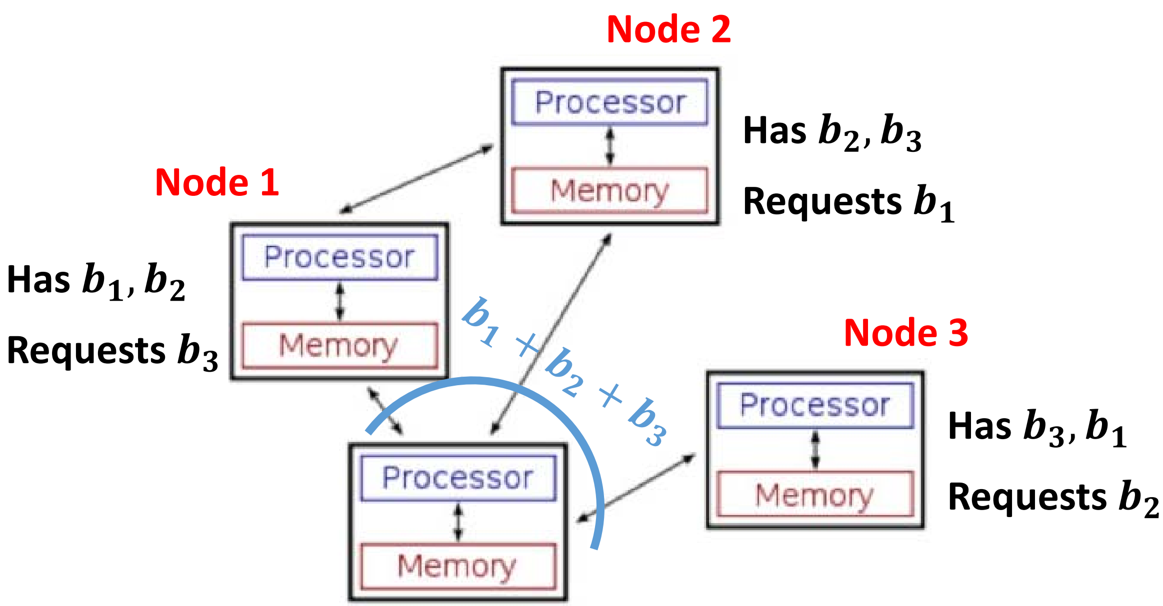


- Data shuffling -> statistical performance, robustness



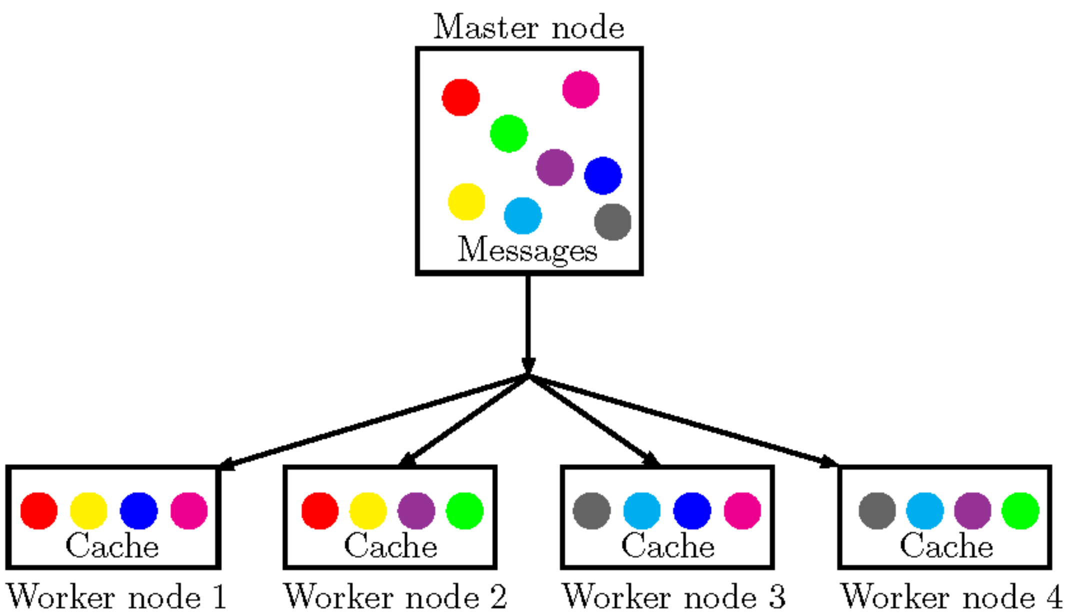
Coding helps!

- Recent trends: using coding
  - Index coding [Birk'98]
  - "Master-workers" structure [Lee'15]
  - "MapReduce" structure [Li'18]
- Redundancy creates coding opportunities
  - Similar to channel coding and network coding
  - Redundancy in computational and storage resources



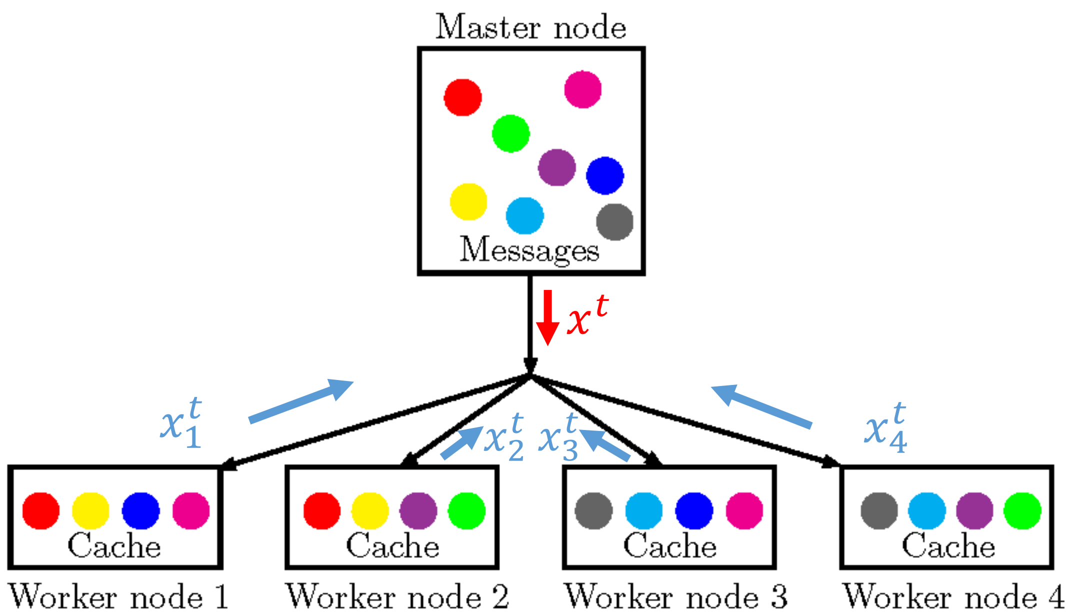
Considered system model

- One master node with all m messages (data) .
- n worker nodes, each worker i with
  - Cache of size s_i.
  - Cache state at iteration t: an indicator $z_i^t \in \{0,1\}^m$ to denote which message is cached for worker i.
- Master node can make broadcast transmissions to n workers.



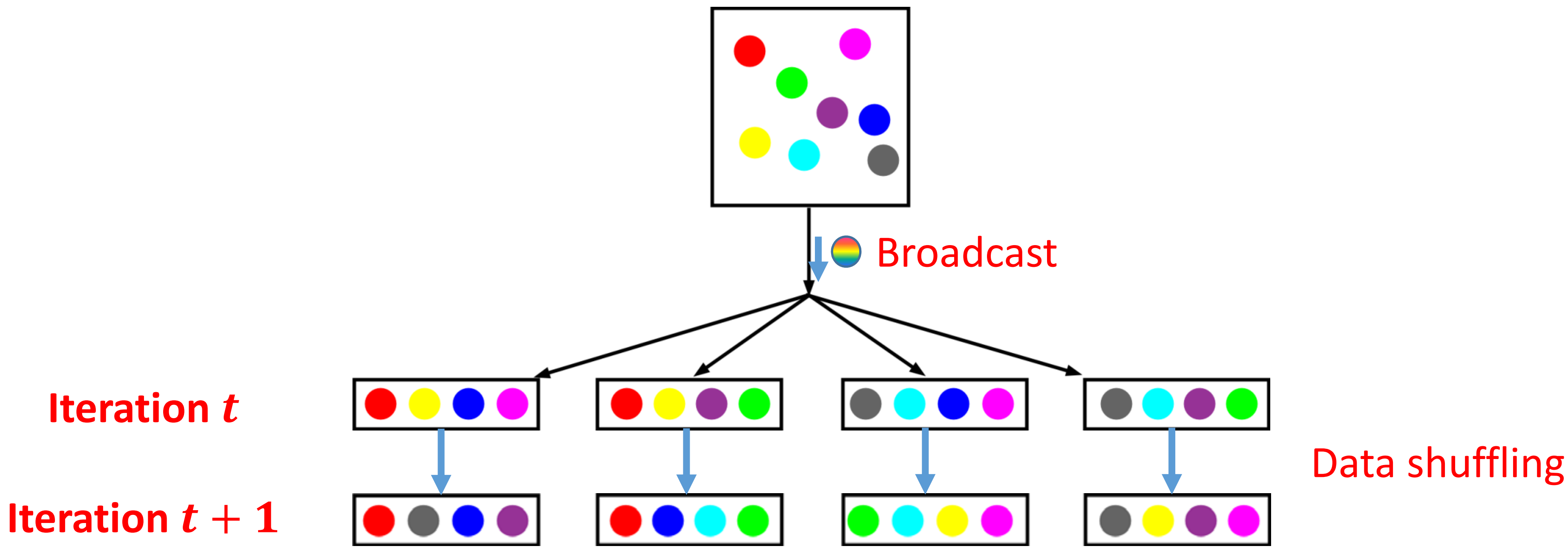
Computing process

- Distributed computational task: $x = g(\{b_j\}_{j\in[m]})$. E.g., classifier.
  - Operate in iterations $t = 1,2, \dots$
  - Local computation: $x_i^t = l_i(x^{t-1}, \{b_j\}_{j\in S_i^t})$. Return back.
  - Aggregation: $x^t = f(x_1^t, x_2^t, \dots, x_n^t)$. Broadcast.
  - Data shuffling: random refresh cache data -> statistical gain.



Computing process

- Distributed computational task: $x = g(\{b_j\}_{j\in[m]})$. E.g., classifier.
  - Operate in iterations $t = 1,2,\ldots$
  - Local computation: $x_i^t = l_i(x^{t-1}, \{b_j\}_{j\in S_i^t})$. Return back.
  - Aggregation: $x^t = f(x_1^t, x_2^t, \ldots, x_n^t)$. Broadcast.
  - Data shuffling: random refresh cache data -> statistical gain.



Design goal

Uniformly at random shuffling
High cost in communication
High statistical gain

No shuffling
No communication cost
No statistical gain

Semi-random shuffling
Low communication cost
Fairly high statistical gain

Our research

What is a good shuffling?

Empirical studies -> Good shuffling

Sufficient difference in cached content

across iterations and workers [Lee2015, Gürbüzbalaban2015]!

Hamming distance metric $H \stackrel{\text{def}}{=} \frac{\sum_{(i,t)\neq(i',t')} H\left(z_{i,t}, z_{i',t'}\right)}{\#\, pairs}$

Hamming distance of cache states, averaged

across all workers and iterations.

Cache states	Worker 1	Worker 2		Worker n
Iteration 1	$[0,1,0,0,\dots,1]$	$[1,0,1,0,\dots,1]$	…	$[0,0,1,1,\dots,0]$
Iteration 2	$[1,1,0,1,\dots,1]$	$[0,1,1,0,\dots,0]$	…	$[0,1,0,1,\dots,1]$
⋮		⋮		
Iteration T	$[1,1,0,0,\dots,0]$	$[0,1,1,1,\dots,1]$	…	$[1,1,1,0,\dots,0]$

What is a good shuffling?

Empirical studies -> Good shuffling

Sufficient difference in cached content

across iterations and workers [Lee2015, Gürbüzbalaban2015]!

Hamming distance metric $H \stackrel{\text{def}}{=} \frac{\sum_{(i,t)\neq(i',t')} H\left(z_{i,t}, z_{i',t'}\right)}{\#\, pairs}$

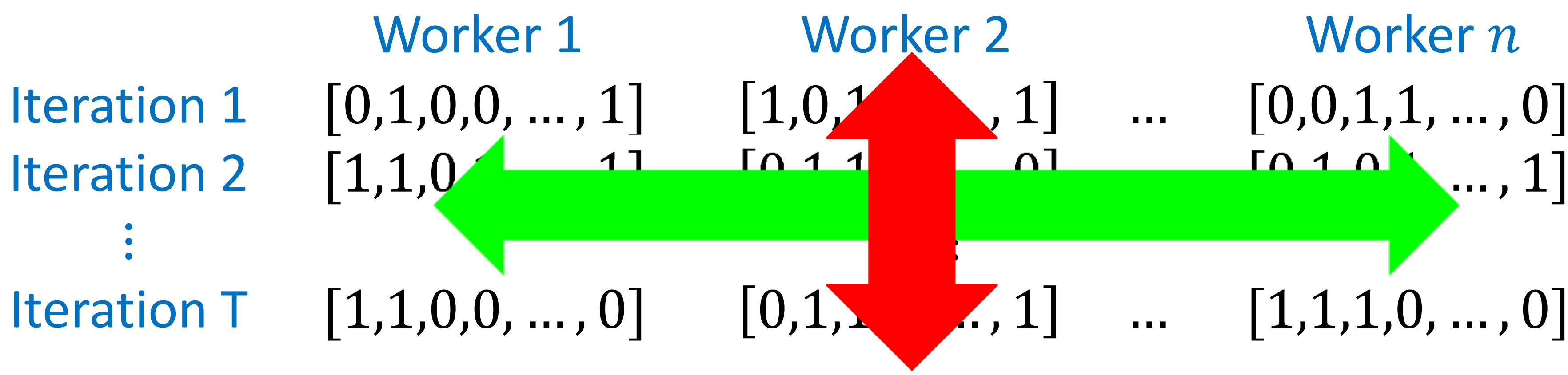
Hamming distance of cache states, averaged

across all workers and iterations.

Cache states	Worker 1	Worker 2		Worker n
Iteration 1	$[0,1,0,0, \dots, 1]$	$[1,0,1,0, \dots, 1]$	…	$[0,0,1,1, \dots, 0]$
Iteration 2	$[1,1,0,1, \dots, 1]$	$[0,1,1, \dots, 0]$	…	$[0,1,0,1, \dots, 1]$
⋮				
Iteration T	$[1,1,0,0, \dots, 0]$	$[0,1,1,1, \dots, 1]$	…	$[1,1,1,0, \dots, 0]$

AVERAGE

Design framework



- ❑ Reduce correlation of cached content across workers

-> **data shuffling constrained coding**, where a message can reach at most c caches

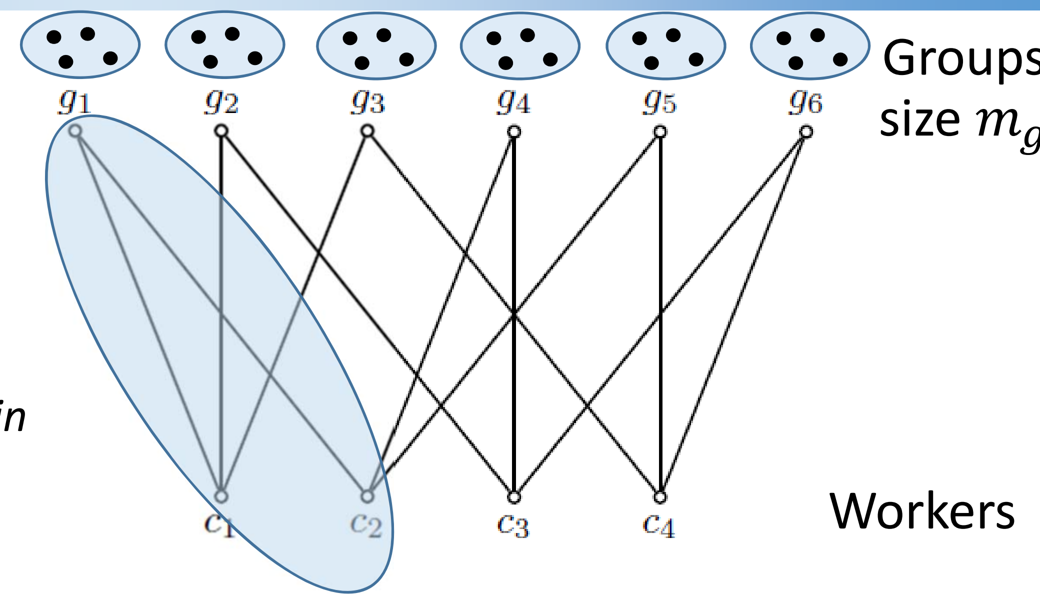
- ❑ Reduce correlation of cached content across iterations

-> **hierarchical structure**

Shuffling scheme design

Outer layer:
- *messages -> groups*
- *workers – group structure*

each worker randomly caches $(1 - 1/r)$ *fraction of messages in each of some certain groups.*



Building block coding

$$b_{j_1} + b_{j_2} + \cdots + b_{j_r}$$

Inner layer: each group
- *constrained coding*
- *random coded transmission*

In total m/m_g transmissions for each shuffling.

Design parameters: outer layer structure, r, m_g, $c \geq \frac{ns}{m(r-1)}$.

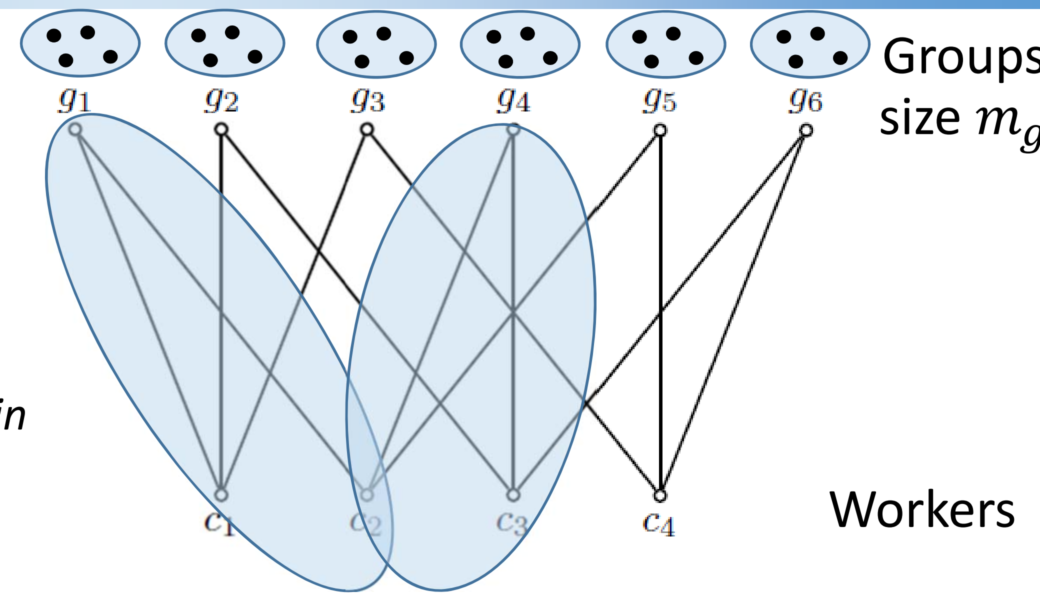
Shuffling scheme design

Outer layer:

- messages -> groups

- workers – group structure

each worker randomly caches $(1-1/r)$ fraction of messages in each of some certain groups.



A message reaches at most a certain # workers

Correlation across workers is reduced!

A worker gets new messages from a certain # groups

Correlation across iterations is reduced!

Data shuffling performance

- Hamming distance

$$H \geq \min\left\{\frac{2s}{em_g\left(1-\frac{1}{r}\right)}, 2\left(s - m_g + \frac{m_g}{r}\right)\right\}$$ (up to $O(s)$)

- Communication gains over classical index coding

  up to $O\left(\frac{ns}{m}\right)$

  Redundancy

  Avg. # copies a message is cached in all worker nodes

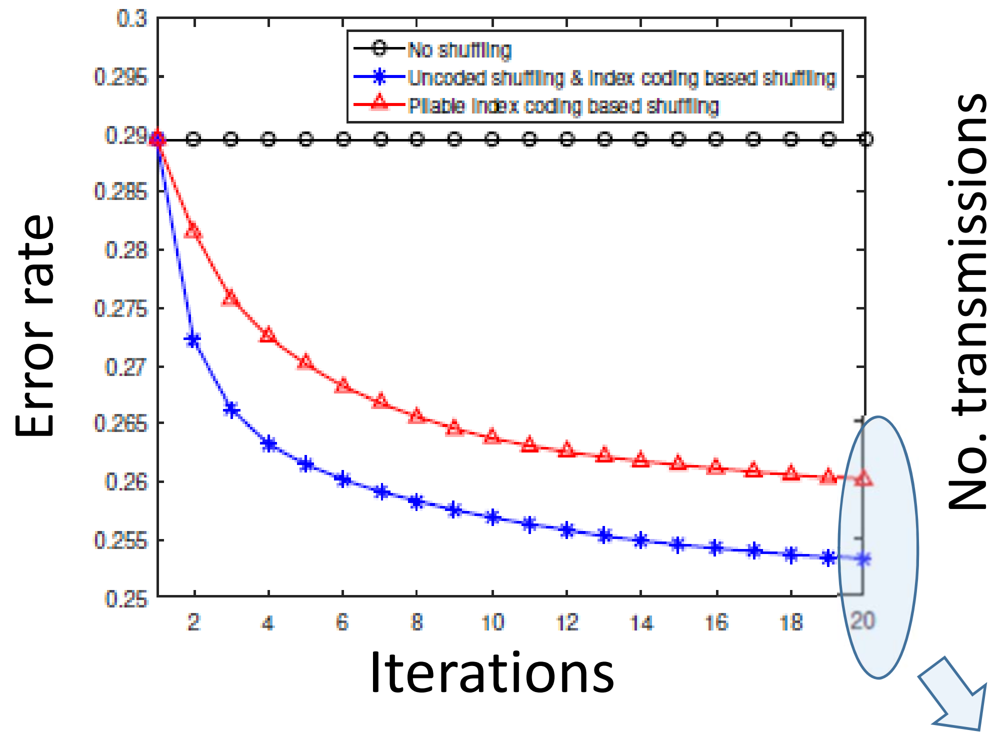
- Preserving semi-randomness

  Initial semi-random dist. (of cached content for all workers) -> semi-random dist. for all iterations

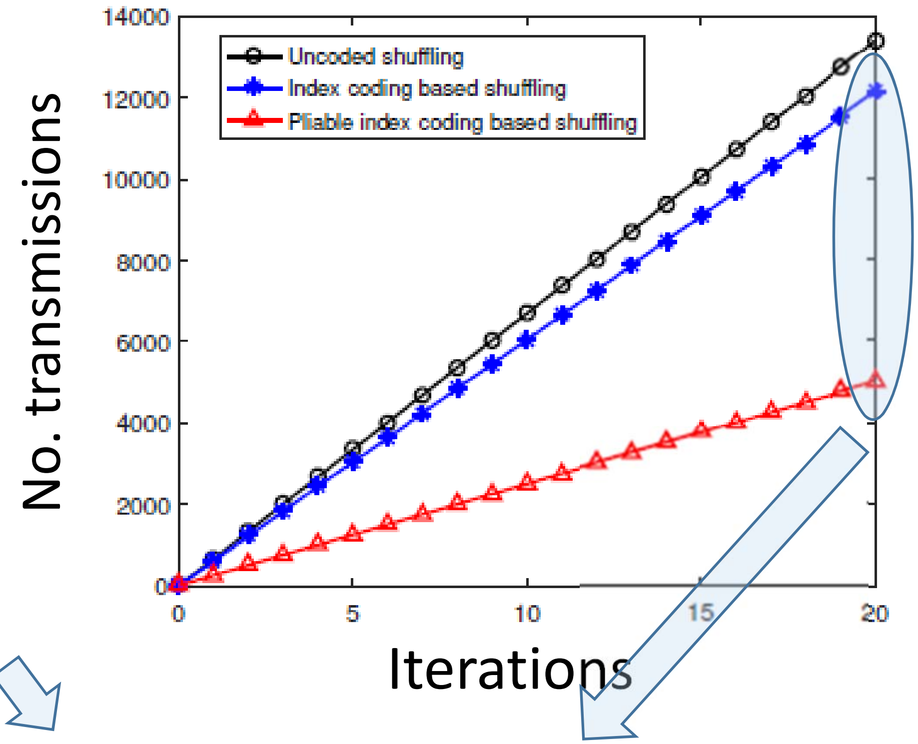
Experimental results

Distributed classification task

Computational performance



Communication cost



Save 60% bandwidth by only 2.6% performance loss

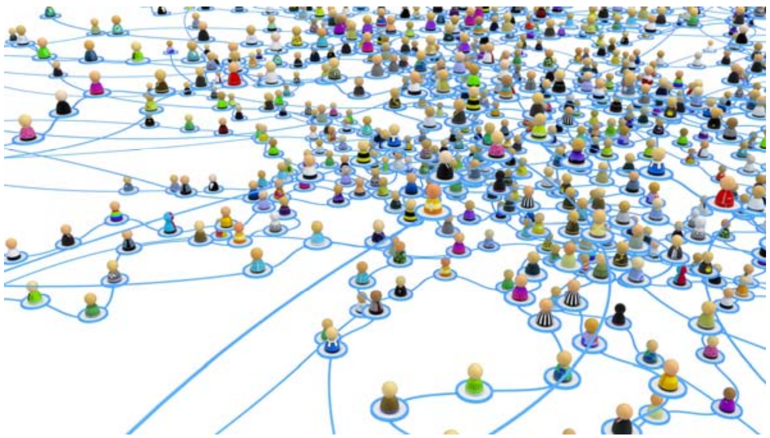
Future work

Recommender systems and learning

Recommender systems + social networks

Bandwidth-aware recommendations

- Loose -> tight-coupling, single -> multi-stage (on going)



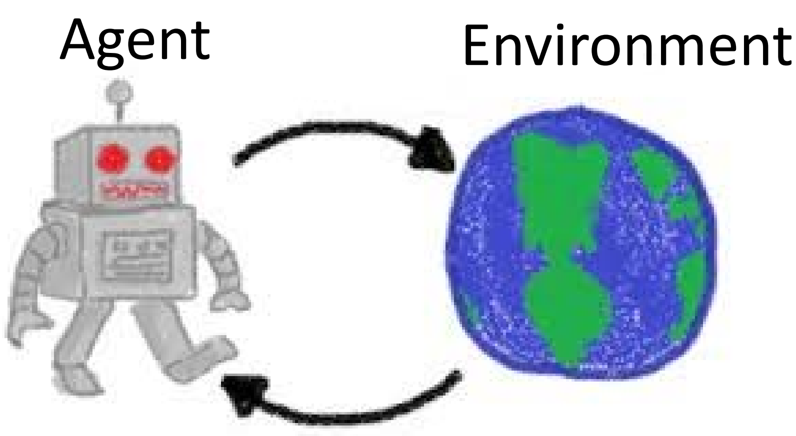
Distributed machine learning

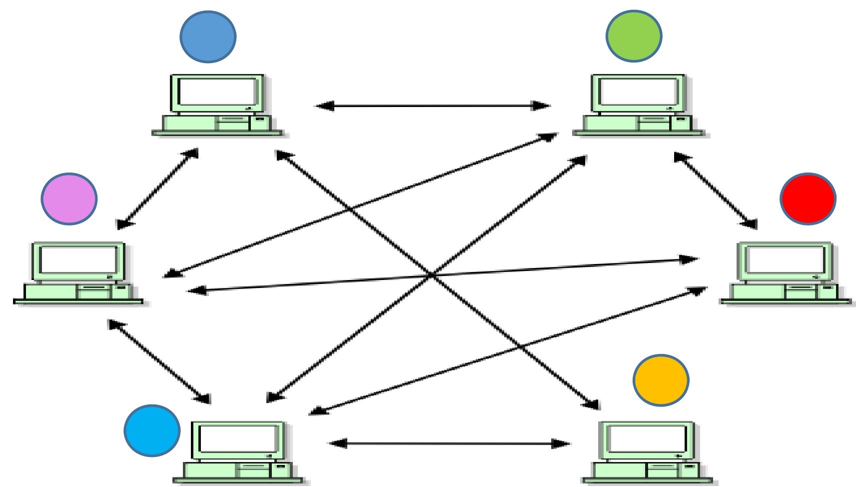
Extend to more computation paradigms

- Boosting, reinforcement learning, evolutionary computing

Communication for distributed computing

- Data locality & task assignment
- Networked structure





Thank you!