UNIVERSITY OF CALGARY

Privacy-Preserving Big Data Publishing

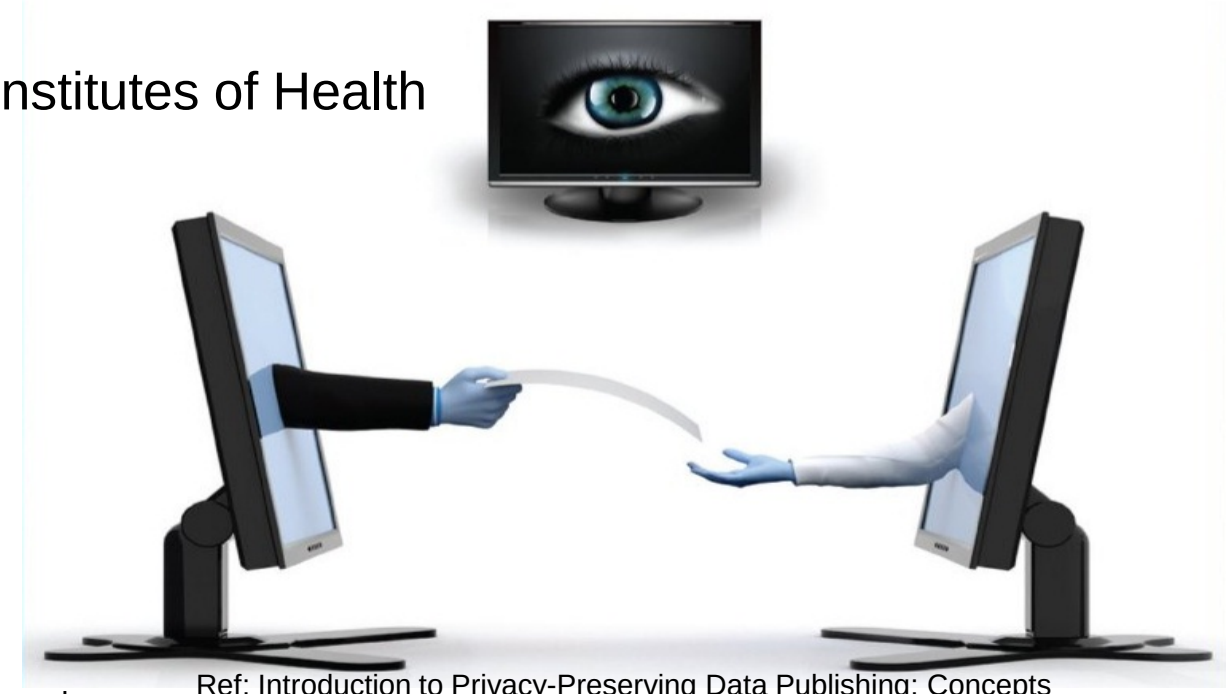
Hessam Zakerzadeh[1], Charu C. Aggarwal[2], Ken Barker[1]

SSDBM'15

[1] **University of Calgary, Canada**
[2] **IBM TJ Watson, USA**

Data Publishing

- OECD* declaration on access to research data
- Policy in Canadian Institutes of Health Research (CIHR)



Ref: Introduction to Privacy-Preserving Data Publishing: Concepts and Techniques (2010).

*- Organization for Economic Co-operation and Development

Data Publishing

Benefits:

- Facilitating the research community to confirm published results.
- Ensuring the availability of original data for meta-analysis.
- Making data available for instruction and education.

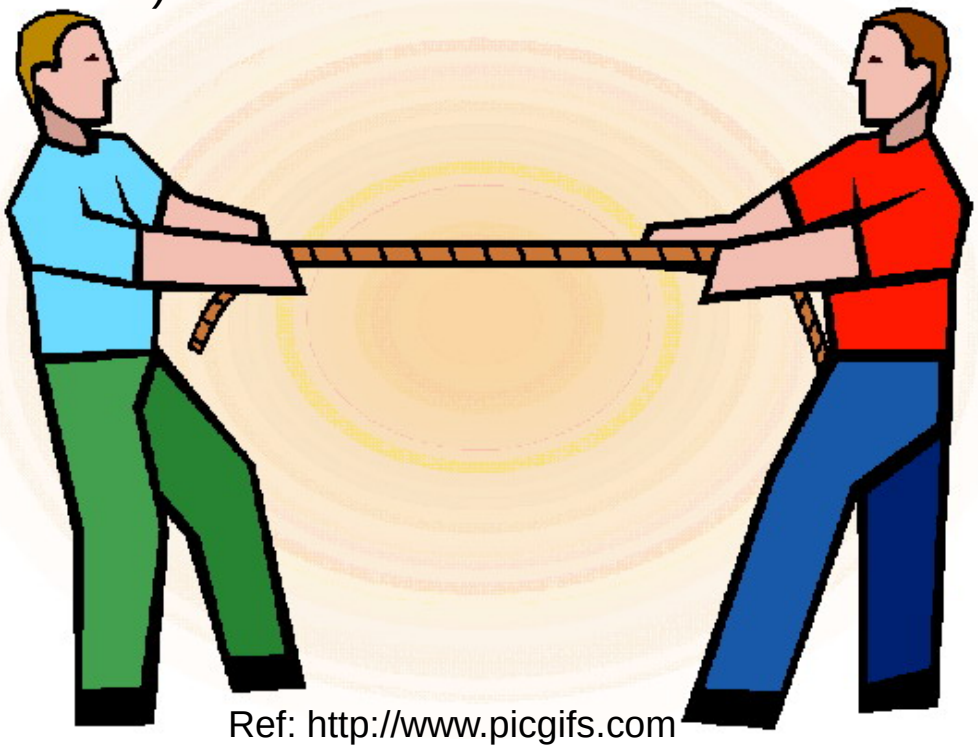
Requirement:

- Privacy of individuals whose data is included must be preserved.

Tug of War

Preserving the privacy of individuals whose data is included (needs anonymization).

Usefulness (utility) of the published data.



Ref: http://www.picgifs.com

Key Question in Data Publishing

How to preserve the privacy of individuals while publishing data of high utility?

Privacy Models

Privacy-preserving models:

- Interactive setting (e.g. differential privacy)
- **Non-interactive** setting (e.g. k-anonymity, l-diversity)
  - Randomization
  - **Generalization**

K-Anonymity

Attributes

- Identifiers
- Quasi-identifier
- Sensitive

K-Anonymity

Table 1: A patient data set

(a) Original data

t#	*SIN*	*Gender*	*Zipcode*	*DOB*	*Disease*
1	12543222	M	12499	30 June, 1985	HIV
2	23988880	M	12423	1 June, 1986	flu
3	34340012	M	13001	12 Sept, 1982	flu
4	12001234	M	13078	17 Sept, 1982	gastritis
5	33336788	F	13223	22 Aug, 1971	miocarditis
6	33789901	F	13009	2 Nov, 1974	miocarditis

(b) 2-anonymized data

EQ

t#	*Gender*	*Zipcode*	*DOB*	*Disease*
1	M	124**	June [1985-1986]	HIV
2	M	124**	June [1985-1986]	flu
3	M	130**	Sept 1982	flu
4	M	130**	Sept 1982	gastritis
5	F	13***	[1971-1974]	miocarditis
6	F	13***	[1971-1974]	miocarditis

L-Diversity

Table 1: A patient data set

(a) Original data

t#	*SIN*	*Gender*	*Zipcode*	*DOB*	*Disease*
1	12543222	M	12499	30 June, 1985	HIV
2	23988880	M	12423	1 June, 1986	flu
3	34340012	M	13001	12 Sept, 1982	flu
4	12001234	M	13078	17 Sept, 1982	gastritis
5	33336788	F	13223	22 Aug, 1971	miocarditis
6	33789901	F	13009	2 Nov, 1974	miocarditis

(c) 2-diverse data

t#	*Gender*	*Zipcode*	*DOB*	*Disease*
1	M	1****	[1982-1986]	HIV
2	M	1****	[1982-1986]	flu
3	M	1****	[1982-1986]	flu
4	*	13***	[1971-1982]	gastritis
5	*	13***	[1971-1982]	miocarditis
6	*	13***	[1971-1982]	miocarditis

Assumptions in state-of-the-art of anonymization

Implicit assumptions in current anonymization:

- Small- to moderate-size data
- Batch and one-time process

Focus on

- Quality of the published data

[redacted]

d?

N

- [redacted]a Big data (in Tera or

[redacted]ogs

- [redacted]s Repeated application

Focus on

- Quality of the published data + Scalability

Naïve solution?

Divide & conquer

- Inspired by streaming data anonymization techniques
- Divide the big data into small parts (fragments)
- Anonymize each part (fragment) individually and in isolation

Naïve solution?

Divide & conquer

- Inspired by streaming data anonymization techniques
- Divide the big data into small parts (fragments)
- Anonymize each part (fragment) individually and in isolation

- What we lose?
  - **Rule of thumb**: more data, less generalization/perturbation (large crowd effect)
  - Quality

Main question to answer in Big Data Privacy

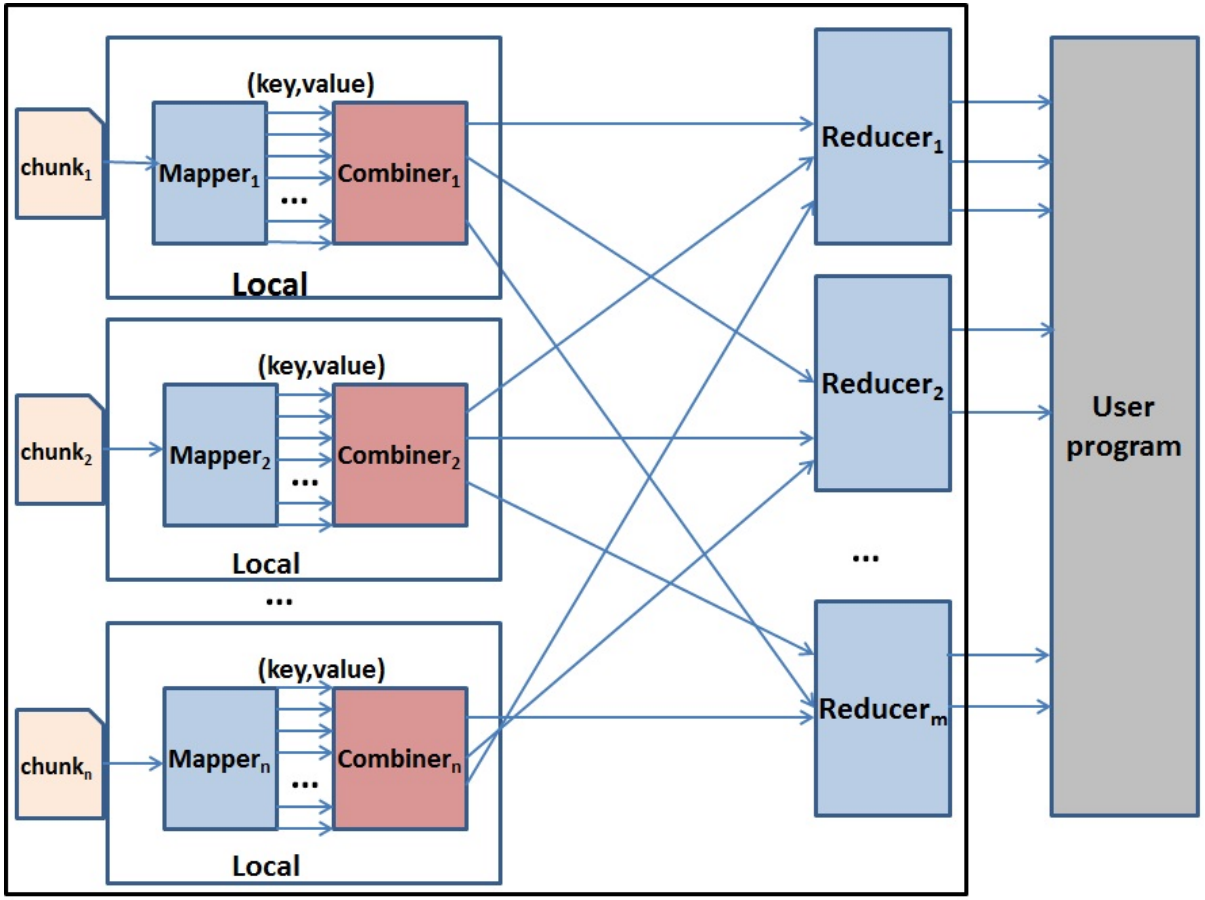
Is it somehow possible to take advantage of the **entire data set** in the anonymization process without losing **scalability**?

Map-Reduce-Based Anonymization

Idea: distribute the high-computational process of anonymization among different processing nodes such that distribution **does not affect** the quality (utility) of the anonymized data.

Map-Reduce Paradigm

Data flow of a mapreduce job



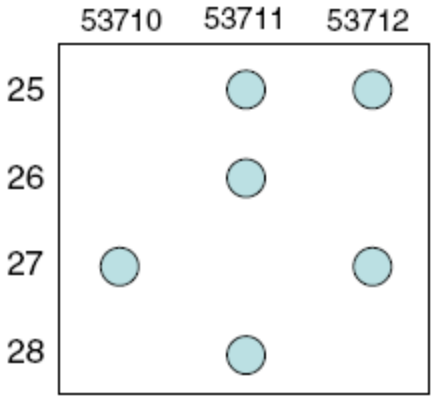
Mondrian-like Map-Reduce Alg.

Traditional Mondrian

- Pick a dimension (e.g. dim with widest range), called **cut dimension**
- Pick a point along the cut dimension, called **cut point**
- Split data into two equivalence classes along the **cut dimension** and at the **cut point**, provided that privacy condition is not violated.
- Repeat until no further split is possible

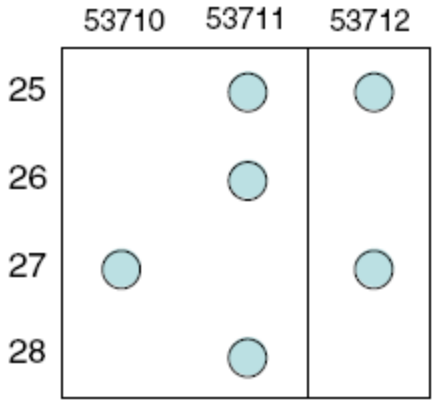
Mondrian-like Map-Reduce Alg.

Traditional Mondrian



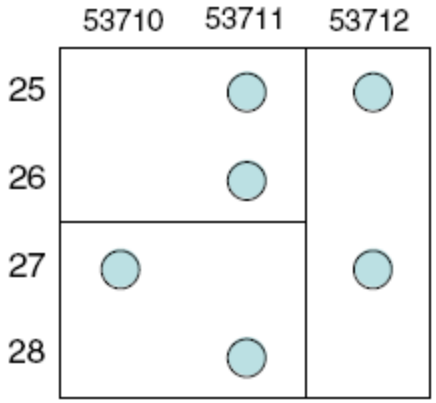
Mondrian-like Map-Reduce Alg.

Traditional Mondrian



Mondrian-like Map-Reduce Alg.

Traditional Mondrian



Mondrian-like Map-Reduce Alg.

Preliminaries:

- Each equivalence class is divided into **at most** *q* equivalence classes.
- A global file is shared among all nodes. This file contains equivalence classes formed so far organized in a tree structure (called *equivalence classes tree*).
- Initially contains the most general equivalence class.

Mapper

Algorithm 1 Mapper in k-anonymity

```
1: KMapper(k,v)
2:    //obtain the global file from distributed file system
3:    eq-id = findFinestEQ(v)
4:    output-value = empty // an array of size dimensionality
5:    foreach dim in v
6:      append pair (dim,1) to output-value
7:    emit(eq-id, output-value)
```

Mapper: Example

iteration 1: only one equivalence class exists (called eq_1)

eq_1 = [1:100],[1:100],[1:100]

Data records:

12, 33, 5

56, 33, 11

12, 99, 5

Mapper's output:

< eq_1, <(12,1),(33,1),(5,1)>>

key

< eq_1, <(56,1),(33,1),(11,1)>> value

< eq_1, <(12,1),(99,1),(5,1)>>

Combiner

Algorithm 2 Combiner in k-anonymity

```
1: KCombiner(k, V)
2:    output-value = empty // an array of size dimensionality
3:    foreach v in V
4:       for i in [1, ..., dimensionality]
5:          add(v[i], output-value[i])
6:    emit(k, output-value)
```

Combiner example

Mapper's output (combiner's input):

$< eq_1, <(12,1),(33,1),(5,1)>>$

$< eq_1, <(56,1),(33,1),(11,1)>>$

$< eq_1, <(12,1),(99,1),(5,1)>>$

Combiner's output:

$< eq_1,$

12	2
56	1

,

33	2
99	1

,

5	2
11	1

$>$

Reducer

Algorithm 3 Reducer in k-anonymity

```
1:  KReducer(k, V)
2:    //all the data records belong to equivalence class eq
3:    //q is maximum number of split
4:    for (i in [1,..., dimensionality])
5:      c-dim=findDimToCut(V,i)
6:      for p in [q,...,2]
7:        cp_1,cp_2,...,cp_{p-1} = findCutPoint_p(V,c-dim)
8:        if (splitting eq at cp_1,cp_2,...,cp_{p-1} does not violate
9:        the k-anonymity)
10:         eq_1,eq_2,...,eq_{p-1} = Cut_p(eq,c-dim,cp_1,...,cp_{p-1})
11:         for j in [1,...,p-1]
12:           emit(eq_j,"1")
13:         exit()
14:   emit(eq,"0")
```

Reducer's output

Combiner's output (reducer's input):

$< eq_1,$

12	2	,	33	2	,	5	2
56	1		99	1		11	1

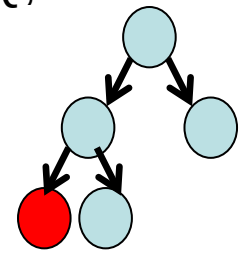
$>$

Reducer's output:

- If eq_1 is splittable:
- <[12:56],[33:45],[5:11], "1"> → Added to the global file
- <[12:56],[45:99],[5:11], "1">
- If eq_1 is un-splittable:
- <[12:56],[33:99],[5:11], "0"> → Added to the global file

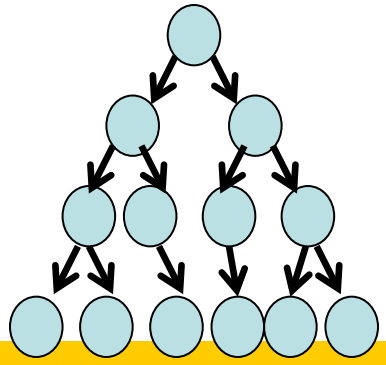
Improvement 1 (Data transfer improvement)

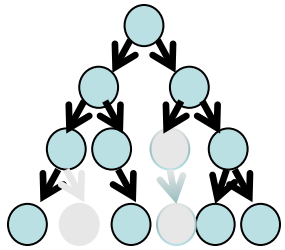
- Mapper outputs only records belonging to splittable equivalence classes.
- Requirement:
  - Global file includes a flag for each equivalence class indicating whether it is splittable or not.
  - If a data record belongs to an un-splittable EQ, do not output it.
  - e.g. [1:100],[1:100],[1:100], "1"
    [12:56],[33:45],[5:11], "1"
    [12:56],[45:99],[5:11], "1"
    [12:30],[33:45],[5:11], "1"
    [30:56],[33:45],[5:11], "0"

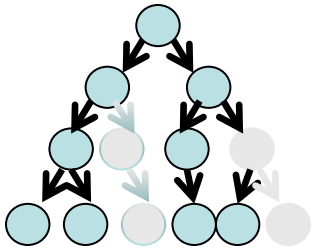


Improvement 2 (Memory improvement)

- What if the global file doesn't fit into memory of mapper nodes?
  - Break down the global file into multiple small files
  - How?
    - Split the big data into small files.
    - Create a global file per small input file.
    - Each global file contains only a subset of total equivalence classes.
    - Each small global file is referred to as global subset equivalence class file (gsec file).







Improved Algorithm

Mapper's output:

< eq_1, <f_1,(12,1),(33,1),(5,1)>>
< eq_1, <f_1,(56,1),(33,1),(11,1)>>
< eq_1, <f_1,(12,1),(99,1),(5,1)>>

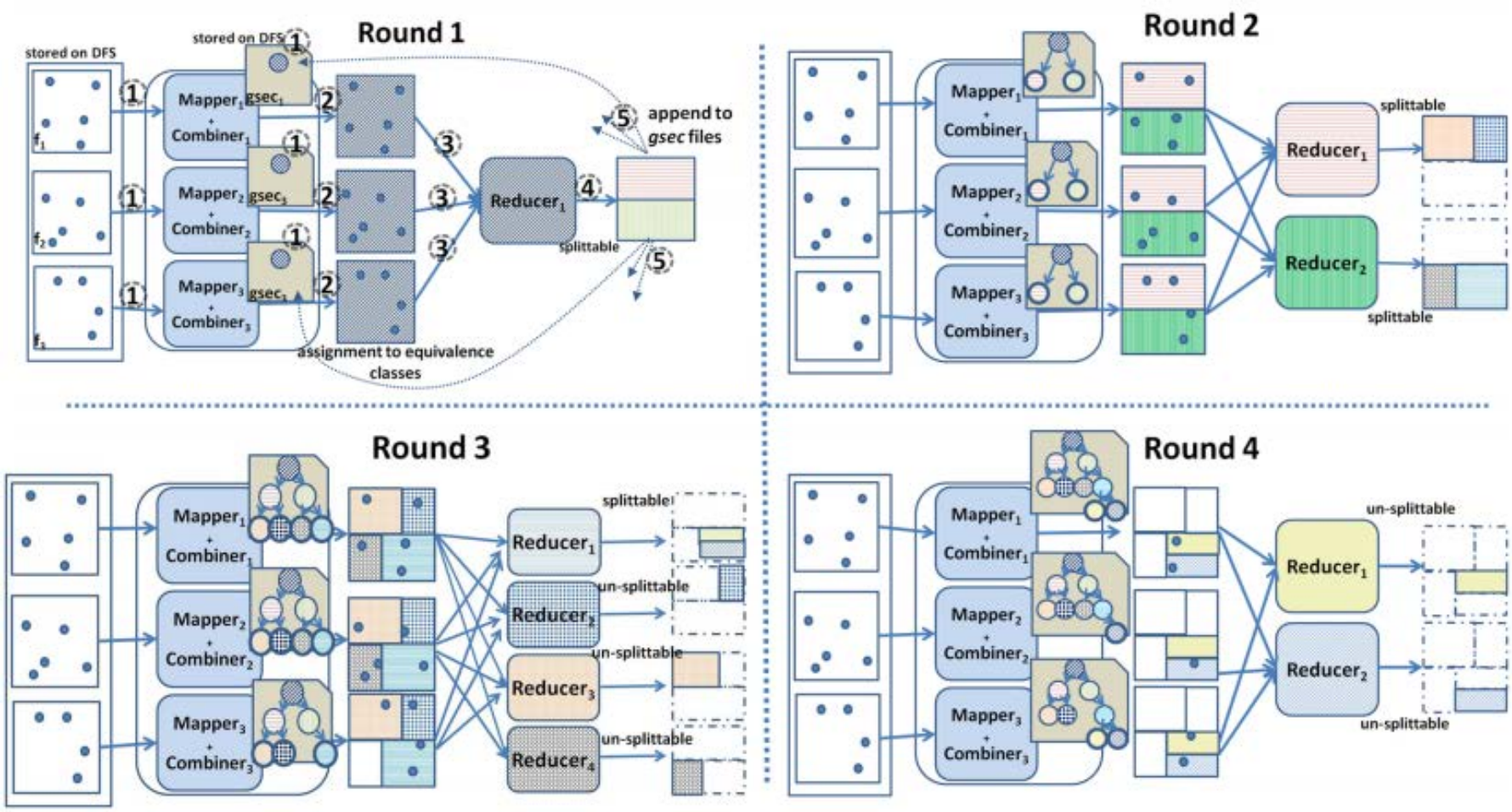
Combiner's output:

< eq_1, <f_1, {12, 2; 56, 1}, {33, 2; 99, 1}, {5, 2; 11, 1}>

Reducer's output:

If eq_1 is splittable:
<[12:56],[33:45],[5:11], "1", f_1 >
<[12:56],[45:99],[5:11], "1",f_1>
If eq_1 is un-splittable:
<[12:56],[33:99],[5:11], "0",f_1>

K,q = 2


Round 1
stored on DFS
stored on DFS
Mapper1 + Combiner1
Mapper2 + Combiner2
Mapper3 + Combiner3
gsec1
gsec2
gsec3
Reducer1
append to gsec files
splittable
assignment to equivalence classes
Round 2
Mapper1 + Combiner1
Mapper2 + Combiner2
Mapper3 + Combiner3
Reducer1
Reducer2
splittable
splittable
Round 3
Mapper1 + Combiner1
Mapper2 + Combiner2
Mapper3 + Combiner3
Reducer1
Reducer2
Reducer3
Reducer4
splittable
un-splittable
un-splittable
un-splittable
Round 4
Mapper1 + Combiner1
Mapper2 + Combiner2
Mapper3 + Combiner3
Reducer1
Reducer2
un-splittable
un-splittable

UNIVERSITY OF CALGARY

Further Analysis

- Time Complexity (per round)
  - Mapper
  - Combiber
  - Reducer
- Data Transfer (per round)
  - Mapper and Combiner (Local)
  - Combiner and Reducer (Across the network)

Experiments

Answer the following questions:

- How well the algorithm scales up?
- How much information is lost in the anonymization process?
- How much data is transferred between mappers/combiners and **combiners/reducers** in each iteration?

Experiments

Data sets

- **Poker data set**: 1M records, each 11 dimensions
- **Synthetic data set**: 10M records, each 15 dimensions, 1.4 GB
- **Synthetic data set**: 100M records, each 15 dimensions, 14.5 GB

Information loss baseline

Each data set is split into 8 fragments. Each fragment is anonymized individualy

State-Of-The-Art

MapReduce Top-Down Specialization (MRTDS) [Zhang et. al'14]

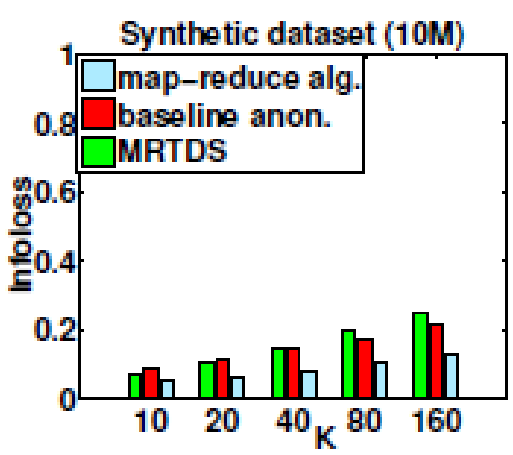
Experiments Settings

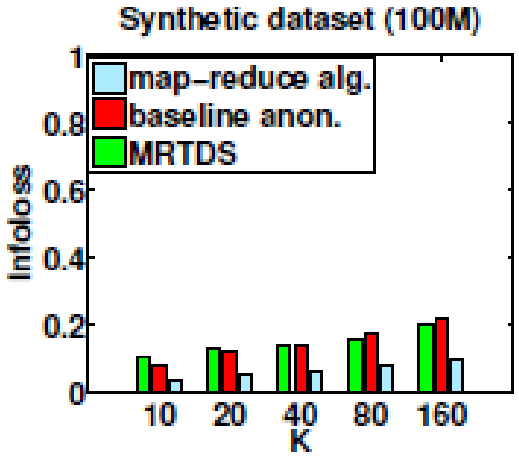
- Hadoop cluster on AceNet
- 32 nodes, each having 16 cores and 64 GB RAM
- Running RedHat Enterprise Linux 4.8

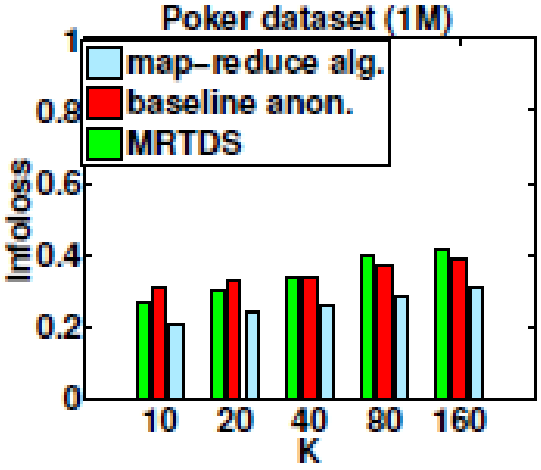
Parameters of Hadoop

Parameter Name	Value
fs.block.size	64MB
io.sort.mb	1024MB
io.sort.factor	50
dfs.replication	3

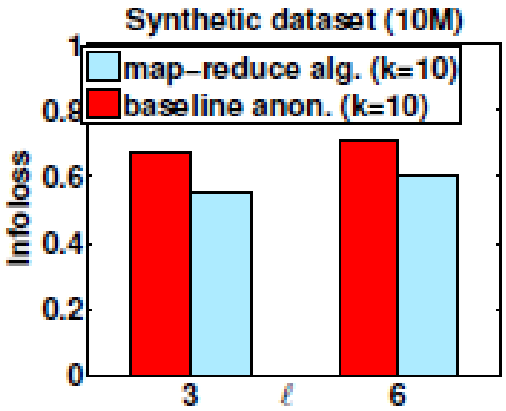
Information Loss vs. K

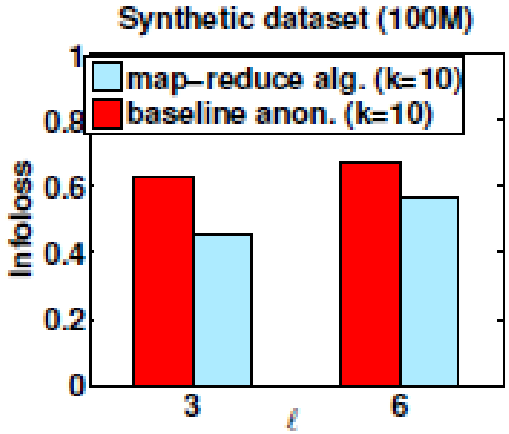


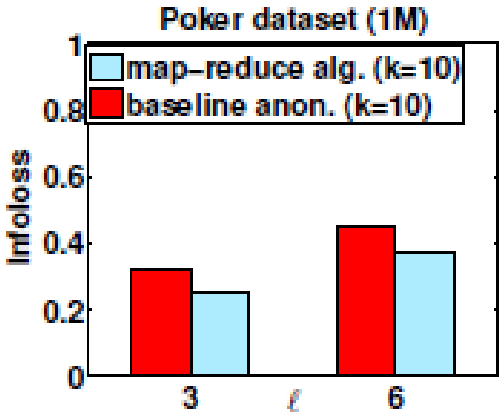




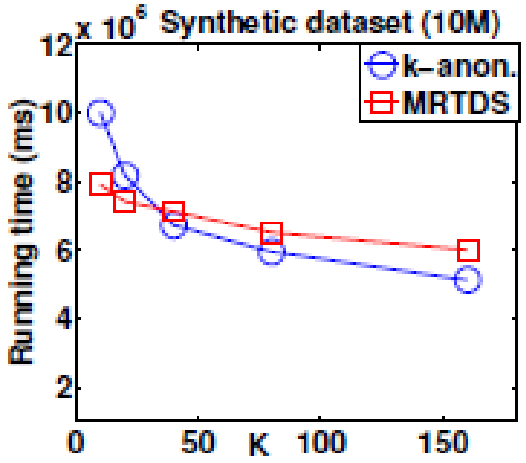
Information Loss vs . L

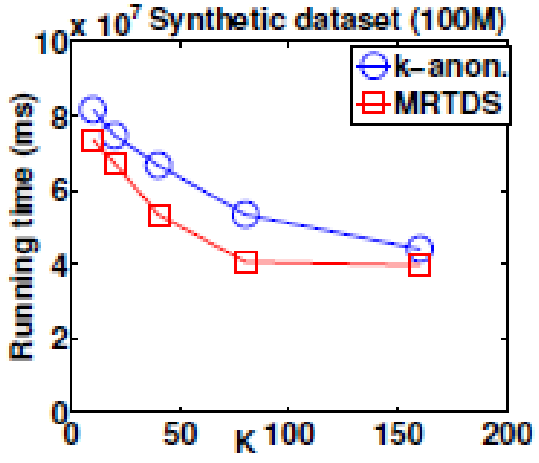


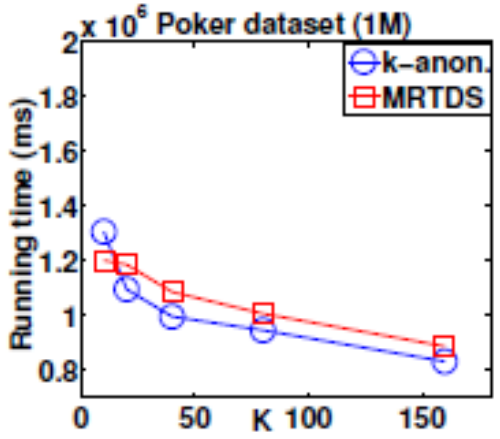


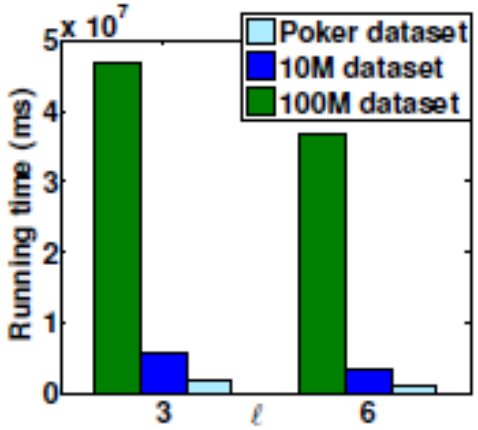


Running time vs. K (L)

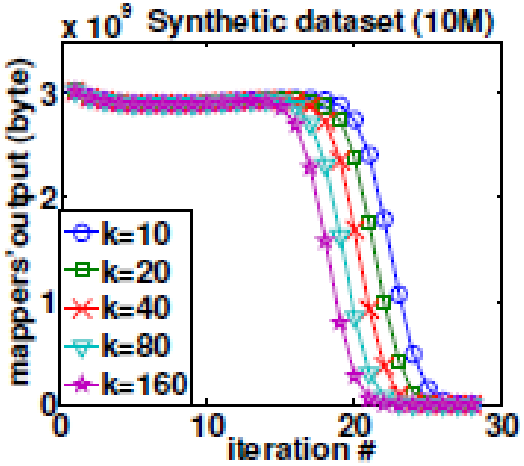


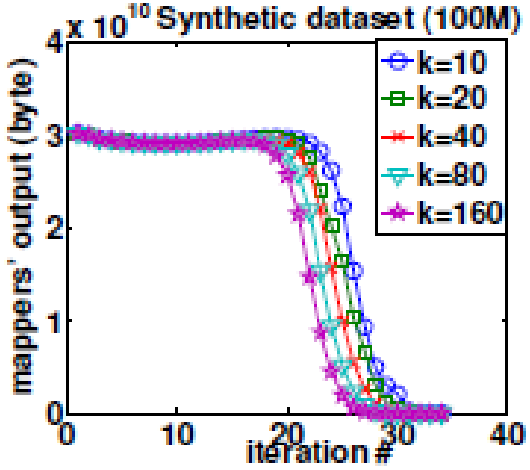


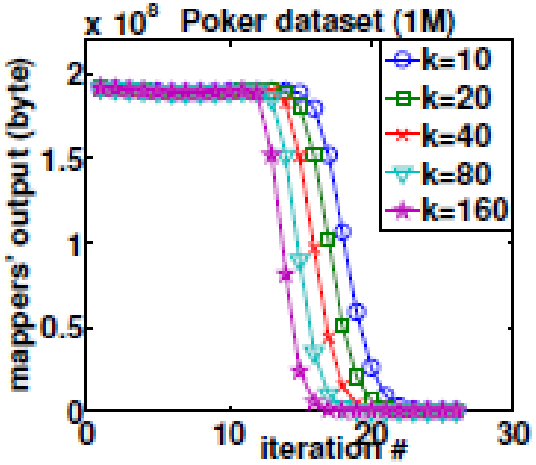




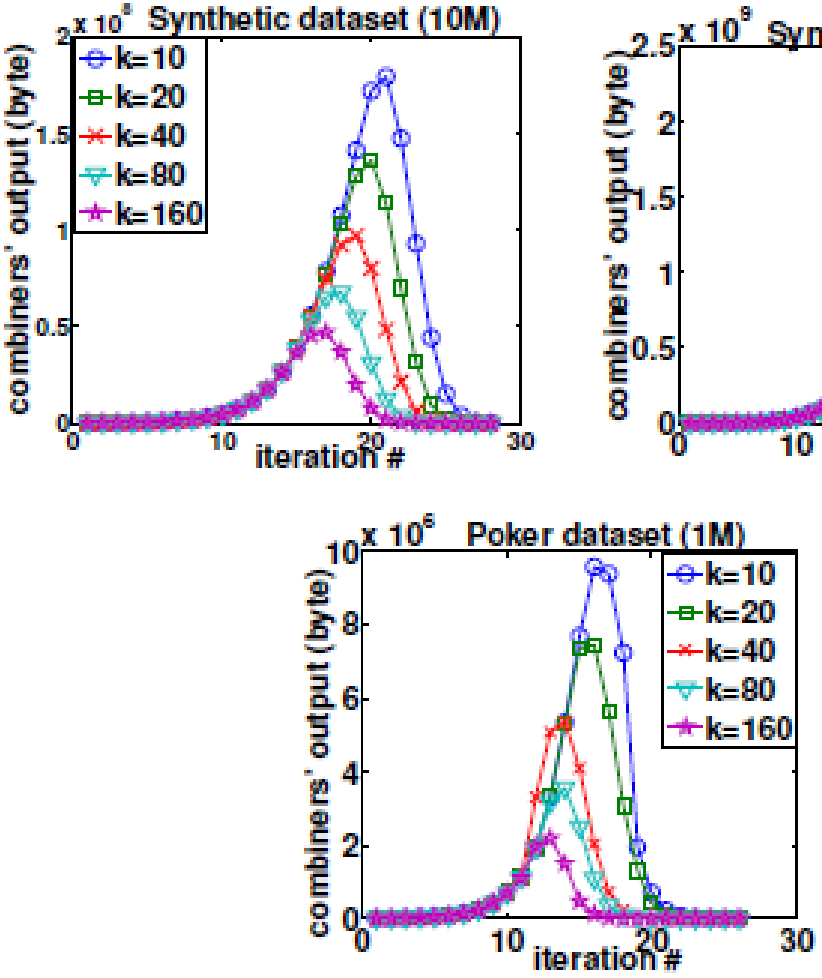
Data Transfer vs. Iteration # (between mappers and combiners)







Data Transfer vs. Iteration # (between combiners and reducers)



Future work

- Extension to other data types (graph data anonymization, set-value data anonymizaiton, etc)
- Extension to other privacy models