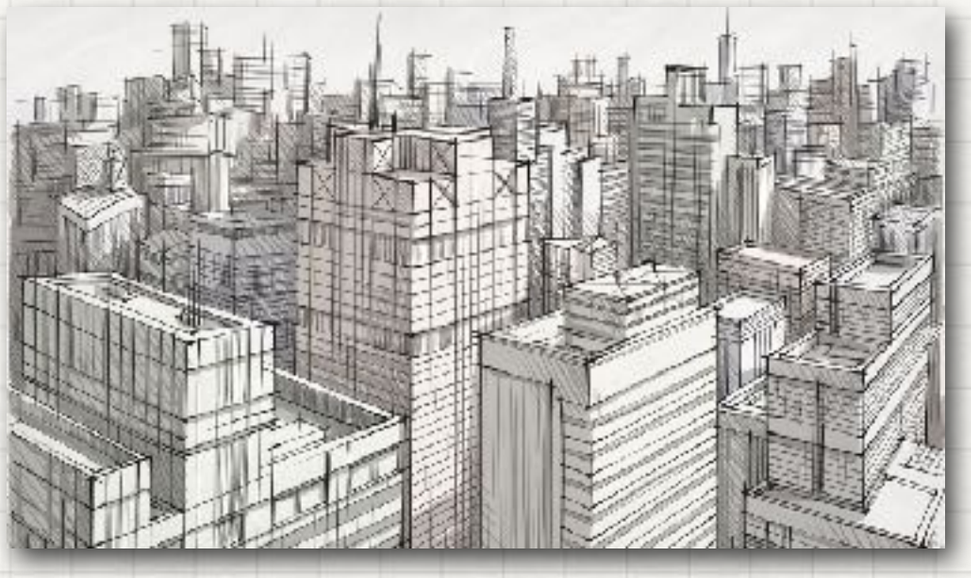
FINDING OUTLIERS IN BIG DATA

KTX



- Dataset Evaluation
- Find Null Values
- Find Meaningful Distinct Values

BLOCK NESTED-LOOP METHOD

INDEX BASED METHOD

Tianhan Yuan
Xiaoyan Zhang
Kunyu Ye

DATASET EVALUATION

50 Data Sets

Smallest Set: auuc-fqzi.tsv 835 bytes 2 rows 23 columns

Biggest Set: rhe8-mgbb.tsv 2.57G 16511056 rows 15 columns

There are some datasets containing more 70 or more columns

There are some datasets contain tremendous null/unspecified values

There are some datasets which we cannot fully understand

FIND NULL VALUES

Tools: Hadoop, Mapreduce

Main Idea:
Traverse the whole file only once
Split file into lines => Split line into fields => Check if field == None

Two sets of Mapreduce:
One can display null value in each line ; Another can show the number of null values in each column and the distribution map; Then compare the result of both to verify.

```
Row1789345 has 14 null columns:31,39,48,41,42,43,44,45,46,47,48,49,50,51
Row1789346 has 14 null columns:31,39,48,41,42,43,44,45,46,47,48,49,50,51
Row1789347 has 14 null columns:31,39,48,41,42,43,44,45,46,47,48,49,50,51
Row1789348 has 14 null columns:31,39,48,41,42,43,44,45,46,47,48,49,50,51
Row1789349 has 14 null columns:31,39,48,41,42,43,44,45,46,47,48,49,50,51
Row1789350 has 14 null columns:31,39,48,41,42,43,44,45,46,47,48,49,50,51
Row1789351 has 14 null columns:31,39,48,41,42,43,44,45,46,47,48,49,50,51
Row1789352 has 28 null columns:1,15,20,21,24,27,30,31,33,36,37,39,4,40,41,42,43,44,45,46,47,48,49,5,50,51,7,9
Row1789353 has 28 null columns:1,15,20,21,24,27,30,31,33,36,37,39,4,40,41,42,43,44,45,46,47,48,49,5,50,51,7,9
Row1789354 has 28 null columns:1,15,20,21,24,27,30,31,33,36,37,39,4,40,41,42,43,44,45,46,47,48,49,5,50,51,7,9
Row1789355 has 28 null columns:1,15,20,21,24,27,30,31,33,36,37,39,4,40,41,42,43,44,45,46,47,48,49,5,50,51,7,9
Row1789356 has 28 null columns:1,15,20,21,24,27,30,31,33,36,37,39,4,40,41,42,43,44,45,46,47,48,49,5,50,51,7,9
Row1789357 has 28 null columns:1,15,20,21,24,27,30,31,33,36,37,39,4,40,41,42,43,44,45,46,47,48,49,5,50,51,7,9
Row1789358 has 28 null columns:1,15,20,21,24,27,30,31,33,36,37,39,4,40,41,42,43,44,45,46,47,48,49,5,50,51,7,9
Row1789359 has 28 null columns:1,15,20,21,24,27,30,31,33,36,37,39,4,40,41,42,43,44,45,46,47,48,49,5,50,51,7,9
TotalNullFields: 27653108
```

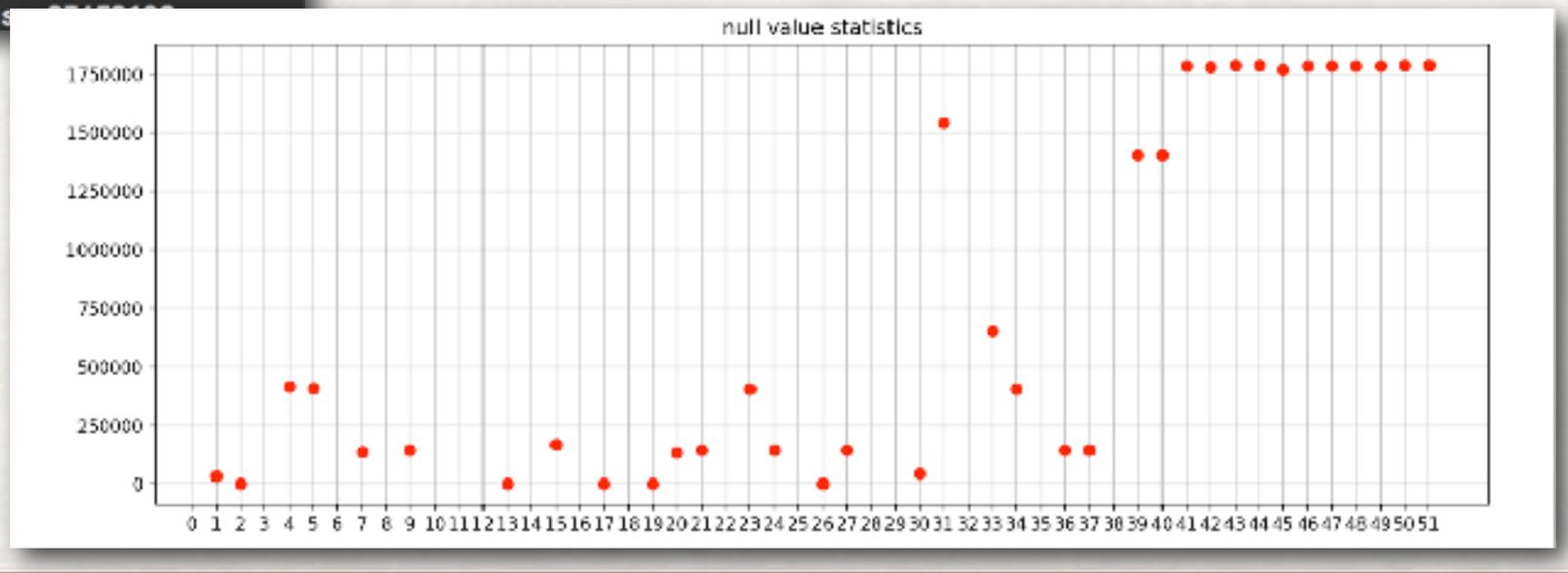
Example: DataSet hy4q-igkk.tsv

Mapper Output:
RowNumber \t
ColumnNumber which
contains null value

```
1789353  ""  Sign/Awning/Marquee - Illegal/No Permit MANHATTAN
    Unspecified MANHATTAN   Unspecified Open    Unspecified
STREET MANHATTAN            2006-12-12T00:00:00
1789354  ""  Illegal Hotel Rooms In Residential Building BROOKLYN
Unspecified     Unspecified BROOKLYN    Unspecified Open    U
2110 WESTBURY COURT BROOKLYN            2006-10-16T00:00:00
1789355  ""  Sign/Awning/Marquee - Illegal/No Permit MANHATTAN
    Unspecified MANHATTAN   Unspecified Open    Unspecified
SOUTH  MANHATTAN            2006-05-09T00:00:00
1789356  ""  Sign/Awning/Marquee - Illegal/No Permit MANHATTAN
    Unspecified MANHATTAN   Unspecified Open    Unspecified
            2006-05-18T00:00:00
1789357  ""  Sign/Awning/Marquee - Illegal/No Permit BROOKLYN
    Unspecified BROOKLYN    Unspecified Open    Unspecified
STREET BROOKLYN             2006-07-13T00:00:00
1789358  ""  Sign/Awning/Marquee - Illegal/No Permit QUEENS          S
Unspecified QUEENS  Unspecified Open    Unspecified         U
QUEENS          2006-01-09T00:00:00
1789359  ""  Sign/Awning/Marquee - Illegal/No Permit QUEENS          S
Unspecified QUEENS  Unspecified Open    Unspecified         U
QUEENS          2006-04-17T00:00:00
1789360  ""  Sign/Awning/Marquee - Illegal/No Permit STATEN ISLAND
Unspecified     Unspecified STATEN ISLAND   Unspecified Open
35 PAGE AVENUE  STATEN ISLAND           2006-03-22T00:00:00
```

```
Column1 has 32275 null values
Column2 has 13 null values
Column4 has 414504 null values
Column5 has 407645 null values
Column7 has 134889 null values
Column9 has 143565 null values
Column13 has 181 null values
Column15 has 166659 null values
Column17 has 20 null values
Column19 has 181 null values
Column20 has 133826 null values
Column21 has 143565 null values
Column23 has 404273 null values
Column24 has 143565 null values
Column26 has 179 null values
Column27 has 143565 null values
Column30 has 43361 null values
Column31 has 1542487 null values
Column33 has 651487 null values
Column34 has 404256 null values
Column36 has 143565 null values
Column37 has 143565 null values
Column39 has 1409203 null values
Column40 has 1409206 null values
Column41 has 1785968 null values
Column42 has 1781789 null values
Column43 has 1788717 null values
Column44 has 1788760 null values
Column45 has 1769492 null values
Column46 has 1786680 null values
Column47 has 1786653 null values
Column48 has 1785810 null values
Column49 has 1786680 null values
Column50 has 1787386 null values
Column51 has 1789133 null values
ToatalNullFields
```

Mapper Output: ColumnNumber which contains null value \t 1



due_date	school_name
2006-07-26T00:00:00	Unspecified
2006-07-26T00:00:00	Unspecified
2007-01-23T00:00:00	Unspecified
2006-08-30T00:00:00	Unspecified
2006-08-30T00:00:00	Unspecified
2006-10-05T00:00:00	Unspecified
2006-09-25T00:00:00	Unspecified
2006-09-15T00:00:00	Unspecified
2006-07-26T00:00:00	Unspecified
2006-07-29T00:00:00	School - IS 285 Meyer Levin
2006-10-18T00:00:00	Unspecified
2006-08-26T00:00:00	Unspecified
2006-10-05T00:00:00	Unspecified

BLOCK NESTED-LOOP METHOD

Tools: Hadoop, MapReduce

General Idea:

finding the distance, D, to the k-th nearest neighbor for each point, then select points with maximum D values.

Map function:

In a dataset of n entries, for each point p, pair it to n-1 entries, each denoted as q.
Key: row number and selected attributes of p.
Value: selected attributes of q

Reduce function:

Calculate and store the manhattan distance D between p and each q into a k-th nearest neighbor list L. Update the list, i.e, delete the furthest neighbor, if a nearer neighbor q is found. Output each p, w.r.t. the greatest D-max in L, i.e., D-max := max(L)

Result collection:

Collect the points with the greatest distances D-max. These points are outliers

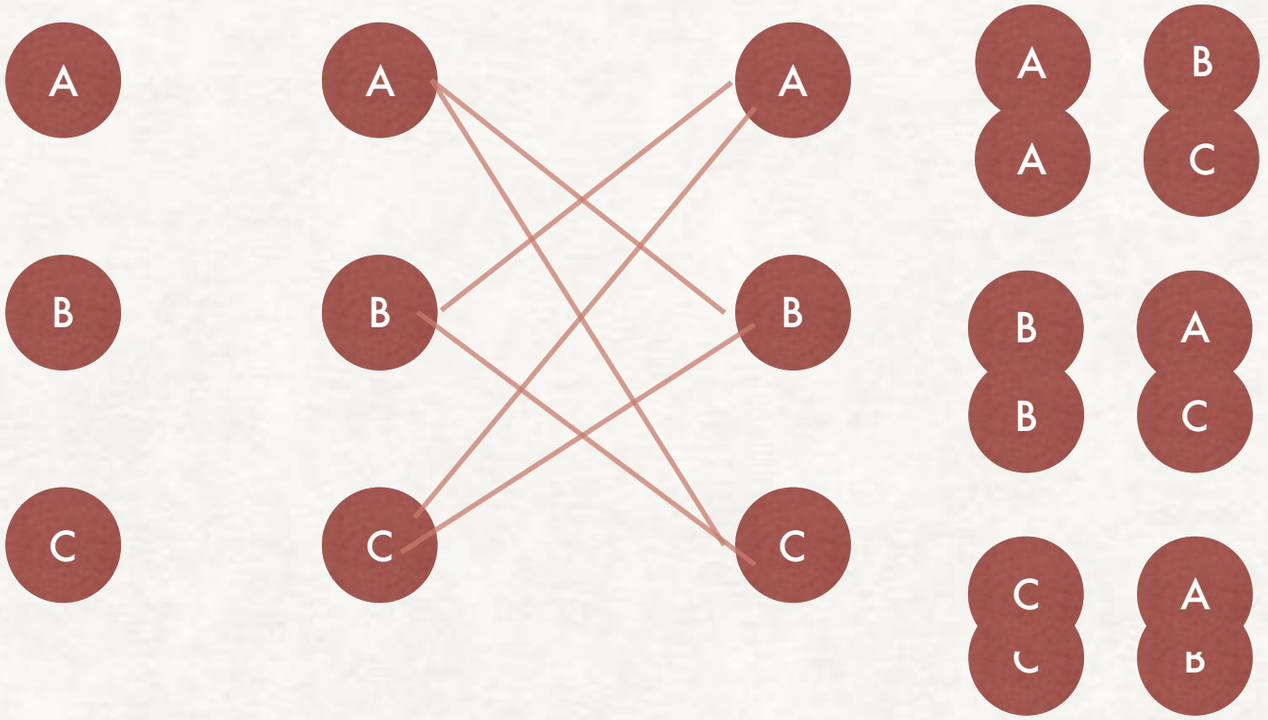
BLOCK NESTED-LOOP METHOD

Original

Mapping

Reducing

Choose Kth biggest Value For each row



A,Dist(A,B),Dist(A,C)……

B,Dist(B,A),Dist(B,C)……

C,Dist(C,A),Dist(C,B)……

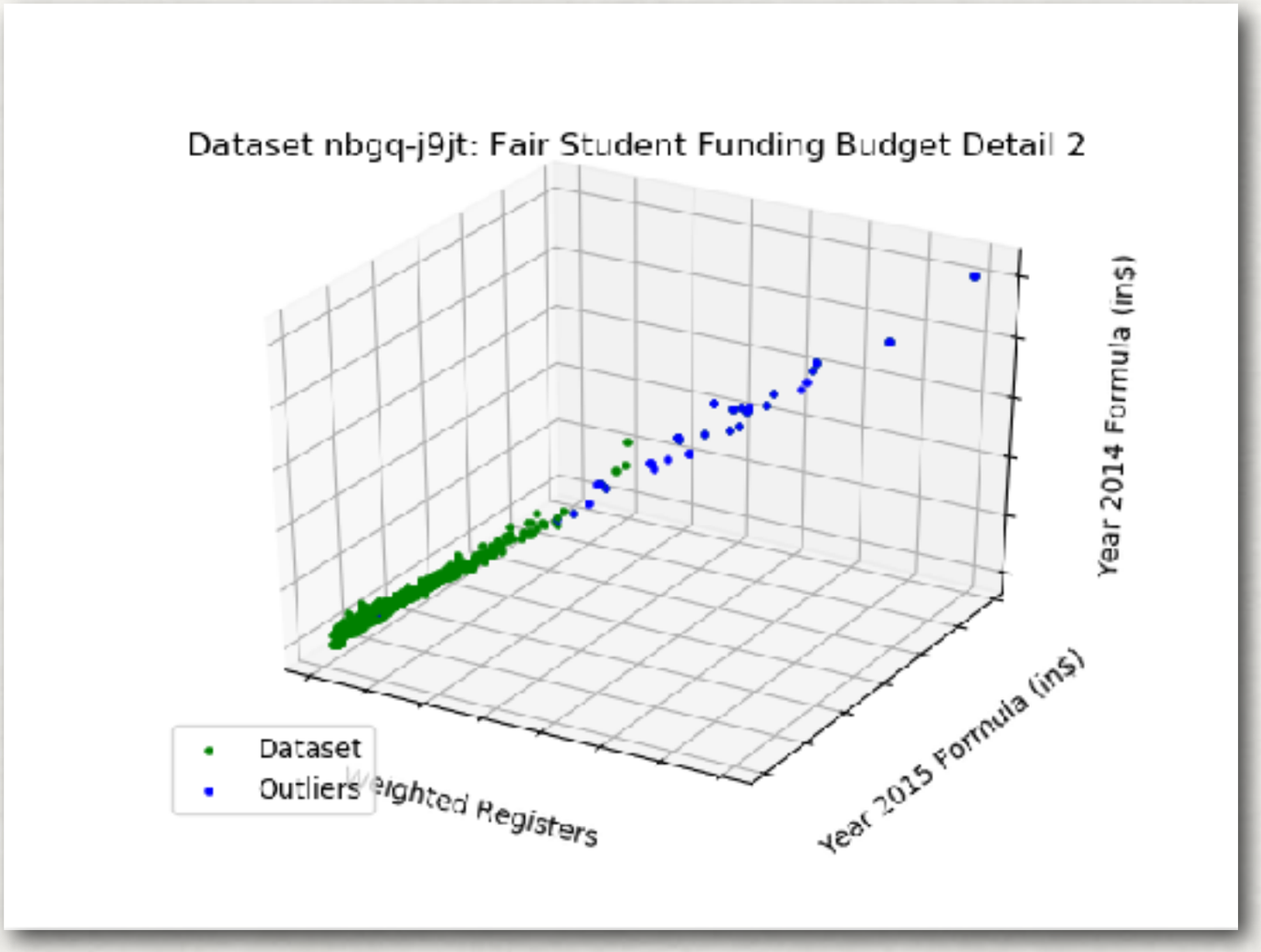
[A, Dist(AK)] [B, Dist(BK)] ……

Example：Elementary School Student Funding Budget, FY14-15

Small dataset with about 1500+ rows, 3 input columns

Configuration: Single Mapper + 10 Reducers

Time Cost: About ***5 minutes***



Example： NYC TAXI DATA ghpb-fpea.tsv

When processing first ***10000*** rows

Configuration: Single Mapper + 10 Reducers

Time Cost: More than ***12 minutes***

When processing first ***50000*** rows

Configuration: Single Mapper + 100 Reducers

Time Cost: About ***1.25 hour***

PROBLEM WITH NESTED BLOCK APPROACH

- Restricted by our log(N*N) algorithm, only one mapper can be used during mapping period, which seriously limit the scalability of this approach.

- In our test, this approach would be infeasible when processing data which is more than 50000 rows, let alone real big data which could be much bigger

SPATIAL INDEX TREE

- A space-partitioning data structure. (r tree, k-d tree)
- Package scikit-learn (k-d tree)
- Implement procedure using Spark:
  - 1 Step: Data preprocessing (read data, map() and filter(), get a rdd)
  - 2 Step: Collect the data and build k-d tree (collect() and sklearn package)
  - 3 Step: Broadcast the k-d tree object (boradcast())
  - 4 Step: For each data, get the k-th nearest neighbor (rdd, sklearn, k-d tree)

RUNNING ON DATASET

- ghpb-fpea.tsv (a taxi related dataset)
  - The file is 184 MB, contains 1.2 million rows.
- Running time—10 minutes.
  - We can get the results in 10 minutes in Spark.
  - For comparison, Nested-loop will run 12 minutes for 10k rows.
- We found the top outliers are meaningful.
  - We choose 2 columns from the taxi data to analyze
  - fare_amount and trip_distance

TOP OUTLIERS

Normal Data

trip_distance and fare_amount are positive correlation

fare_amount	trip_distance	passenger_count	pickup_longitude	lpep_dropoff_datetime	lpep_pickup_datetime	vendorid	tolls_amc
4	0.5	1	-73.965065002441406	2013-12-06T12:14:06	2013-12-06T12:11:05	1	
52.5	14.16	5	-73.958465576171875	2013-11-23T02:36:01	2013-11-23T01:31:07	2	
5	0.6	1	-73.923065185546875	2013-12-13T10:30:26	2013-12-13T10:26:02	1	
9.5	1.84	1	-73.964912414550781	2013-10-05T11:29:47	2013-10-05T11:19:04	2	
8.5	1.69	1	-73.902473449707031	2013-12-12T13:42:23	2013-12-12T13:33:58	2	
24.5	6.71	5	-73.951698303222656	2013-11-24T04:12:53	2013-11-24T03:46:53	2	

TOP OUTLIERS

Top 5 outliers

fare_amount	trip_distance	passenger_count	pickup_longitude	lpep_dropoff_datetime	lpep_pickup_datetime	vendorid	tolls_an
9999.99	0	5	-73.937843322753906	2013-12-24T11:30:50	2013-12-23T12:37:59	2	
5048.5	0	3	-73.937782287597656	2013-11-21T11:54:24	2013-11-21T03:02:20	2	
3569	0	2	0	2013-08-03T16:57:31	2013-08-02T18:04:20	2	
3259.5	0	1	0	2013-08-08T22:25:33	2013-08-08T09:50:45	2	
1900	0	4	-73.937797546386719	2013-09-07T09:33:29	2013-09-06T18:17:09	2	

Top 20 outliers

fare_amount	trip_distance	passenger_count	pickup_longitude	lpep_dropoff_datetime	lpep_pickup_datetime	vendorid	tolls_am
258.5	101.11	1	-73.830352783203125	2013-12-01T18:49:00	2013-12-01T16:36:50	2	

A FURTHER IMPROVEMENT

For the data of 10 million rows

- K-means to select the candidate outliers
  - Only the data points that are far away from the cluster center are possible to be outliers
- Calculate the k-th nearest neighbor of the candidate outliers
  - Get the candidate points rdd
  - Only calculate the nearest neighbor of the small rdd
  - The rest of the operations is same as before

COMPARE THE RESULTS

- We compare the results of the improved algorithm and the index tree algorithm.
  - For the top 10 outliers, the results is completely same
  - For the top 20 outliers, there are 1-2 different outliers