Fast Fine-Grained Air Quality Index Level Prediction Using Random Forest Algorithm on Cluster Computing of Spark

Chuanting Zhang and Dongfeng Yuan

Shandong University, Jinan, China

August 12, 2015

Outline

1. Introduction
2. Implementation of Radom Forests Algorithm on Spark
3. Experiment & Results Analysis
4. Conclusion

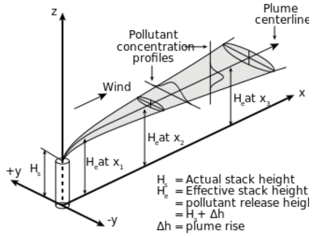
Background

- What are the pollutants in the air?
  - NO2/SO2/PM2.5/PM10
- Why it matters?
  - 1 in 8 deaths linked to air pollution (WHO)
- Reality
  - Building an air quality monitor station is difficulty

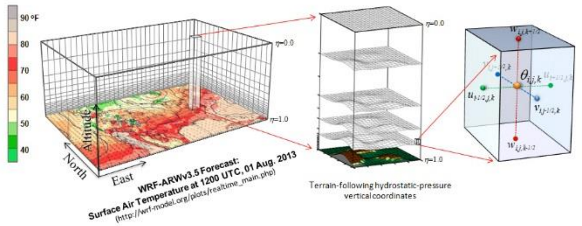
All pictures are from Google-- www.google.com

Challenges

Traditional methods do not work well



Gaussian plume model



Community Multi-scale Air Quality model

Difficult to decide the application conditions and many key parameters are arduous to obtain

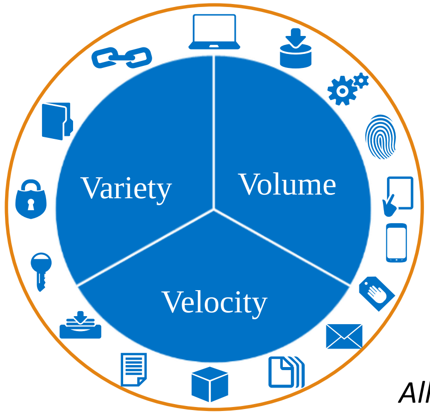
Limited to a few gasses: CO_2 and CO

Sensors for detecting aerosol are not portable: PM10

A long period of sensing process, 1-2 hours

New techniques are facing big data challenges

Data mining and machine learning techniques play a critical role in air quality index prediction



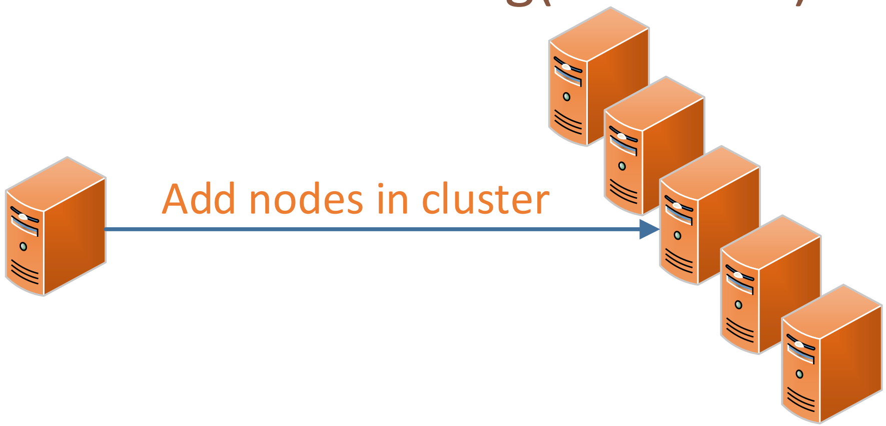
- ✓ Millions even billions records in DB
- ✓ Data is generated quickly
- ✓ Many kinds of data can be used to predict AQI

All pictures are from Google-- www.google.com

How to deal with the big data challenges in AQI prediction?

Vertical scaling(scale-up)

Horizontal scaling(scale-out)



- Optimal data structure design(Hash/RB Tree)
- Parallel programming(OpenMP/MPI)

- Fault tolerance, concurrency
- Distributed platforms like Apache Hadoop and Spark

We focus on the **design and implementation of traditional random forests algorithm based on Apache Spark** and use this algorithm to do AQI prediction, at last, we test the algorithm's **scalability** when deal with big data set.

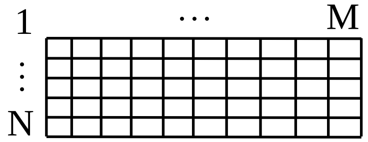
Outline

1. Introduction
2. Implementation of Radom Forests Algorithm on Spark
3. Experiment & Results Analysis
4. Conclusion

Random forests algorithm

- Random forests are an ensemble classifier that consists of many decision trees, and output the class that is the mode of the output by individual trees--CART.

Random sample



- ✓ Row: sample N with replacement
- ✓ Column: sample m attributes without replacement($m = \sqrt{M}$)

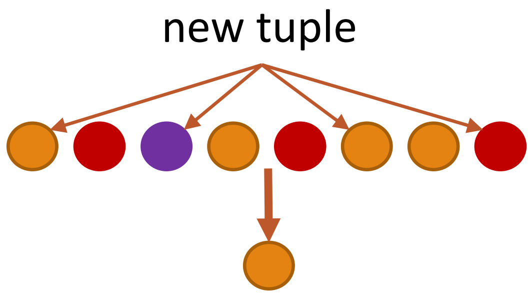
Impurity measurement

$$Gini(D) = 1 - \sum_{i=1}^{k} \left(\frac{|C_i|}{|D|}\right)^2$$

$$Gini_A(D) = \frac{|D_1|}{|D|} Gini(D_1) + \frac{|D_2|}{|D|} Gini(D_2)$$

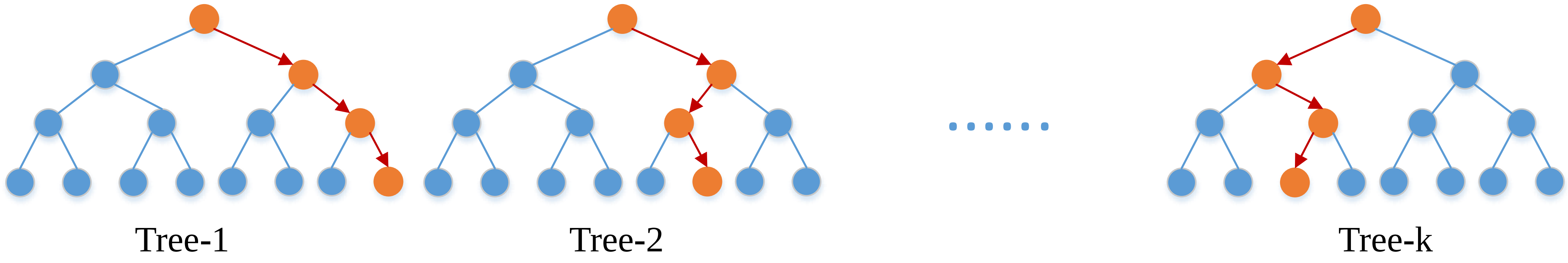
- ✓ The attribute that with the minimum Gini index is selected as the splitting attribute

Vote of classifiers



- ✓ Classification that has the most votes is the final prediction

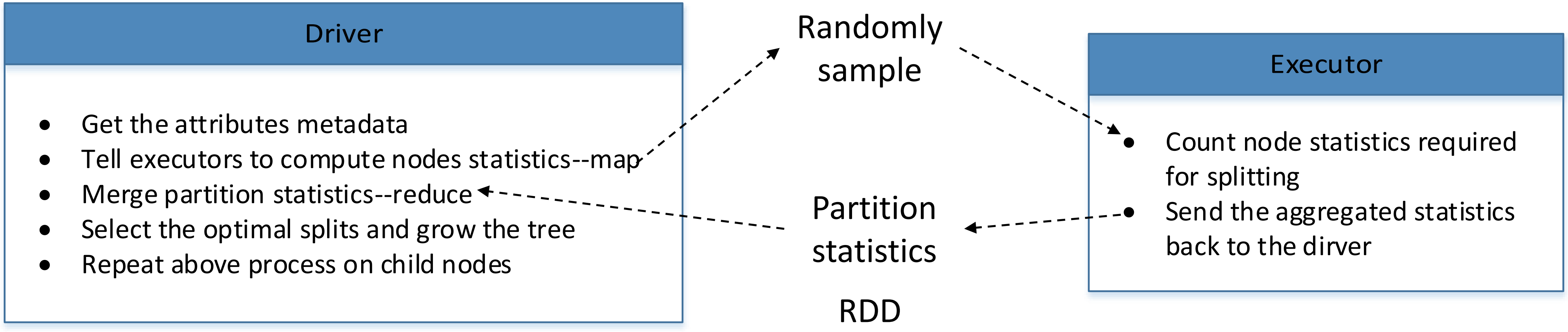
Advantages



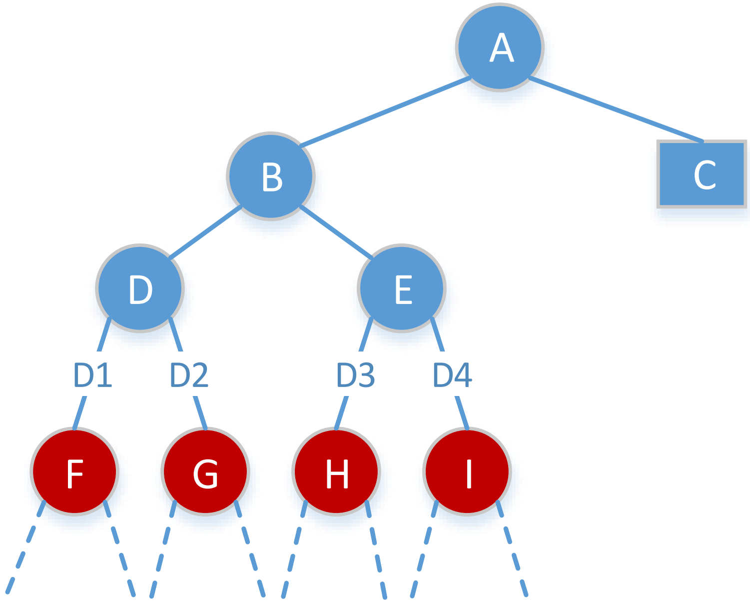
- It is one of the most **accurate** learning algorithms available.
- It gives estimates of **what variables are important** in the classification.
- It has an effective method for estimating **missing data** and maintains accuracy when a large proportion of the data is missing.

Distributed tree training in Spark

- The idea is from Google' PLANET (Parallel Learner for Assembling Numerous Ensemble Trees) and Sequoia Forest.
- Individual trees are built node by node and level by level in the driver node.
- At each iteration, individual executors compute partition statistics that is required to determine node splits.



Map and Reduce



- Mappers:

✓ For each node store statistics of the data entering the node

$$< \mathrm{NodeID}, |C_k|, |D_n|, k >$$

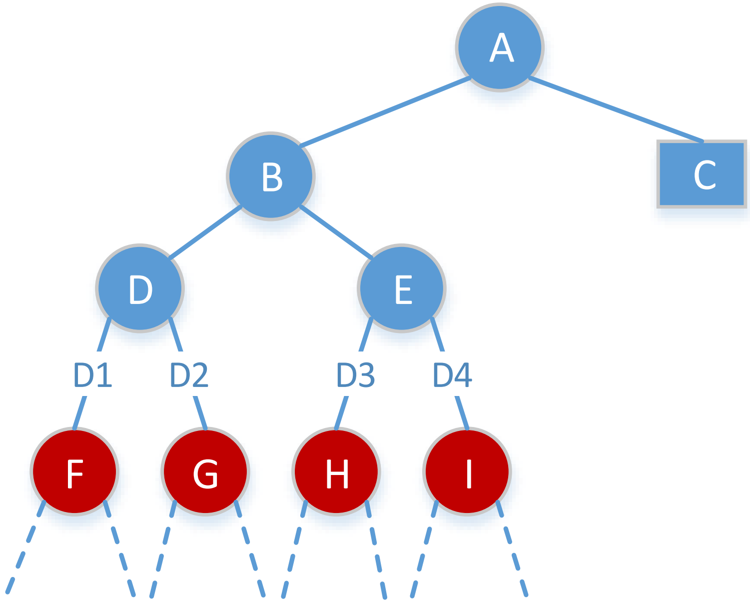
For example: $< F, |C_1|, |C_2|, |D| = 100, k = 2 >$

✓ For each split store statistics

$$< \mathrm{NodeID}, \mathrm{Split}, |C_k|, |D_n|, k >$$

For example: $< F, A_1 < 3, |C_1|, |C_2|, |D| = 20, k = 2 >$

Map and Reduce

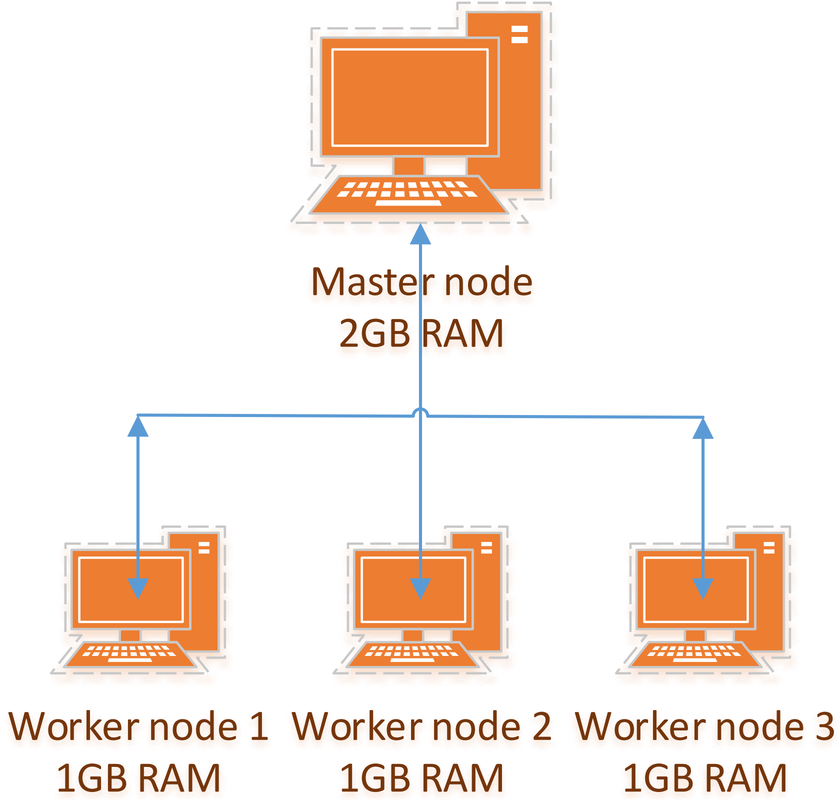


- Reducers:
  - ✓ Load all the $< \mathrm{NodeID}, |\mathrm{C_k}|, |\mathrm{D_n}|, \mathrm{k} >$pairs and aggregate per node statistics.
  - ✓ Load all the $< \mathrm{NodeID}, \mathrm{Split}, |\mathrm{C_k}|, |\mathrm{D_n}|, \mathrm{k} >$data and aggregate every possible split statistics.
  - ✓ For each NodeID, output the best split (locally).
- Driver
  - ✓ Collects outputs from all reducers <split.NodeID, attribute, value, Impurity>, for each node decides the best split(globally).
  - ✓ If D* is small enough, build the subtree locally on driver to speed up the whole process. Else run map and reduce jobs.

Outline

1. Introduction
2. Implementation of Radom Forests Algorithm on Spark
3. Experiment & Results Analysis
4. Conclusion

Experiment environment and data set description



Configuration of a 4-node cluster

- A 4-node cluster.
- The data set comes from the public data source of MSRA. It is comprised of 1 year 36 air quality monitor stations' data of Beijing.

Temperature	Weather	Wind	Pressure	Humidity	PM2.5
-5	Sunny	3	1031	46	M

- There are six class labels, {Good, Moderate, Unhealthy for Sensitive, Unhealthy, Very Unhealthy, Hazardous}.
- We use the former 5 attributes to predict the level of PM2.5.

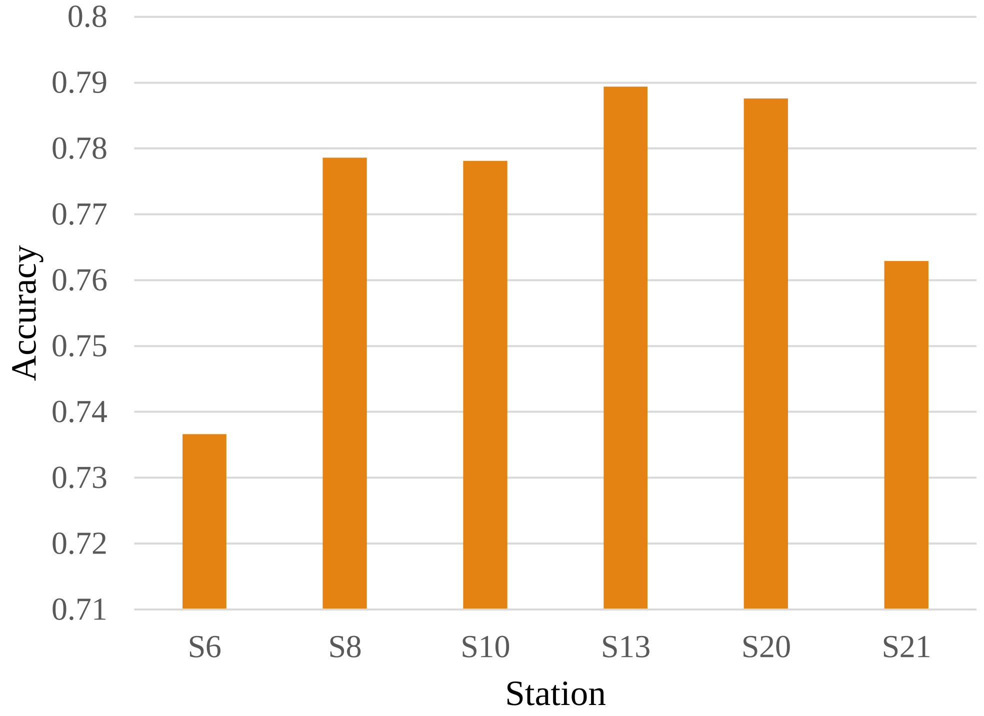
Experimental results

Confusion matrix

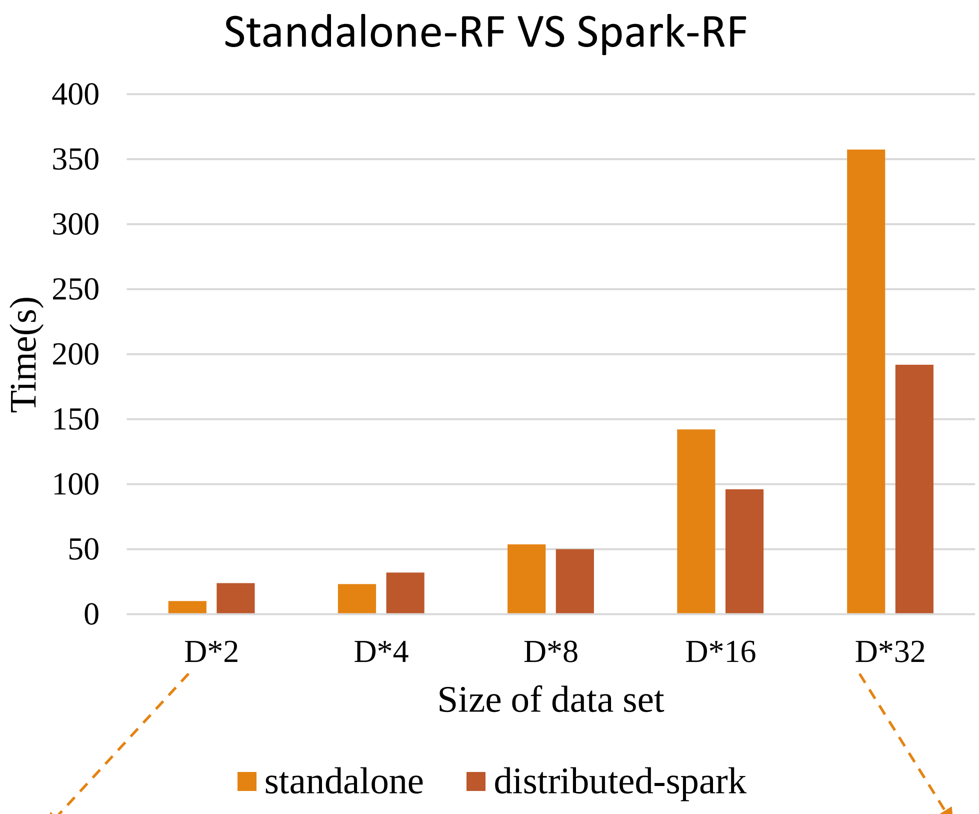
Truth	Predictions					
	G	M	US	U	VU&H	Recall
G	1088	157	13	3	2	0.8614
M	169	755	237	27	11	0.6297
US	10	98	966	278	26	0.7012
U	10	38	72	1584	184	0.8390
VU&H	5	1	3	59	964	0.9341
Precision	0.8487	0.7197	0.7483	0.8112	0.8121	0.7925

- In the DB, we select one of the 36 stations and it's data as test dataset, and the other stations data as training set.
- The accuracy is about 0.79, all the precisions in prediction are above 0.7.

Results of different tests

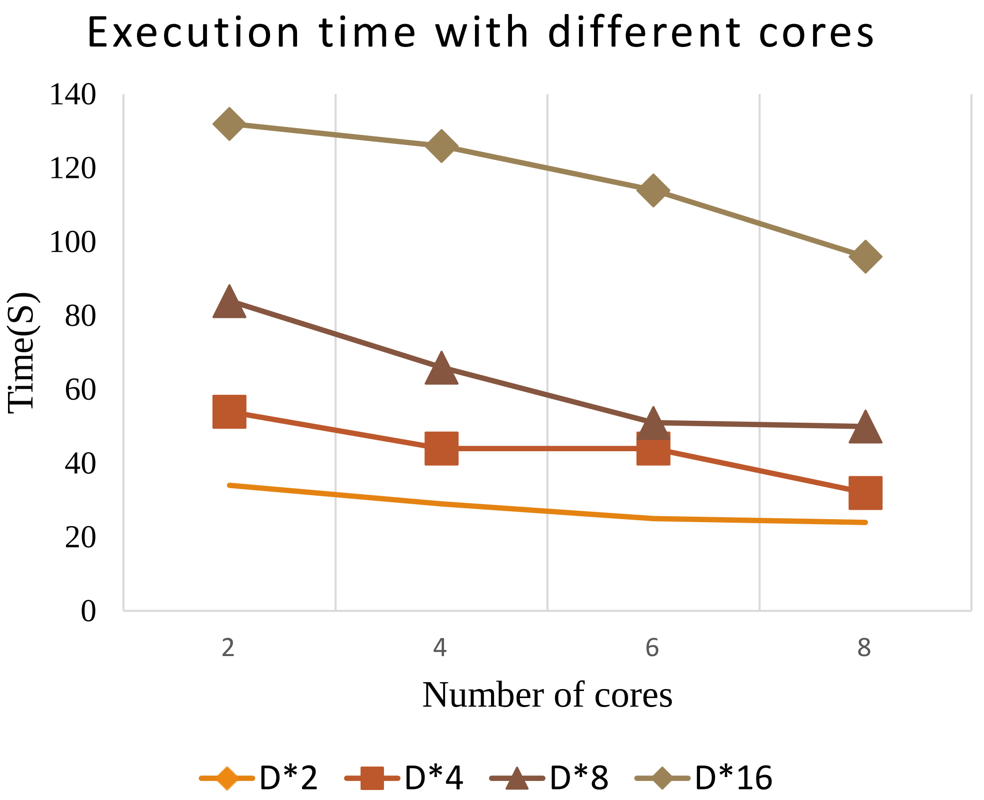


Scalability for big data



3 million

108 million



Outline

1. Introduction
2. Implementation of Radom Forests Algorithm on Spark
3. Experiment & Results Analysis
4. Conclusion

Conclusion

- This paper investigated a fast parallelized AQI level prediction method using random forests algorithm on Apache Spark. The individual trees are built node by node and level by level. Besides, a locally sub-tree building scheme is used to accelerate the processing speed.
- Experimental results showed that the parallelized random forests algorithm achieved relatively high accuracy in prediction. The results also proved the algorithm's effectiveness and efficiency when dealing with big data set.

Thanks for your attention!

chuanting.zhang@gmail.com