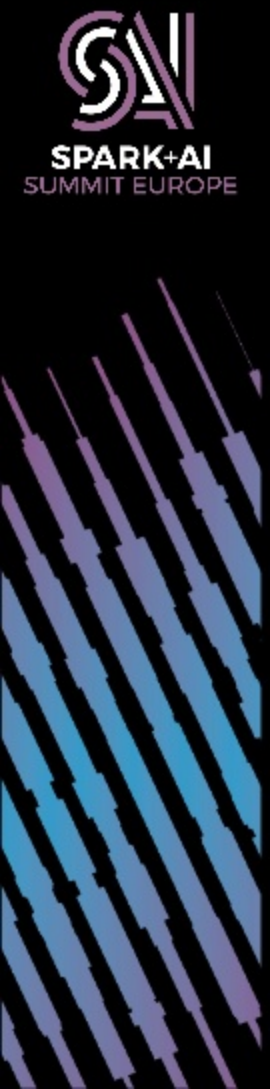


Spark-ITS: Indexing for Large-Scale Time Series Data on Spark

Liang Zhang (lzhang6@wpi.edu)
Data Science Dept.,
Worcester Polytechnic Institute

#SAISEco5

Data Science Research Group @ Worcester Polytechnic Institute

Prof. Elke A. Rundensteiner

Prof. Mohamed Y. Eltabakh

Liang Zhang

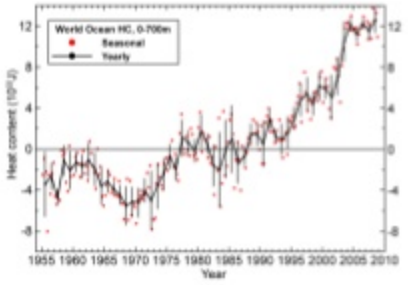
Noura Alghamdi

Liang Zhang, Noura Alghamdi, Mohamed Y. Eltabakh, Elke A. Rundensteiner. *TARDIS: Distributed Indexing Framework for Big Time Series Data*. Proceedings of 35th IEEE International Conference on Data Engineering **ICDE**, 2019

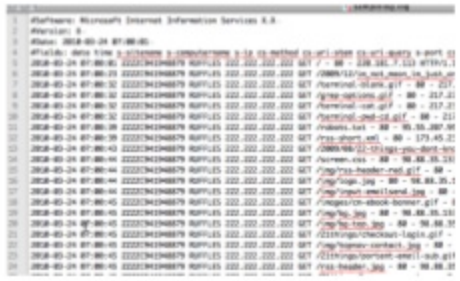
Outline

- Motivation
- Background
- Spark-ITS Framework
  - Overview
  - Index Construction
  - Query Processing
- Performance Evaluation

Time Series are Continuously Produced Everywhere



Climate data



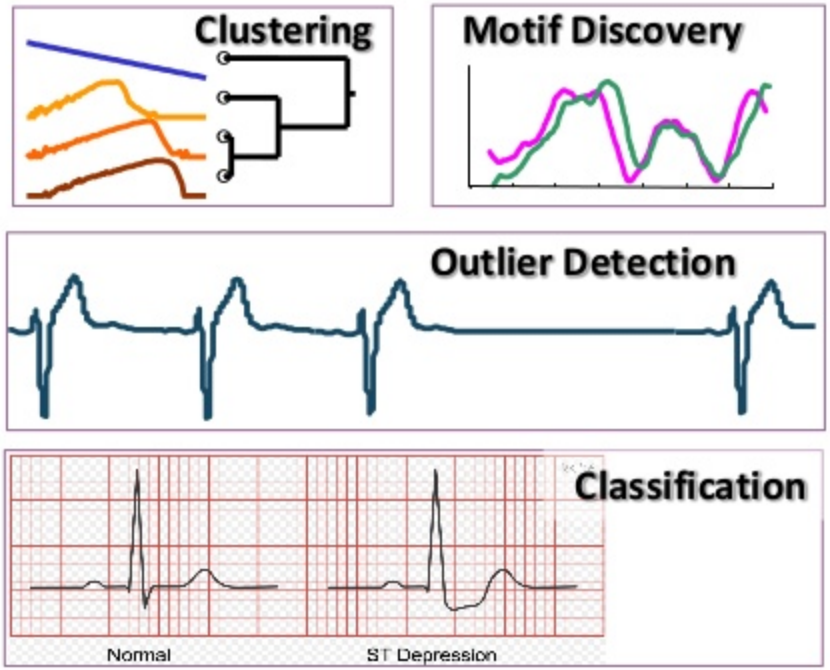
Web log

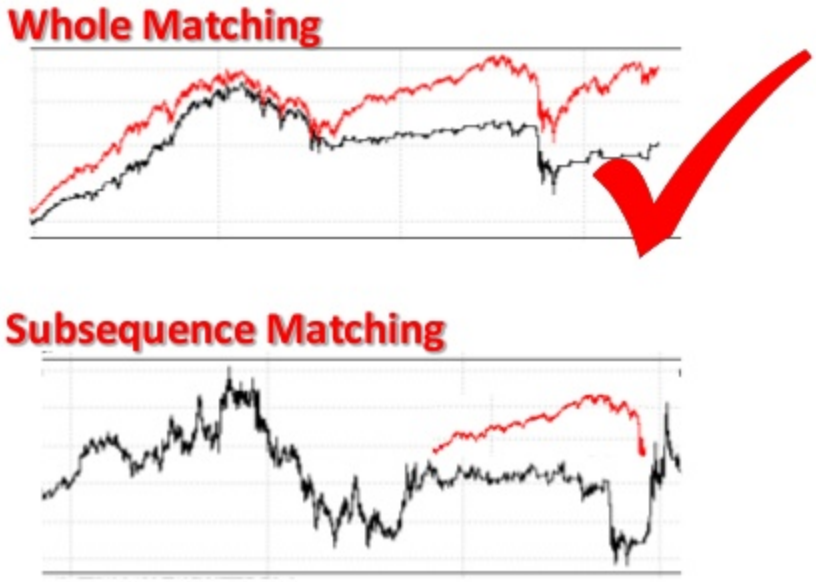
EEG

Stock price

- How to deal with billions of time series?

Almost all Time Series Data Mining Tasks rely on **Similarity Query**





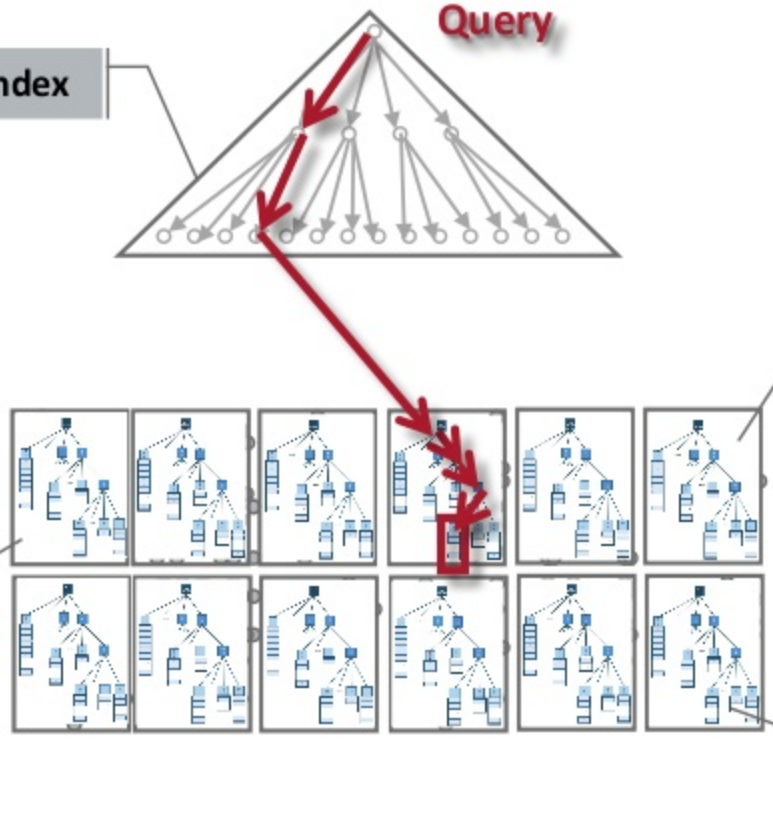
Esling, Philippe, and Carlos Agon. "Time-series data mining." *ACM (CSUR)* 45.1 (2012): 12.

Spark-ITS

- A new **Index Tree** and an effective **Signature** to simplify the cardinality conversion and keep better similarity
- A **Distributed Index Framework** to support **large-scale** time series dataset
- Efficient algorithms for **Exact Match** and **kNN Approximate** queries process

Spark-ITS Overview

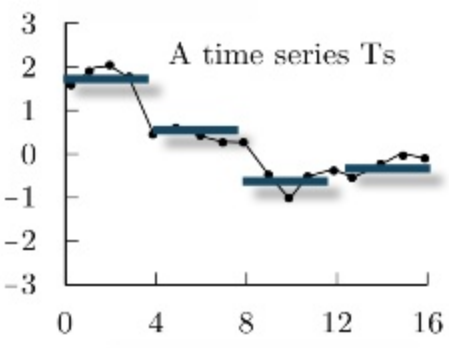
1. Sampling
2. Node Statistic
3. Build Index Tree
4. Assign Partition ID



1. Construct Local Structure
2. Construct Bloom Filter

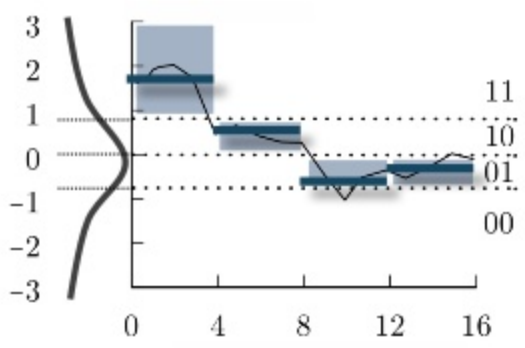
1. Read and convert data
2. Shuffle data

Background: iSAX Representation



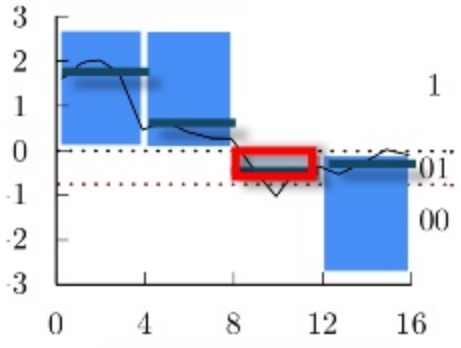
A time series of length 16

PAA representation with 4 segments



SAX representation with 4 segments and ***cardinality*** 4

[11,10,01,00]



iSAX representation with 4 segments and ***variable cardinality***

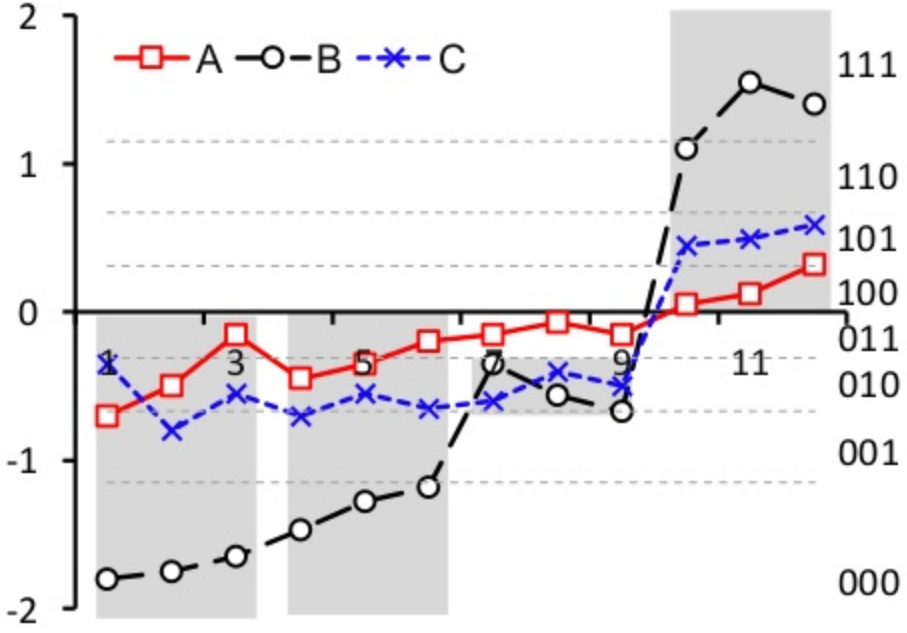
$[1_2, 1_2, \mathbf{01_4}, 0_2]$

PAA: Piecewise Aggregate Approximation
iSAX: indexable Symbolic Aggregate approXimation

Shieh, Jin, and Eamonn Keogh. "iSAX: indexing and mining terabyte sized time series." *SIGKDD* ACM, 2008.
Camerra, A., Palpanas, T., Shieh, J., & Keogh, E. "iSAX 2.0: Indexing and mining one billion time series." ICDM, 2010

Word-level Similarity

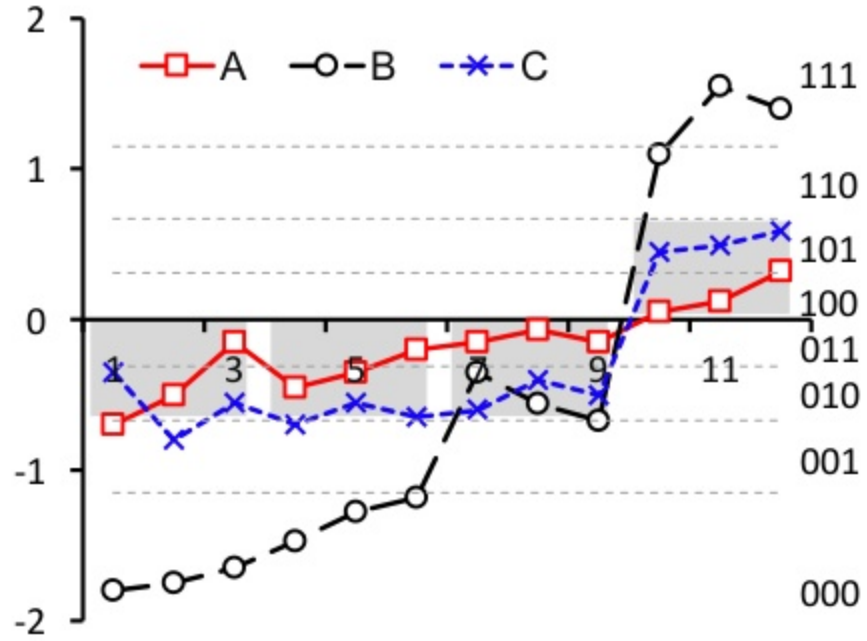
State-of-the-art: Character-level Similarity



A: $[0_1, 0_1, 011_3, 1_1]$
B: $[0_1, 0_1, 010_3, 1_1]$
C: $[0_1, 0_1, 010_3, 1_1]$

B and **C** are similar

Proposed: Word-level Similarity

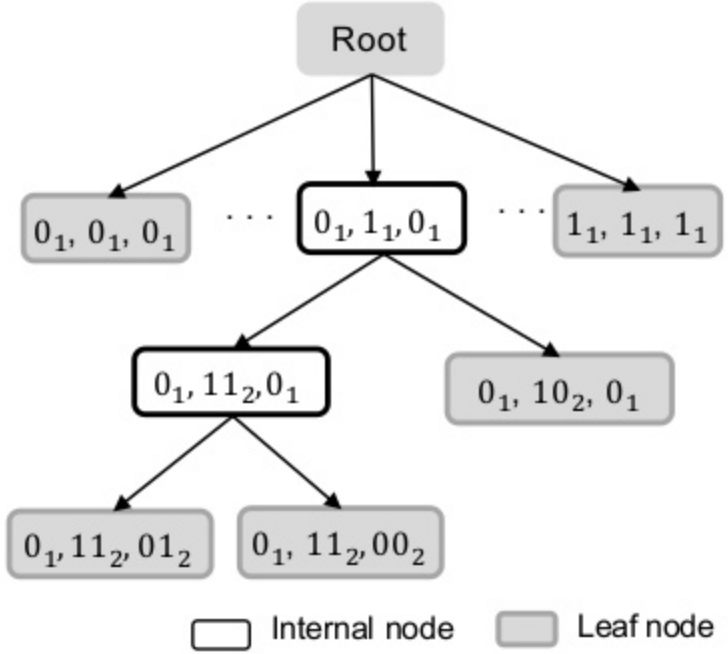


A: $[01_2, 01_2, 01_2, 10_2]$
B: $[00_2, 00_2, 01_2, 11_2]$
C: $[01_2, 01_2, 01_2, 10_2]$

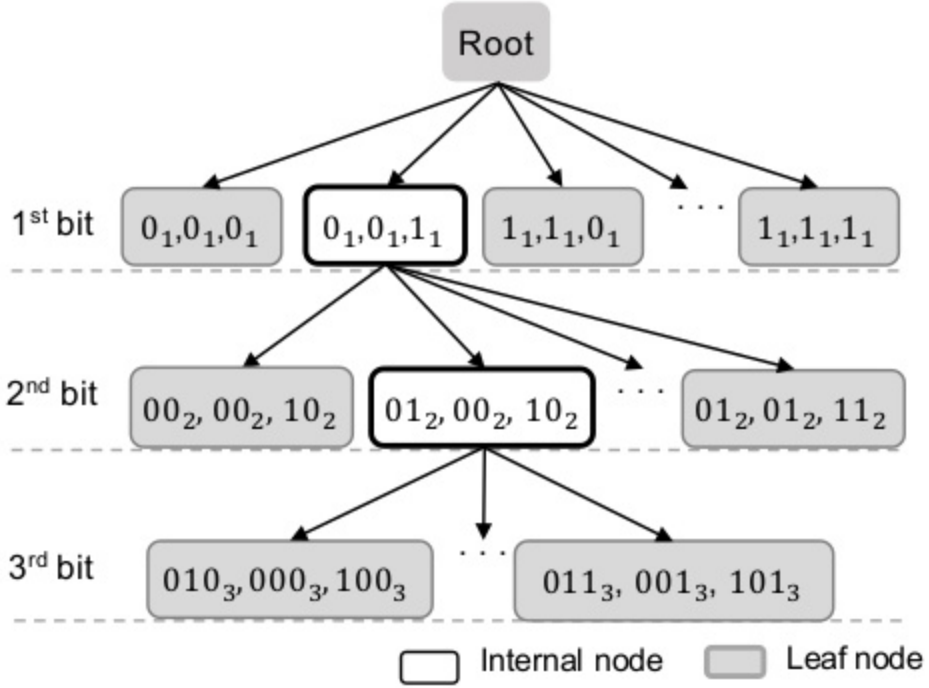
A and **C** are similar

New Index Tree Supports Word-level Similarity

State-of-the-art: iSAX Binary Tree



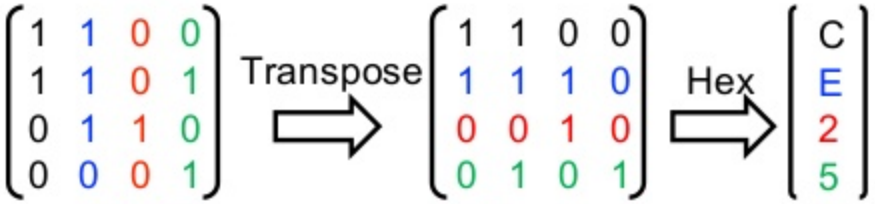
Proposed: iSAX-T K-ary Tree



iSAX-T(Transpose) Signature

Time series:

[1100, 1101, 0110, 0001]



					iSAX-T
SAX(T,4,16) =	{1100,	1101,	0110,	0001}	= CE25
SAX(T,4,8) =	{110,	110,	011,	000 }	= CE2
SAX(T,4,4) =	{11,	11,	01,	00 }	= CE
SAX(T,4,2) =	{1,	1,	0,	0 }	= C

Outline

- Motivation
- Background
- Spark-ITS Framework
  - Overview
  - Index Construction
  - Query Processing
- Performance Evaluation

Global Index[1/4]: Sampling

1 Terabyte

Sampling

100 G

Map

iSAX-T(b bits), Freq:1		
1256ae3e	,	1
0134ef45	,	1
234567ae	,	1
1256ae3e	,	1
1256ae3e	,	1
234567ae	,	1
237867ae	,	1
024567ae	,	1
6243371e	,	1
.....		
452167ef	,	1

0.9 G

Reduce

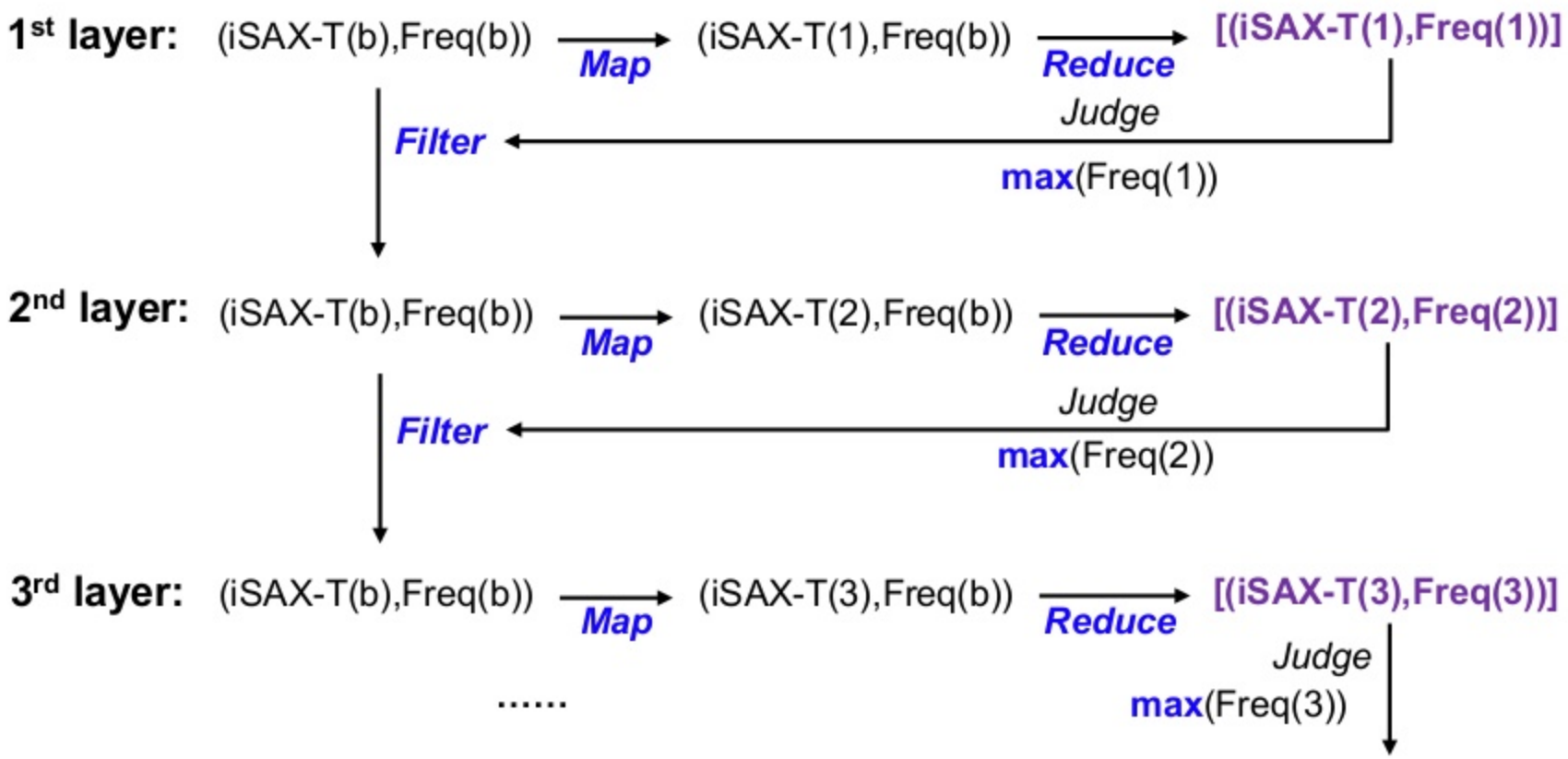
iSAX-T(b bits), Freq(b bits)		
1256ae3e	,	23
0134ef45	,	2
234567ae	,	20
024567ae	,	4
237867ae	,	10
.....		
452167ef	,	10

0.1 G

Word counting *MapReduce* process

Segment Number: 8, so use 2 letters to represent 1 bit
Initial cardinality: **b bit level**
The data size is based on 1 billion time series with 256 length

Global Index[2/4]: Node Statistic



Global Index[3/4]: Build Tree

1st layer (iSAX-T, Freq)

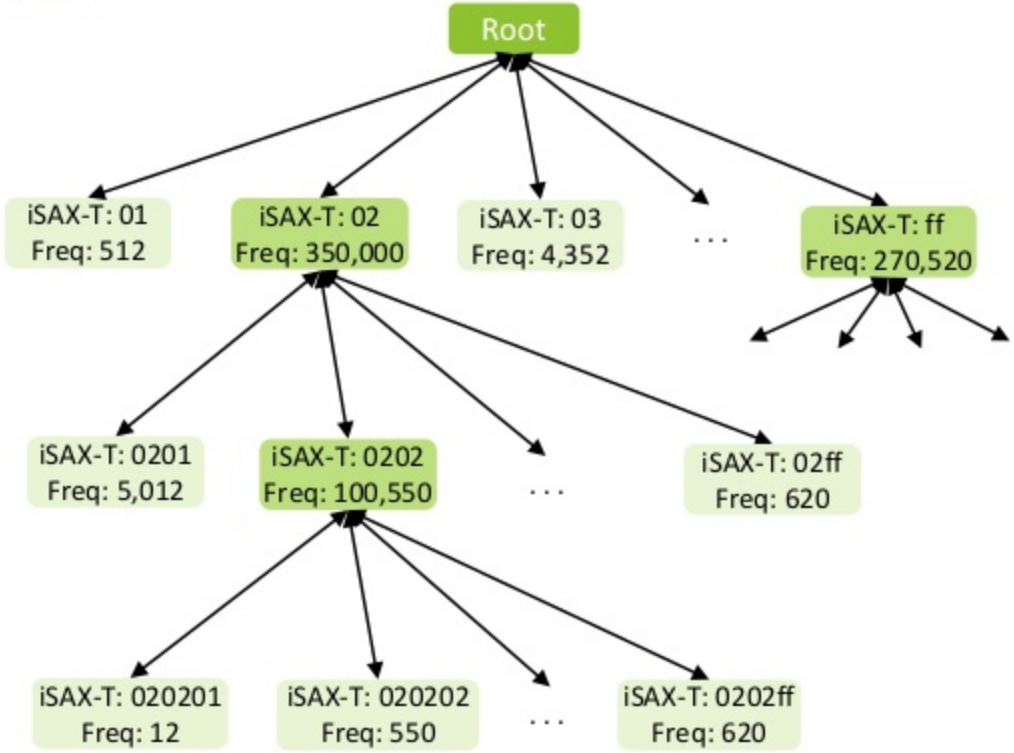
- ("01", 512)
- ("02", 355,000)
- ....
- ("ff", 270,520)

2nd layer (iSAX-T, Freq)

- ("0201", 5,012)
- ("0202", 100,550)
- ....
- ("ffff", 10,520)

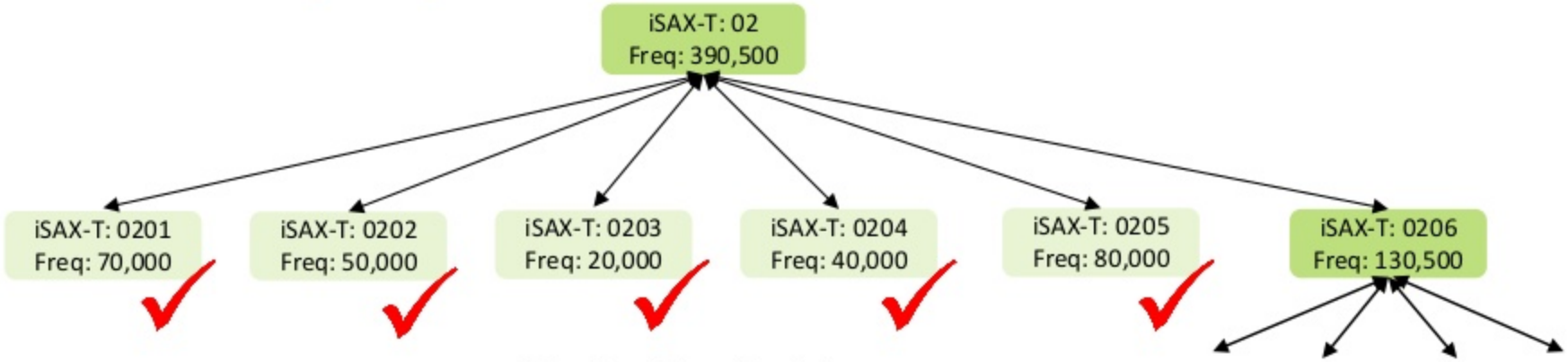
3rd layer (iSAX-T, Freq)

- ("020201", 12)
- ("020202", 550)
- ....
- ("0202ff", 620)



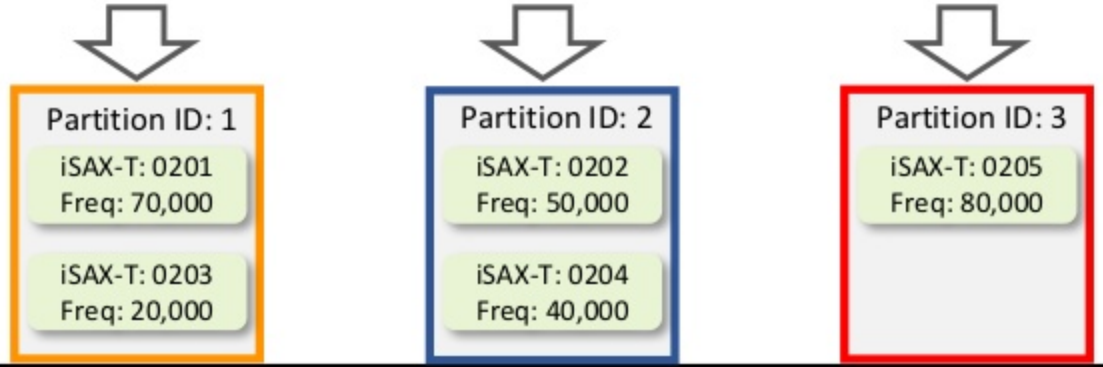
Segment number: 8
Partition Capacity: 100,000

Global Index[4/4]: Assign Partition Id to Leaf Nodes



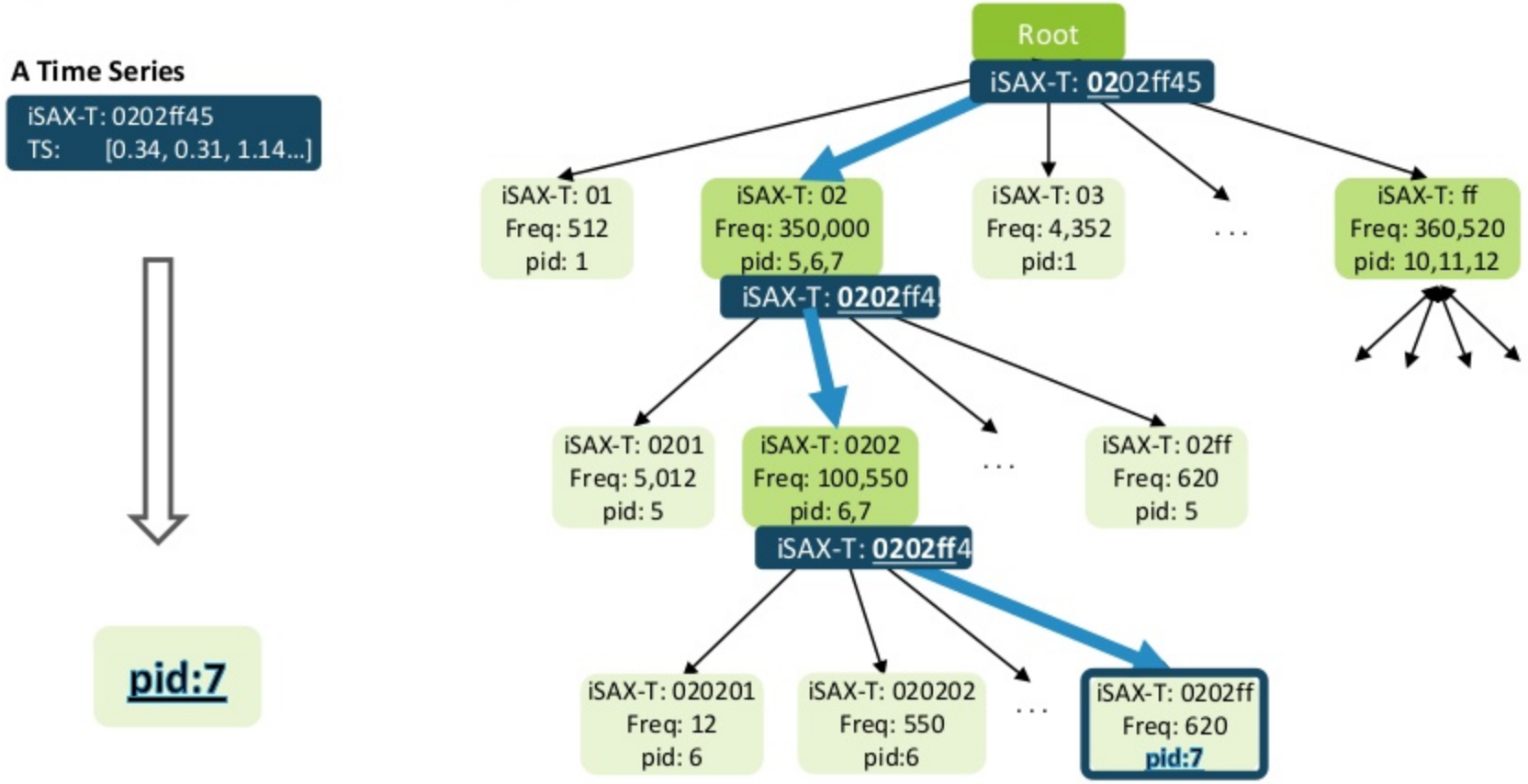
Bin Packing Problem:

How to fit a set of nodes in the smallest numbers of partitions?



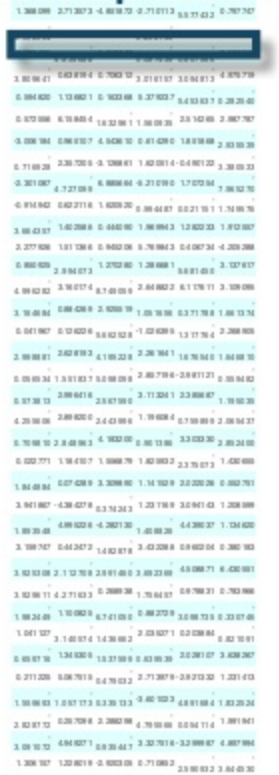
Partition capacity: 100,000

Repartition: Wrap Global Index as the Partitioner

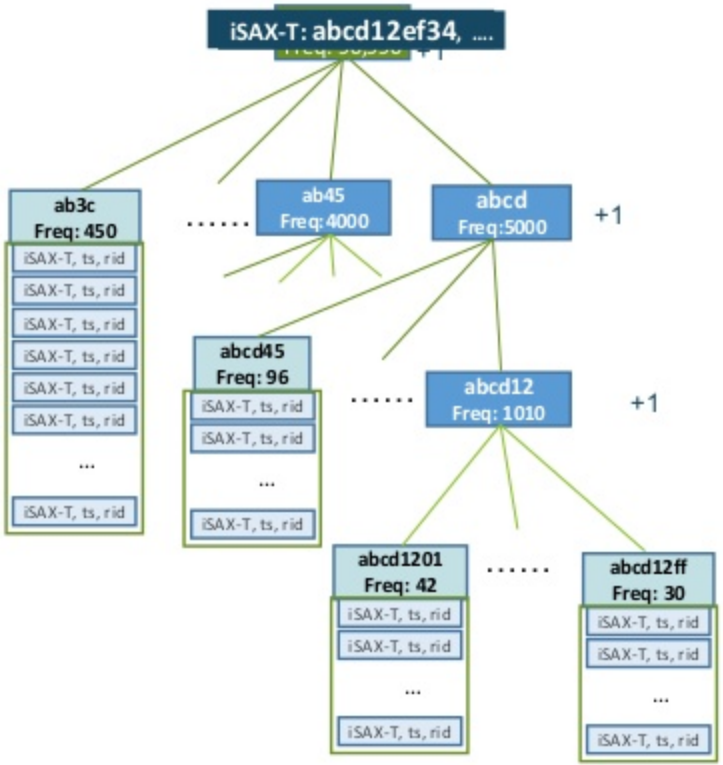


Local Index: Construction Within Each Partition

Time series in one partition



Local Index



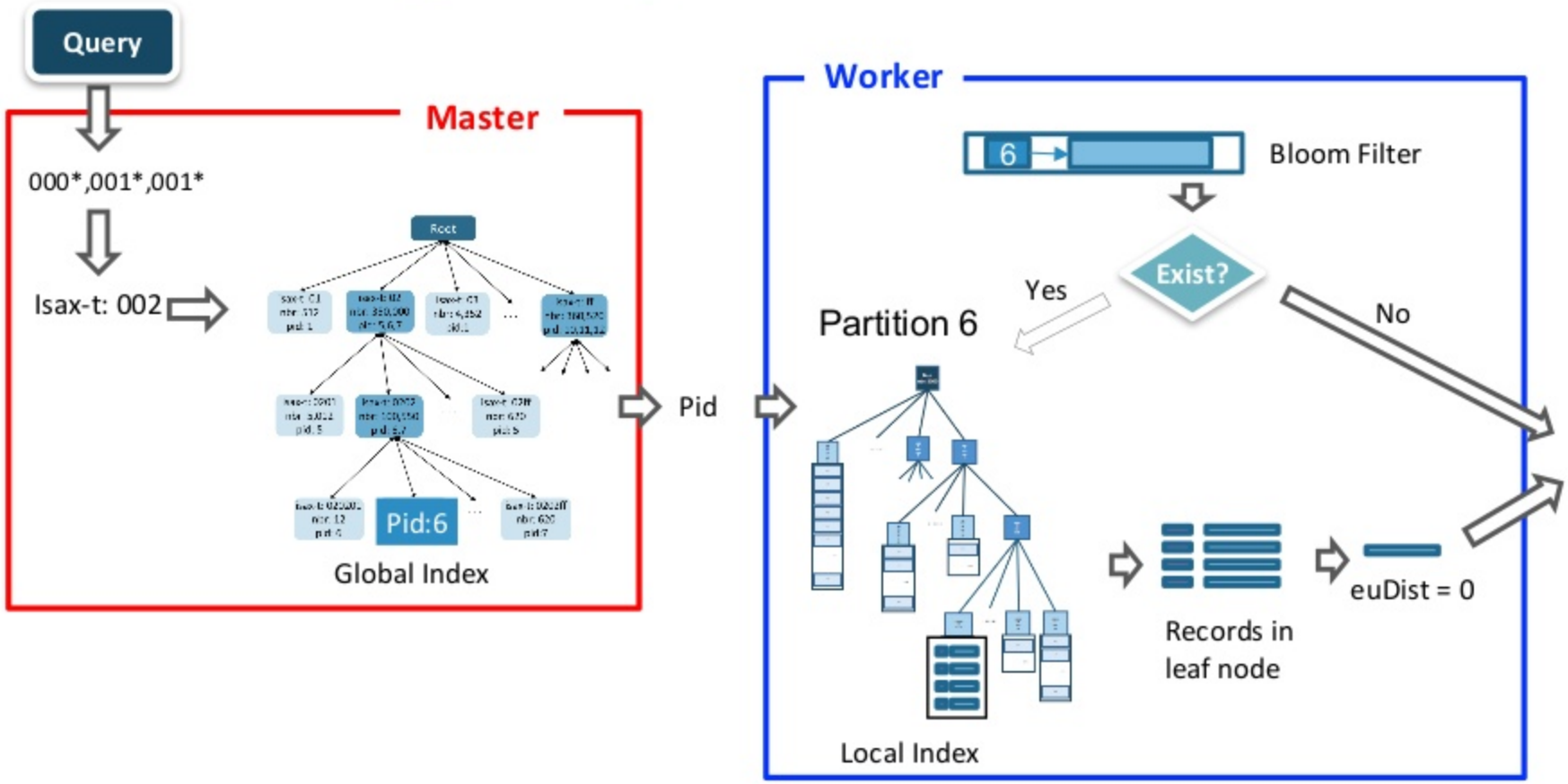
Bloom Filter

Partition capacity: 100,000
Node split threshold: 1000
Segment Number: 8

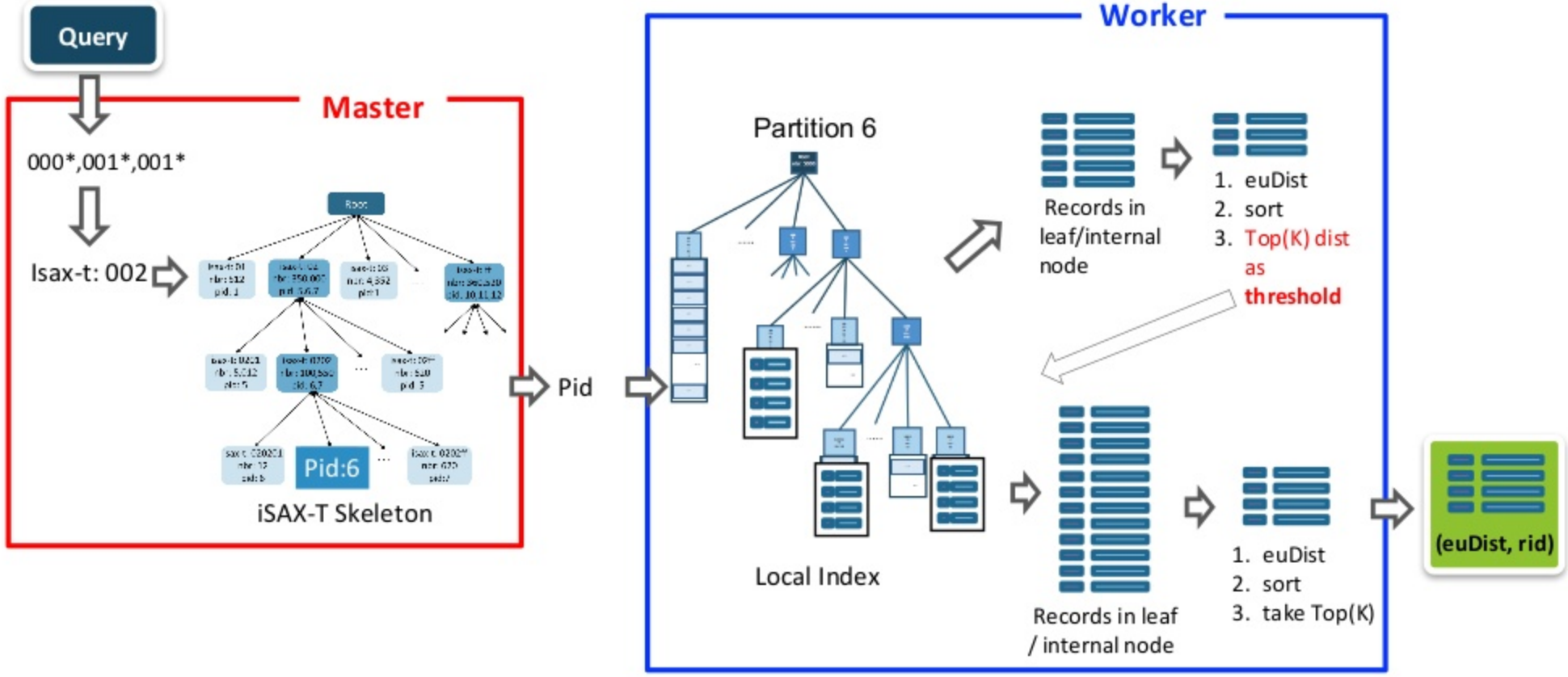
Outline

- Motivation
- Background
- Spark-ITS Framework
  - Overview
  - Index Construction
  - Query Processing
- Performance Evaluation

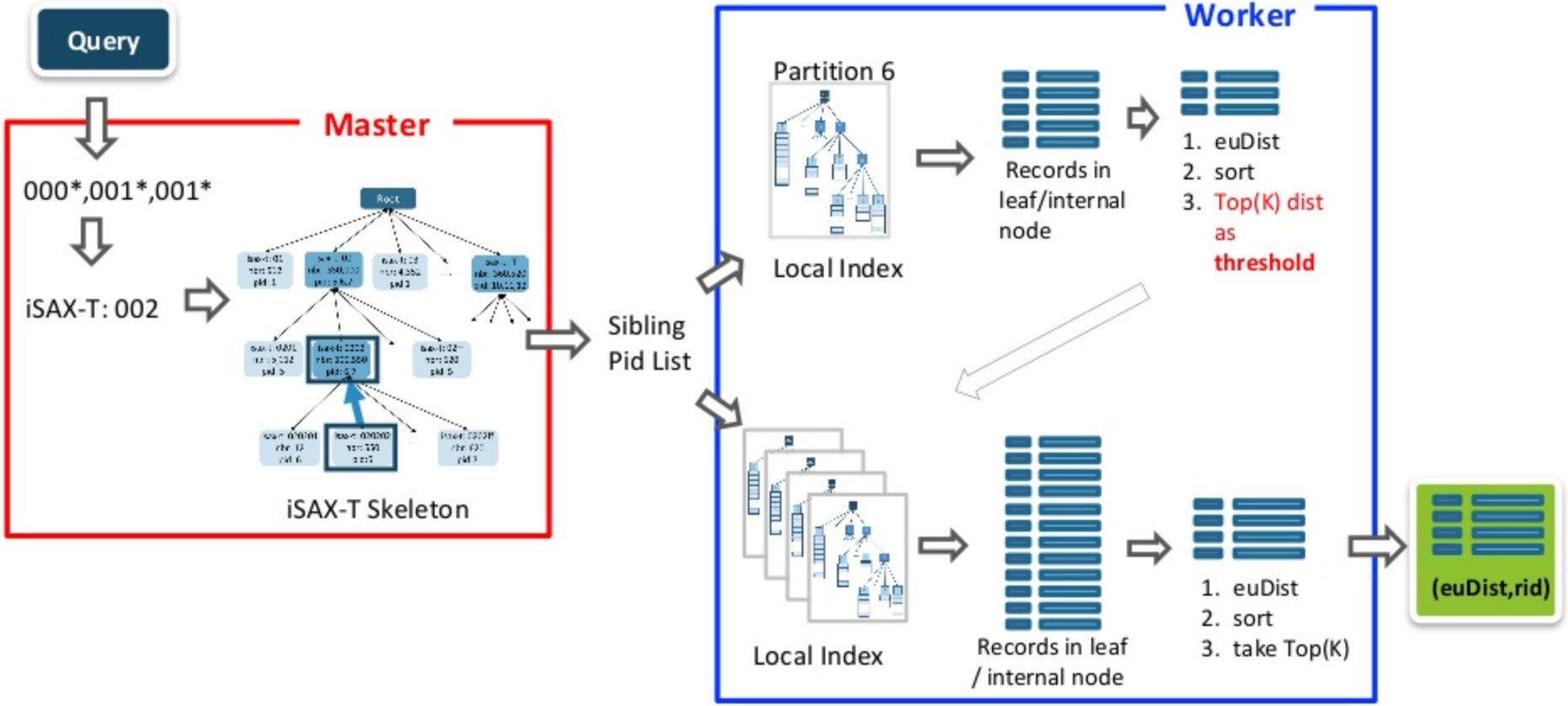
Exact Matching Query



KNN Approximate Query: One Partition Access



KNN Approximate Query: Multi-Partitions Access



Outline

- Motivation
- Background
- Spark-ITS Framework
  - Overview
  - Index Construction
  - Query Processing
- Performance Evaluation

Experimental Setup

HW&SW	Configuration
Spark	2.0.2, Standalone mode
Hadoop	2.7.3
Platform	Ubuntu 16.04. LTS
HW	2 nodes, each node consist of 56 Xeon E5 processors, 500G RAM, 7TB SATA hard drive

Dataset	Size	Length
Random Walk	1 billion	256
Texmex[1]	1 billion	128
DNA[2]	200 million	192
Noaa Climate[3]	200 million	64

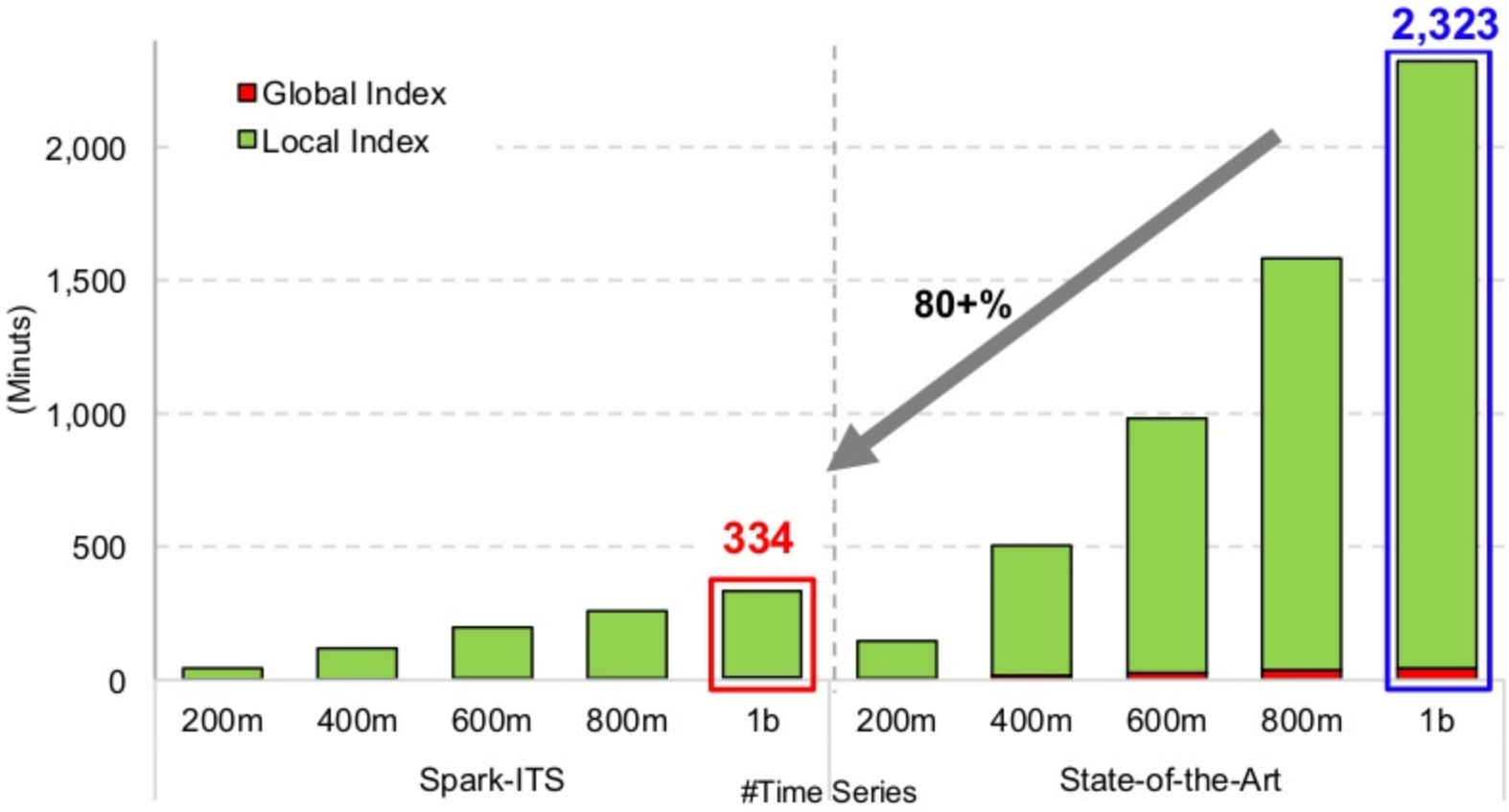
	Baseline	Spark-ITS
Initial cardinality	512	64
Word length	8	8
Sampling percent	10%	10%
Leaf node split threshold of Local index	1000	1000

State-of-the-Art: Yagoubi, Djamel-Edine, et al. "DPiSAX: Massively Distributed Partitioned iSAX." *ICDM 2017*
The initial cardinality of the baseline system is the default value and it needs a large initial value to guarantee enough bit level for binary split.

The dataset is normalized
Each point is saved as float format

Source:
1. http://corpus-texmex.irisa.fr/
2. https://genmone.ucsc.edu
3. https://www.ncdc.gov/

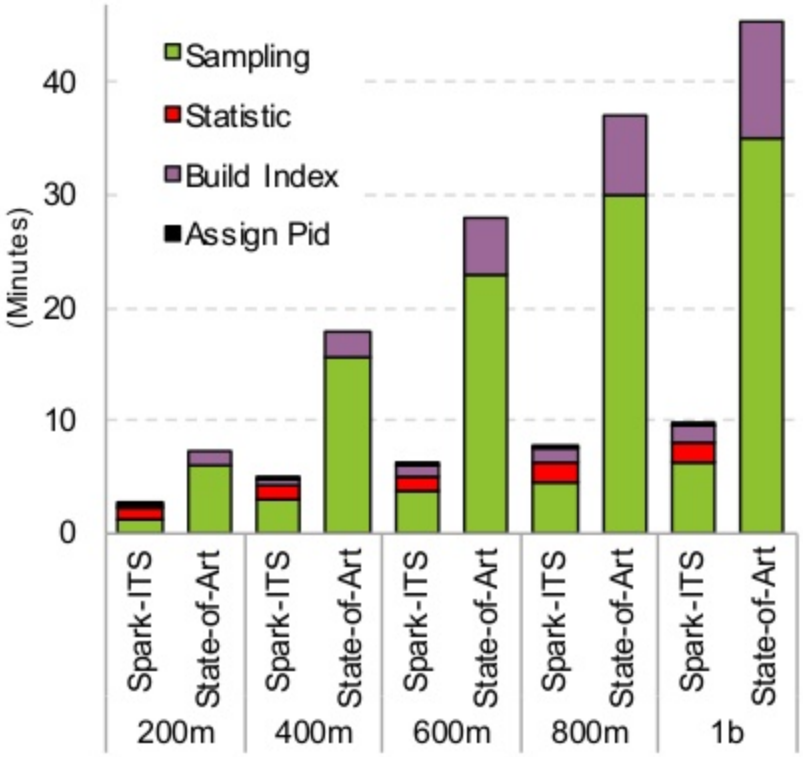
Index Construction Time



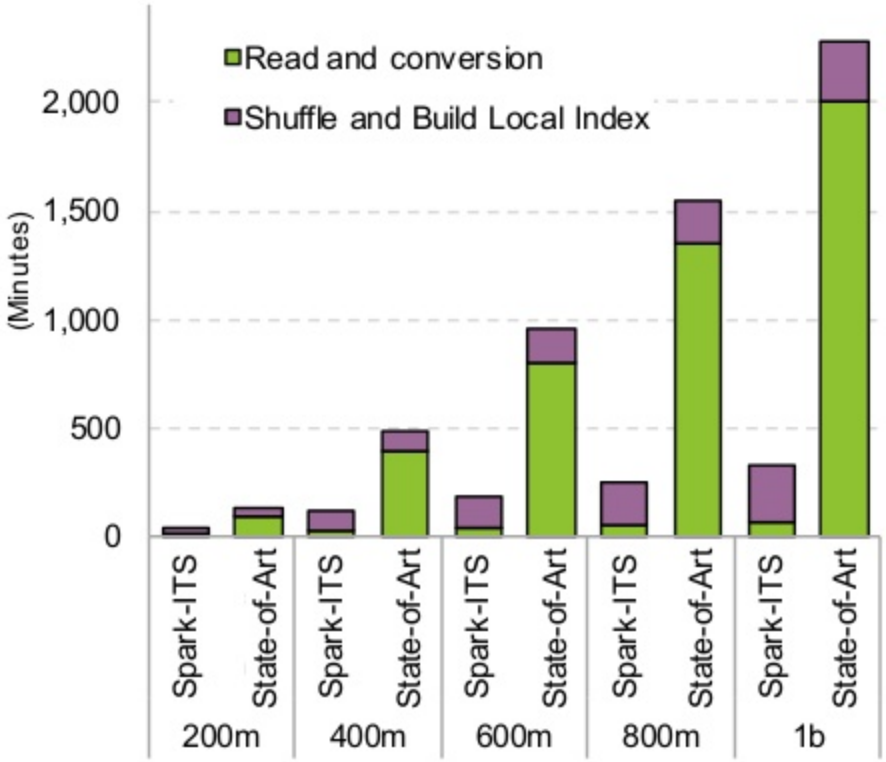
Dataset: Random Walk Benchmark

Index Construction Time: Breakdown

Global Index Time Breakdown

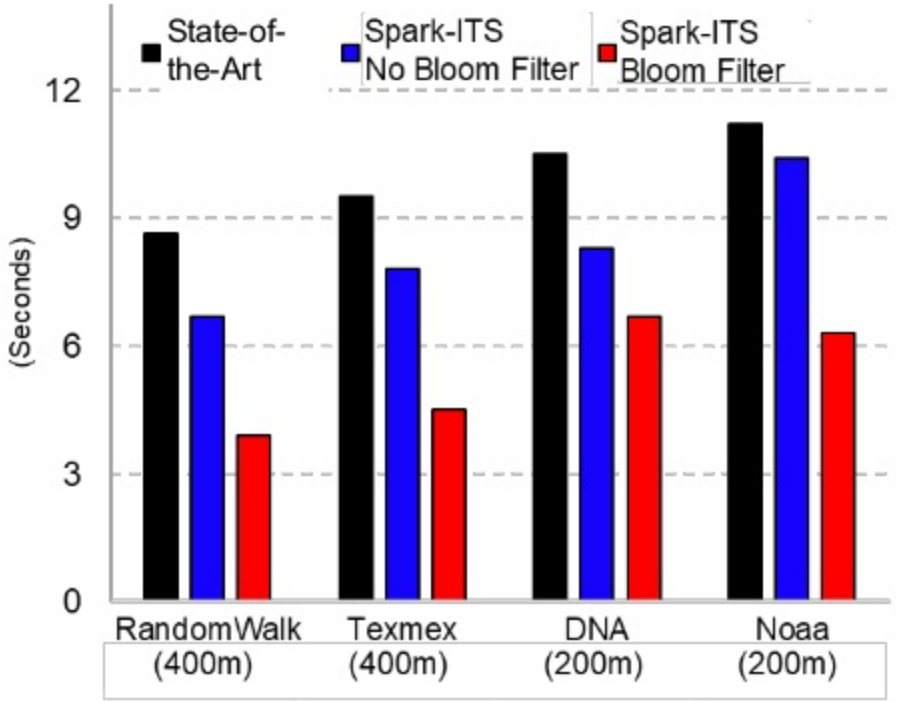


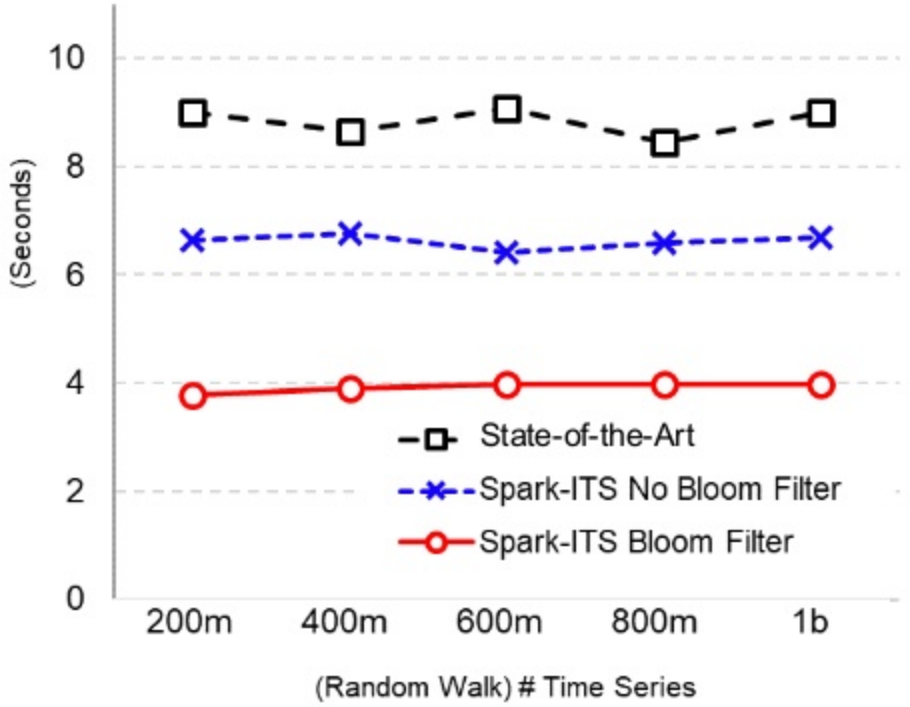
Repartition and Local Index Time Breakdown



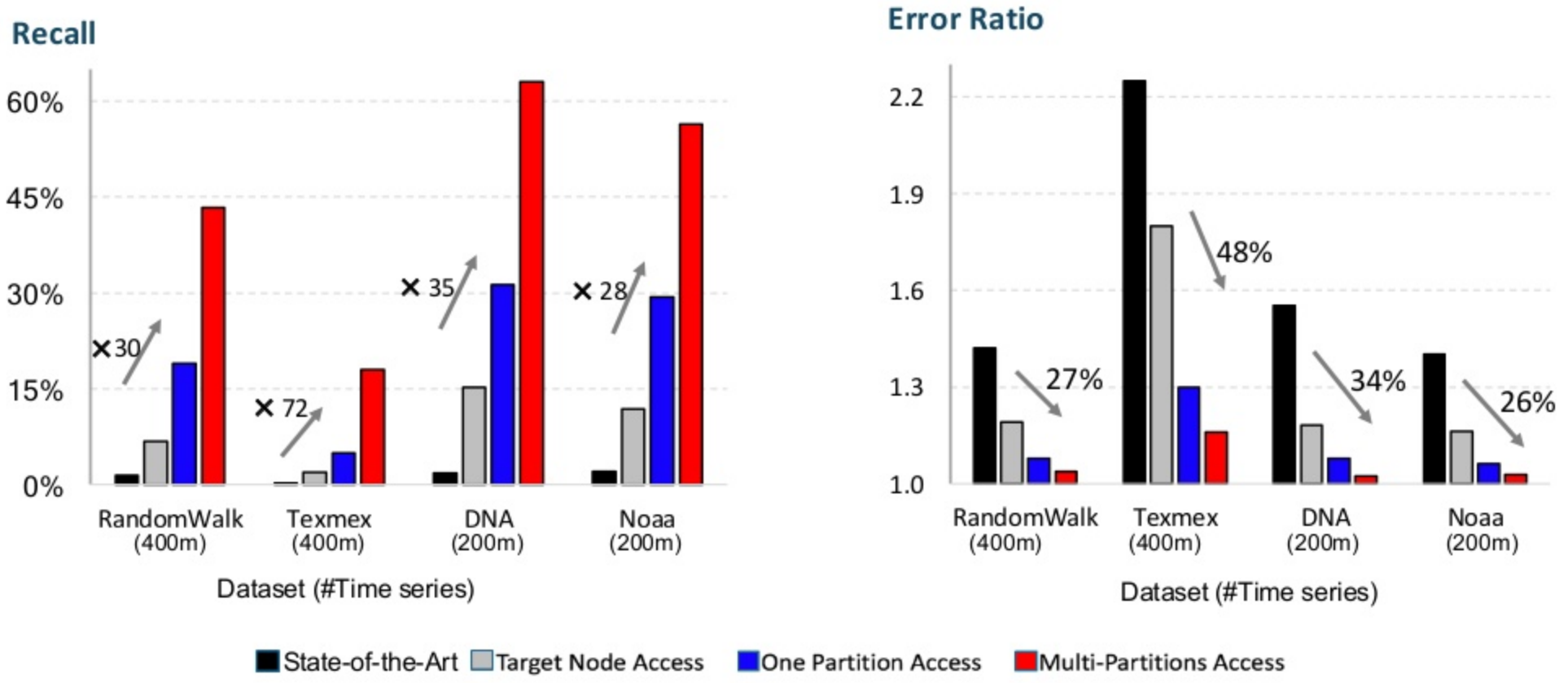
Dataset: Random Walk Benchmark

Exact Matching Query





kNN-Approximate Query Performance



Conclusion

- Index Tree
  - Large fan-out decreases the depth of leaf nodes
  - Keeps better similarity at Word-level
  - The signature simplifies the conversion of cardinality
- Spark-ITS: Index Construction
  - Block-sampling and node statistic collection to fast build global index
  - Synchronously build local indices within a partition
  - Constructs Index faster 80+%.
- Spark-ITS: Query
  - Exact Matching: the time decreases by 50%.
  - kNN approximate: the accuracy increases more than 10 fold.

Acknowledge Funding from...

Xianjin Tech Co., Ltd.

Saudi Arabian Cultural Mission

WPI Computer Science Dept.,

NSF CNS: 305258 II-EN

NSF CRI: 0551584