Structural Monitoring and Maintenance, *Vol. 7, No. 4 (2020) 345-365*
DOI: https://doi.org/10.12989/smm.2020.7.4.345

Big data platform for health monitoring systems of multiple bridges

Manya Wang[1,2a], Youliang Ding*[1,2], Chunfeng Wan[1,2b] and Hanwei Zhao[1,2c]

[1]Key Laboratory of C&PC Structures of the Ministry of Education, Southeast University, Nanjing 210096, China
[2]Department of Civil Engineering, Southeast University, Nanjing 210096, China

(Received April 23, 2020, Revised June 5, 2020, Accepted June 9, 2020)

Abstract. At present, many machine leaning and data mining methods are used for analyzing and predicting structural response characteristics. However, the platform that combines big data analysis methods with online and offline analysis modules has not been used in actual projects. This work is dedicated to developing a multifunctional Hadoop-Spark big data platform for bridges to monitor and evaluate the serviceability based on structural health monitoring system. It realizes rapid processing, analysis and storage of collected health monitoring data. The platform contains offline computing and online analysis modules, using Hadoop-Spark environment. Hadoop provides the overall framework and storage subsystem for big data platform, while Spark is used for online computing. Finally, the big data Hadoop-Spark platform computational performance is verified through several actual analysis tasks. Experiments show the Hadoop-Spark big data platform has good fault tolerance, scalability and online analysis performance. It can meet the daily analysis requirements of 5s/time for one bridge and 40s/time for 100 bridges.

Keywords: structural health monitoring; bridge; big data; Hadoop-Spark platform; data processing

1. Introduction

According to the 7th Digital Universe report released by EMC in 2014, it is estimated the total amount of data in the world will reach 44ZB by 2020, and it will exceed to 8.5 billion TB in China. How to use big data methods to fully exploit the value of various data has become a hot topic in current research (García-Valls *et al.* 2018).

Big data management systems are used in many areas (Jagadish *et al.* 2014, García-Valls *et al.* 2018). Mass data were collected to predict the probability of extreme value generation in structural health monitoring (SHM) system (Okasha and Frangopol 2012). Mayo Clinic Healthcare uses a near real-time big data platform to simplify the management work, providing great convenience for daily operation and maintenance (Chen *et al.* 2017). Combined with mobile devices, the operation safety evaluation system for Chinese subway construction personnel was established (Guo *et al.* 2015).

*Corresponding author, Professor, E-mail: civilding@seu.edu.cn
[a] Ph.D. Student, E-mail: mywang_seu@163.com
[b] Professor, E-mail: wan@ seu.edu.cn
[c] Assistant Professor, E-mail: wudizhw_0@126.com

ISSN: 2288-6605 (Print), 2288-6613 (Online)

Since the 1960s, the application of health monitoring systems has received extensive attention and implementation. SHM systems have been installed on many infrastructure buildings and bridges. The evaluation of bridge service state is mainly based on the structural dynamic response such as natural frequency, vibration mode, stiffness and damping. However it is not suitable for local damage identification and service performance degradation assessment. The data collected by SHM can accurately reflect the structural service status (Memmolo *et al.* 2018). The evaluation framework of serving state for bridge structure can be presented based on the RAMS theory (Zhao *et al*. 2018). How to use machine algorithm combined with SHM data to analyze structural characteristics becomes a hot issue (Nick *et al.* 2015). A variety of machine learning algorithms are used in identification of structural damage and component response prediction (Guo *et al.* 2014, Beltempo *et al.* 2015, Nguyen *et al.* 2018). Structural damage can be identified using bridge health monitoring data combined with machine learning methods (Nick *et al.* 2015, Cao *et al.* 2016 and Sayed *et al.* 2017). There are some researches evaluate structural dynamic responses based on the reliability and life cycle theory (Han 2017) and the correlation between structural vehicle induced vibration and vehicle speed (Zhao *et al.* 2016, Ding *et al.* 2016, Ding *et al.* 2017). The genetic algorithm and neural network (Zhao *et al.* 2019) are combined to develop an expert system for bridge damage fuzzy evaluation (Furuta *et al.* 1996). An in-service concrete bridge condition assessment system based on expert system was also studied (Kawamura *et al.* 2003).

Since the vast data collected by SHM system is increasing day by day, while there are various data abnormalities, some machine learning methods have also been introduced into data processing (Ding *et al.* 2018), such as signal denoising filtering (Zhao *et al.* 2015) and clustering method (Ren *et al.* 2004). For the abnormal processing of structural health monitoring data, deep learning LSTM neural network (Liu *et al.* 2020), compressed sensing (Bao *et al.* 2012) and combined Bayesian compressed sensing (Huang *et al.* 2016) are used to recover data in wireless sensor networks. The structural vibration response data for random deletions is compensated by sparse matrix distribution (Yang *et al.* 2016). Neural networks and support vector machines (Huang *et al.* 2010) are very suitable for nonlinear mapping relationships, so they are more dominant in data complements.

SHM system has evolved toward intelligent and digital, while the storage of high-speed railway bridge health monitoring data remains in an original way. The amount of data daily collected by bridge health monitoring sensor can be several GB, while the processing and analysis cannot be completed in time. Because of the disconnection between data storage and analysis, most data analysis works are still limited in traditional way. Most health monitoring systems can only analyze historical data, cannot deal with the real-time monitoring data processing. Although there are some work that builds simple cluster based on Hadoop, and analyzes structural response using big data methods, few of them can be used in operational bridge monitoring systems. This paper proposes a multifunctional Hadoop-Spark big data platform for health monitoring system of Nanjing Dashengguan Bridge.

There are still some main problems in SHM big data analysis:

(1) The amount of data daily collected by SHM system of one high-speed bridge can be several GB, while the data processing and analysis are usually processed once a month. This leads to a delay to obtain effective information which reflects the structural service status.

(2) The sensing system collects multiple monitoring data. There is high-degree of correlation among these data, which can be effectively disclosed by data mining over big-data platform.

(3) Nowadays the storage of high-speed railway bridge health monitoring system data remains in a simple compressed form. This is not conducive to rapid data mining and analysis.

(4) Since most SHM can only analyze historical data, the management and maintenance of

infrastructure are still mainly based on operational inspection, which means the real-time bridge security warnings cannot be performed.

(5) There is no integrated safety performance evaluation system based on bridge health monitoring data, as the data collection, storage, analysis and application modules are separated from each other. The Hadoop-Spark big data platform can bring together all the modules building a systematic and structured bridge safety performance evaluation system.

2. Key points in big data platform for SHM system

In order to solve the problem of analysis delay and data storage in SHM system, big data platform needs several modules which take charge of massive data storage, fast computing, deep computing and online analysis. Currently most big data platforms are built on Hadoop clusters. Hadoop is an Apache open-source software framework written in Java language for distributed storage and distributed processing. It provides solutions for big data processing and analysis (White 2012). The basic computing framework of Hadoop is MapReduce, while it is not suitable for big data platform for SHM system. Therefore, selecting applicable components for computing framework, data storage module, and real-time processing module is very necessary.

2.1 Computing framework selection

Building a big data platform requires suitable framework environment (Zhang *et al.* 2019). MapReduce and Spark are commonly used parallel computing frameworks. Apache Hadoop and Spark platform are used in the advanced traffic management system (Praveen *et al.* 2020). The MapReduce framework can realize many functions including distributed storage, job scheduling, load balancing, fault-tolerant balancing, and processing in parallel programming, helping to achieve batch processing of massive amounts of data. The MapReduce task contains five steps: input, split, map, shuffle, and reduce (Fig. 1). However, it can only supports Map and Reduce operations. The operation principle of distributed system is shown in Fig. 2. After the client submits task job, it is divided into multiple tasks running on several data nodes. Job tacker runs on NameNode and is responsible for coordinating tasks running on different DataNodes based on the feedback information from Task Tracker. After each processing, data sequence must be written into the distributed file system. The exchange of data through HDFS will lead to a lot of IO operations, which may cause the inability of making full use of computer memory. At the same time, MapReduce does not apply to iterative calculations, interactive and streaming processing.

Spark is an open source universal distributed memory computing framework on UC Berkeley AMP Lab. The analysis speed is 10-100 times faster than MapReduce. Spark Core is the core of computing framework, and it has four main modules (Fig. 3): Spark-SQL, Spark-Streaming, MLlib, and GraphX. SparkSQL is a module for processing structured data; Spark-Streaming is used for streaming data analysis; MLlib is a Spark machine learning library, providing machine learning algorithms for models; GraphX is a module for graph calculation and mining. Spark provides Cache mechanism to support multiple iterations of calculation and data sharing. It can reduce the IO overhead of data reading, and store intermediate results in memory as much as possible, which greatly improving computational efficiency.

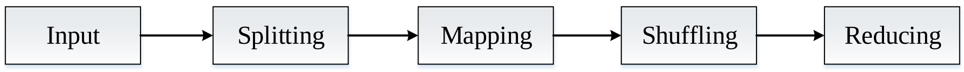


Fig. 1 Flow chart of MapReduce

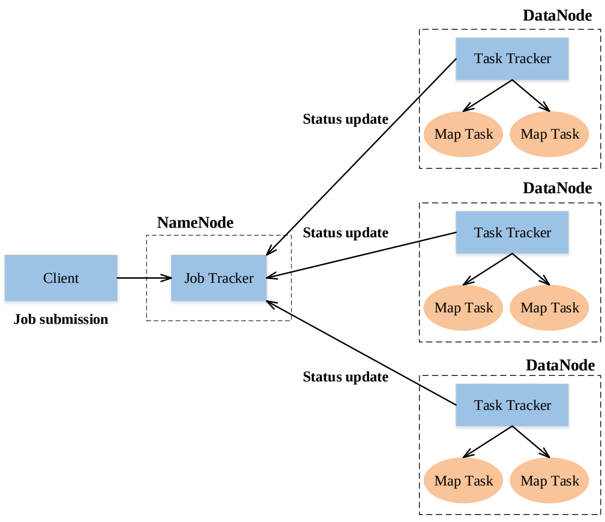


Fig. 2 Distributed system processing flow

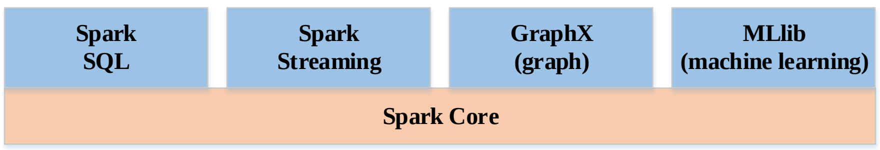


Fig. 3 Main components in Spark

To figure out the calculation difference between Hadoop MapReduce and Spark computing frameworks, we did the Word Count Program test. The task aim is to count the word frequency in 1GB English text file using different computing clusters. We deploy the Hadoop cluster and the Spark cluster separated under the same hardware conditions. The test was conducted for ten rounds, and the time consumption is shown in Fig. 4.

We can conclude that:

1) The average time spent on Hadoop computing framework is seven times than that on Spark platform. Although they both have distributed computing frameworks, Spark has a far better performance in computing than MapReduce in Hadoop environment.

2) Because data is distributed, the location of the data has an impact on task execution time, and the impact on Hadoop framework is more obvious. Therefore Spark is more controllable in computing time than Hadoop when performing large-scale tasks.

3) Unlike the data processing in Hadoop framework, the Spark framework only reads data once. This makes it more suitable for machine learning tasks with multiple iterations.

The computing performance comparison between MapReduce and Spark is listed in Table 1. MapReduce can support high-fault big data batch processing, but due to the framework limitations, intermediate operations require disk IO processing. It is not suitable for iterative computing tasks like machine learning. Spark's memory-based computing makes it more efficient, and it has a large library of machine learning algorithms which is useful for further analytical modeling. The offline data analysis module. In order to achieve a fast analysis of the SHM system, we choose Spark for calculation framework, combined with Hadoop environment.

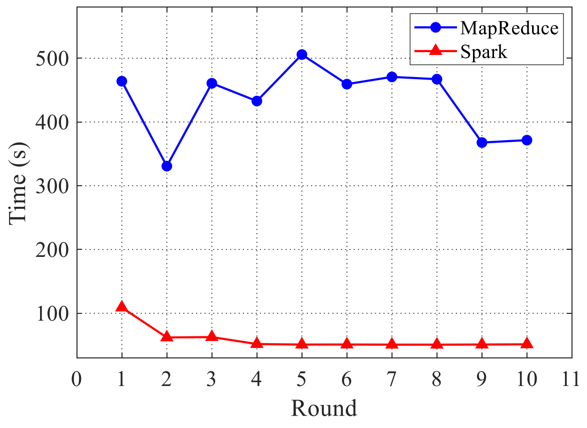


Fig. 4 Word Count task time consuming for 1GB file

Table 1 Computing performance comparison of MapReduce and Spark

	MapReduce	Spark
Advantages	a) Good scalability: The computational performance keeps increasing linearly with the number of compute nodes. b) High fault tolerance: When the calculation node is faulty, the task can be automatically transferred without manual operation.	a) Good calculation performance: Suitable for iterative calculations. b) Provide a large number of APIs, support Java, Scala, Python, R four languages. c) It can be used in HADOOP environment, read and write HDFS/ HBase, and integrate with YARN to realize resource scheduling
Disadvantages	a) Only MAP and REDUCE operations are supported b) Low computational efficiency: Calculation requires a lot of IO operations c) It is not suitable for iterative calculations, interactive processing and streaming	a) High memory consumption: If the amount of processed data is too large, abnormal conditions such as overflow.

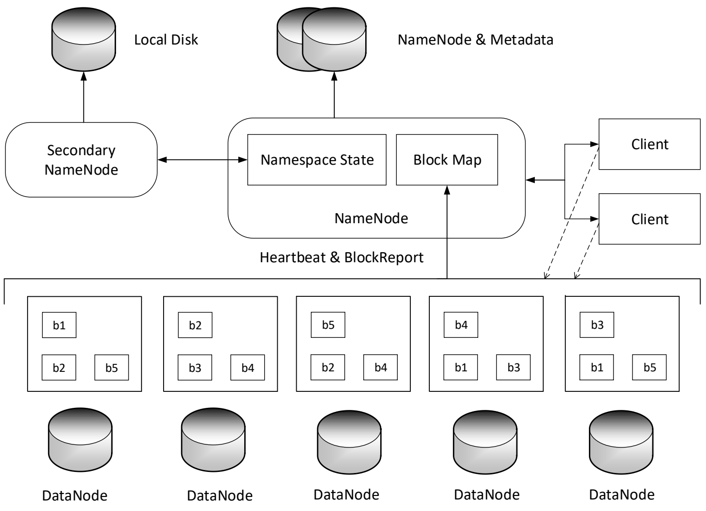


Fig. 5 Schematic diagram of HDFS

2.2 Distributed data storage

In order to store massive data, a big data management platform for SHM systems needs a distributed database and file system.

2.2.1 Distributed file system

GFS (Google File System), HDFS (Hadoop Distributed File System), Ceph and FastDFS are commonly used distributed file systems. HDFS is an open source implementation of GFS and is a core subproject of the Hadoop ecosystem. HDFS uses master/slave architecture for data storage, which composed of four main parts: Client, NameNode, DataNode and Secondary NameNode. Fig. 5 shows the system architecture diagram of HDFS.

Client is a user-oriented terminal, which mainly accomplishes four tasks: 1) file segmentation; 2) interaction with NameNode to get file location information; 3) interaction with DataNode to read or pass in data; 4) Provide commands for HDFS management and access. NameNode is a cluster management, and DataNode executes the commands. The Secondary NameNode is used to assist NameNode to share the workload. If an emergency occurs, it can assist to reply to the NameNode. In HDFS system, monitoring data is stored in blocks, and in case of malfunction it has a multi-copy mechanism. In addition to these basic storage needs, HDFS can also be used for streaming data processing. It can unify online data storage and offline data reading paths on the platform, facilitating the conversion between real-time online processing and offline analysis modules.

In conclusion, HDFS has the following two notable features, which makes it suitable for big data platform:

a) High fault tolerance: As a distributed file system, files on HDFS are shared and stored on several DataNode servers, while each slice of each file in HDFS clusters can be saved in multiple backups (default 3 copies). This specific mechanism guarantees the data storage security.

b) Convenient file access: HDFS provides a unified directory tree to locate files. Client only needs to specify the directory tree path when accessing the file, without getting the specific file physical location.

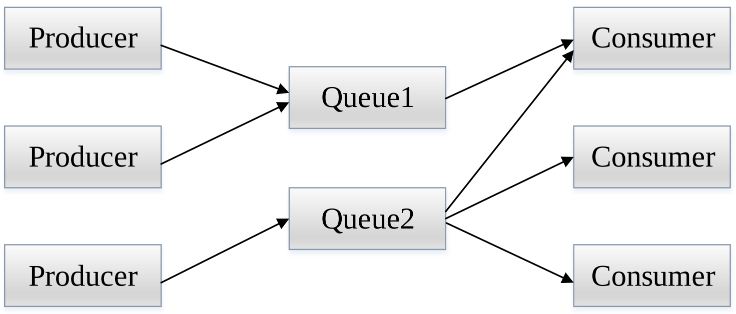


Fig. 6 Information release subscription mode

2.2.2 Distributed database

Distributed database is built on distributed files system providing storage function for massive data sources. Relational databases such as Oracle, MySQL, and Microsoft SQL Server are widely used in various industries. For big data-based SHM system, distributed relational database clustering can be realized through open source middleware. The current widely used database form is NoSQL. Its flexible data model can effectively support Web 2.0 applications with powerful horizontal scalability (O'Connor *et al.* 2016). A typical NoSQL database contains four types: key-value database, column family database, document database, and graphical database. Among them, the column family database HBase is more suitable for SHM massive data storage.

HBase has these characteristics:

1) Large capacity: A table can have hundreds of millions of rows, millions of columns.
2) Column-oriented: It has list-oriented (cluster) storage and access control and column (cluster) independent retrieval.
3) Sparse list: Because empty columns do not take up storage space, tables can be very sparse.
4) No specific mode: Each row has a primary key and any number of columns. Columns can be dynamically added as needed. Different rows in the same table can have distinct columns.
5) Single data type: The data in HBase is a string, there is no specific type.

To sum up, different distributed databases or files can be selected according to different platform storage requirements. Since the data collected by SHM system is structured with a clear naming and storage format, HDFS with high reliability, scalability and fault-tolerance system can meet all the needs. HDFS uses a master/slave architecture for data storage.

2.3 Real-time processing implementation

In order to solve the problem of traditional monitoring data processing lag, it is necessary to implement online analysis on big data platform. The real-time processing module can realize the fast reading and analysis of data. It contains data caching and streaming calculations.

2.3.1 Real-time data caching

The data cache is implemented by publishing the message queue of the subscription mode. The publishing subscription mode is shown in Fig.6. This mode provides flexible control of data. The producer inputs instant data into the queue, then the consumer can flexibly acquire data for reprocessing.

Table 2 Cache performance comparison among Kafka, RabbitMQ and Redis

	Kafka	RabbitMQ	Redis
Characteristics	a) High throughput, low latency: It can process massive messages per second with a latency of at milliseconds; b) Scalability: Kafka clusters support hot expansion; c) Persistence reliability: Messages are persisted to local disk and data backup can prevent data loss; d) Fault tolerance: It allows nodes in clusters to fail; e) High concurrency: It supports thousands of clients to read and write simultaneously.	a) Reliability: It uses transmission confirmation and release confirmation to ensure reliability; b) Flexible routing: Messages rout through Exchange before message enters the queue; c) High availability: queues can be mirrored on machines in the cluster; d) Multiple protocols: It supports multiple message queue protocols, such as STOMP, MQTT.	a) Provide several data types such as list, set, zset, hash; b) Support Master-Slave data backup mode; c) Support data persistence: Cash data can be loaded again when restarting.

Table 3 Real-time processing performance comparison

	Spark-Streaming	Storm
Real-time	Small batch processing: Generate batches according to interval duration. The general processing time is between 0.5s and 2s.	Fully real-time processing, triggering a calculation by entering one data. Its minimum delay is 100ms.
Throughput capacity	Comparatively large: the batch throughput is higher than the calculation of real-time triggering. It uses mobile computing module, not mobile data.	It is slightly worse than Spark-Streaming. It uses mobile data instead of mobile computing module.
Fault tolerance mechanism	Transform logic by storing RDD mode: If the data is calculated incorrectly from the A data set to the B data set, since there is a calculation logic of A to B, it can be recalculated directly from A to generate B.	Acker (ack/fail message confirmation mechanism) confirmation mechanism ensures that a tuple is fully processed

Kafka, RabbitMQ, and Redis components can both cache real-time data, the cache performance differences between them is shown in Table 2. Compared to other components, Kafka is lightweight, high-throughput, low-latency, high-scalability, and long-lasting. At the same time, it can also match Spark-Streaming and Storm, which makes it very suitable for SHM big data platform.

2.3.2 Real-time processing

At present, the main real-time processing modules are Spark-Streaming and Storm. Their real-time processing performance characteristics are shown in Table 3. Storm clusters can only be used for real-time calculations, mainly for pure real-time processing scenarios. Spark-Streaming is a main part of Spark. Spark provides a unified solution for real-time computing on clusters, such as streaming computing, graph computing and machine learning.

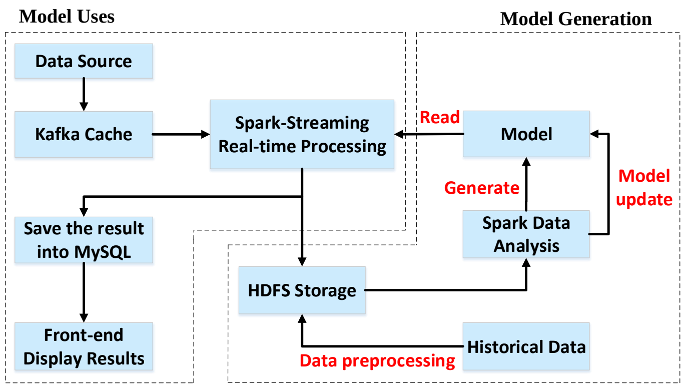


Fig. 7 Framework of big data platform

3. Development of Hadoop-Spark based big data platform

3.1 System design

In order to meet actual application requirements of big data platform, it is necessary to select appropriate functional modules to implement data storage and analysis functions in Hadoop cluster (Landset *et al.* 2015). Multifunctional Hadoop-Spark big data platform for high-speed railway bridges proposed in this paper contains the following components:
(1) Distributed file storage system: HDFS;
(2) Resource scheduling and management system: YARN;
(3) Distributed computing framework: Spark;
(4) Real-time stream computing framework: Spark Streaming;
(5) Distributed coordination service: Zookeeper;
(6) Relational Database: MySQL.

The overall framework of big data platform is shown in Fig.7. It consists of two parts, model generation (offline data analysis) and model uses (online data analysis). The model generation module mainly analyzes the historical data. Spark reads data into memory for analysis and modeling, then generates early warning model, while data are stored in the HDFS distributed file system. The model usage module mainly processes real-time bridge monitoring data and performs early-warning analysis. Real-time data first enters into Kafka cache, and then Spark-Streaming reads data and processes them. At the same time, Spark-Streaming can read the model generated by Spark for bridge warning decision. This framework makes full use of historical data analysis results and combines online and offline modules well.

The overall architecture of big data platform machine is shown in Fig. 8. The machine is divided into three parts: 1) data storage and analysis cluster, 2) Kafka cache cluster and 3) data display cluster. The data storage and analysis cluster mainly installs Hadoop and Spark components. It can realize data storage and analysis functions (Spark-streaming component can realize real-time data stream processing); Kafka cluster implements data caching function; the data display cluster is a WEB daemon for data display.

In terms of data storage module, data collected by health monitoring system of high-speed railway bridge is structured with a clear naming and storage format. Hadoop distributed file system

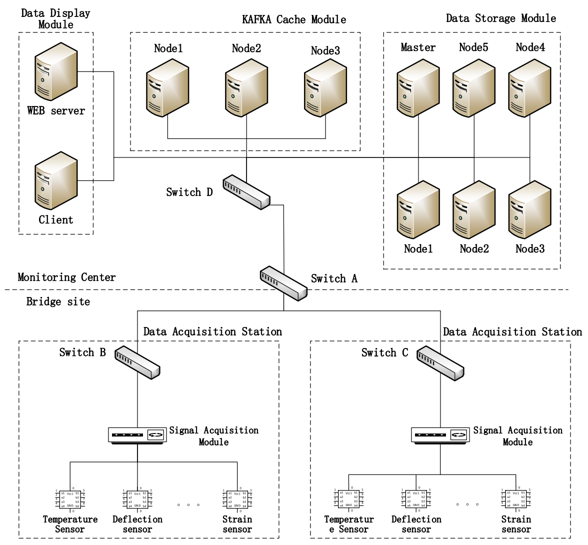


Fig. 8 Schematic diagram of machine architecture

(HDFS) with high reliability, scalability and fault-tolerance system can meet all the storage needs. In HDFS system, monitoring data is stored in blocks, and the multi-copy mechanism is used in case of malfunction. In addition to these basic storage needs, HDFS can also be used for streaming data processing. It can unify the online data storage and offline data reading path on the platform, facilitating the conversion between real-time online processing and offline analysis modules.

3.2 Early warning module design

To handle the big data collected by high-speed railway bridge health monitoring system, a real-time intelligent early warning method based on steaming data is proposed. The Hadoop-Spark big data platform mainly relies on Spark for real-time data analysis, and a circular parallelized Spark task chain is formed (Fig. 9). It consists of four steps: (1) data cache in Kafka; (2) data preprocessing in Spark-Streaming; (3) safety warning: Calculate the predictive model of real-time data and compare it with the stored warning thresholds in Spark-Streaming; (4) Data storage. Store all the efficient cleaned data and calculation results in HDFS. The whole process is continuously cycled in the platform according to set processing interval. It breaks the original sequential analysis and realizes the parallelization of Spark task chain and significantly reduce calculation analysis time.

4. Experimental verification

The big data platform for health monitoring systems has high requirements in terms of safety, stability, scalability, and timeliness of response:

(1) Good scalability to store and analyze massive data collected every day;

(2) Durable stability to provide sufficient storage space to support multi-class data analysis requirements in big data platform;

(3) The simple query response time of big data platform should be in the order of seconds, and

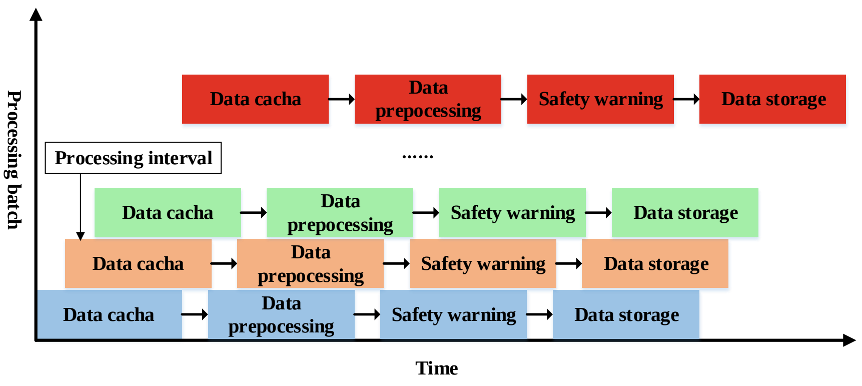


Fig. 9 Real-time data parallel processing pipeline diagram

the average response time of complex data analysis task should be in the order of minutes.

Based on the previous design, while using the monitoring data of Nanjing Dashengguan Bridge, a big data platform is arranged for actual operation management. And the big data platform performance was tested through the processing of actual data.

4.1 Overview of the Dashengguan Bridge

Nanjing Dashengguan Bridge is a six-lane high-speed railway bridge across the Yangtze River. The Shanghai-Chengdu and the Beijing-Shanghai Line are respectively on upstream and downstream sides, while the subway is divided into both sides. The design speed of main line on the bridge is 300km/h, which makes it the highest-speed railway bridge in the world. The main bridge is 1272 m long and adopts (108+ 192+336+336+192+108) m six-span continuous steel truss arch structure. The steel truss arch of the main span is 336 m long and 84 m high, with a ratio of 1/4. The whole bridge has installed 116 sensors on 21 sections, monitoring important structural response indicators including temperature, wind speed and direction, humidity, strain, deformation and train speed. For example, deflection monitoring points are set at 8 sections over the bridge, located in each span and at the 1/4 and 3/4 spans of the main span (Fig. 10).

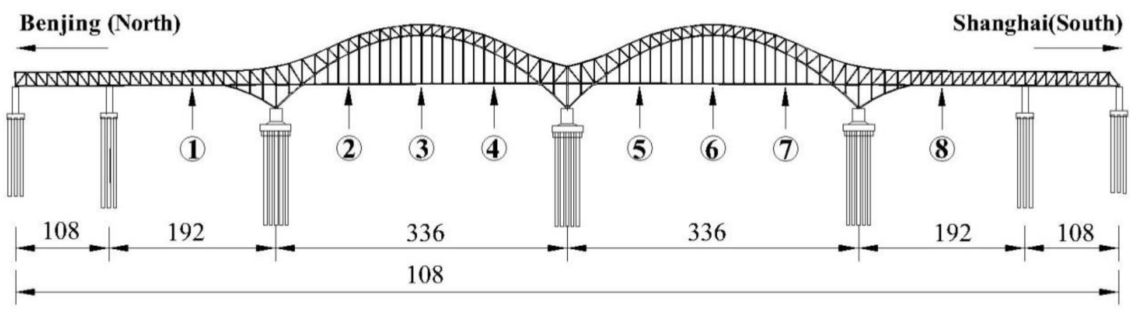
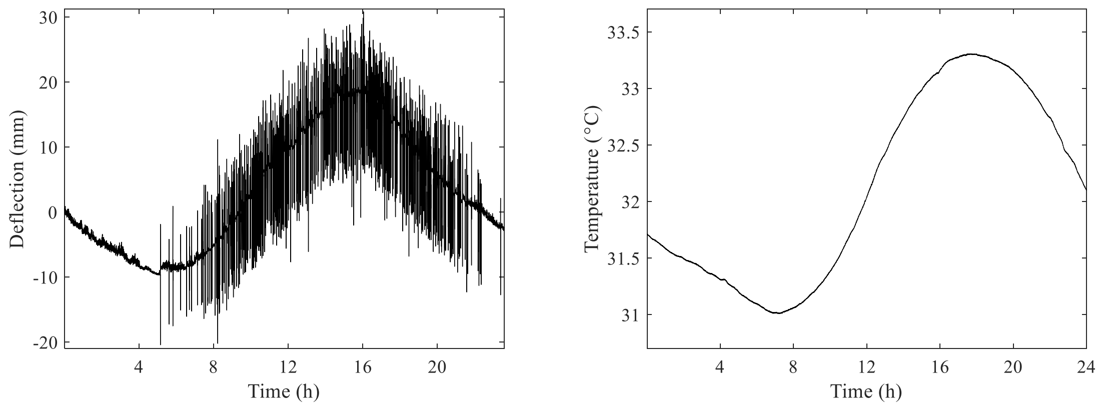


Fig. 10 Dashengguan Bridge elevation and deflection monitoring distribution

Table 4 List of sensors

Monitoring data category	Sensor type	Data to be collected
Environmental load	Temperature and humidity sensor	Temperature and humidity
	Wind speed and direction sensor	Wind speed and direction
Train load	Speedometer	Train speed
	Total station	Deflection
	Strain gage	Strain of the structure
Structural response	Accelerometer	Proper acceleration
	Vibroscope	Vibration amplitude
	Displacement meter	Displacement



(a) Bridge deflection (b) Ambient temperature

Fig. 11 Deflection and ambient temperature diurnal curve

Table 4 shows the contents of high-speed rail bridge health monitoring sensor system. In addition to the basic monitoring projects listed in the table, vehicle and ship collisions, fires, explosions, and earthquakes are also needed to be monitored dealing with extreme conditions.

4.2 Experiment environment

The big data platform for health monitoring systems of multiple bridges is deployed in the lab LAN environment to ensure the quality of network communication. Hadoop shares a device cluster with Spark. The Hadoop cluster is a master node with five storage nodes, while the Spark cluster is a master node with five working nodes.

To test the performance of Hadoop-Spark big data platform, 2.5 million pieces of Dashengguan Bridge monitoring data in 2015 were selected to perform temperature effect analysis and prediction

Table 5 Machine configuration of Hadoop-Spark clusters and Kafka clusters

		Hadoop-Spark clusters	Kafka clusters
Software version	Hadoop 2.6.0-CDH5.7.0, Spark 2.3		Kafka 0.9
Computer	Master node	CPU: Quad-core processor with 2.1GHZ Memory size：16GB Hard disk：500GB Quantity：1 Operating system：Ubuntu 14.04	CPU：Quad-core processor with 2.1GHZ Memory size：2GB Hard disk：500GB Network card：Marvell Yukon 88E8057 PCI-E Gigabit Ethernet Controller Quantity：3 Operating system：Ubuntu 14.04
	Slave node	CPU：Quad-core processor with 2.1GHZ Memory size：8GB Hard disk：500GB Quantity：5 Operating system：Ubuntu 14.04	
Switch	Model：Cisco SF90D-16 Configuration Information：16-Port 10/100 Mbps Ethernet Switch Quantity：1		None

over the whole bridge. Experiments examine the fault tolerance, scalability, online analytical performance, and the differences between big data platforms and stand-alone computing.

Structural temperature effect analysis is an important part of the study of bridge structure response characteristics. The diurnal curve of the mid-span section vertical deflection exhibits distinct sinusoidal characteristic, which is consistent with the trend of environmental temperature. This indicates a strong correlation between environmental temperature and main span vertical deflection (Zhao *et al.* 2019). In addition, the daily variation curve of vertical deflection has intensive fluctuations caused by traffic from 7 to 22 o'clock during high-speed train operation. Regression analysis and early warning analysis can be performed while analyzing the structural temperature effects. Therefore the temperature effect analysis of deflection field of Nanjing Dashengguan Bridge was selected to test the performance of the Hadoop Spark big data platform.

4.3 Offline computing performance comparison: Hadoop-Spark vs. stand-alone Python

The experiment implements the machine learning algorithms on Hadoop-Spark and Python respectively. In order to ensure uniform device configuration in this experiment, Spark adopts the independent cluster mode, while the Python program is executed on master node of Spark cluster. Since the scikit-learn based Python program is not a distributed computing framework, it needs longer computation time. In the decision tree regression test, Spark framework takes about half the

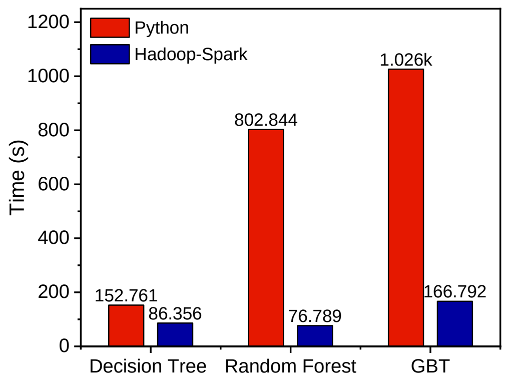


Fig. 12 Average time-consuming comparison of 2.5 million dataset regression tasks

time of Python calculations. While in more complex tasks, random forest and Gradient Boost Tree (GBT) regression, Spark is respectively 10 times and 6 times faster than Python. It can be seen that big data platform has the advantage of fast and stable parallel processing compared with ordinary stand-alone computing. The big data platform gets rid of the stand-alone computing method, and greatly shorten computing time and improved computing efficiency by establishing device clusters. In this way, short-period data processing can be realized and the problem of data delay analysis in health monitoring system can also be solved.

4.4 Fault tolerance experiment

The big data platform of health monitoring systems of multiple bridges requires unified centralized collection, management, analysis and early warning for a number of high-speed railway bridges across the country. The monitoring data collected every day by multiple bridges can be several TB. It means the platform needs to be very stable. Therefore, the fault tolerance performance should be tested. The platform nodes downtime experiment uses temperature response linear regression of measuring points all over the bridge. Fault-tolerant experiments are divided into two types: fault tolerance of computing nodes and management nodes. Then compare two experiences results. If the results are the same whether the platform computing nodes is down or not, it indicates the big data platform is fault-tolerant and can meet daily operation requirements.

During the node fault tolerance experiment sequentially add one computing node into down state in the second, fourth, and seventh rounds. The results show:

(1) Hadoop-Spark big data platform can get the same results during downtime.

(2) The first round of training is time consuming due to computing environment initialization and data preprocessing.

(3) In the second, fourth and seventh rounds, one computing node is sequentially added into down state. Due to the need for data recovery through the replica mechanism and fault-tolerant recovery of Spark computing tasks, the time spent on the same task after downtime is significantly increased.

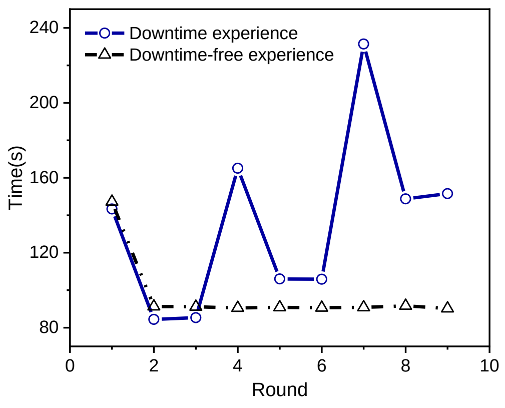


Fig. 13 Fault tolerance experiment

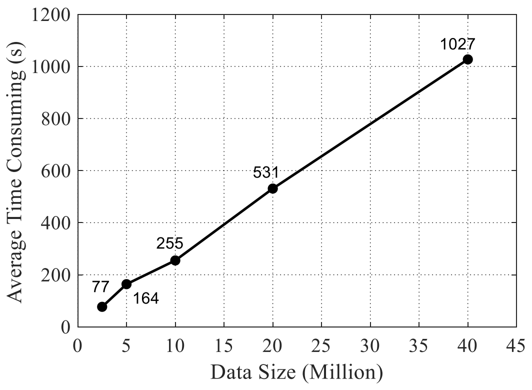


Fig. 14 Average time-consuming of random forest regression tasks for different data sets

(4) As the number of compute nodes decreases, the calculation time increases. It proved that the speed of distributed parallel processing is much faster than single machine processing under the same hardware configuration conditions. In distributed environment, the execution speed of large-scale tasks can be improved by expanding clusters sizes.

4.5 Scalability experiment

The experiment builds a simulated real-time input environment and creates data sets of different sizes based on 6.99 million pieces of health monitoring data from June to October 2015. According to statistics, the annual monitoring data of Dashengguan Bridge does not exceed 40 million, so the experimental data scale is larger than the annual data volume of one single high-speed railway bridge.

Table 6 Correspondence between data volume and storage space

Data Size / Million	2.5	5	10	20	40
Storage Space / GB	0.81	1.62	3.25	6.5	13

In this experiment, data were used to perform random forest regression task on temperature effect analysis of bridge deflection field. There are 9 measuring points in the whole bridge, which means one specification of data in the experiment needs to be calculated 9 times. The average time consumption of different data amounts is shown in Fig. 14.

The result shows that there is a significant positive correlation linear relationship between data size and regression task time. When data size is 40 million, the model training takes about 17 minutes and the deflection prediction task of whole bridge takes 154 minutes, which is acceptable for large-scale offline data analysis of high-speed railway bridges. The relationship between data size x and average time consumption y under current equipment cluster conditions is defined as:

$$y = 0.2519x + 20.375 \tag{1}$$

Its coefficient of determination R2 is 0.9988. With the goal of predicting the whole bridge deflection field within 24 hours, the average consuming time of one-predictive model task is limited to 9600 seconds according to this linear model, while the platform can complete an analysis task for a 3.8 billion data set. The calculation time can shorten by adjusting the startup parameters in Spark, improving or extending hardware configuration of cluster. It shows the Hadoop-Spark big data analysis system has good scalability performance.

Today, the big data frameworks pose new challenges in terms of real-time data storage and processing. Creating a robust platform to serve such infrastructures with minimum hardware or software failures is a key challenge (Chouliaras *et al.* 2019). The scalability of the big data platform is the guarantee for massive data storage in SHM system. It changes the traditional single mode of relying on compressed data and hard disk storage. At the same time, the storage capacity and stability of the big data platform are enhanced by allocating storage tasks through hardware clusters.

4.6 Online performance test

The Hadoop-Spark big data platform real-time analysis tasks include: 1) data source read-in, 2) data pre-processing, 3) reading prediction models, 4) real-time safety warning, 5) warning results and data storage. According to previous big data platform structure design, the pre-processed data is stored into HDFS, and the results of online safety warning are saved to the MySQL database.

(1) Kafka's cache performance

The Kafka cache system is the guarantee of the collected data successfully be read into real-time analysis system. Assuming there are 100 bridges health monitoring systems simultaneously collecting data and transmitting to the Hadoop-Spark platform, every bridge creates one piece of information, than collect the consuming time for Kafka to read in 100 pieces information. The average time of 10 rounds of testing is 0.038 seconds, which is much less than the data acquisition time of 1s. This means the platform can meet the data reading requirement of 100 bridges simultaneously at 1 Hz.

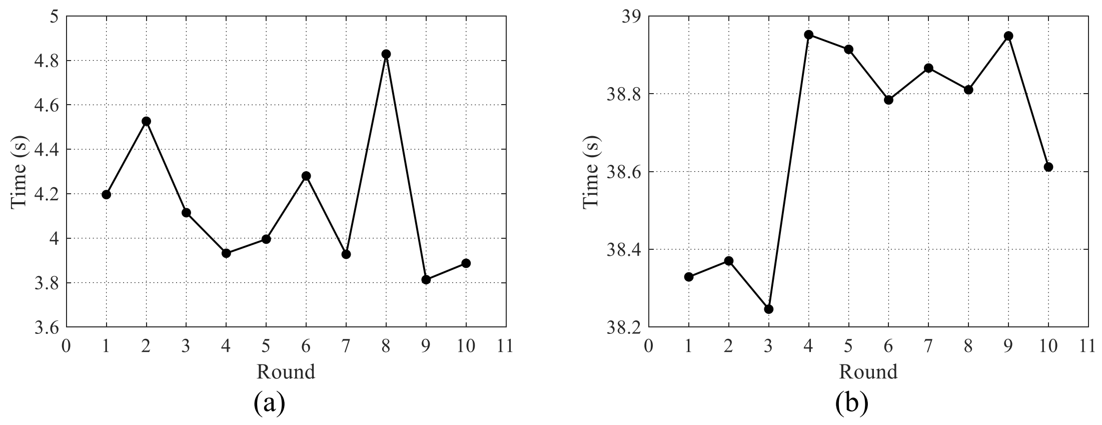


Fig. 15 Spark-Streaming computing performance: (a) One bridge at 5s intervals, (b) 100 bidges at 40s intervals

(2) Real-time analysis performance of Spark-Streaming

In order to test the analysis performance of Spark-Streaming, the simultaneous analysis test of one bridge and 100 bridges was carried out based on the temperature effect analysis of the deflection field. When analyzing one single bridge, the data set containing all monitoring data is generated every second, and the current experimental conditions support microbatch analysis at 5-second intervals. When the analysis target is expanded to 100 bridges, the platform needs to accept information from 100 different data sources simultaneously, it can perform small batch analysis at 40 second intervals. Based on the above two cases, the platform is tested in 10 rounds of real-time analysis. The results is shown in Fig. 15. The average time for real-time analysis of 10 rounds for one bridge is 4.15 seconds, while the average time for 100 bridge is 38.68 seconds. In both experiments, the processing time of real-time data is smaller than the interval of small batch processing. It shows the configuration of Hadoop-Spark big data platform can meet the real-time analysis requirements. Meanwhile some methods can be adopted to shorten the interval time of small batch processing, such as expanding the scale of distributed system and improving the hardware configuration of cluster.

In summary, the performance of the Hadoop-Spark big data platform has been tested and confirmed that it can meet operational conditions. In terms of distributed computing, the computational speeds of two distributed frameworks, Spark and Hadoop MapReduce, are compared. The result shows Spark has obvious advantages compared with Hadoop MapReduce. Then we compare the modeling speed of Spark and Python by using machine learning multiple iteration analysis. Compared to the current popular Python language programs, Spark still has a great advantage in the computing speed. Two levels of experiments have been conducted on the online analysis performance, indicating that Spark Streaming has good micro-batch processing capabilities to meet all the real-time data processing needs based on machine learning methods. Considering all the computing needs, the Spark distributed computing framework is chosen as computing engine. Combined with Spark Streaming for streaming data, the Hadoop-Spark big data platform can meet current offline analysis and real-time analysis requirements.

5 Conclusions

This paper proposes a Hadoop-Spark platform based on Dashengguan Bridge health monitoring system. By combining Hadoop and Spark computing frameworks, it can avoid the problem of low efficiency and inflexible analysis algorithm design when only using Hadoop computing framework. The platform realizes near real-time analysis and off-line analysis of high-speed railway bridge health monitoring system. After comparing advantages and disadvantages of various big-data technologies and methods, HDFS (Hadoop Distributed File System) is selected for data storage and Spark is selected for data modeling when processing off-line data. What's more, Kafka is chosen for data cache and Spark-streaming is chosen for data reading and data processing when dealing with real-time data. Finally, through the analysis of data experiment, the established big-data platform is superior in the off-line computation performance, real-time online performance, expansibility and fault-tolerance ability. It means the big data platform can meet the actual operational needs.

The big data platform solves the main problems in SHM big data analysis. The hardware cluster device architecture meets the data storage requirements and provides a more stable, extensible and convenient data storage structure. And it has more convenient and fast computing power than stand-alone mode, it can realize data analysis in a shorter period and solve the problem of delay analysis caused by massive data accumulation. In the traditional SHM system, the data analysis relies on manual work. It cannot carry on real-time data analysis, and realize timely safety warning work. To solve this problem, the Hadoop-Spark big data platform adopts the online real-time data processing module, which can not only complete the real-time data preprocessing but also carry out the structural security early-warning analysis.

Although the Hadoop-Spark big data platform satisfies the basic computational analysis functions at this stage, it still needs further structural optimization. The existence and efficiency of such a platform is dependent upon the underlying storage and processing engine (Satti, *et al*. 2020). The current preprocessing of bridge health monitoring data still requires manual intervention. More algorithms based on big data can be developed in the future to realize fully automatic identification and cleaning of abnormal data. In addition, the current safety warning function mainly uses the health monitoring sensor system deployed on structure for service performance evaluation and early warning work. As our future work, the unstructured regular inspection transcripts and monitoring information will be added into the Hadoop-Spark big data platform to assist in structure status assessment and build a complete automatic management platform. Furthermore the current safety early warning system of high-speed railway bridge mainly focuses on the structural responses. It doesn't include the bridge usability security alert function. So information such as weather and hydrological forecast will also be added into Hadoop-Spark big data platform, driving safety warnings will be implemented based on these data.

Acknowledgements

The research described in this paper was financially supported by the Distinguished Young Scientists of Jiangsu Province (No. BK20190013) and the National Natural Science Foundation of China (Grants. 51978154 and 52008099).

References

Bao, Y.Q., Li, H., Sun, X.D., Yu, Y. and Ou, J.P. (2012), "Compressive sampling-based data loss recovery for wireless sensor networks used in civil structural health monitoring", *Struct. Health Monit.*, **12**(1), 78-95. https://doi.org/10.1177/1475921712462936.

Beltempo, A., Cappello, C., Zonta, D., Bonelli, A., Bursi, O.S., Costa, C. and Pardatscher, W. (2015), "Structural Health Monitoring of the Colle Isarco Viaduct", *Workshop on Environmental Engergy and Structural Monitoring Systems,* 7-11. https://doi.org/10.1109/EESMS.2015.7175843.

Cao, B.Y., Ding, Y.L., Zhao, H.W. and Song, Y.S. (2016), "Damage identification for high-speed railway truss arch bridge using fuzzy clustering analysis". *Struct. Monit. Maint.*, **3**(4), 315-333. https://doi.org/10.12989/smm.2016.3.4.315.

Chen, D., *et al.* (2017), "Real-time or near real-time persisting daily healthcare data into HDFS and ElasticSearch index inside a big data platform", *IEEE T. Ind. Inform.*, **13**(2), 595-606. https://doi.org/10.1109/TII.2016.2645606.

Chouliaras, S. and Sotiriadis, S. (2019), "Real-Time Anomaly Detection of NoSQL Systems Based on Resource Usage Monitoring", *IEEE T. Ind. Inform.*, **16**(9), 6042-6049, https://doi: 10.1109/TII.2019.2958606.

Ding, Y.L., Wang, C., Zhao, H.W., Yue, Q. and Wu, L.Y. (2016), "Vehicle-bridge resonance analysis of dashengguan bridge based on vibration acceleration monitoring", *J. Railway Eng. Soc.*, **33**(9), 48-54. https://doi.org/10.1061/(ASCE)CF.1943-5509.0000932.

Ding, Y.L., Zhao, H.W., Deng, L., Li, A.Q. and Wang, M.Y. (2017), "Early Warning of Abnormal Train-Induced Vibrations for a Steel-Truss Arch Railway Bridge: Case Study", *J. Bridge Eng.*, **22**(11), 05017011.1-05017011.12. https://doi.org/10.1061/(ASCE)BE.1943-5592.0001143.

Ding, Z.Y., Mei, G., Cuomo, S., Li, Y.X. and Xu, N.X. (2018), "Comparison of estimating missing values in IOT time series data using different interpolation algorithms", *Int. J. Parallel Program.*, **48**(3), 534-548. https://doi.org/10.1007/s10766-018-0595-5.

Furuta, H., He, J.H. and Watanabe, E. (1996), "A Fuzzy Expert System for Damage Assessment Using Genetic Algorithms and Neural Networks", *Comput. - Aided Civil Infrastruct. Eng.*, **11**(1), 37-45. https://doi.org/10.1111/j.1467-8667.1996.tb00307.x.

García-Valls, M., Abhishek, D. and Vicent, B. (2018), "Introducing the new paradigm of social dispersed computing: applications, technologies and challenges", *Journal of Systems Architecture,* **91**, 83-102. https://doi.org/10.1016/j.sysarc.2018.05.007.

García-Valls, M., Calva-Urrego, C. and García-Fornes, A. (2018), "Accelerating smart eHealth services execution at the fog computing infrastructure", *Future Generation Computer Systems*, S0167739X17327425. https://doi.org/10.1016/j.future.2018.07.001.

Guo, J.Q., Xie, X.B., Bie, R.F. and Sun, L.M. (2014), "Structural health monitoring by using a sparse coding-based deep learning algorithm with wireless sensor networks", *Personal & Ubiquitous Comput.*, **18**(8), 1977-1987. https://doi.org/10.1007/s00779-014-0800-5.

Guo, S., Luo, H. and Yong, L. (2015), "A big data-based workers behavior observation in china metro construction", *Procedia Eng.*, **123**, 190-197. https://doi.org/10.1016/j.proeng.2015.10.077.

Han, S.H. (2017), "Optimal safety valuation of high-speed railway bridges based on reliability assessment and life-cycle cost concept", *Int. J. Steel Struct.*, **17**(1), 339-349. https://doi.org/10.1007/s13296-014-0165-7.

Huang, Y., Beck, J.L., Wu, S. and Li, H. (2016), "Bayesian compressive sensing for approximately sparse signals and application to structural health monitoring signals for data loss recovery", *Probabilist. Eng. Mech.*, **46**, 62-79. http://dx.doi.org/10.1016/j.probengmech.2016.08.001.

Huang, Y.W. *et al.* (2010), "Lost strain data reconstruction based on least squares support vector machine", *Measurement Control Technol.*, **29**, 8-12. http:// doi.org/10.2991/icacsei.2013.159.

Jagadish, H.V., Gehrke, J., Labrinidis, A. and Papakonstantinou, Y. (2014), "Big data and its technical challenges", *Communications of the ACM*, **57**(7), 86-94. https://doi.org/10.1145/2611567.

Jeong, S., Zhang, Y.L., O'Connor, S., Lynch, J.P., Sohn, H. and Law, K.H. (2016), "A NoSQL data management infrastructure for bridge monitoring", *Smart Struct. Syst.*, **17**(4), 669-690. http://doi.org/10.12989/sss.2016.17.4.669.

Kawamura, K., Miyamoto, A., Frangopol, D. M. and Kimura, R. (2003), "Performance Evaluation of Concrete Slab of Existing Bridges Using Neural Networks", *Eng. Struct.*, **25**(12), 1455-1477. https://doi.org/10.1016/S0141-0296(03)00112-3.

Landset, S., Khoshgoftaar, T.M., Richter, A.N. and Hasanin, T. (2015), "A survey of open source tools for machine learning with big data in the Hadoop ecosystem", *J. Big Data*, **2**(1), 1-36. http://doi.org/10.1186/s40537-015-0032-1.

Liu, H., Ding, Y.L., Zhao, H.W, Wang, M.Y. and Geng, F.F. (2020), "Deep learning-based recovery method for missing structural temperature data using LSTM network", *Struct. Monit. Maint.*, **7**(2), 109-124. https://doi.org/10.12989/smm.2020.7.2.109.

Memmolo, V., Pasquino, N. and Ricci, F. (2018), "Experimental characterization of a damage detection and localization system for composite structures", *Measurement*, **129**, 381-388. https://doi.org/10.1016/j.measurement.2018.07.032.

Nguyen, C.U., Huynh, T.C. and Kim, J.T. (2018), "Vibration- based damage detection in wind turbine towers using artificial neural networks", *Struct. Monit. Maint.*, **5**(4), 507-519. https://doi.org/10.12989/smm.2018.5.4.507.

Nick, W., Asamene, K., Bullock, G., Esterline, A. and Sundaresan, M. (2015), "A study of machine learning techniques for detecting and classifying structural damage", *Int. J. Machine Learning Comput.*, **5**(4), 313-318. https://doi.org/10.7763/IJMLC.2015.V5.526.

Okasha, N.M. and Frangopol, D.M. (2012), "Integration of structural health monitoring in a system performance based life-cycle bridge management framework", *Struct. Infrastruct. Eng.*, **8**(11), 999-1016. https://doi.org/10.1080/15732479.2010.485726.

Praveen, D.S. and Raj, D.P. (2020), "Smart traffic management system in metropolitan cities", *J. Ambient Intelligence and Humanized Comput.*, 1-13. https://doi.org/10.1007/s12652-020-02453-6.

Ren, D.M., Rahal, I. and Perrizo, W. (2004), "A vertical outlier detection algorithm with clusters as by-product", *International Conference on Tools with Artificial Intelligence*, (2004), 22-29. https://doi.org/10.1109/ICTAI.2004.22.

Satti, Fahad Ahmed, *et al.* (2020), "Ubiquitous Health Profile (UHPr): a big data curation platform for supporting health data interoperability", *Computing*, 1-36. https://doi.org/10.1007/s00607-020-00837-2 .

Sayed, M.A., Kaloop, M.R., Kim, E. and Kim, D. (2017), "Assessment of acceleration responses of a railway bridge using wavelet analysis", *KSCE J. Civil Eng.*, **21**(5), 1844-1853. https://doi.org/10.1007/s12205-016-1762-0.

White, T. (2012), "Hadoop: the definitive guide", *O'Reilly Media Inc. Gravenstein Highway North*, **215**(11), 1-4. http://dx.doi.org/10.9774/GLEAF.978-1-909493-38-4_2.

Yang, Y. and Nagarajaiah, S. (2016), "Harnessing data structure for recovery of randomly missing structural vibration responses time history: sparse representation versus low-rank structure", *Mech. Syst. Signal Pr.*, **74**, 165-182. http://dx.doi.org/10.1016/j.ymssp.2015.11.009.

Zhang, W.J. and Huang, Y.P. (2019), "Using big data computing framework and parallelized PSO algorithm to construct the reservoir dispatching rule optimization", *Soft Computing*, **24**(11), 8113-8124. http://doi.org/10.1007/s00500-019-04188-9.

Zhao, H.W., Ding, Y.L., An, Y.H. and Li, A.Q. (2016), "Transverse dynamic mechanical behavior of hangers in the rigid tied-arch bridge under train loads", *J. Perform. Constr. Fac.*, **31**(1), 04016072. https://doi.org/10.1061/(ASCE)CF.1943-5509.0000932.

Zhao, H.W., Ding, Y.L., Geng, F.F. and Li, A.Q. (2018), "RAMS evaluation for a steel-truss arch high-speed railway bridge based on SHM system", *Struct. Monit. Maint.*, **5**(1), 79-92. https://doi.org/10.12989/smm.2018.5.1.079.

Zhao, H.W., Ding, Y.L., Li, A.Q., Ren, Z.Z. and Yang, K. (2019), "Live-load strain evaluation of the prestressed concrete box-girder bridge using deep learning and clustering", *Struct. Health Monit.*, 147592171987563. https://doi.org/10.1177/1475921719875630.

Zhao, H.W., Ding, Y.L., Nagarajaiah, S. and Li, A.Q. (2019), "Longitudinal displacement behavior and girder end reliability of a jointless steel-truss arch railway bridge during operation", *Appl. Sci.*, **9**(11), 2222. http://doi.org/10.3390/app9112222.

Zhao, L. and Yin, A.J. (2015), "High-order partial differential equation de-noising method for vibration signal", *Math. Method. Appl. Sci.*, **38**(5), 937-947. https://doi.org/10.1002/mma.3119.

TY