International Journal of Grid and Distributed Computing
Vol. 9, No. 11 (2016), pp.95-110
http://dx.doi.org/10.14257/ijgdc.2016.9.11.08

Research on Hybrid Distributed Computing System Based on Embedded System

Li Zhe[1], Mu Dejun[1], Zhang Tianfan[1] and Guo Lantian[1]

[1.] *Department of Automatic Control, Northwestern Polytechnical University, Xi'an 710072, China*
lizhe_hbeu@vip.163.com, alitasoft@hotmail.com

Abstract

With the rapid development of embedded system and Internet of Things technology, embedded system and smart device based on embedded system is collecting huge amounts of data, and corresponding data processing and application method have been greatly changed, different from the traditional big data and cloud computing focus , local processing and application become important trend. This paper tries to take Cortex-A7 as the main system with the stronger ability , Hadoop distributed computing system is deployed in embedded system and could meet the demand of the future data process with directly managing the resource-constrained sensors, embedded systems and smart devices . Successfully deploying over 20GB data in the test, the system is verified that it can complete most of the functions of data processing cluster , and can also manage the collected sensors and embedded system terminal ,with better research and market promotional value.

Keywords*: Embedded systems; Hadoop clusters; Distributed system; Parallel computing*

1. Introduction

According to IBM 90% of the world's data is generated in less than two years. This reflects the development of big data, take Google for example, it has handled more than 20PB of data every day **Error! Reference source not found.**. These huge amounts of data usually consist of business transaction data, social network data and database processing **Error! Reference source not found.**. Facing the challenges of big data, we need more data bandwidth and powerful data processing system. Typical super computer is very expensive and has high consumption, for example the upgrading project "Titan" from United States Oak Ridge National Laboratory's is at a cost of 90 million dollars and has 7 million Watts consumption **Error! Reference source not found.**. Simply the method of hardware investment is unable to meet the requirements of big data processing. Different from the traditional big data and cloud computing focus, local processing and application become the important trend. And the computing power of the latest embedded systems reaches the 5TFLOPS/W, but the unit costs less than $10. More and more enterprises will shift from high-end server to low-end hardware platform for large scale cluster **Error! Reference source not found.**.

This paper tries to take Cortex-A7 as the main system with the stronger ability, Hadoop distributed computing system is deployed in embedded system and could meet the demand of the future data process with directly managing the resource-constrained sensors, embedded systems and smart devices . Successfully deploying over 20GB data in the test, the system is verified that it can complete most of the functions of data processing cluster , and can also manage the collected sensors and embedded system terminal ,with better research and market promotional value.

ISSN: 2005-4262 IJGDC

2. Embedded Systems are Profoundly Changing the Big Data World

With the rapid development of embedded system and technologies such as Internet, data collection, transmission and analysis of patterns are undergoing profound changes and its application models are also changing, resulting in big data opportunity for embedded systems firms **Error! Reference source not found.**.

Bigdata is a concept which deals with storing, accessing and analyzing large dataset. This can be applied to any field where data is larger or easier to change structures. The general impression in big data is difficult to imagine the huge amount of data generation, and it includes text messages, temperature sensor data and video obtained from surveillance cameras and all kinds of information. A recent example of this as reported by the BBC is the Amsterdam fire department using the business intelligence (BI) information to provide Amsterdam with more accurate risk assessment to ensure that the fire-fighting force is able to be deployed in the right place.

Whether it is a home automation system of intelligent vehicle management system, embedded systems are exerting profound influence **Error! Reference source not found.**.In home automation system, collect user's home appliances access data for few months using various embedded devices. Device sends this data to cloud, where you apply data analytics, machine learning *etc* to predict customer's actions according to per user's past behavior. But in the smart vehicle management, embedded devices in your car send the vehicle information is to the cloud. These data is analyzed and you automatically get alert regarding servicing, improvements *etc* and these making cars smarter.

About IoT (Internet of Things), it's all about Bigdata analytics along with embedded systems. For example, a shopping mall which has hundreds of shops those sell many kinds of goods. The big sensor network can be deployed to cover the entire Mall to support customer gathering process automation. These sensors will start to record the customer's shopping tracks and also record that which product began to be sold and which product has never been sold. Malls of this size may be deployed more than 2,000 sensors, they record data in every second and some of the sensors shoot video and record images, these tiny data gathering becomes a very large data collection and call it the Bigdata. Then these data is stored in its local memory and every ten or more minutes these data will be transferred to the data center or cloud. And the mall through the data center or cloud center to use big data analysis tools and develop strategies to sell unsolvable products and improve product margins. Therefore business decision is based on the collection and processing of personal data by these sensors. Most of these sensors are based on embedded system and it constitutes a complex relationship developing between the bigdata and creating the IoT, and it is inseparable from the cloud and the bigdata.

Aggregate data analysis from embedded devices and data analysis system based on embedded system are changing the world of data analysis **Error! Reference source not found.**. IBM MessageSight chief architect Andrew Schofield dedicated to promoting BI points out that a new turning point——BI technology allows embedded technology less manual intervention and to enhance the efficiency of their business processes and gain competitive advantage. The case of M2M, general practice is to include all data collected and displayed on a large dashboard and use global angle of view for centralized management. This seems to be the best way, but it is not the most intelligent way of application information. Equipments can find each other, interconnect, share information, and can make local decision. Business systems use this information to support intelligent applications and cross data analysis and did not necessarily need to connect to the data center or the other nodes. Because it can be directly in embedded cluster and the part node to obtain the necessary information, and its dynamic structure can be adjusted according to the different needs for their own.

At TED&CIT talk Dr.John Barrett from the Cork Institute introduced the expectation and growth of embedded technologies , by 2032,for the ordinary people in their daily lives, the contact smart things reach to a staggering 3000~5000, this highly frequent contact will change the existing marketing and also change existing data processing model.

Database puts Big Data on Mobile and Embedded Devices **Error! Reference source not found.** . ITTIA announces the big data capabilities of ITTIA DB SQL, a modern database for smart embedded systems with limited local resources. With this unique technology, a large data set can be distributed across a wide array of devices, with the potential to store millions of rows per device. Applications can run queries on individual devices or pool data together in a back-end system for data warehousing and data mining. In this way, developers benefit from highly reliable embedded database software with a strong track record in mission-critical systems.

These embedded ecosystems are generally composed of many devices that divide up a big data problem. Some of these data are private data for the device holder, while other data is shared with select peers. ITTIA DB SQL uses replication and synchronization to efficiently distribute local data with other devices and back-end databases according to the application's business logic and policies. Any number of peers can participate in replication, allowing an ecosystem to scale up as more devices are added. By providing cross platform data management method, the system can be compatible with all kinds of platform and processor architecture, which can support MIPS, PowerPC, ARM, x86, and so on.

3. Design of Hadoop Distributed Computing based on Embedded System

3.1. Hybrid Distributed Data Processor System Based on Embedded System

Consideration local data process and system redundancy, design of hybrid distributed data processor system based on embedded system, Figure 1:

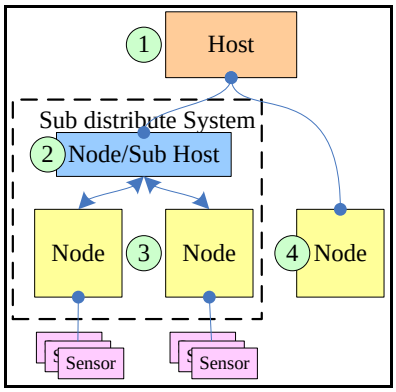


Figure 1. Hybrid Distributed Data Processor System based on Embedded System Concept Map

The system consists of four components: host, sub host, sensor node and normal compute nodes. Their composition based on the C/S model of distributed structure on the macroscopic structure, but it allows multiple nodes to build a processing system, even the node itself is an independent system.

As described in chapter2, a node may manage multiple sensors, and an area that may contain multiple nodes and within the region for data processing can reduce the data transmission process can also improve the efficiency of regional——this hybrid model to subvert the common structure of distributed data processing system, because the data are

no longer unilaterally from the host to each compute node, compute node can also be its own sensors or other subsystems for the data flow to the host, to more fully develop the system of each node's capability, improving the adaptability of the system.

3.2. Research on Parallel Computing System based on Embedded

Our team research on embedded systems starting from 2011. Early research was focused on multi-cascade control based on embedded system board for industrial applications, before gradually turning research on distributed systems and parallel computing, Figure 2.

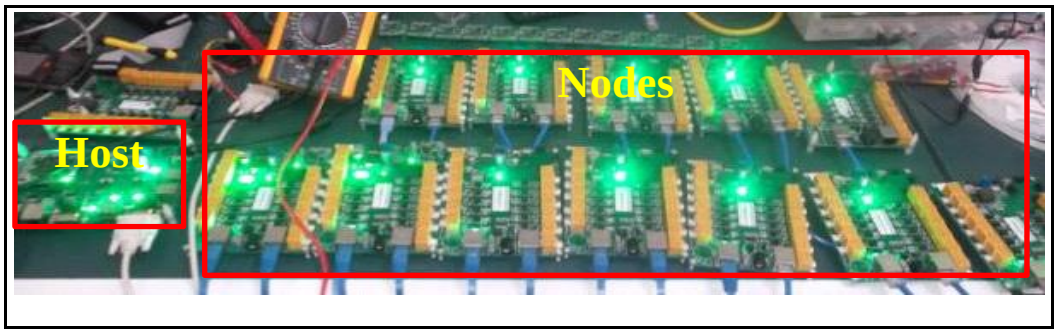


Figure 2. Have 1 Host and 12 Node Cascade Control System

In Figure 2 is the prototype, which is comprised of 1 host and 12 nodes in a cascading manner and each node has the 12 outputs and corresponding sensor. Using the Cortex-M3 architecture of embedded systems and type of MCU is STM32F103ZET6 and STM32F103C8T6. The host is responsible for global management and system status monitoring, other machine deal with specific instructions and sends information collected by sensors to host.

Early use two serial port as input and relay output port cascade, and use dual-NIC in a subsequent version to replace the old system. Theoretically access 5,000 nodes and can cover a very wide range of areas. Figure 3.

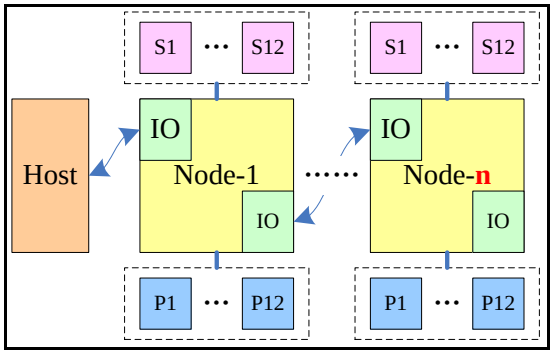


Figure 3. Cascade Mode

Cascade model shortcoming is also very obvious: high latency, restricted communication bandwidth, the relay node failure will invalidate all subsequent nodes. In use ethernet network through a switch, the most of these shortcomings can be eliminated. Figure 4:

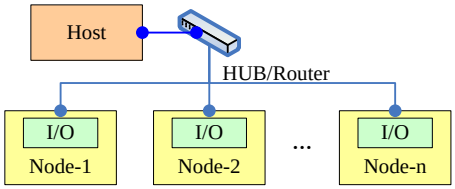


Figure 4. Star Topology

The fact that hosts and nodes are connected by HUB/Router can significantly reduce delay and improve the system reliability. And multilevel HUB cascading can enhance coverage, but this mode requires additional equipment

In the case of setting one host, it can connect 20 nodes with independent control of 255 devices/sensors. System total computing capacity can reach up to 1890DMIPS **Error! Reference source not found.**, the core consumption only needs 10W and just costs $300. This could meet the requirement of real-time control and deal with more complex tasks. In order to further tap the potential of embedded system, the "micro-parallel computing systems" based on Cortex-M3 is focused on research and building .Each node size is only 5*5cm.

The distributed memory MIMD architecture unit is consisted of five MCU, and each Processor has 64MB SRAM memory. The host and other machine are connected by SPI, FSMC bus and memories are connected through platform – specific. On the basis of the system a study on parallel image processing algorithms **Error! Reference source not found.** is made, and then the research of its scheduling of parallel processing system and operation mechanism **Error! Reference source not found.**. Multiple "micro-parallel computing systems" over a network connection which consists of clusters can greatly enhance the overall computing power. Prototype system can contain 255 nodes and increase computing power to 22950DMIPS (theoretical value).

But limited by the Cortex-M3 architecture itself and MCU internal resource constraints, it is difficult to stand the runtime environment required to support large data processing such as Hadoop, and system-wide technology development to bear independent labor costs. So it has not yet conduct in-depth research in this area, turning to the more processing power for embedded architectures and systems. Then type Cortex-A7/A8 high performance architecture for embedded system is determined in the study and records company A10/A20 systems is selected as the basis of embedded data systems, the project named "Medusa", Figure 5:

Figure 5. A20 Embedded Systems based on Cortex-A8

A10/A20 frequency 1.2GHz, A20 has two processors, graphics processor for ARM Mali-400, 1GB DDR3 memory, onboard 8GB Flash and supports SATA external storage, can support large data systems such as the Hadoop runtime environment required.

3.3. The Hadoop Distributed Computing System based on Embedded System Design

Google is undoubtedly the great pioneer in the field of data, which has published the Google File System (GFS) **Error! Reference source not found.**, MapReduce**Error! Reference source not found.**, Bigtable**Error! Reference source not found.** the three papers by researchers and the industry as Google three classic papers. MapReduce now have released 2004 on behalf of the big data, and Google did not stop with MapReduce, subsequently announced a Percolator, Pregel, Dremel, Spanner and many articles, and become a cornerstone of the treatment and research of large numbers. Bigtable inspired the birth of millions of NoSQL databases, such as Cassandra and HBase. Google has produced huge of effect on that world , today this effect not only dose not be reduced, instead of development of the new theory, and new algorithm, and new schema and application environment , except for Google's constantly development and perfection itself , many other company and institutions also followed its footsteps to develop many excellent products, of which Apache Foundation by development of distributed system schema Hadoop**Error! Reference source not found.** is an excellent representation.

HDFS is the core of Hadoop and our prototype system it is based on the Linux operating system, using master-slave structure. Typically, HDFS has a Master node and Slave nodes. NameNode process is on the Master node, maintaining information for the whole cluster, DataNode in the Slave node process, is responsible for data storage. In the HDFS, it forms the local storage space on all nodes into a distributed file system, providing a global view of the storage nodes graphs. HDFS split a large file into number of pieces of each size of 64MB, distributed among the different nodes, Figure 6:

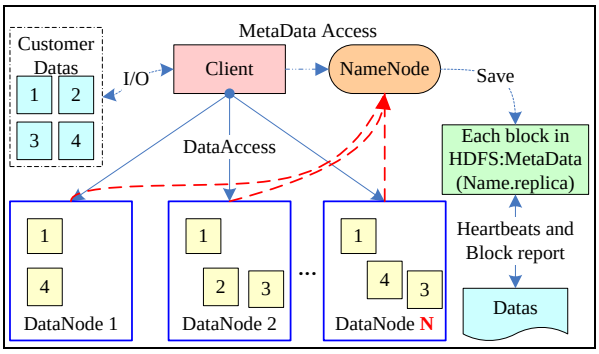


Figure 6. HDFS Distributed Storage Framework

Based on HDFS and studied in this paper, the embedded system platform also uses the master-slave structure, Figure 7:

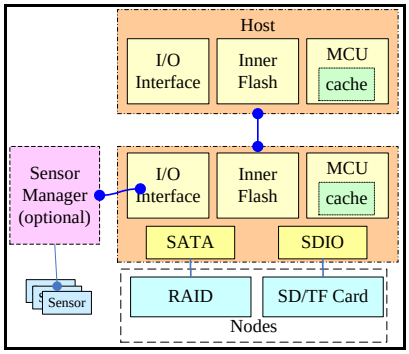


Figure 7. Embedded Parallel Computing System Architecture

Master node is responsible for management of the system; monitoring the status of Slave nodes, and assigning used ranges of memory for each Slave node it manages. Global general configuration, a master node, the node's stable situation will severely affect the performance of the entire system-once the master node outage, causing the entire system to a halt. In practice, often setting up multiple copies of master node, public online hot-spare is used to improve the fault tolerance of the system. Data storage nodes Slave nodes, usually also maintain a local data index table. System can add Slave nodes to implement system level extension.

In the framework of Master-Slave, master node has been in a listening state, avoiding direct communication between the Slave nodes in order to reduce the communication cost. In the process of running, slave nodes constantly reports current health status and load conditions to the master node, when a node is down or overloaded, it will be dispatched by the master node, and the later will share data with other nodes of this node, or through the way of adding a new node to adjust the Key-value storage system Bigtable, HBase is a typical master-slave structure.

4. Deploying Hadoop on Embedded System

4.1. The Basic Structure of the System

Lab environment uses Cortex-A7 for embedded parallel computing system architecture Hummingbird (Hummingbird) A20 system as a platform for implementation, the system has 2GB inner memory, 2GB main memory, nodes are connected through a Gigabit network switch; by extending the Flash memory each node reaches HDFS storage system, the original design of the system for 1 host 48 from machine to form a cluster, two clusters to form a cluster. In the pilot phase 31 nodes consist of Hadoop cluster hardware, Figure 8:

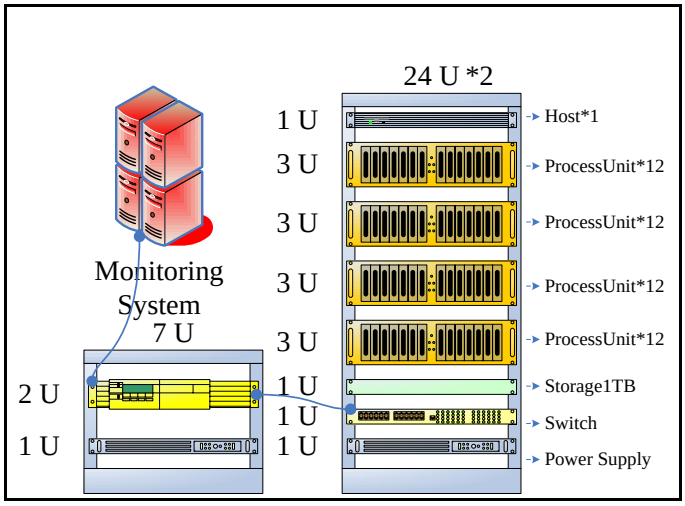


Figure 8. "Medusa "System Frame Diagram

4.2. Deployment of the Operating System

Differences of embedded system platform, experiments use lightweight Lubuntu OS, Lubuntu is small, fast, low power consumption, low demands for hardware resources, and supports both X86 and ARM architectures. Use TF card to start HummingBird and Lubuntu.

First, create an Ubuntu VMware system, the Lubuntu.CardImage file is copied to the system. Select a formatted TF card, access Ubuntu system, then *fdisk-l* command to view

and check the mount TF card information, you can use the *dd* command to burn and write TF card if it is mounted correctly:

:~/Desktop$ dd if = Lubuntu.CardImage of=/dev/sdb bs=1M

After about 15 minutes, if there are similar to 3904897024bytes (3.9GB) copied indicates that programming is completed, you need to try these steps to complete the download. After the completion of the TF card inserted into the HummingBird, electric starter on, enter your user name and password to log on the system (*linaro* is the default user name and password).

4.3. Node Configuration

Experimental deployment of Hadoop clusters composed of 31 nodes, 1 Master NameNode, 31 Slave as DataNode. All nodes based on Lubuntu systems via a LAN connection between nodes, and data exchange, address configuration as shown in Table1:

Table 1. Hadoop Cluster Address Configuration Table

Operation System	Node Name	IP Address
Lubuntu	Master.Hadoop	192.168.1.101
Lubuntu	Salve1.Hadoop	192.168.1.102
…	⋮	⋮
Lubuntu	SalveN.Hadoop	192.168.1. N

4.4. Configured SSH without Password Authentication

SSH Secure Shell "Secure Shell Protocol" is created based on the transport layer and application layer, designed for remote login sessions and other Web services protocols that provide security. Configured through SSH without password authentication, Hadoop nodes in the cluster can be achieved with no password to log in.

In the Master node using the *ssh-keygen* command to generate a password less key pair:

ssh-keygen –t rsa –P ''

To append the *id_rsa.pub authorized_keys* authorized:

cat ~/.ssh/id_rsa.pub >> ~/.ssh/authuorized_keys

Modifying the SSH configuration file:

:~$ vi /etc/ssh/sshd_config

Set up after the restart the SSH service, for settings to take effect:

/etc/init.d/ssh restart

Local configuration is complete, and then copies the public key to Slave1.Hadoop and Slave2.Hadoop machines; use the *SSH-copy-ID* command to transfer the public key to the remote host:

ssh-copy-id root@Slave1.Hadoop

4.5. Install JDK

Hadoop is based on Java, and therefore need to install JDK on embedded platforms. Note that is supported by the JDK is not commonly used for embedded systems; you need to use the JDK version for ARM platform amendment, *JDK-8-EA-b36e-Linux-arm-hflt-29_nov_2012.tar.gz* is used here, unzip the tar command and configuration-related environment variables.

4.6. Deployment of Hadoop

Complete Hadoop-based preparations after deployment. Experimental use of Hadoop version is *Hadoop-1.1.2.tar.gz*. Copy it to the */usr* directory, configuration commands are as follows:

tar –xzvf hadoop-1.1.2.tar.gz

mv hadoop-1.1.2 hadoop

In the "*/usr/Hadoop*" below create a *tmp* folder, and add the Hadoop installation path in the environment variable.

mkdir /usr/hadoop/tmp

Modify */etc/profile*, added to the environment variable path of Hadoop, the command is:

export HADOOP_HOME=/usr/hadoop

export PATH=$PATH:$HADOOP_HOME/bin

Edit */usr/Hadoop/conf/Hadoop-env.sh* to add the absolute address of the JDK into the configuration file.

Edit */usr/Hadoop/conf/core-site.XML* configuration system of HDFS name, address, and port number.

Edit */usr/Hadoop/conf/hdfs-site.XML* configuration backup of HDFS, default is 3.

Edit */usr/Hadoop/conf/mapred-site.XML* configure a job to monitor the address and port

Modify the Hadoop MapReduce in the configuration file; configure the Job Tracker address and port.

Edit /usr/Hadoop/conf/masters file to configure the primary node address

Edit /usr/Hadoop/conf/slaves file from the node address

Since then need to configure more than one file:

1) configuration Hadoop-env.sh

The JDK absolute address is added to the configuration file:

:/usr/hadoop$ vi /usr/hadoop/conf/hadoop-env.sh

export JAVA_HOME=/usr/hadoop/conf

2) configuration core-site.XML file

Modify the Hadoop core configuration file core-site. XML, HDFS master is configured here (the NameNode) address and port number

3) Configure the hdfs-site.XML file

Modify the Hadoop HDFS configuration, configure backup defaults to 3.

4) configuration mapred-site.XML file

Modify the Hadoop MapReduce in the configuration file; configure the Job Tracker address and port.

5) configure masters files

6) configure slaves file (Master host-specific)

After completing the steps above, Hadoop is configured on the Master node, and then configured Hadoop configuration information is sent to the Slave1.Hadoop and Slave2.Hadoop nodes:

:~$ scp –r /usr/hadoop root@Slave1.Hadoop:/usr/

Use Hadoop namenode – format command, format the HDFS file system, Figure 9


```
14/12/21 17:53:05 INFO namenode.NameNode: STARTUP_MSG:
/************************************************************
STARTUP_MSG: Starting NameNode
STARTUP_MSG:   host = Master.hadoop/211.85.9.240
STARTUP_MSG:   args = [-format]
STARTUP_MSG:   version = 1.1.2
STARTUP_MSG:   build = https://svn.apache.org/repos/asf/hadoop/common/branches/b
ranch-1.1 -r 1440782; compiled by 'hortonfo' on Thu Jan 31 02:03:24 UTC 2013
************************************************************/
14/12/21 17:53:05 INFO util.GSet: VM type       = 32-bit
14/12/21 17:53:05 INFO util.GSet: 2% max memory = 19.33375 MB
14/12/21 17:53:05 INFO util.GSet: capacity      = 2^22 = 4194304 entries
14/12/21 17:53:05 INFO util.GSet: recommended=4194304, actual=4194304
14/12/21 17:53:12 INFO namenode.FSNamesystem: fsOwner=root
14/12/21 17:53:12 INFO namenode.FSNamesystem: supergroup=supergroup
14/12/21 17:53:12 INFO namenode.FSNamesystem: isPermissionEnabled=true
14/12/21 17:53:12 INFO namenode.FSNamesystem: dfs.block.invalidate.limit=100
14/12/21 17:53:12 INFO namenode.FSNamesystem: isAccessTokenEnabled=false accessK
eyUpdateInterval=0 min(s), accessTokenLifetime=0 min(s)
14/12/21 17:53:12 INFO namenode.NameNode: Caching file names occuring more than
10 times
14/12/21 17:53:12 INFO common.Storage: Image file of size 110 saved in 0 seconds
.
14/12/21 17:53:13 INFO namenode.FSEditLog: closing edit log: position=4, editlog
=/usr/hadoop/tmp/dfs/name/current/edits
14/12/21 17:53:13 INFO namenode.FSEditLog: close success: truncate to 4, editlog
=/usr/hadoop/tmp/dfs/name/current/edits
14/12/21 17:53:13 INFO common.Storage: Storage directory /usr/hadoop/tmp/dfs/nam
e has been successfully formatted.
14/12/21 17:53:13 INFO namenode.NameNode: SHUTDOWN_MSG:
/************************************************************
SHUTDOWN_MSG: Shutting down NameNode at Master.hadoop/211.85.9.240
************************************************************/
```

Figure 9. Format the HDFS File System

Implementation of *shart-all.sh* started Hadoop, Figure 10:

```
starting namenode, logging to /usr/hadoop/libexec/../logs/hadoop-root-namenode-M
aster.hadoop.out
211.85.9.241: starting datanode, logging to /usr/hadoop/libexec/../logs/hadoop-r
oot-datanode-Slave1.Hadoop.out
The authenticity of host '211.85.9.240 (211.85.9.240)' can't be established.
ECDSA key fingerprint is fb:0d:0c:9c:1f:c1:fd:4e:c4:85:54:4b:c6:46:94:5c.
Are you sure you want to continue connecting (yes/no)? yes
211.85.9.240: Warning: Permanently added '211.85.9.240' (ECDSA) to the list of k
nown hosts.
211.85.9.240: starting secondarynamenode, logging to /usr/hadoop/libexec/../logs
/hadoop-root-secondarynamenode-Master.hadoop.out
starting jobtracker, logging to /usr/hadoop/libexec/../logs/hadoop-root-jobtrack
er-Master.hadoop.out
211.85.9.241: starting tasktracker, logging to /usr/hadoop/libexec/../logs/hadoo
p-root-tasktracker-Slave1.Hadoop.out
root@Master:/usr/hadoop#
```

Figure 10. Start the Hadoop Clusters

Completed the deployment of Hadoop in an embedded cluster, the research team of Medusa system was completed in May 2015 initial deployment, Figure 11:

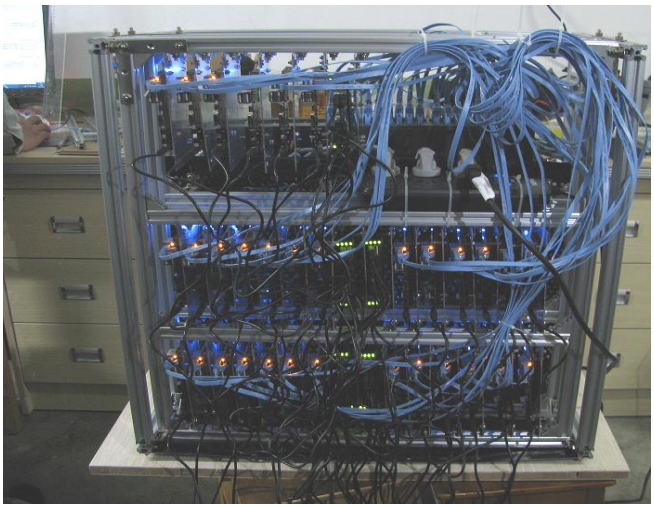


Figure 11. Successful Deployment of Hadoop "Medusa" Embedded Custer System

4.7. Batch Node Management

One by one manually configured node is not realistic in practice, so write management scripts allow bulk semi-automatic configuration Hadoop nodes information:

```
#!/bin/bash
sleep 1s
echo "Starting connection..."
sleep 1s
echo "Starting send files.."
scp /usr/hadoop/conf/core-site.xml /usr/hadoop/conf/mapred-site.xml /usr/hadoop/
conf/masters root@Slave1.Hadoop:/usr/hadoop/conf/
scp /usr/hadoop/conf/core-site.xml /usr/hadoop/conf/mapred-site.xml /usr/hadoop/
conf/masters root@Slave2.Hadoop:/usr/hadoop/conf/
scp /usr/hadoop/conf/core-site.xml /usr/hadoop/conf/mapred-site.xml /usr/hadoop/
……
conf/masters root@SlaveN.Hadoop:/usr/hadoop/conf/
scp /usr/hadoop/conf/core-site.xml /usr/hadoop/conf/mapred-site.xml /usr/hadoop/
echo "All files copy successfuly!"
```

5. System Testing

A number of different sized sample data for testing is prepared, at last two 20 million text data (about 20GB) distributed sorting test are operated, three parts consist of the test procedure: the RandomSelectMapper and RandomSelectReducer are for data extraction; ReudcerPatition is for data division, and SortMapper and SortReducer are for the feedback and data output of the results.

After the host starts, the *start-all.sh* command on the host side is used to open Hadoop distributed cluster system, after a node is initialized, the one by one will be in the main node of the supervision, with 31 nodes connected to the cluster, such as Figure 12:

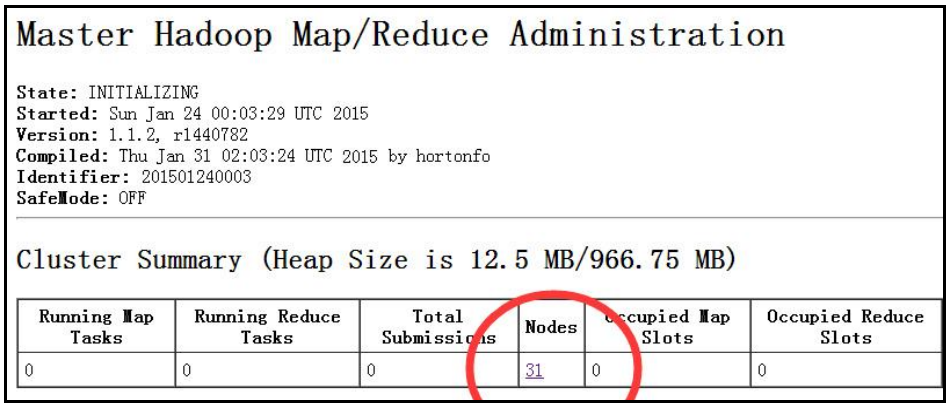


Figure 12. Map/Reduce Administration

Each node maps to external storage space remaining under the Linux *system/usr/Hadoop/tmp/DFS* path, so as to constitute a global space 56.26GB in a distributed manner. Use activity monitoring tool (Running Job) on a primary node or Terminal browser view the status of the current mandate (Running in the browser on the 192.168.1.132:50070 Job here), such as Figure 13.:

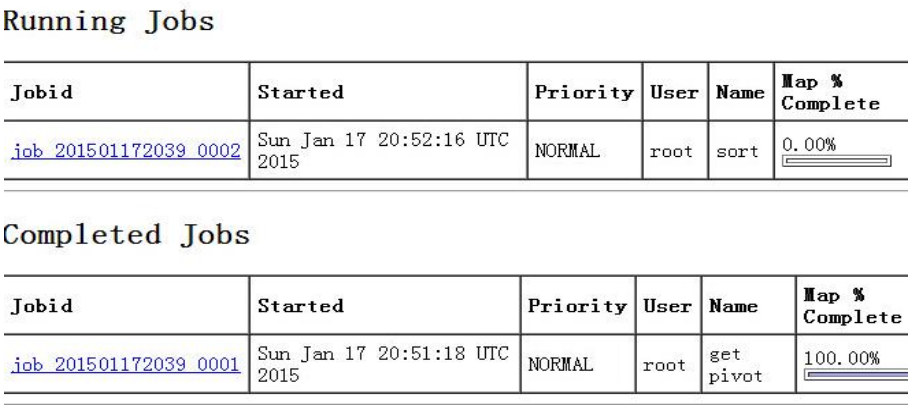


Figure 13. View the Task Execution Status on the Primary Node

When the data is processed, will exit from the node and displays its results in the monitor Terminal, when the Map-Reduce process has been completed, node data will summarize the results of the output current of the primary node, such as Figure 14:

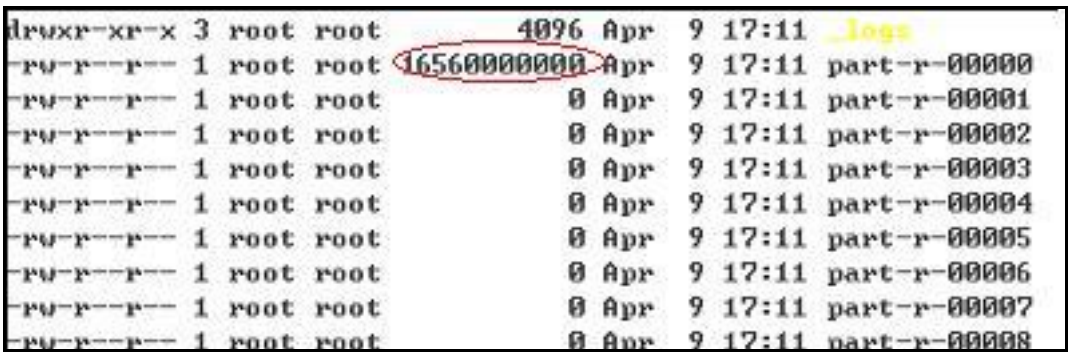


Figure 14. Results gathered in the Master Node from Client Nodes to a Results File

In Figure 14 part-r-00000 is the final summary results file, and part-r-00001~00031 files are files from the node, when data collection is complete from nodes file is empty

6. Summarization and Prospect

Through variety of research of embedded system platform , eventually in Cortex-7 schema of A10/A20 system the built of Hadoop distributed parallel calculation platform is achieved , can also match the limited resources of sensor and other embedded system to implement big data analysis, but system still exists more problem: due to problem of bootloader and kernel cut , current operating system, and data and data processing program are residing in disk , and Flash Card is main to disk, and it reads and writes more slowly than the internal memory, and in terms of program optimization ,it is still dominated by single threaded, not optimized for hardware, overall system performance is greatly affected; And A10/A20 is not special-processing chip, which itself carries the HD H.264 codec has not been applied. Future research platforms mostly handle on embedded systems core and program optimization and around Hadoop platform optimization aspects.

At the time of writing, relevant research field has undergone a significant change:

In March 2014 and January 2015. NVIDIA has released Tegra K1[15] ,X1 that is NVIDIA's first mobile processor and has the same advanced features & architecture as a modern desktop GPU while still uses the low power draw of a mobile chip, power consumption 5W. They have the same schema with 'Titan' and the 'Tianhe-2A' supercomputer, have small size just about 127mm square, and computing power achieved amazing 365GFLOPS. TK1 is NVIDIA's embedded Linux development platform featuring a Tegra K1 SOC and fully capable of complex calculations and data processing requirements.

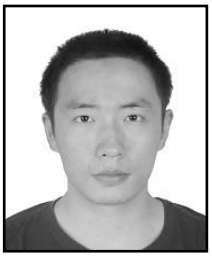
In the Hadoop platform performance optimization, many domestic and foreign researchers do a lot of work to do a lot of work. MIT and the University of Manchester researchers improved MapReduce performance on multi-core hardware [16]. The literature [17] proposed MapReduce on Cell Broadband Engine performance optimization techniques, and researchers at the University of Wisconsin used Cell Sort algorithm, giving full play to hardware capabilities, which greatly improved the performance of sorting. MapReduce performance on multi-core hardware improvements also includes in literature [18]. In addition, the GPU core number and frequency are of continuous improvement, University of Texas at Austin and other researchers at research institutions, tart research[19] on how to improve implementation of GPU to improve MapReduce performance , and extending the field of application of MapReduce. Tsinghua University and researchers at the IBM lab put Ma CG [20] at the source code level to provide portability of the CPU and GPU programming, greatly improving the MapReduce programming. Researchers at Ohio State University for multi-core environments proposed MATE program structure and environment, not only reduce the memory consumption but also the performance is far surpassed that of Hadoop, and Phoenix [21].

In terms of scheduling, literature [22] tried to use a priority-based scheduling strategy to improve efficiency of MapReduce. Literature [23] proposed the MapReduce optimized implementation based on MPI, using MPI-3 new features such as MPI Reduce Local to get 25% of the performance on the cluster of 127 nodes. Purdue University [24] researchers take the method of hunger-by loosening the synchronization requirements of schedule (eager scheduling) to improve efficiency of the MapReduce task [25] .Barcelona Supercomputer Center and researchers at the IBM Watson laboratory research scheduling strategy[26], with a view to improve performance. in literature [28] researched in heterogeneous processors and new algorithm for task scheduling in heterogeneous cluster environment to ensure that parallel task performance is not affected by the negative impact of heterogeneous environments.

References

[1] J. Dean and S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters", Communications of the ACM, vol. 51, no. 1, **(2008)**, pp. 107-113.
[2] Informatica. "Big data unleased", http://www.informatica.com/downloads/ 1601_big_ data_wp.pdf,**(2011)**
[3] L. Zhe, M. Dejun and G. Lantian, "Hybrid Scheduling of Embedded Multi-Processor System", Journal of Northwestern Polytechnical University, vol. 33, no. 1, **(2015)**, pp. 50-56.
[4] W. Shan, W. Hui-Ju and Q. Xiong-Pai, "Architecting Big Data: Challenges, Studies and Forecasts", Chinese Journal of Computers, vol. 34, no. 10, **(2011)**, pp. 1741-1752.
[5] S. Bush, "Big data opportuntiy for embedded systems firms", http:// www.electronicsweekly.com/news/design/embedded-systems/big-data-opportuntiy-for-embedded-systems-firms-2013-04/,**(2013)**.
[6] P. Karimkhan, "How can we relate big data to embedded systems", http://www.quora.com/How-can-we-relate-big-data-to-embedded-systems-Or-what-is-the-role-of-embedded-systems-in-big-data, **(2014)**.
[7] J. Tee, "Aggregating big data from embedded devices is changing the analytics world", http://www.theserverside.com/feature/BI-Opportunities-with-Embedded-Systems- Data, **(2014)**.
[8] ITTIA, "ITTIA DB SQL Puts Big Data on Mobile and Embedded Devices", http://www.ittia.com/news/press/2013/apr/big-data-embedded-devices, **(2013)**.
[9] STMicroelectronics, "STM32 Datasheet", http://www.st.com/web/ catalog/mmc/FM141/ SC1169/SS1031/ LN1565/PF164495?s_searchtype=partnumber, **(2014).**
[10] L. Jun, K. Xin-xian, Z. Tian-fan and L. Zhe, "Multiprocessor in Missile Guidance Systems and Research of Binarization Algorithms of Parallel Algorithms", Modern Defence Technology, vol. 43, no. 2, **(2015)**, pp. 103-109,164.
[11] S. Ghemawat, H. Gobioff and S. T. Leung, "The Google File System", In Proceedings of the 19th ACM Symposium on Operating Systems Principles , vol. 37,no. 7,**(2003)**, pp. 29-43.
[12] J. Dean and S. Ghemawat, "MapReduce: Simplied Data Processing on Large Clusters", Communications of the ACM, vol. 51, no. 1, **(2008)**, pp. 107-113.
[13] F. Chang, J. Dean and S. Ghemawat, "Bigtable: A Distributed Storage System for Structured Data", In ACM Transactions on Computer Systems , vol. 26,no. 2,**(2008)**, pp. 1-14.
[14] Apache. Apache Hadoop", http://hadoop.apache.org/,**(2015).**
[15] NVIDIA. Tegra K1 Mobile Processor", http://www.nvidia.cn/object/ tegra-k1- processor - cn.html, **(2015).**
[16] De Kruijf M, Sankaralingam K, "MapReduce for the cell broadband engine architecture", IBM Journal of Research and Development, vol. 53, no. 5, **(2009)**, pp. 1-12.
[17] Becerra Y, Beltran V, Carrera D, Gonzalez M, Torres J, "Ayguade E. Speeding up distributed MapReduce applications using hardware accelerators", Proceedings of ICPP **(2009)**, pp. 42-49.
[18] Ranger C, Raghuraman R, Penmetsa A,Bradski G, Kozyrakis C, "Evaluating MapReduce for multi-core and multiprocessor systems", Proceedings of HPCA **(2007)**, pp. 13-24.
[19] Stuart JA, Chen CK, Ma KL, Owens JD, "Multi-GPU volume rendering using MapReduce", Proceedings of HPDC, **(2010)**, pp. 841-848.
[20] Hong CT, Chen DH, Chen WG, Zheng W, Lin H, "Writing parallel program portable between CPU and GPU", Proceedings of PACT **(2010)**, pp. 217-226.
[21] Jiang W, Ravi VT, Agrawal G, "A Map-Reduce system with an alternate API for multi-core environments", Proceedings of CCGRID **(2010)**, pp. 84-93.
[22] Sandholm T, Lai K, "MapReduce optimization using regulated dynamic prioritization", Acm Sigmetrics Performance Evaluation Review **(2009)**, pp. 299-310.
[23] Hoefler T, Lumsdaine A, Dongarra J, "Towards efficient MapReduce using MPI", Springer Berlin Heidelberg **(2009)**, pp, 240-249.
[24] Kambatla K, Rapolu N, Jagannathan S, Grama A, "Asynchronous algorithms in MapReduce", Proceedings of CLUSTER **(2010)**, pp. 245-254.
[25] Polo J, Carrera D, Becerra Y, Torres J, "Performance-Driven task co-scheduling for MapReduce environments", Proceedings of NOMS **(2010)**, pp. 373-380.
[26] Polo J, Carrera D, Becerra Y, Beltran V, Torres J, "Performance management of accelerated MapReduce workloads in heterogeneous clusters", Proceedings of ICPP **(2010)**, pp. 653-662.
[27] Papagiannis A, Nikolopoulos DS, "Rearchitecting MapReduce for heterogeneous multicore processors with explicitly managed emories", Proceedings of ICPP **(2010)**, pp. 121-130.

Authors



Zhe Li, Zhe Li, PhD of Department of Automatic Control, Northwestern Polytechnical University (NWPU).Work at the Hubei Engineering University, lecturer. Major in: Consistency control of multi-agent in complex environment, Multi-Rate with multi-agent systems and Distributing control system.