Journal of Geoscience and Environment Protection, 2017, 5, 28-40
http://www.scirp.org/journal/gep
ISSN Online: 2327-4344
ISSN Print: 2327-4336

Study and Analysis of the High Performance Computing Failures in China Meteorological Field

Xiaoxia Chen, Jing Sun*

National Meteorological Information Center, China Meteorological Administration, Beijing, China
Email: *sunj@cma.gov.cn

How to cite this paper: Chen, X.X. and Sun, J. (2017) Study and Analysis of the High Performance Computing Failures in China Meteorological Field. *Journal of Geoscience and Environment Protection*, **5**, 28-40.
https://doi.org/10.4236/gep.2017.512002

Received: November 2, 2017
Accepted: November 28, 2017
Published: December 1, 2017

Abstract

China Meteorological Administration (CMA) has a long history of using High Performance Computing System (HPCS) for over three decades. CMA HPCS investment provides reliable HPC capabilities essential to run Numerical Weather Prediction (NWP) models and climate models, generating millions of weather guidance products daily and providing support for Coupled Model Inter-comparison Project Phase 5 (CMIP5). Monitoring the HPCS and analyzing the resource usage can improve the performance and reliability for our users, which require a good understanding of failure characteristics. Large-scale studies of failures in real production systems are scarce. This paper collects, analyzes and studies all the failures occurring during the HPC operation period, especially focusing on studying the relationship between HPCS and NWP applications. Also, we present the challenges for a more effective monitoring system development and summarize the useful maintenance strategies. This step may have considerable effects on the performance of online failure prediction of HPC and better performance in future.

Keywords

HPC, Failure, Numerical Weather Prediction, Real-Time Monitoring, Resource Management

1. Introduction

1.1. High Performance Computing System in CMA

CMA HPCS is managed by National Meteorological Information Center (NMIC) at CMA in Beijing, China. There are two national subsystems and other 7 regional subsystems locating in different provincial meteorological bureaus.

The supercomputer is an IBM Flex P460 system, consisting of 1786 compute nodes with 57,152 compute cores, and 5.4 petabytes storage. It provides a total peak performance of 1.7 Petaflops. In NMIC, there are two identical subsystems, one for production (Uranus), and the other for research (Neptune). For each subsystem, there are 560 computer nodes with 17,920 cores, 77 terabytes of memory and 1730 terabytes storage. Each compute node contains 4 Power7 compute cores (3.3 GHz). 58 computer nodes have large memory with 256 GB, while others with 128 GB memory. Access to compute resources is managed by LoadLeveler. The IBM General Parallel File System (GPFS) is used to support I/O operations. The compute clusters and storage clusters are connected by InfiniBand, with the speed of 160 Gb/s for either way.

The CMA HPCS are served for all the users in the CMA campus and other provincial bureaus, including users from National Climate Center and Numerical Weather Prediction Center and other operational centers.

1.2. CMA HPCS Performance

We analyze a dataset of 44-month CMA HPCS workload traces and investigate the users' waiting patterns, which include all the jobs submitted from Nov. 2013 to Jun. 2017. We also analyze the resources usage by week, by month and by year. The subsystems that we analyzed are Uranus and Neptune which are located in NMIC and served for CMA users with high utilization. From the previous data collections, we can see the following characteristics:

1) High CPU Utilization: CPU utilization is an important indicator of the system performance.

From Figure 1, we can see that at the very beginning of the 8 months, the CPU average utilization of Uranus is around 51%. In the 9th month, both Uranus and Neptune were in use; all the research work were assigned to Neptune; the average CPU utilization of Uranus decreased immediately, with a figure of

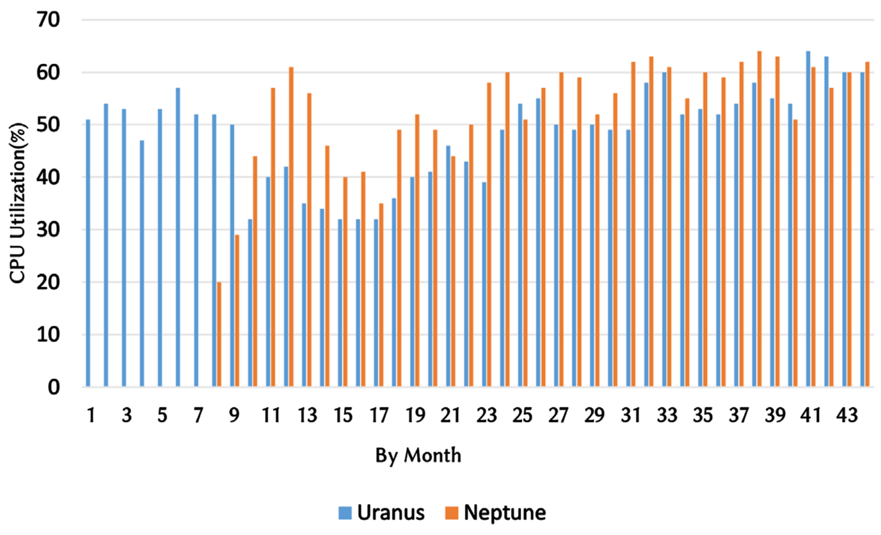

Figure 1. CPU monthly utilization.

36%, while 47% for Neptune. In the following two years, both of the systems have been in high utilization, with 54% and 59% respectively.

2) Small jobs are majority: Every month, there are about 1.8 million jobs which are submitted, meaning that about 60,000 jobs are submitted every day. 84.81% jobs run with a single core, only 5.36% jobs need 32 cores each time, which is followed by those jobs needing 8 cores, with 3.79%. If we take those jobs with one core out of the calculation to analyze the parallel jobs, we will discover that, 65% of jobs run with cores which are less than 32; 12% of jobs ask for cores lying between 33 and 128; 23% of jobs need more than 128 cores.

3) Short Job Waiting Time: 96.15% of the jobs run within 3 minutes. Only 1.5% jobs need to wait for less than 10 minutes to get the resource. We also analyze that for those jobs requiring to wait more than 30 minutes are either out of the user's resource quota or high frequency of job submission by the same user.

4) Top 20 Users & Resources Usage: There are over 200 users, with top 10 users taking up 46.1% resource, top 20 users 70.5%. High Time Slots are those time periods which have high CPU utilization. And production users use more cores in high time slots (Figure 2).

5) Application Usage Distribution: When it comes to application usage, we notice that Beijing Climate Center Climate System Model (BCC_CSM) accounts for 36.3%, while Global and Regional Assimilation and Prediction System (GRAPES) for 25.1%. The third application goes to WRF, with 17%. The top 5 application shows that the resources utilization is very typical.

6) "Round" Estimates for Wall Clock Request Values: Users tend to choose rough time for the estimation for Wall Clock Requested (WCR), a similar user behavior has been found in other HPC machines [1] [2]. We discover that jobs requiring more compute cores with less wall clock time overall have more chance of avoiding long-time waiting in the queue, comparing with those requiring fewer compute cores with more wall clock time.

7) Busy Time Slots: For production system, the busy time slots are very evident for the reason that numerical weather prediction models run by different

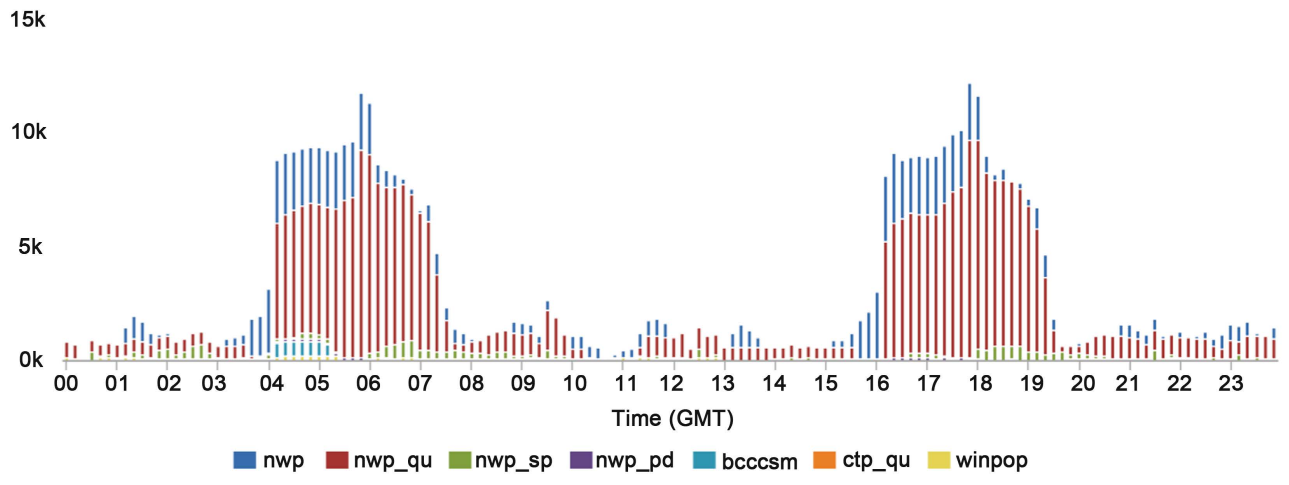

Figure 2. Production users using more cores in high time slots.

forecast hours; while for research system, the CPU utilization is relatively stable with high usage because scientists can run the jobs automatically for research work (Figure 3).

The world lives online. We all depend on IT hardware to keep everything going. Reliable HPC resources are imperative to production running. Many researchers have pointed out the importance of analyzing failure data. Failure analysis is the process of collecting and analyzing data to determine the cause of a failure, often with a goal of determining corrective or preventive actions.

1) System Failures

Failure rates in high performance computing systems rapidly increase due to the growth in system size and complexity. Hence, failures became the norm rather than the exception. Different approaches on HPCS have been introduced, to prevent failures or at least minimize their impacts. On-demand resilience is proposed to work as an approach to achieve adaptive resilience in HPCS. The

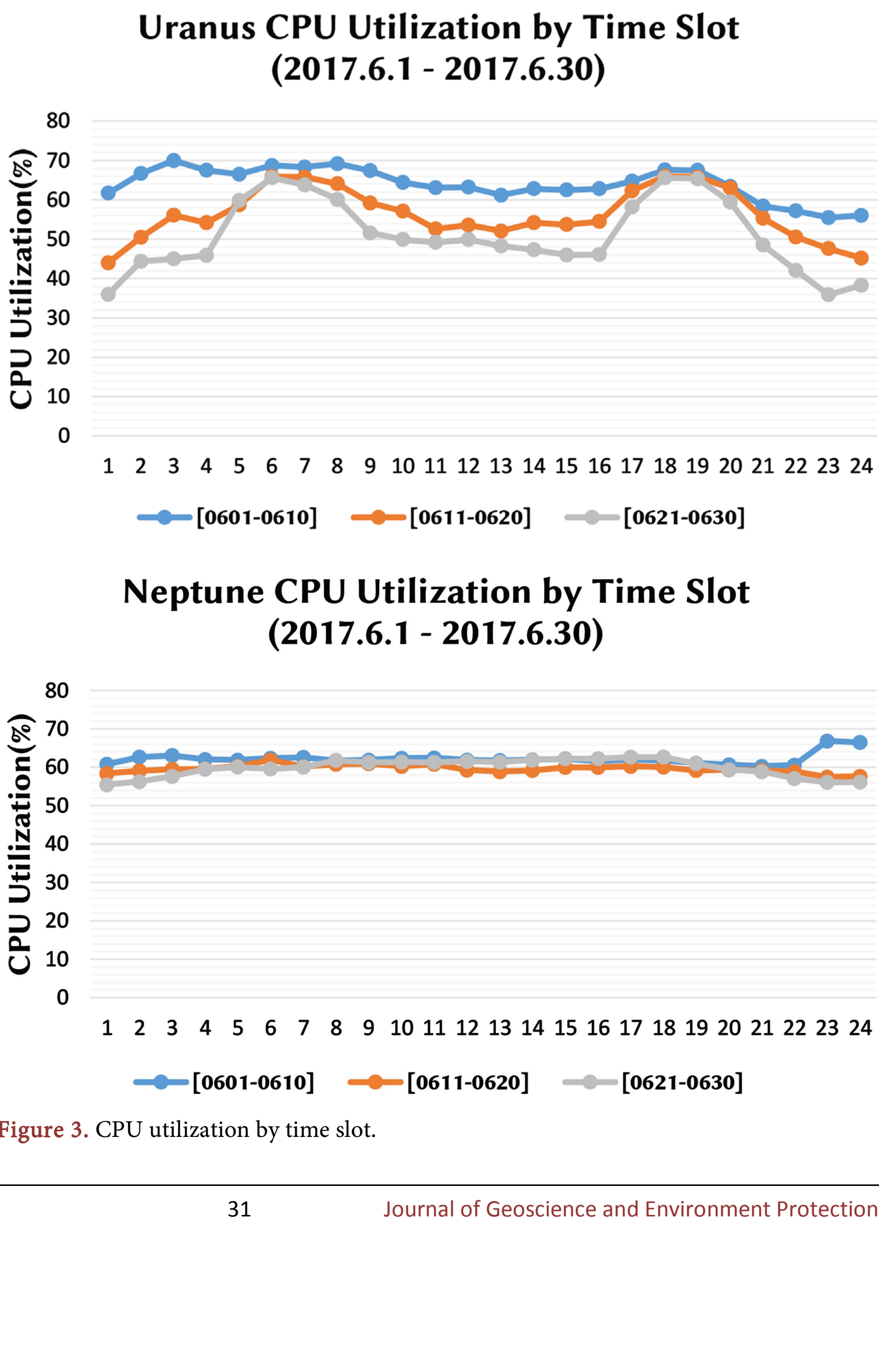

Figure 3. CPU utilization by time slot.

HPCS is considered in its entirety and resilience mechanisms such as check-pointing, isolation, and migration. To mitigate a large number of failures occurring at various layers in the system, to prevent their propagation, and to minimize their impact, all of these are very essential for daily maintenance. In the case of failures that are estimated to occur but cannot be mitigated using the proposed on-demand resilience approach, the system administrators will be notified in view of performing further investigations into the causes of these failures and their impacts.

The system failures come from software, hardware, user operations, network, and environmental problems (e.g. power outages). For each failure, the data includes start time and end time, the system, node and jobs affected, as well as categorized root cause information.

We collect and analyze a dataset of 44-month failures of CMA Uranus and Neptune. Hardware failures are easy to track as we make a record with any hardware replacement with manufacture. (Neptune was installed later and put into operation 8 months after Uranus.)

Figure 4 shows the number of failures per month, starting from production time. Failures number is comparatively low at the first several months. The failure number actually grows over a period of nearly 18 months, before it eventually starts dropping. The reason most likely is that many problems in hardware, software and configuration are only exposed by real user code in the production workloads [3].

There are some research work in understanding the correlations between system failures and workload. Study shows that there is a correlation between a system's failure rate and its workload [4], since in general usage patterns workload intensity and the variety of workloads is lower during the night and on the weekend. When it comes to CMA's HPC systems, as Figure 3 indicates that it has been very busy over the period of the lifetime. We cannot see the relationship

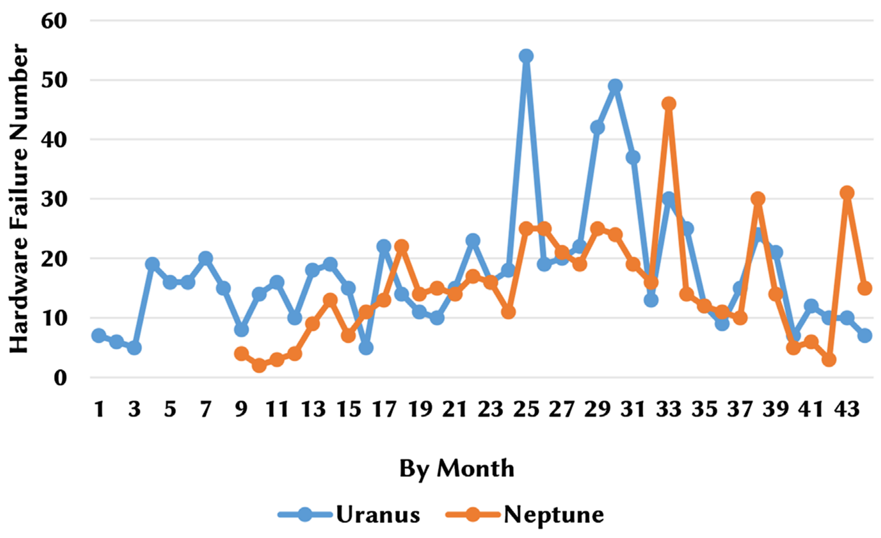

Figure 4. Hardware failures by month.

between the failure rate and its workload intensity. But we analyze the system outage and find that the regular maintenance before the flood season often brought a high percentage of motherboards' replacement.

The curve in Figure 5 corresponds to the hardware failures observed during the first 44 months. For Uranus, 82% of hardware failures come from computer nodes, with the backplane, IB card, internal HDD, FC Card, Ent Card, CPU and Memory; 12% of failures are from storage, with replacement of disk; other failures such as switch, power supply, controller, leaf module and spine module count for 3%. The similar distribution can also be found in Neptune (Figure 5).

When it comes to software failures and other errors caused by users' behaviors, there are two datasets, which represents our development in monitoring and management systems deployed at two different stages. The failure record at the first stage was written by the operation staffs, writing each failure in the shared files. The current failure record is from the monitoring system which stores the historical data. Figure 6 shows the software failures distribution starting from May 2016 to Jun. 2017. IO waiting takes the largest part, with around 68% and 58% of the total failures for Uranus and Neptune. Abnormal GPFS status, abnormal LoadLeveler status and memory overflow are the top failures. Other failures including local file system of the important nodes (login nodes and scheduler nodes) exceed the threshold value, connection and process errors.

2) Overview of NWP Applications and Its Failures

Weather forecasting is regarded as an important element affecting the socioeconomic welfare of the globe—a situation demanding a highly reliable weather forecasting system. In response, CMA scientists and researchers use computationally demanding NWP models to calculate the atmospheric movements and physical processes that cause weather changes. There are currently more than 30 operational NWP models in total, including GRAPES_GMF, GRAPES_GDA, T639_GMF, T639_GDA, Haze and etc.

From the Table 1 and Figure 7, we can see that the NWP models are not working alone. The results of T639_GMF is the source data of other models,

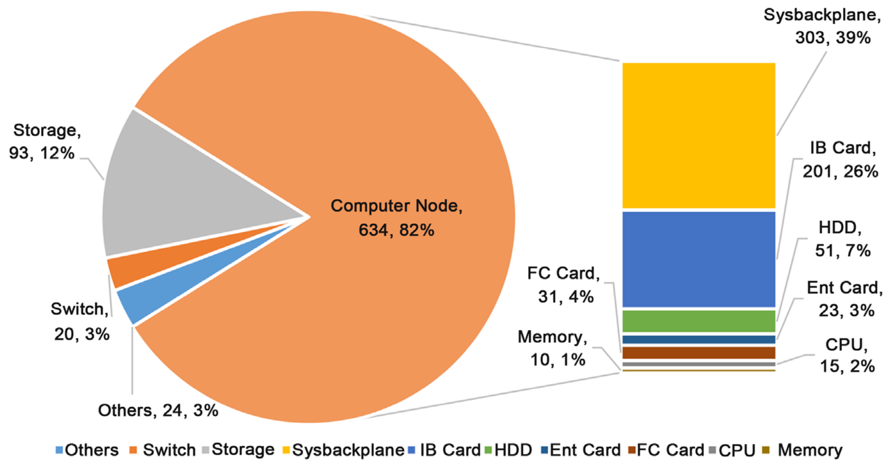

Figure 5. The distribution of hardware failures of Uranus from Nov. 2013 to Jun. 2017.

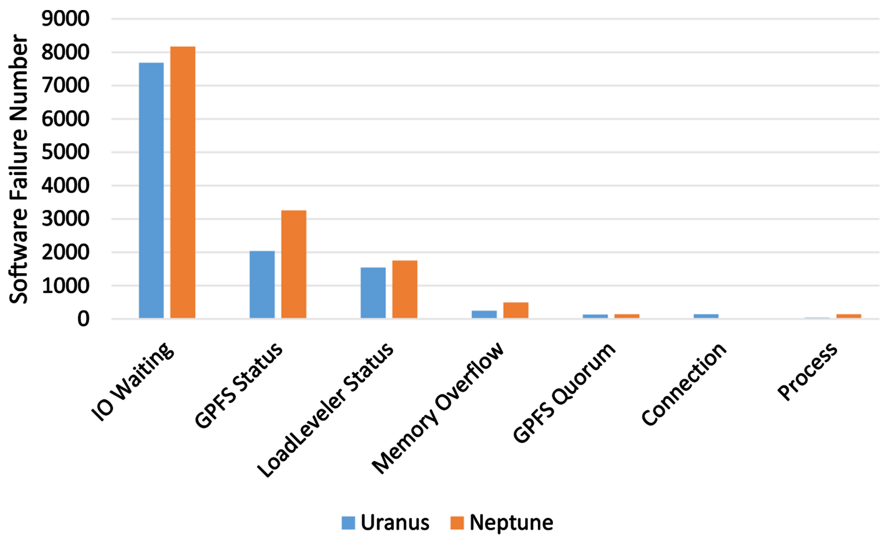

Figure 6. Software failures distribution from May 2016 to Jun. 2017.

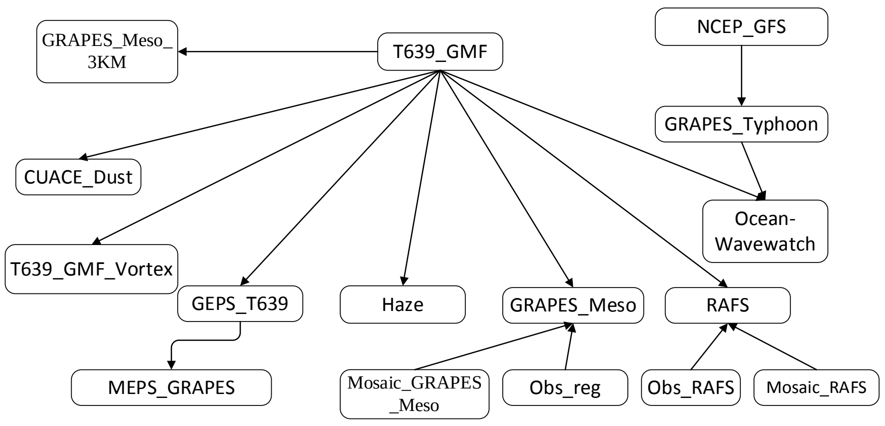

Figure 7. The interconnection among the current NWP model.

Table 1. The overview of each NWP system.

USER	SYSTEM	Cycle/Run Frequency (UTC)			
		00	06	12	18
nwp_op	CUACE_DUST	05:30-07:05		18:30-20:10	
	GRAPES_MESO	03:20-06:55		15:20-18:01	
	GRAPES_GDA	07:00-08:10	13:00-14:10	19:00-20:10	01:00-02:10
	GRAPES_GMF	03:30-06:50		15:30-19:00	
	GRAPES_GDA_DIAG	07:30-08:40	13:30-14:40	19:30-20:45	01:30-02:45
	T639_GMF	02:59-04:53	10:45-12:06	14:59-16:55	22:15-23:42
	T639_GDA	09:00-10:00	13:10-14:10	21:00-22:00	01:10-02:10
	FIRE_Forecast			20:00-20:05	
	Verify	06:30-07:00			
	Obs_Retrieve	03:10-03:13	09:10-09:13	15:10-15:13	21:10-21:13

such as Haze, CUACE_Dust. Hence, the failure of the NWP models will affect other models' operation.

For production system, CMA has been utilizing the SMS or ECFLOW to establish the whole process for NWP and Climate models. From the record of the failures occurred within 44 months, there are 1191 failures. Among them, they are classified into different categories, including NWP models, network, servers' related failure, data, file system and others. Segmentation fault, necessary data delay and bug in program design account for a very large part of the failures. Node related failures including the failures of the HPC hardware and ftp servers' disconnections is another contributor. Thanks to the real-time monitoring system and quick response of the failures, HPC cluster seldom influence the operation of NWP model for weather forecast products.

2. CMA HPC Operation Monitoring System

Through the HPC resource management system, we know very well about the HPC performance. In order to make the best of the system, we work hard on the improvement of the HPC monitoring system. For one thing, user's behavior and Scheduling Policy values a lot, for the other thing, to monitor the HPC is the base for maintenance. We have accumulated a lot of experience or a plethora of solutions to ensure that quality is optimized and downtime is minimized.

HPC Operation Monitoring System is used to monitor the following aspects of the system. We monitor the item listed as below:

1) Hardware and Software of Nodes: CPU, memory, IB network interface card slot, fiber channel.

2) Resources Management: busy queue & busy jobs.

3) GPFS File System: usage, status.

4) InfiniBand: connection status, IB card.

5) Users abnormal actions: Such as running the job with high demand of memory at the submit node; command file with an infinite loop which will run out the resources of the node to make other jobs being held.

6) Special Nodes including logon nodes, service nods, management nodes and I/O nodes.

7) Process Monitor: SMS server (rpc process), LoadLeveler status.

8) I/O nodes: waiting status.

3. Challenges

There are several challenges we face in order to provide high reliability and better service for our users in meteorological field, including how to find the failure chain, how to reduce failure detection rate and improve false alarm rate, how to take actions before affecting the application, all in a word, to improve user experience and the efficiency of maintenance.

1) A Failure Chain

Some errors are isolated from others, to fix them directly will solve the problem; while some failures are correlated to others. A sequence of successive failures may occur sometimes.

How to improve the failure prediction based on failure correlations and analyze the propagation pattern of failures in different system layer, quantify failures' impact within their effectiveness zone, all of these are the challenges for monitoring and maintenance.

2) Failure Detection Rate and False Alarm Rate

Detection Rate [5] represents the percentage of failed drives that are predicted correctly as failed. False Alarm Rate is another important metric which represents the fraction of good drives that are miss-classified as failed.

For example, we use the mmdiag command to query various aspects of the GPFS internal state for troubleshooting and tuning purposes. This information can be very helpful in troubleshooting deadlocks and performance problems. For each thread, the thread name, wait time in seconds, and wait reason are typically shown. From **Figure 6**, we can see a high proportion of IO waiting. However, the alerts of the long wait are momentary values, which do not represent the real situation. But there are some cases which the long waiting time are the indicator of the failure of IO node. So how to increase the failure detection rate and reduce the false alarm rate is the challenge.

3) Failure Discovery Lag behind the Application Performance

Each system has the "cluster master" server which collects syslog data from all other nodes in the cluster. In CMA HPCS, there are two management servers which run periodic scripts to collect the data for checking the health status. The cluster master then forwards the logs to a dedicated monitoring server system, which filters the incoming data and analyzes the data and then makes alerts to the operations staff using the monitoring website. All the syslog is stored for further research.

Such monitoring method has proved useful for handling most day-to-day issues, as it works very well for alerting on well-characterized issues with easy-to-parse error logs. However, it provides little information which cannot be easily caught using system logs, especially for the application issues. It is easy to check the users' job descriptions, including the jobs submitted time, started time, ended time, input scripts, output and error files. It limits the ability to analyze other types of issues, such as the always-challenging question—"Why is my job so slow" "Why is the system so slow". Poor job performance is not noticed until it has affected other related jobs. To monitor the system's latent failure before it influences the application is our great challenge.

4. Solutions and Maintenance Strategies

There is no such thing as a "best" monitoring tool; rather, the best tool is the one that we can use and understand and the one that solves our problems.

From the **Figure 8**, it is clear that what to monitor:

1) Hardware monitoring—the processor, memory, local disk, fan, adapter of the computer nodes and the I/O nodes, the disk array and the switches.

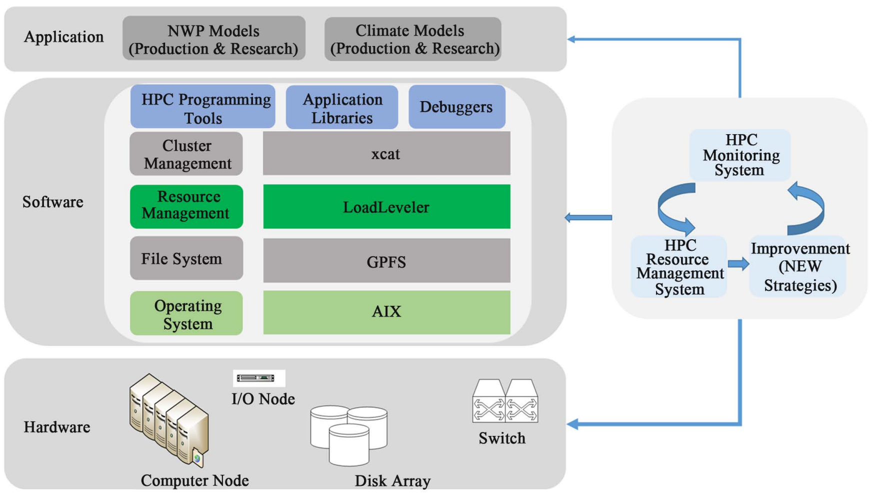

Figure 8. Architecture of the CMA HPC maintenance system.

2) Software monitoring—operating system, file system, resource manager (job scheduler), cluster management HPC programming tools, application libraries and debuggers, keep track on any new packages or versions.

3) Application—there are NWP and climate models running on the HPC. Focus on the jobs, the user experience (when the job was submitted, when the job started, how long it sat in the queue, how many jobs are in the queue at any one time, which queues have the most jobs waiting, the most popular day of the week and time of day jobs are submitted) and workflow execution.

There is a HPC monitoring system, to check the status of normal computing nodes, normal IO nodes, special computing nodes, login nodes, management nodes. We also use the HPC resource management system to analyze the resource usage to provide better services for users. For example, to exploit a new queue for those special jobs under circumstances, to dispatch more compute nodes to the queue which is always very busy, to adjust the quota limit for certain users who behave badly.

Based on the previous work of maintenance and HPC resource support, we argue that a minimum set of maintenance strategies including scheduled inspection, scheduled preventive maintenance and emergency maintenance (Figure 9).

1) Scheduled Inspections

Regular health check by running the scripts and pushing the data to the web-based monitoring system with 7 × 24 operators is one of the most effective way to guarantee the system's high performance and reliability.

Failure Finding Inspections: inspect the equipment on a scheduled basis to discover failures. If the equipment is found to be failed, initiate corrective maintenance. Fix the errors when it fails. Restart the related service or replace the hardware. Almost all hard drive manufacturers have implemented Self-Monitoring, Analysis, Reporting Technology (SMART) in their products [6], which monitor internal attributes of individual drives and raise an alarm if any attribute exceeds a pre-defined threshold. For example, disk storage or local disk usage will raise an alarm if the usage exceeds a pre-defined threshold.

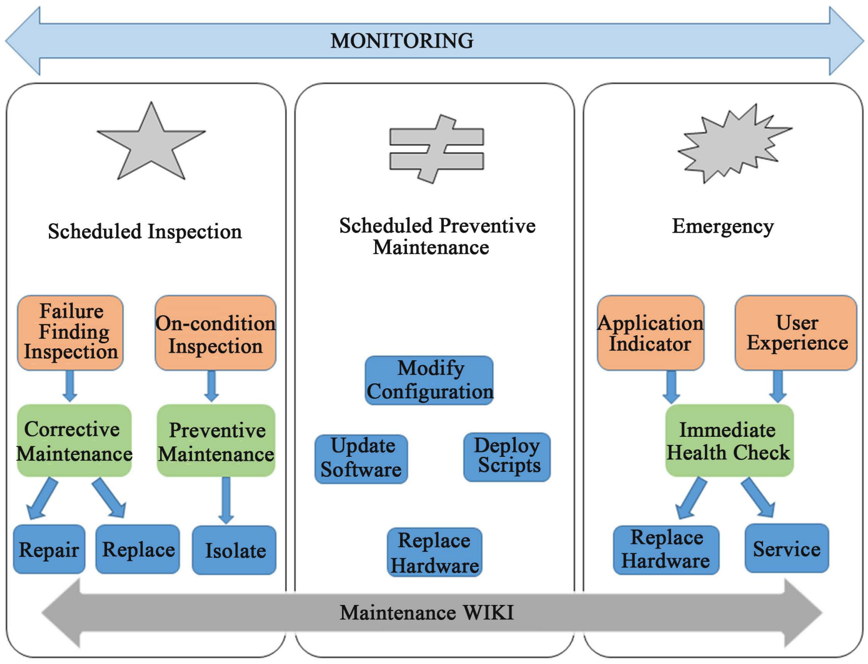

Figure 9. Maintenance strategy map.

On-Condition Inspections: Inspect the equipment on a scheduled or ongoing basis to discover conditions indicating that a failure is about to occur. If the equipment is found to be about to fail, initiate preventive maintenance. Isolate the nodes with latent fault such as multiple times of mmfs error log or SYSVMM may signal the file system is bad or some service is abnormal. So the first action is to isolate the node. When a part of an application running on a node that seems to respond very slowly or likely to fail, which may lead to failure of the whole application, fault tolerant techniques, such as replication and erasure code are often used. For example, an error of a computing node may cause the unavailability of the access of the file system, IO node waiting for itself causes a deadlock, which slows down the whole system.

2) Scheduled Preventive Maintenance

Failure avoidance is done by taking a preventive action [7]. There is one regular maintenance before flood season. Through the shutdown action, some replacement of the hardware, software update, scripts deployment will be executed. Each time, each step and the amount of time, replay actions will all be considered in advance.

Techniques such as pinging a node or running a small script on a node can inform the master whether a node is alive. An alternative way to do this is to have the master node to see if a node is alive. If the command runs successfully, the node is "alive". This can also capture slow-running nodes that have some issues and can't complete the command for a long time.

3) Emergency Maintenance

The above two strategies cannot cover all the failures discovery, user experience and application execution server as a plus. Some latent failure are not lethal if there are no jobs running on the nodes. For example, the adapter of the disk array chassis can survive with only one in case of the outage of another one in most circumstances. But with some exception both of them go shutdown, causing the whole disk array offline. This is a chain failure, making the file system unavailable for the operational models to read to delay its run. Take another example, hundreds of jobs failed to be submitted through SMS while they were submitted successfully through manual way, which never happened before. After checking the whole system, we finally found that the local file system of scheduler node had been full. It was the subtle mistake that cause the breakdown of the operational application. After that we figured that there had been alarm for the threshold of the scheduler node which was set aside. Then we change our monitoring policy demanding each operator to check the previous errors occurred before his or her duty week. Both technology and management strategies work together can improve the performance and reliability of the system.

4) Maintenance WIKI

With the rapid development of HPC in CMA, we have built different systems to satisfy the increasing demand either in maintenance or in management filed. Collect and analyze failure information for improving the monitoring and resource management design. Through the record of operator, we have established a maintenance wiki, which enables other engineer to search for the possible solutions when error occurs. With the help of the cloud desktop technology, we are able to check the system anywhere anytime. Wechat is one of the most popular application, which catches and sends the errors to the operator directly, serving as a quick and time-saving method.

5) User Guide and Scheduling Policies

User habits can make influence on the HPCS. From the top level, we make a set of principles to follow, such as how to submit the jobs, how to write the command jobs. Adjust the scheduling policies according the resource usage and performance.

As specified on user guide, the jobs' running time that exceeds their WCR limits would be forced to terminate hence yield incomplete results. Some users don't write the WCR limits while some users take the incentive to overestimate their jobs' WCR to prevent unexpected termination.

As a result, the larger WCR estimates, in principle, increase the job's queuing time as it reduces the chance of fitting in smaller but vacant space [8]. We encourage user to choose values of WCR that are more exact, which could reduce the efficiency of backfilling for better packing.

5. Concluding Remarks

In this paper, we have illustrated the system failures and application failures. The main aims were to study the relationship between resource and application, to increase the performance of HPC, and to improve the reliability of HPC.

Combined with the analysis of monitoring system and resource management system, we summarize a set of principle to follow to better predict the failures and make an effective user guideline and scheduling policy.

As a future work we can include the effect of changing the policy. Other direction is to continue to expand and analyze the failure dataset to discover the latent failures. A third direction is to apply the learning model to the online detection system to fix the problems before it affects the application.

Acknowledgements

The projects presented in this paper are supported by the National Key Research and Development Program of China (Grant No. 2016YFA0602102) and China Special Program for Meteorological Research in the Public Interest (Grant No. GYHY201306062).

The research used the data from High Performance Computer Management System, High Performance Monitoring System, NWP on Duty System. We would like to thank all the above data source applications.

References

[1] Tsafrir, D., Etsion, Y. and Feitelson, D. (2005) Modeling User Runtime Estimates. *JSSPP'*05: *Job Scheduling Strategies for Parallel Processing*, Boston, 19 June 2005, 1-35. https://doi.org/10.1007/11605300_1

[2] Tsafrir, D., Etsion, Y. and Feitelson, D.G. (2007) Backfilling Using System-Generated Predictions Rather Than User Runtime Estimates. *IEEE Transactions on Parallel and Distributed Systems*, **18**, 789-803. https://doi.org/10.1109/TPDS.2007.70606

[3] Schroeder, B. and Gibson, G. (2010) A Large-Scale Study of Failures in High-Performance Computing Systems. *IEEE Transactions on Dependable and Secure Computing*, **7**, 337-350. https://doi.org/10.1109/TDSC.2009.4

[4] Sahoo, R.K., Sivasubramaniam, A., Squillante, M.S. and Zhang, Y. (2004) Failure Data Analysis of a Large-Scale Heterogeneous Server Environment. 2004 *International Conference on Dependable Systems and Networks*, Florence, 28 June-1 July 2004, 772-781.

[5] Fouz, F. and Hadi, A.A. (2016) Detecting Failures in HPC Storage Nodes. *International Journal of Scientific and Engineering Research*, **7**.

[6] Allen, B. (2004) Monitoring Hard Disks with Smart. *Linux Journal*, No. 117, 74-77.

[7] Zhu, B., Wang, G., Liu, X., Hu, D., Lin, S. and Ma, J. (2013) Proactive Drive Failure Prediction for Large Scale Storage Systems. 2013 *IEEE* 29*th Symposium on Mass Storage Systems and Technologies* (*MSST*), Long Beach, 6-10 May 2013, 1-5. https://doi.org/10.1109/MSST.2013.6558427

[8] Fang, B., Guan, Q., Debardeleben, N., Pattabiraman, K. and Ripeanu, M. (2017) LetGo: A Lightweight Continuous Framework for HPC Applications under Failures. *Proceedings of HPDC'*17, Washington DC, 26-30 June 2017, 14 p. https://doi.org/10.1145/3078597.3078609