International Journal of Grid Distribution Computing
Vol.7, No.4 (2014), pp.41-48
http://dx.doi.org/10.14257/ijgdc.2014.7.4.05

The Comparison of Checkpoint Technology in the Cloud

Li-Jie Cui[1], Hui He[2] and Guang-Yi Tang[3]
[1,3]*School of Software and Engineering, Harbin University of Science and Technology, Harbin, China*
[2]*Harbin Institute of Technology, Harbin, China*
[1]*andyclj1977@163.com*, [2]*hehui@hit.edu.cn*, [3]*tanggy818@126.com*

Abstract

Improving the fault tolerance ability of a cluster system is an important technology. Using a checkpoint in a multi-machine system is a hot topic in China and abroad. This study introduces common checkpoint technologies and algorithms, as well as presents a common improvement strategy and a comparative analysis.

Keywords: *checkpoint, fault-tolerance, Condor, SNS, CL*

1. Introduction

Being the important tools to improve the fault-tolerance ability of clusters and grids, research on checkpoints technology has been a hot area. In this paper, we give a detailed survey of checkpoints theories. We studied 10 mainstream checkpoints algorithm of their advantages and disadvantages, and analyzed the basic principles and their applying environment, and presented the improving strategy.

2. Checkpoint Concrete-setting Technology

2.1. Setting Theory

Checkpoint design and rollback recovery in a system during normal operation in setting checkpoints save system coherency state of time, and perform correlation tracking and recording. After a fault occurs, the related process rolls back the fault repatriation system to a coherency state. When the state recovers, the checkpoint restarts, and the system also recovers.

2.2. Checkpoint Setting and Recovery

2.2.1. Process Data and User Stack: When a checkpoint is set, an unoptimized and completely preserved mode (*i.e.*, a full checkpoint) copies data from the starting address to the newly assigned maximum address to a checkpoint file. During recovery, the data segment in the checkpoint file is copied back to the original position of the address space. The user stack process in the preservation and restoration stages is similar to the aforementioned scenario. When restoring a user stack during the current call process of the user stack operation, returning to the cover of the user stack contents prevents the return address and the local data structure from causing the program to run errors. This condition ensures that the useful content of the current user stack is not covered, and thus, a user stack can be used back to back.

2.2.2. Context -related Items: Checkpoints and context switching-related items, including the program counter (PC), the processor status word (PSW) register, and the stack pointer (SP), can use the "sigsetjmp" system call to save an environment variable. space. The user stack process in the preservation and restoration stages is similar to the aforementioned scenario.During recovery, the corresponding system calls "siglongjmp" to restore the variable.

2.2.3. Related Signal Information: When the system calls "sigsetjmp" or "siglongjmp" to save or restore context-related items simultaneously, it also saves or restores the signal shielding code and the signal SP. Setting checkpoints through system calls "sigaction" and "sigpending" obtains the signal processing function handle and the pending signals, respectively. It records in the data section of a signal information table, which is in the data segment with the preservation. During recovery in the segment of the data copy back, the signal processing function handle and the pending signals recover consistently while they are taken out of the signal information table through the corresponding system calls.

2.2.4. Active File Information: We define the file information table for the preservation, restoration, and recording of file information activities. In the normal process of implementation, we supervise "open-read-write-close-files" operating system calls and maintain the corresponding file information items in the information table. In setting the checkpoint, the file information table is preserved as a part of the data segment. After data recovery, we open the file according to the original accessing mode, restore the file descriptor, and then read and write the pointer based on the activities of the file information recorded in the file information table.

3. Common Checkpoint Algorithms

In high-performance parallel computing systems, two types of checkpoints exist: single process and distributed checkpoints. The former saves the final state of a process to a document of the external memory and guarantees the recovery process from the external documents, thus this checkpoint continues running correctly. Meanwhile, the latter is composed several single process checkpoints. Checkpoint technology in high- performance parallel computer systems can cut and recover parallel program states through various protocol designs that use a single process checkpoint technology.

Distributed checkpoint algorithms are divided into synchronous and asynchronous checkpoint algorithms.

In a distributed asynchronous checkpoint algorithm, the program is processed independently to save itself in a running state. The recovery process requires coordinated rewinding into the appropriate checkpoint time, thus ensuring that the entire program is restored to a consistent checkpoint global state. The advantage of this algorithm is that it enables a distributed process with the greatest degree of autonomy. Meanwhile, its disadvantages are as follows. First, this algorithm requires a large space because each process needs to save the checkpoint file for a certain time to be able to return to a completely consistent global state. Second, the program recovery process may be rewound and may exhibit the domino effect.

The present study focuses on a representative single process checkpoint Condor [1] algorithm and a distributed synchronous checkpoint algorithm. SNS and CL are discussed in detail in the succeeding sections.

3.1. The signal process Condor algorithms

A single process checkpoint is the process address space at a given moment in a snapshot, which is the state of a process at a certain time. The state of the process is defined to ensure that the process correctly performs the necessary information after migration. The state of the process should include process data, user stack contents, the PC, the PSW register, the SP content, file information activity, and interrupt signal information.

The process must preserve the completeness of the process must preserve the completeness of the information of the decision process behavior factor of the process state and the process environment to achieve the correct rollback recovery. The process state is divided into non-volatile and persistent states. The former includes text segment, data segment, and operating system kernel structure. The latter refers to disk space implementation and related contents. The process environment refers to the process of the operating system environment, including accessing various resources such as exchange areas, file systems, and communication channels.

Core and content preservation is necessary to process a user area by employing the “setjmp” system call to save the processes of the PC and the SP pointer. Next, the process that opens the file open mode and the pointer offset should be saved. Finally, the stack segment of the data segment is processed, and the code section information is combined into one implementation of a checkpoint file. The recovery of the program running state is the implementation program of the checkpoint file only. The checkpoint file is executed, the program is returned to the checkpoint time operation state, and the “longjmp” system call recovery processes of the PC and the SP pointer are employed to continue process execution.

But it has two limitations. First, users must have the source code to set checkpoints. When only the executable code is available, users are unable to use the checkpoint function because the user program needs the checkpoint system static link.Second, the program cannot be used to save and restore the distributed program because a single process checkpoint algorithm does not run the use and process derivative of the user, and does not process communication-related system calls.

3.2. Synchronization of the Distributed SNS Algorithms

As a distributed synchronous checkpoint algorithm, the SNS algorithm program only needs to perform the process checkpoint file as it runs back. The program state is saved by the following process responsible for the PC. First, all process checkpoints begin broadcasting information (Mb) from the PC to the program. Any process that receives a message from Mb stops running. A message (Ms1) is sent to all recipients after the PC process. When the PC receives all the processes that send Ms1, it then processes the broadcast message, Mchk. This procedure starts the second time synchronization. Any process can receive the news immediately after Mchk at the local checkpoint. The checkpoint file sends the message Ms2 to the PC after formation. The PC receives all the processes that send Ms2. Finally, it processes the broadcast message, Me. When the process receives a message from Me, the old checkpoint file is removed to continue the operation.

The advantage is that the SNS algorithm is easy to implement without any cross news hour save program running state.

The disadvantage is that in the program status that saves the requirements in the two global synchronization processes, the program stops for long periods when using the checkpoint system: Cocheck, Mist, and Fail-Safe-PVM.

3.3. Synchronization of the Distributed CL Algorithms

The CL [2] algorithm program state is preserved by the following PC process. First, any process stops running because of its direct connection to machine processes that broadcast the message, Mb. Simultaneously, the PC processes the local checkpoint. Any process K receives a message from Mb. If the checkpoint operation has not yet started, then process K stops running and directly connects the machine processes to the broadcast message Mb. The local checkpoint is then initiated. When the process in the K checkpoint operation begins, the receiver sends a checkpoint control message n for preservation. When the K process checkpoint file forms, it receives directly connected machine-processed messages sent by the Mb. K and PC processes occur simultaneously to send a message to Ms. When the PC process receives all the programs that have sent Ms messages, all process checkpoint files produced are marked. The PC should then broadcast the message Me to each process. Finally, the old checkpoint file is processed, thus ensuring that the PC process continues to run. The K process receives a message from Me. This process also deletes the old checkpoint file, and then continues to run.

The advantage is that different algorithms with SNS do not have global synchronization overhead and have a complex topological structure for a distributed system.

The disadvantage is that when the machine has a fully connected structure, the time complexity of the algorithm is too high.

4. Checkpoint Improvement Strategy

Decreasing the checkpoint file information and the time delay of the two aspects of the checkpoint algorithm provides improvements [3]. In this section, we will separately reduce checkpoint save information representative algorithm incremental, programmer guide, and compiler-assisted and hidden delays, as well as improve the operation of the parallel representative algorithm of the main memory. Thus, the CLL copy is written.

4.1. Common Technology and Algorithm Improvement

4.1.1. Incremental Mode: Incremental mode from the decreased checkpoint size angle optimization saves important checkpoint means. The basic idea is to save the data segment because the last checkpoint has been modified or newly assigned to each page. Pages without modifications have been stored at the previous checkpoint file. An incremental mode requires holding a majority of the checkpoint files. At the checkpoint, if the checkpoint file number is beyond the user-written maximum checkpoint file number, then the checkpoint files are merged. During restoration, the checkpoint file is stored in the memory area content back to back according to the time sequence from front to back.

Incremental checkpoint is the use of pages of virtual memory management. The complete process is as follows. In the checkpoint settings, all pages in the program space are set to “read-only.” If the program in a subsequent operation needs to write a read-only page, it produces an error. Simultaneously, a corresponding processing program is recorded on the page, and the setting becomes “read and write.” Thus, the program can modify the page. The next checkpoint should be set and the processor state should be saved. All pages are modified without the need to preserve the entire program.

Application environment is as follows. The program should be applied and retained several times for checkpoint file occasions, such as rollback debugging or the asynchronous checkpoint algorithm. The incremental algorithm in the program has a good local when no obvious improvement effect exists.

4.1.2. Checkpoint Cache: Also known as the main algorithm, the checkpoint operation delay arises from the process operation state of the disk copy that copies memory time[4]. The process is as follows: 1) freeze preservation process user process operation, the process operation is copied to a memory backup space; 2) recovery processes run; and 3) dedicated replication processes responsible for the memory to disk backup copy space is started.

Disadvantage: The need for greater physical memory space is relatively large. Otherwise, memory and disk swapping will cause significant algorithmic delay.

4.1.3. Writing the Replication Algorithm: The written copy is based on the memory page protection mechanisms of most computer systems. The checkpoint allows the memory of each page and the preservation of the speech concurrency. In the checkpoint setting, the control of a separate thread is responsible for the memory page in the checkpoint, and the program continues to run. To preserve checkpoints in the process, the original program is stored on an unsaved page for the write operation. The system becomes the corresponding page to a dedicated buffer, while allowing the original program on the page. When the control thread places such a memory page, the content is saved to allow spot checks by employing the buffer backup. This strategy is an effective means to optimize the system, and the checkpoint overhead is decreased considerably.

Defect: From the beginning, the set checkpoints to the content are completely preserved in stable storage. This condition results in relatively long checkpoint latency, and in some cases, long checkpoint delay results in high expense for the system.For example, in a transaction processing system used to be presented rapidly, the checkpoint can be completed after the external output results.

4.1.4. The CLL Algorithm: The CLL algorithm writes the copy algorithm for memory backup space by using buffer technology. The algorithm and write replication algorithm is the difference between the state process and the saving process. Two replication processes are started.

Copy process 1 is responsible for the process operation state page copied to the memory backup space.Copy process 2 is the circular queue buffer management for memory space backup.The backup space cache process state is copied to the disk.

4.1.5. Compiler-assisted: The compiler-assisted memory area elimination method employs an automatic analysis to exclude the memory area. Through data flow analysis, the compiler cannot recognize all dead or clean memory areas, but can identify most of such memory areas. Combining such information with the programmer's guide can significantly improve performance. In this manner, the programmer only needs to point to a program memory area to be excluded in the compiler to insert the corresponding function, and decide on the specific exclusion that the memory region needs to work on.

4.1.6. Programmer Guide: User memory areas are mainly excluded by providing a certain function to the user. The program user who explicitly points out the dead memory or clean region is wrong about the regional information to be saved, in addition to providing functions to cancel the previous exclusion operation. By employing these two functions together, the user increases or decreases checkpoint flexibility to save memory space and decrease checkpoint overhead storage. However, this condition clearly increases the burden of the users.

4.2. Comparison of the Improved Algorithm

The comparison of algorithm is as the Table 1.

Table 1. The Comparison of Checkpoint Algorithm

Algorithm	Basic idea	Algorithm evaluation
User participation	Keep the self-checkpoint modified or have a new assignment for each page. Unmodified pages have been previously saved in the checkpoint file.	When the program has good locality, it clearly improves. This phenomenon applies to more moments of the checkpoint file for the reservation program.
Compiler-assisted	Providing programmer function to identify explicitly that the memory is not a preserved area in the program; providing function to cancel the previous exclusion action.	Flexible in saving memory space and time overhead. Yet, the algorithm should increase the burden on the programmer and should be error prone.
Checkpoint cache	Automatic analysis by the compiler occurs in regions of memory exclusion knowledge. The cleanest memory area should be saved. Block time sets up checkpoints after the backup program to continue. Simultaneously, the DMA checks the contents stored in the external equipment.	Users can participate in the binding (CAME algorithm). Implementation may be excluded from the region to find and correct user error automatically. The memory is large enough to decrease significantly the overhead of the checkpoint setting and the memory for caching, and thus, improvement is not obvious.
Copy on write	The protection mechanism of most computer systems based on memory page (i.e., the checkpoint) allows memory pages to preserve the concurrency of speech.	The overhead reduction is substantially set to the checkpoint, but with sufficient write operation time. The costs of time and space also increase.
The CLL	The algorithm employs cache technology in the backup memory space of copy on write algorithm. Two copy processes are initialized in the course of the preservation process.	Memory backup space is smaller than the process space. However, the overhead space increases and time is delayed when copying a memory page.

5. Checkpoint Algorithm Limitations

The current checkpoint algorithm generally exhibits the following limitations.

Operating system compatibility: The current, widely used operating system does not provide external access to all system process running mechanisms. This condition requires checkpoint algorithms to use special techniques in the operating system to obtain external access to necessary processes in the core region running on different operating systems. By using different techniques, the checkpoint algorithm for operation system compatibility and portability is significantly reduced.

Inability to back up file systems: The current state of the checkpoint algorithm in the bar procedure volume returns to the last checkpoint time to run and does not correspond to the recovery of the file system state and the state changes in the document. Thus, the execution results in error rollback.

6. Conclusions

This study assessed 10 kinds of current domestic and foreign popular checkpoint algorithms. It introduced the checkpoint technology and the single process (Condor), and distributed the checkpoint algorithms. The distributed checkpoint algorithms were subdivided into synchronous checkpoint algorithms (*i.e.*, CL, SNS) and asynchronous checkpoint algorithms for comparison. We proposed a checkpoint algorithm improvement strategy. Based on the decrease in checkpoint preservation of information, we introduced the assessment of incremental, user participation, and compiler-aided algorithms. To improve operation parallelism, we introduced assessed memory, write the copy, and CLL algorithms. This approach improved the strategy for a detailed comparison. The algorithm introduced the evaluation process. We provided a detailed analysis of the principle of the algorithm, its advantages and disadvantages, and its application to environmental analysis. Finally, we analyzed the existing limitations of the current checkpoint algorithm.

References

[1] M. Litakow and M. Solomon, "Supporting checkpointing and process migration outside the unix kernel", In Proc. USENIX Winter92, San Francisco, CA, (**1992**) pp. 283-290.
[2] K. Chandy and L. Lamport, "Distributed snapshots: determine globle states of distributed systems", ACM Trans on Computer Systems, vol. 3, no. 1, (**1995**), pp. 63-75.
[3] C. Yang, H.-l. Zhang, Z.-x. Tian, B.-x. Fang, "The Research of Fault-tolerant Mechanism of Grid Computing Based on Checkpoint Algorithm", Microelectronics and Computer, (**2006).**
[4] W. Zhang and H. He, "Fault Tolerance for Conjugate Gradient Solver Based on FT-MPI", Studies in Informatics and Control, ISSN 1220-1766, vol. 22, no. 1, (**2013**), pp. 51-60.

Authors

Li-Jie Cui, she is a lecture in school of software, Harbin University of Science and Technology, China. She was born on February 1977. She achieved her Master degree in software engineering in 2005 at Harbin Institute of Technology. Her research emphasizes on information technology, software engineering and network security.

Hui He, she received the B.S., M.S. and Ph.D. degree in computer science from Harbin Institute of Technology, Harbin, China. Since September 1999, she has been with the School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China, where she became an Associate Professor in October 2007. Her research interests include network computing, network security.

Guang-Yi Tang, he is a lecture in School of Software, Harbin University of Science and Technology, China. He was born on August 1980. He achieved his Master degree in computer software and theory in 2007 at the central china normal university. His research emphasizes on workflow, interoperation and open source software development.