A Difference Comparison Framework Based On Provenance

Liuqi Zhao

Jinan University, Guangzhou 510000, China

13229407712@163.com

Abstract

With the rapid development of computer technology computers are becoming more and more convenient. Computers store many similar data files, such as different version caused by the configuration of software updates, multi-version files . it is not easy for user to identify the differences between these files quickly. The provenance records the history of the data, which can effectively provide the user with the details of information, but the provenance also has its own disadvantages.It need a lot of extra storage space.This paper proposes a difference comparison framework based on provenance (DCP) ,which can quickly locate and find the data between different versions. The difference. According to the experimental results, the method can quickly obtain version difference data accurately with small cost, and can effectively improve work efficiency.

Keywords

Provenance, Multi-version data , file searching.

1. Introduction

Multi-version files are becoming common as users use computers to handle various transactions In their daylife. Multiple versions of the document and frequent modifications result in the storage of a large number of similar documents in the computer[1]. However, it is difficult for users to distinguish between documents and it is hard to obtain documents efficiently which has several versions . Therefore, how to obtain the difference between documents quickly is an important part of improving work efficiency. In the software development process, professional software developers will have special documents to describe the differences in software development versions. However, because of the few addition in documents ,lack of corresponding specifications. When naming documents, PC user commonly used numbers such as 1.0, 2.0, etc., or file names plus time to indicate file differences. but for file differences , few people can completely remember the differences for multiple versions of the document. The only way to figure out difference for users is to open the file and compare them. This method is not only inefficient but also confusing. When the file size is large, it is more difficult to find the document. The current file search tools are almost always based on file name searches which do not have the ability to distinguish between version files quickly.

The provrnance is a kind of data that describes data, and it includes how the data is generated. What kind of tools generated data. What is the history data of the data, and the environment when the data is generated[2], provenance is suitable for detecting intrusions[3], restoring systems[4], file searching[5] etc.using the provenance,user can quickly find the correlation and difference between the files. So provenance can be a kind of way to solve multi-version documentation problems.

Therefore, this article provides a method for finding differences in files with high file similarity and multi-version files. By using provenance and file comparison technology, user can quickly and efficiently obtain the difference between multiple versions of files. In the case of a large files, the provenance is used to directly obtain the difference of the file data, and the content comparison method is used when the file is small. Those menthod can Improve work efficiency.

2. Related Work

2.1 System library

Due to the popularity of personal computers, the files being modified constantly, and the continuous updating of application software often leads to a large number of similar files in personal computers. With the delay of time, most people often have difficulty distinguishing the gap between two files. When users need to obtain a document, they may get a bunch of similar files and it is difficult to distinguish the difference between them, which resulting in inefficiency. Currently, the operating system provides its own way to help organize the query of personal documents, such as Windows OS provides search function, which according to the file name., file type and the creation modification time, Similarly, Finder in MacOS X has the same function. However, in most cases, users only remember inaccurate attributes. Similar creation time and similar file names , those informations do not get access to the differences between files. Which caused efficiency between file queries. Unix Diff is another tools.which can quickly compare the differences between two files. However, this function also has certain limitations. It can only be used to compare documents. It cannot be used to compare other file formats. Similarly, it is only applicable to small files. When the file is too large, not only will the time increase dramatically. There is also a risk that the memory is not enough to compare.

2.2 P-index.

P-index[6] is a tool for finding data files based on provenance. Its functions are divided into two kinds of accurate search and fuzzy search. Accurate search is based on the file name and time to determine a file content, but the authors suggest that in real life, users can only give a fuzzy information. Instead of the accurate time, file information. According to this feature, the author proposes to use the provenance to obtain the correlation information between the files. After the user queries the approximate range, the P-index provides the file information with strong relevance to the user, which can greatly improve the efficiency of file search. This method can effectively improve the accuracy of finding files.but for multi-version files, although the file information can be provided, the difference of the files cannot be obtained at one time,it is difficult for the user to figure out the file they want immediately in a plurality of highly correlated files. P-index unable gat access to the comparison of file differences.

2.3 Dac-Man.

Dac-man[7] is a tool for finding differences between data, but it is suitable for high-performance distributed cluster computing environments. It scans all the files when it load and then compares the data to find the differences between the data and cache them all at once in the disk. Since initialization requires one-time scanning and caching of all files, this method is costly and the initialization time is long. However, in high-performance computing clusters, where data is numerous. If you compare the differences between files when needed, it takes a lot of time. but the author caches the difference contents of the version files at one time, you can directly obtain the version file differences. It has a great time advantage in the subsequent queries.

3. Design

This paper proposes a content variability method based on provenance. The current method of differential detection mainly relies on direct comparison to obtain version differences. This method has high efficiency when the data is small, but when the content of the file is gradually increased and the amount of data is increased, it has a very High latency, comparison directly by using diff is also only for text files, and is not supported for files in other formats. So we introduce the provenance data. The provenance data can collect the history of the data and easily obtain the change information of the file data, but the current provenance data is collected at the Linux kernel. The provenance data will inevitably collect other relevant environmental information. As a result, the data overhead is huge. Research[8]shows that when there are many small files, the provenance data can be more than 10 times larger than the provenance data, and when the single data is large, the provenance data is

relatively less expensive. Therefore, when there are many small files, the time spent collecting and the space overhead are huge. We choose to ignore the provenance data of small files. When there are more large files, the overhead is relatively small. We collect the provenance data. By combining the direct comparison of provenance data and files, we can get our method of difference in provenance content. This article extends the data provenance collection module. By adding a hook module in the Linux kernel, the system process is intercepted, and the read and write operations are intercepted, and the process of adding and subtracting files is obtained and output to the database for storage. As shown in our document difference framework. Our framework is mainly divided into four parts.see Fig. 1.

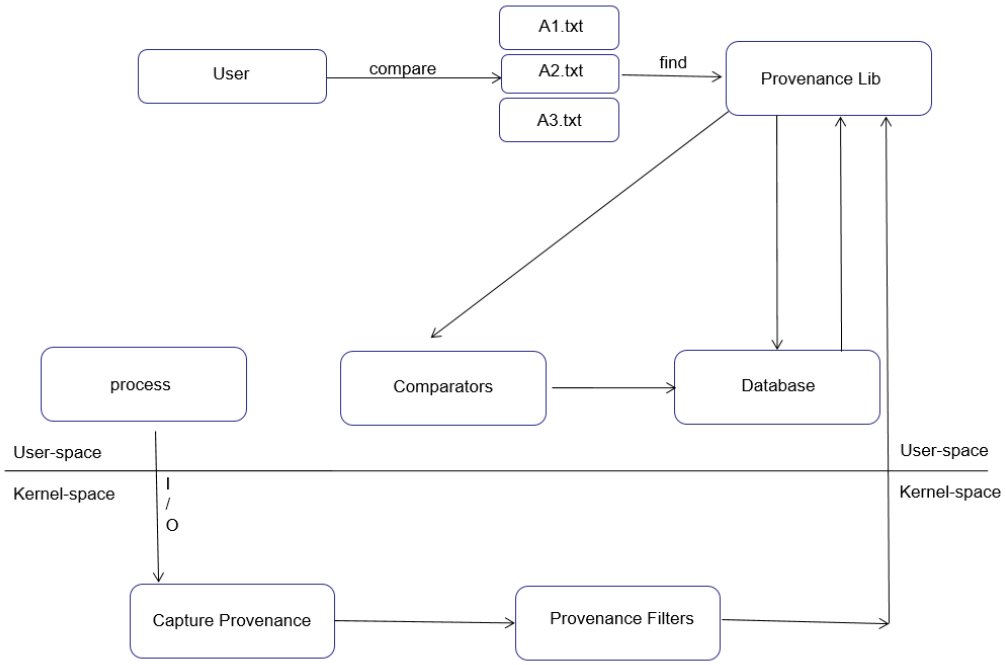

Fig. 1 Difference comparison framework based on provenance

3.1 Provenance library.

The Provenance library is consisted with indexes. The role of the index is that when user need to compare multiple files, The Provenance library can query whether there is file difference information in the database. If there is relevant information, it can be obtained directly in the database. If not, the files need to be compared, and then the comparison data is stored in the database, and the file information is added to The Provenance library. The cache is persistent because the results are saved on disk. Hence, it is only limited by the available space of the filesystem. They are never deleted except if the user explicitly clears the cache

3.2 Comparators.

Since the provenance data is filtered and does not collect all the provenance information of the small file, there is no corresponding difference data in the provenance library. Therefore, when there is no information to compare files in the provenance information database, it is necessary to compare the two files. And save the file label to The Provenance library and save the data information to the database.

3.3 Database.

The database is used to store information about file differences. When needed, it can be access directly.

3.4 Provenance filter.

The provenance information filter is used to filter data with too much provenance information such as small files to ensure low overhead of the system. The filtered data is compared using diff in

subsequent comparisons. This is because small files are less expensive in diff than provenance data. Since the direct comparison of diff can only be used for text format, but the provenance information can be used to compare other formats, such as executable files, we query the source code of the executable file through the provenance information, and compare the source code of the executable file. To determine the difference in the executable. This technology will be implemented in the future.

4. Evalution

In this section, we evaluate the performance of Difference comparison framework based on provenance (DCP) in the context of datasets. We evaluate our DCP with a computer server.It has Intel(R) Xeon(R) CPU E5-2403 0 @ 1.80GHz CPU , 8GB memory,500 GB hard disk with the system version is CentOS 7 linux 3.10.94.

We used two ways to test the workload of the framework. By compiling the linux kernel and using the filebench test tool. Compiling the kernel version is the kernel linux 3.10.94, which operates on multiple files to generate a large number of object files and executable files as well as documentation. Consume many CPU resources. Filebench is a tool for testing system performance. It can simulate the load of the server to test system performance. The experiment uses Fileserver, Varmail and Webserver .configuration see Table 1.

Table 1 Configuration of three different load

Load	Fileserver	Webserver	Varmail
File number	10000	1000	50000
Average file size	128K	16K	16K
IO size	1M	1M	16K
Load	Fileserver	Webserver	Varmail

We evaluate the system performance by using DCP. By testing the filebench, the experimental results are shown in Figure 2. The framework reduce both throughput and IOPS. This is because the information collected in the kernel makes IO path longer. The path reduces its performance, but the overall performance is reduced by no more than 10%. Since the provenance collected once, the subsequent comparison only needs to read the provenance in the database, which is acceptable. Similarly, for compiling the Linux kernel, under normal circumstances, the compiler system kernel needs 1464s, and in the DCP framework, the compiler kernel needs 1514s, which increases the time overhead by 3.4%. This paper considers that the cost of comparison with documents is equally acceptable. Since the source data is collected, the differences between the files can be obtained directly. It is faster to compare directly with diff, but it requires the cost of the system.

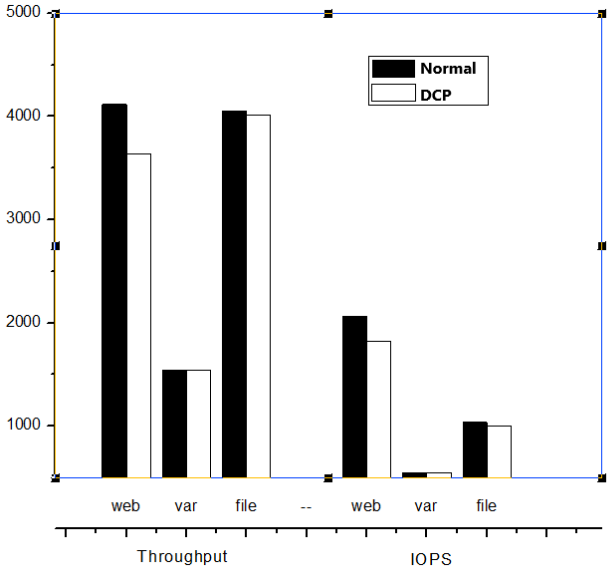

Fig. 2 Throughput and IPOS in DCP and normal

5. Conclusion

In this paper, we design DCP a version-differences detection tool based on provenance information to detect differences between multi-version documents. Compared to the traditional direct comparison of document content, this method has the feature of finding the difference of the version more quickly and at a lower cost. Ability to identify differences in multiple file formats. There are still many shortcomings in this paper. For example, the provenance data collects the whole system provenance information. It can be collected for a certain file or a certain format.

References

[1] Lyman, Peter & R Varian, Hal. (2003). How Much Information. Retrieved from http://www.sims.berkeley.edu/how-much-info-2003 on January 14th, 2005:2-3

[2] Buneman, P, Khanna, S., Tan, W.C.: Why and where: a characterization of data provenance. In: International Conference on Database Theory (ICDT) , 2001:316–330

[3] Xie Y, Feng D, Tan Z, et al. Unifying intrusion detection and forensic analysis via provenance awareness[J]. Future Generation Computer Systems, 2016, 61(C):26-36.

[4] Yamamoto K, Kuriyama T, Shigemori H, et al. Provenance Based Retrieval: File Retrieval System Using History of Moving and Editing in User Experience.[J]. 2011:618-625

[5] Shah S, Soules C A N, Ganger G R, et al. Using Provenance to Aid in Personal File Search.[C]// Usenix Technical Conference. 2007:171-184.

[6] Liu J , Feng D , Hua Y , et al. P-index: An Efficient Searchable Metadata Indexing Scheme Based on Data Provenance in Cold Storage[J]. 2015. 10.1007/978-3-319-27140-8_41

[7] Ghoshal D , Ramakrishnan L , Agarwal D: Dac-Man: Data Change Management for Scientific Datasets on HPC systems SC 2018: 72:1-72:13

[8] Jayapandian M , Chapman A , Tarcea V G , et al. Michigan Molecular Interactions (MiMI): putting the jigsaw puzzle together[J]. Nucleic Acids Research, 2007, 35:566-571.