1-22-2008

FARMER: A Novel Approach to File Access Correlation Mining And Evaluation Reference Model for Optimizing Peta-Scale File System Performance

Peng Xia
Wuhan National Laboratory for Optoelectronics, China

Dan Feng
Wuhan National Laboratory for Optoelectronics, China

Hong Jiang
University of Nebraska-Lincoln, jiang@cse.unl.edu

Lei Tian
University of Nebraska-Lincoln, tian@cse.unl.edu

Fang Wang
Wuhan National Laboratory for Optoelectronics, China

University of Nebraska–Lincoln, Computer Science and Engineering
Technical Report TR-UNL-CSE-2008-0001
Issued Jan. 22, 2008

FARMER: A Novel Approach to File Access Correlation Mining And Evaluation Reference Model for Optimizing Peta-Scale File System Performance

Peng Xia*, Dan Feng*, Hong Jiang#, Lei Tian*#, Fang Wang*
*Computer College, Huazhong University of Science and Technology, Wuhan, China
*Wuhan National Laboratory for Optoelectronics, China
#Department of Computer Science and Engineering, University of Nebraska-Lincoln
sharp@smail.hust.edu.cn, {dfeng,wangfang}@hust.edu.cn
{jiang,tian}@cse.unl.edu

Abstract

File correlations have become an increasingly important consideration for performance enhancement in peta-scale storage systems. Previous studies on file correlations mainly concern with two aspects of files: file access sequence and semantic attribute. Based on mining with regard to these two aspects of file systems, various strategies have been proposed to optimize the overall system performance. Unfortunately, all of these studies consider either file access sequences or semantic attribute information separately and in isolation, thus unable to accurately and effectively mine file correlations, especially in large-scale distributed storage systems.

This paper introduces a novel File Access coRrelation Mining and Evaluation Reference model (FARMER) for optimizing peta-scale file system performance that judiciously considers both file access sequences and semantic attributes simultaneously to evaluate the degree of file correlations by leveraging the *Vector Space Model (VSM)* technique adopted from the *Information Retrieval* field. We extract the file correlation knowledge from some typical file system traces using FARMER, and incorporate FARMER into a real large-scale object-based storage system as a case study to dynamically infer file correlations and evaluate the benefits and costs of a FARMER-enabled prefetching algorithm for the metadata servers under real file system workloads. Experimental results show that FARMER can mine and evaluate file correlations more accurately and effectively. More significantly, the FARMER-enabled prefetching algorithm is shown to reduce the metadata server latency by approximately 24-35% when compared to a state-of-the-art metadata prefetching algorithm and a commonly used replacement policy.

1 Introduction

Exploiting file and block correlations to benefit performance has become an increasingly common practice in the design and optimization of intelligent storage systems today. Most related studies focus on extracting semantic knowledge (including file and block correlations) to guide and facilitate various performance enhancing strategies (such as prefetching, caching, data layout and security-awareness, etc.). Representative studies that exploit semantic knowledge to enhance storage system performance include Active-disk [2] , Self-* storage [3], Semantically-Smart Disk System (SDS) [4], Object-Based architecture [5], etc,. Moreover, the file system level can provide more useful and insightful information about access sequences and semantic attributes (e.g., process id, user id, application, metadata, and certain file properties) than can the block level because of the elaborate and rich I/O interface between storage applications and file systems. Therefore, mining file correlations can be very beneficial for exploiting application semantics and has been widely used for performance optimization in file systems. Unfortunately, it is nontrivial to explore semantic knowledge in file systems effectively and accurately because various factors affecting this knowledge exploration may be intricately related with one another as demonstrated later in this paper (Please see the Section 2 for details)

The main approaches to mining file correlations can be classified into two categories: access sequence mining and file semantic attribute mining. By tracing file system activities, several studies [6, 7, 8, 9, 10, 11] show that file accesses are strongly correlated to their preceding ones.

On the other hand, by extracting semantic attributes from file systems, semantic attribute mining approaches [12, 13, 14, 15] provide flexible associative accesses to documents, programs, object codes, images and other files contained by the system automatically. Recently, Daniel Ellard et al. [13] presented a very interesting method to infer the correlation between semantic attributes and file properties by using a decision tree technique. Nevertheless, this approach is limited to predicting certain file properties (e.g., read-only, size, etc.) from semantic attributes.

File correlations are typically more difficult to mine, and

thus richer, than block correlations because the former are impacted or co-impacted by far more factors than the latter, as a result of the interface and interactions between applications and the file system that are far richer than those between data blocks and storage devices. In such complex interface and interactions, storage applications perform file operations with various indications of access sequences and semantic attributes. Further, while file correlations are richer than block correlations, the former can also have more negative impact if incorrectly inferred than the latter because block correlations are typically inferred through I/O scheduling and block access patterns that are relatively deterministic. Moreover, our preliminary experimental studies also indicate that these two aspects are mutually influencing factors on file correlations. Therefore, we are led to firmly believe that access sequence mining alone, without combining the benefit of semantic attributes, can not fully reveal file correlations, especially in large scale distributed storage system, while semantic attribute mining without considering access patterns is equally inadequate to infer file correlations.

Unfortunately, all of the existing methods for mining file correlations either solely rely on file access sequence or only take into account semantic attributes in isolation, thus possibly failing to fully exploit the potentially important correlation between access sequences and semantic attributes that in turn may reveal more accurate file correlations. This motivates us to propose a more powerful mining approach in this paper, called a File Access coRrelation Mining and Evaluation Reference model (FARMER), that can discover more complex file correlations by judiciously and effectively combining access sequence mining with file semantic attribute mining. FARMER takes into account both file access sequences and semantic attributes simultaneously to evaluate file correlations and uses a directed, weighted correlation-graph to capture file access correlations. FARMER's correlation graph is weighted with the file correlation degree that is evaluated by the *Vector Space Model (VSM)* technique adopted from the *Information Retrieval (IR)* field [17, 18].

In this paper, we begin by presenting statistical evidences from several typical distributed file system traces to indicate that access sequence and semantic attribute have strong collective and joint impact on file correlations. We also incorporate FARMER into a practical platform - HUSt [16] to validate our designed goal that FARMER is a useful and efficient tool to infer file correlations with reasonable overhead. We utilize FARMER to improve the intelligence in a metadata prefetching algorithm and optimize the file layout in HUSt. Other potential applications of FARMER such as security, reliability and consistency are also discussed and pointed out as our future work. Furthermore, we conduct extensive experiments to determine which semantic attributes or combinations of semantic attributes provide positive contributions and which others provide negative ones. Based on these experimental results, our FARMER is shown to more effectively and accurately mine and evaluate file correlations than existing evaluation algorithms more effectively and accurately. More significantly, the FARMER-enabled prefetching algorithm is shown to reduce the metadata server latency by approximately 24-35% when compared to a state-of-the-art metadata prefetching algorithm and a commonly used replacement policy.

The remainder of this paper is organized as follows. In the next section we briefly discuss further motivation for our research and provides some background information.. In section 3, we present our file correlation mining and evaluating model to infer file correlations. Section 4 discusses how to take advantage of file correlations revealed by FARMER for several potential applications. In Section 5, we introduce a case study that utilizes FARMER to a real storage system to improve the intelligence in prefetching and discuss our experimental results. Section 6 reviews representative research works in the literature that are more relavant to our proposed work and Section 7 concludes the paper.

2 Motivation and Background

Access sequence and semantic attributes are the two best known factors influencing file correlations. We have reasons to believe that these two factors are strongly correlated and by judiciously combining them we can more effectively and accurately mine file correlations. For example, when a user executes *gcc* to compile a set of source files, it will generate the object and executable files for the corresponding source files. The interesting fact is that files are often generated in the same access sequence and eventually deposited to the same directory. It is intuitive from this example that there are strong correlations among these source files with hints provided by user id, program id, access sequence and directory information. Therefore, it is possible and necessary to utilize the hints provided by a combination of file attributes and access sequences, such as those in the above example, to improve the accuracy of inferring correlations between the source files.

In this section, we present a discussion on some intuitive and statistical evidences to illustrate the effectiveness of integrating these two factors of file to infer file correlations and thus further motivating our research.

2.1 Intuitive Scenarios

Initially, we can consider the following intuitive scenarios that provide hints to file correlations:

- Files accessed by the same user tend to have strong correlations, because each user has an access domain in which files possess strong correlations.
- Individual program typically access the same files in the same order, and thus files invoked by individual

programs tend to have strong correlations.
- It is common for a user to deposit related files in one specific directory, thus leading to strong correlation among files stored in the same directory.
- A frequent access sequence typically indicates that the involved files are frequently accessed together. Thus, files belonging to one frequent access subsequence tend to have a strong correlation.

The above scenarios include such hints as frequent access sequence that belongs to the access sequence factor, and the user, program and directory information that belong to the semantic attribute factor. All of these hints can be used to infer file correlations. Furthermore, since semantic attributes can be used to filter out unrelated access sequences to narrow the mining scope and improve the precision of inferring correlations among files, files with equal or similar statistics obtained from their corresponding hints/factors are most likely to be strongly correlated.

2.2 Statistical Evidences

Although above four scenarios are intuitive, they indicate that both access sequence and semantic attributes can apparently be associated with file correlations, which inspires us to conduct further experiments to verify this association with real file system traces.

We analyze four typical traces – LLNL, INS and RES, and HP, taken from different distributed file system application environments:

- **LLNL trace** [19] traces several typical parallel scientific applications, which have heavy I/O demands with data accesses of varying size. The LLNL trace was collected from a large Linux cluster with more than 800 dual-processor nodes at the Lawrence Livermore National Laboratory (LLNL). It consists of 6403 trace files with a total of 46,537,033 I/O events.
- **INS Trace** and **RES Trace:** Drew Roselli and Thomas Anderson [20] traced two groups of Hewlett-Packard series 700 workstations running HP-UX 9.05. INS was collected from twenty machines located in laboratories for undergraduate classes. RES was collected from 13 machines on the desktop of graduate students, faculty, and administrative staff of their research projects.
- **HP Trace:** The HP trace is a 10-day file system trace collected on a time-sharing server with a total of 500GB storage capacity and 236 users at the HP Lab [21].

Based on these collected traces, we quantify the association between file correlations and their possible influential factors (i.e., access sequence and semantic attributes). We keep track of access sequences for different semantic attributes separately, and then compute the probability of inter-file accesses within these different sequences.

The probability of inter-file access of a file *A* to another file *B* refer to the likelihood of file *B* being accessed given that file *A* has been accessed. This is also called file successor probability. Our observation shows that if the average access probability is large, the corresponding file correlation is strong. By contrast, if there is no association between file correlations and semantic attributes, the access probability tends to be independent of the semantic attributes. For example, if there is no association between file correlations and semantic attributes, the access probability when considering a semantic attribute say process P that is meant to 'filter out' unrelated file access sequence will not differ from the access probability when none of the semantic attributes is considered. Therefore, by comparing the probabilities of inter-file accesses for different sequences, we can quantify the influence on the association between file correlations and semantic attributes by different semantic attributes.

Figure 1 compares the probabilities of inter-file accesses for different sequence. There are three important observations drawn from this figure. The first observations is that the inter-file access probabilities due the same attribute in different traces are different. For example, in RES trace, the *pid* attribute corresponds to a 37.6% access probability, but a probability of 52.7% results in the HP trace for the same attribute. The second observation is that even in the same trace, different attributes lead to different inter-file access probabilities. For example, in the HP trace, the probability corresponding to the *file path* attribute (55.2%) is larger than that for the *uid* attribute (45.8%). The last observation is that when none of the attributes is considered, the access probability is the lowest in all the traces among all the cases considered.

From the statistical evidences we learn that we clearly stand to gain benefits from finding an appropriate method to judiciously combine the access sequence factor and the semantic attribute factors into an integrated scheme to mine and evaluate file correlations. We also learn that different attributes or attribute combinations have different influence on inferring file correlations.

Motivated by the above preliminary investigations and observations from experimental studies, we propose an integrated model for file correlation mining and evaluation, which is detailed in the next section.

3 FARMER

In this section, we provide the details of the proposed File Access coRrelation Mining and Evaluation Reference (FARMER) model to quantify and evaluate file correlations. We start with an architectural overview of the FARMER model and its mining and evaluating approaches, followed by a discussion on how to address the issues of building FARMER to mine file correlations. Finally, we analyze the validity and efficiency of FARMER.

3.1 The FARMER Architecture

As shown in the previous section, access sequence and semantic attributes collectively and jointly impact file correlations more profoundly than can either alone. Nevertheless, to the best of our knowledge, very few studies have been conducted to integrate access sequence and semantic attributes to infer file correlations since it is difficult to quantify semantic attributes and estimate the extent to which they impact file correlations.

The proposed FARMER model is composed of a four-stage process of *Extracting*, *Constructing*, *Mining* & *Evaluating* and *Sorting*, as shown in Figure 2. FARMER provides a "black-box" approach to inferring file correlations without any assumption and modification to the interface between applications and the file systems (storage front-end). Therefore, FARMER is general and independent of the front-end. More specifically, the *Extracting* module collects file request information; the *Constructing* module is deployed to construct a weighted and directed graph representing file access sequences; the *Mining* & *Evaluating* module, which houses the core mining algorithm of FARMER called CoMiner, is responsible for mining and evaluating file correlations; and the *Sorting* module organizes the quantified inter-file correlations appropriately to facilitate performance enhancing strategies that exploit file correlations obtained by FARMER. Our core algorithm *CoMiner*, embedded in the *Mining* & *Evaluating* module, mines and evaluates semantic attributes and access sequence cooperatively. By leveraging the *Vector Space Model (VSM)* [17] and similarity estimation techniques, *CoMiner* quantifies semantic attributes and evaluates file correlations. Based on the evaluation, FARMER can infer inter-file correlations with a relatively high degree of accuracy in distributed storage systems.

The functions of and workflow among the four FARMER modules are elaborated below:

- **Stage 1: *Extracting*.** In this stage, we collect file attributes such as timestamp, file name, user, group, program information, etc. by extracting from each file request. A set of such attributes that identify a certain file request pattern help mine and evaluate file correlations effectively and accurately.
- **Stage 2: *Constructing*.** Once the appropriate file attributes are obtained, a weighted, directed correlation-graph is constructed to represent file access sequences. This graph consists of a set of directed edges and a set of nodes, where a node represents an accessed file and a directed edge that starts from a predecessor node and ends at a successor node represents an access order. The weight on each edge equals the value of correlation degree between the predecessor and the successor.

  If a newly requested file is already in the graph, only the inter-file access count is increased and the file correlation degree is updated accordingly. Obviously, if the weight on an edge is large, the corresponding nodes (files) are likely correlated. Therefore, this graph representation is able to capture access sequences and mine file correlations.
- **Stage 3: *Mining* and *Evaluating*.** This stage contains the core of FARMER. After Stage 1 and Stage 2, the file attributes and frequent sequence information between the current file and its successors have been extracted from the file requests. FARMER then mines and integrates this information to evaluate file correlations.

  The file correlation degree obtained through mining and evaluation for each file-successor pair is recorded in the corresponding *Correlator List* that is associated with each file having one or more successors. In addition, this list contains relevant information extracted in Stage 1 and 2 of certain successors of the current file and is indexed by these succesors' file IDs. Subsection 3.2 presents in details the FARMER core mining and evaluating algorithm, CoMiner, which also includes a description of the Correlator List and it's update operations.
- **Stage 4: *Sorting*.** In this stage, the *Correlator List* for each file is sorted and organized by file correlation degree. Consequently, each file with one or more successors is associated with a sorted *Correlator List* in deceasing order of the inter-file correlation degree from head to tail. Thus, if a file is closer to the head in the *Correlator List*, there will likely be a stronger correlation between the file and its owner.

This is an iterative process that repeats itself for each incoming request. Therefore, it is possible to infer file correlations by automatically and dynamically mining and evaluating the semantic attributes and access sequence information contained in the requests.

3.2 The Core Mining and Evaluating Algorithm - *CoMiner)*

Existing file correlations mining algorithms include semantic attribute mining and access sequence mining. These two approaches focus on the statistical analysis for access history information in their respective fields and mine these hints by using various data mining techniques to infer file correlations. However, their effectiveness is limited by either the lack or difficulty of integrating the hints of semantic attribute and access sequence to evaluate and quantify file correlations.

To combine semantic attributes with access sequence to maximize the efficiency and accuracy of inferring file correlations, we propose *CoMiner*, a method that leverages the state of the art approaches including the *Vector Space Model (VSM)* techniques in the *Information Retrievel* area to mine and evaluate file correlations quantitatively in storage systems. In particular, in order to estimate the similarity of

semantic attributes, we utilize *Semantic Distance* to denote how far apart two files are semantically in the correlation-graph. We also identify the validity of file correlations by specifying an appropriate threshold.

CoMiner mainly consists of three steps: (1) Mining and quantifying similarity of semantic attributes and access frequency; (2) Evaluating file correlations by using similarity of semantic attributes and access frequency; and (3) Filtering out weak or false file correlations. A pseudo code describing the process of *CoMiner* is presented as Algorithm *CoMiner*.

Algorithm 1 *CoMiner*

Parameters:
Semantic Distance between files sd
access frequency f
file correlation degree e
Input:
A request file f,
successor files of request file $successor$,
valid threshold $max\_strength$
Output:
Correlator List l
for $i = 0$ to $i < successor.length$ **do**
  compute sd
  compute f
  compute e
  if $e > max\_strength$ **then**
    $l \leftarrow pair(successor[i].id, e)$
  end if
  $l.sort$
end for

3.2.1 Semantic Attribute Mining

File semantics can be exposed at various levels, such as definitional, associative, structural, behavioral, environmental level or through other information related to the files [23], where various hints can be obtained. Many of these hints can help improve the precision of inferring file correlations. *SD graph* [9] presents *Semantic Distance* and attempts to use this concept to estimate the degree of similarity between two files. However, effectiveness of *Semantic Distance* in *SD graph* is limited to only exploiting access sequence, thus failing to quantify the rich semantic attributes that can potentially improve the similarity measurement between files.

An approach, called *Vector Space Model (VSM)*, is widely and successfully deployed in the area of information retrieval [17] for text representation and searching. Inspired by *Vector Space Model (VSM)*'s successes, we believe that it can be deployed in our CoMiner to estimate file correlations accurately. In *Vector Space Model (VSM)*, a vector represents a text document and basic vector operations are used to compute similarity between two documents. In adopting *VSM* to our *CoMiner*, vectors are used to represent files and similarity estimation algorithms are used to quantify the similarity between semantic attributes that can be extracted from a set of metadata attributes representing a file [16].

More specifically, a vector represents a file and each item of the vector represents one particular attribute of the given file. Vectors are stored as columns of a single matrix. Then basic vector operations can be used to evaluate *Semantic Distance* between files. The computation of *Semantic Distance* is based on a common similarity computation function:

$$sim(A, B) = \frac{|A \cap B|}{|\max(A, B)|} \quad (1)$$

In this function, set *A* and set *B* represent *semantic vectors (SVs)* spaces of files, $sim(A, B)$ represents the semantic distance of two *SVs*. Let $A = \{A_1, A_2, \ldots, A_n\}$ and $B = \{B_1, B_2, \ldots, B_n\}$. Table 1 demonstrates how to transform file semantic entries to semantic vectors. Here, a semantic vector item A_x corresponds to one attribute, such as the user id. Therefore, semantic distance of files can be defined by $sim(A, B)$.

User	Process	Host	File Path
$user1$	$p1$	$host1$	$/home/user1/paper/a$
$user1$	$p2$	$host1$	$/home/user1/paper/b$
$user2$	$p3$	$host2$	$/home/user2/c$

$$\Downarrow$$

$A = \{user1, p1, host1, home, user1, paper, a\}$
$B = \{user1, p2, host1, home, user1, paper, b\}$
$C = \{user2, p3, host2, home, user2, c\}$

Table 1: transform file semantic entries to semantic vectors.

Divided Path Algorithm (DPA) vs. Integrated Path Algorithm (IPA). According to Function 1, we can compute the semantic distance between files. All semantic attributes except for the file path attribute, present their corresponding values directly; To handle the more complex file path attribute, two methods can be used compute the *Semantic Distance*. The first one is to parse a full file path to several subdirectories. Each subdirectory is represented as one item in *SV*. We call this method *Divided Path Algorithm (DPA)*. The other approach is to regard the entire file path as one item, which is called the *Integrated Path Algorithm (IPA)*. Table 2 depicts the application of Function 1 to calculate the *Semantic Distance* in *DPA* and *IPA* respectively. The value contained at the intersection between the vectors *A* and *B* acts as the numerator of the function. The denominator is the max count of items of *A* or *B*.

Divided Path Algorithm	Integrated Path Algorithm
$sim(B, C) = 1/7$	$sim(A, C) = 0.25/4$
$sim(A, C) = 1/7$	$sim(B, C) = 0.25/4$
$sim(A, B) = 5/7$	$sim(A, B) = 2.75/4$

Table 2: *DPA* vs *IPA*

Table 2 demonstrates how *DPA* and *IPA* can be used to compute *Semantic Distance* among files. For *DPA* on the left, each subdirectory or attributes is represented as one item in a vector. Therefore, $\max(A, B)$ is 7. Moreover, 5 attribute items between *A* and *B* are the same, which means that $A \cap B$ equals 5. For *IPA* on the right, the similarity of directories is computed first. For A and B, the maximal count of subdirectories is 4, and the intersection of directories is 3. Thus, the directory similarity is $3/4 = 0.75$. Then, since the entire file path is regarded as a single item, $\max(A, B)$ is 4 and $A \cap B$ is 2.75.

However, the drawback of the *DPA* algorithm is that if files are in a deep directory, the directory attribute becomes the main influential factor in determining the result while other attributes such as process and user IDs are significantly weakened. As a result, file correlations that are actually strong by virtue of the weakened attributes will be considered weak and thus filtered out. For example, consider A that is an executable file and B that is a library file linked by A. Although the intersection of file path between them is null, their correlation is nevertheless very strong. The file correlation degree of file A and B computed by *DPA* will surely be smaller than the specified *max strength* (threshold), thus resulting in file B being removed from file A's *Correlator List*. Based on this analysis, we decide to use the *IPA* algorithm to compute the *Semantic Distance*.

3.2.2 Access Sequence Mining

There is pronounced regularity in file access sequence, a well-known observation in file systems. This observation can help discover file correlations between two or more files. *Probability Graph* and *Semantic Distance (SD)* keep frequent counts of file accesses that follow within a window of a specific length. In these two techniques, all the successors of a file are assigned the same importance. The limitation of this approach is that it fails to distinguish the importance of the successors which have different access distance. To overcome this limitation, we apply the *Liner Decremented Assignment algorithm* [11] to count the inter-file access number, where the farther the access distance between file and its successor is, the weaker their file correlation will be.

CoMiner keeps track of the predecessor's, successor's information of a file, and computes the access frequency $F(A, B)$ for a pair of files. Here, $F(A, B) = N_{AB}/N$, where N_{AB} is the number of times that file *B* is the successor of *A*. *N* is the total access count of file *A*. Thus, $F(A, B)$ represents the frequency of accesses in which file B is a successor of A. A high value of $F(A, B)$ means that if access to file A occurs, file B is very likely to be accessed soon. So, we use $F(A, B)$ to describe file correlations. For example, given an access sequence of *ABCD*, *B* is a closed successor of *A*, 1 will be added to N_{AB}. For *C* and *D*, their access distance from *A* are 2 and 3 respectively and, according to the linear decremented assignment the additional value is 0.9 for *C*, and 0.8 for *D*.

3.2.3 Computing File Correlation Degree

The file correlation degree for files x and y is defined as:

$$R(x, y) = sim(x, y) \cdot p + F(x, y) \cdot (1 - p) \qquad (2)$$

In Function 2, $sim(x, y)$ and $F(x, y)$ represent semantic distance and access frequency between x and y respectively. p is a tunable weight that can be adjusted to an appropriate value to judiciously combine semantic distance and access frequency to more effectively exploit inter-file correlations.

Semantic distance is a function of several semantic attributes such as user id, process id, host id and file path. Once these attributes are determined, they are rarely modified. However, access frequency varies with access count of a file and its successors. So, the access count information for each file and its successors must be updated in time to compute the latest access frequency.

3.2.4 Threshold for Valid File Correlation Degree

An important issue to consider is the threshold for valid file correlation degree. If the correlation degree between two files is very small, such a correlation may not be valid in that the two files may only occasionally inter-access and their correlation, if present at all, may merely be random and offers very little to be exploited for performance improvement. For example, two otherwise unrelated files may belong to a random access sequence, with a file correlation degree of 0.0001 as evaluated by FARMER. This correlation degree is so weak that it is generally meaningless to be considered for an exploitable file correlation. Therefore, in order to describe the validity of file correlations, we define *strength* to measure the file correlation degree. Moreover, we specify a valid threshold for correlation degree, denoted as *max_strength*. After evaluating the correlation degree, we compare the file correlation degree of a candidate file with *max_strength*. If the file correlation degree is smaller than *max_strength*, this correlation will be considered invalid and thus filtered out, and vice versa.

3.3 Efficiency of FARMER

Compared with existing file access sequence mining algorithm such as *SD Graph* , *Probability Graph* and Nexus, FARMER can more effectively and accurately infer file correlations in distributed storage systems while requiring less overhead. These algorithms need to keep the correlative information for every file during the process of graph building, whereas FARMER does not need to maintain any correlative information for weak correlations due to its filtering ability. In practice, only active file correlations are updated and thus FARMER needs much smaller memory footprint to store file correlations information. Moreover, in terms of time complexity, FARMER is more efficient than the approaches

mentioned above because less correlative information needs to be processed. Therefore, we argue that FARMER is much more efficient than these existing algorithms.

4 FARMER Applications

FARMER as a mining and evaluating tool can infer and reveal hidden file correlations that can be potentially used and exploited by a number of performance enhancing strategies directly or indirectly, as explained in more details in this section.

4.1 FARMER-enabled Prefetching

The file correlation information mined and evaluated by FARMER can be used to help prefetch files data, especially file metadata accurately in large distributed storage systems. It is a well recognized fact that metadata accesses and operations account for a majority of all I/O operations in a typical storage system [20], because hundreds of thousands of pieces of file metadata need to be updated simultaneously, which often means that metadata servers are severe performance bottlenecks of distributed storage systems. This metadata bottleneck has been addressed generally in two directions. The first is to use multiple metadata servers to coordinate the metadata requests to metadata servers for load balancing. The second is to reduce the overhead incurred by metadata operations by improving storage cache hit ratio. FARMER helps exploit inter-file correlations and offers an enhanced prefetching algorithm to reduce the overhead incurred on metadata servers and effectively alleviate the bottleneck effect of metadata servers.

In general, a conservative prefetching algorithm attempts to avoid prefetching inaccuracy and cache pollution by reducing the frequency and amount of prefetching. By contrast, an aggressive prefetching algorithm prefers to hoard more entries to scale up overall system performance. In modern storage systems, more than 50% of all I/O operations are related to metadata access [20] while the typical size of file metadata is no more than 5% of the size of file data [24]. This observation implies that storage systems stand to gain greatly by aggressive metadata prefetching while incurring relatively small mis-prefetching penalties. However, the benefit of aggressive prefetching can be quickly offset by mis-prefetching penalties if it is not accurate and brings in too many unrelated metadata files. To alleviate this problem, FARMER filter out unrelated or weakly correlated files from *Correlator List* by comparing the correlation degree with a valid correlation degree threshold *max_strength*. By appropriately configuring the *max_strength* threshold, FARMER can potentially optimize the prefetching size at a minimal mis-prefetching penalty.

The capability of metadata operations for metadata server plays a critical role in scaling up performance in a peta-scale distributed storage system. We identify demanding requests and prefetching requests by setting a request attribute and provide a priority-based request-scheduling model, as elaborated in Section 5. In particular, a metadata server uses two request queues to guarantee the availability of service for the demand requests queue that is of higher priority than the prefetching request queue.

4.2 FARMER-enabled File Data Layout

File correlations can also be exploited to improve the efficiency of file data layout. We can merge several small files into one group to scale up the overall system performance by enhancing the correlative file data locality. The average file size of modern workstation cluster is 108 KB - 189 KB [24], so file data layout has a great impact on the batched I/O operations that, as a result of exploiting file correlations and thus data locality, are transformed from random I/Os to sequential I/Os, thus significantly improving data access performance.

Several design issues should be considered while exploiting file correlations to optimize file data layout. One of the most important issues is to determine which files should be integrated into one group. We can use the sorted *Correlator List* of each file to address this issue. However, if file data are frequently modified, the data layout management of such a grouping scheme will become very complex. Therefore, as an initial attempt, only read_only files are considered to be stored in the same group.

Metadata servers can organize files based on inter-file correlations and file attributes. After evaluating and sorting based on file correlation degree to obtain a group of strongly correlated files in the *Correlator List*, a metadata server may try to allocate these files in one group contiguously. Thus, whenever the predecessor is accessed, its correlated files are batch read into the cache by a single I/O request.

4.3 FARMER-enabled Security and FARMER-enabled Reliability

File correlations can also be exploited to improve storage system security and reliability, for example, in cases such as secured delete and denial of malicious access [25]. In intelligent secure storage systems, once a user configures rule-based accesses for a file or directory, this rule may be applied to other files that have strong file correlations with this file or directory automatically. In addition, file replication and the corresponding consistency management can also take advantage of file correlations by grouping files with strong inter-file correlations in the same logical replica group. Each backup and recovery task on a replica group can be an atomic operation so that we can guarantee the strong consistency of files in the same replica group.

5 Case study: FARMER-enabled Prefetching for Improved Accuracy on HUSt

We have discussed several useful potential applications of exploiting file correlations mined and evaluated by FARMER in the previous section. Here, we apply FARMER to improve the intelligence of the prefetching algorithm in our object-based storage system – HUSt to verify the feasibility and effectiveness of our algorithm. In this section, we will present the FARMER framework and how it works with our system. To evaluate the benefits and overheads of the FARMER-enabled prefetching algorithm (FPA) and demonstrate FARMER's applicability to a wide range of workloads, we use four typical traces taken from distributed file system (including LLNL, INS, RES, and HP traces). Based on the four distributed file system traces (see Section 2 for more detailed descriptions of these traces), we conduct experiments to show the impact of FARMER on performance. We compare FPA with Nexus (because Nexus performs better than any of the existing algorithms for metadata prefetching [11]), and LRU (because LRU is the most commonly-used algorithm for cache replacement) in terms of the cache hit ratio, prefetching accuracy and average response time.

5.1 The HUSt System Architecture

Figure 4 shows the architecture of HUSt with FARMER integrated. The system comprises three major components: (1) **MDSs** are mainly responsible for managing files' metadata information and security authorization. The metadata information of files and objects are stored in the Berkeley DB. (2) **OSDs** are actual storage depositories for object data, and provide the object-based interface for clients' accesses. (3) **Clients** runs applications and provide general access interfaces for applications.

In order to support FARMER, we have added two major components into HUSt: *extractor* and *mining and evaluating utility*. *extractor* is a file-type specific filter that takes as input the request for a file from a client and outputs the corresponding semantic vector of this file. The functionality of *mining and evaluating utility* that implements the *CoMiner* is to mine and evaluate file correlation according to semantic vector. The *mining and evaluating utility* also interacts with the Berkeley DB to store the file correlation information such as *Correlator List* for the files.

5.2 Impact of FARMER Parameters on Design Decisions

Through analyzing experimental results obtained from the HUSt prototype implementation of FARMER, we evaluate how various FARMER parameters impact some design decisions.

5.2.1 Weight Factor p

To find out what value of weight p can achieve the best performance, we conduct some experiments to evaluate the performance of the overall system as a function of p (varying from 0.0 to 1.0 with step of 0.1). Figure 3 shows the cache hit ratios of the FARMER-enabled prefetching algorithm as a function of the *max strength* with different p values of 0, 0.3, 0.7 and 1, respectively. We notice that when the weight factor p is 0.7, the cache hit rate reaches the highest value. So, the default value of p is configured 0.7 in our subsequent experiments.

5.2.2 Attribute Combination

In section 2, analysis of different attribute combinations demonstrates that not all of file attributes have the same effect on file correlation. By analyzing the experimental result, we can determine attribute combinations that are more effective than others and potentially identify the most influential combinations.

In Table 5, the first column enumerates all combinations of the four attributes (User ID, Process ID, Host ID and File path) in the HP Trace, the third column enumerates all combinations of the four attributes (User, Process, Host and File ID) in the INS and RES trace. The second, fourth and fifth columns show the cache hit ratios associated with different attribute combinations. The last row presents the result when considering all the semantic attributes. From Table 5, we observe that the difference of cache hit ratio among different attribute combinations range from 0.1% to about 13%.

This result proves our conjecture that different attribute has different contribution to file correlation evaluation. Therefore, we can adopt a optimum semantic combination to improve prefetching accuracy and cache hit ratios.

5.2.3 Valid threshold

Valid threshold – *max strength* may affect the prefetching policy. We run the prefetching algorithm experiment under the HP traces. In this experiment, two files, with their file correlation degree larger than the validity threshold, will be prefetched to the system cache. The range of *max strength* varies from 0.0 to 1.0. A larger *max strength* corresponds to more a conservative prefetching policy. From Figure 6, we can see that if the *max strength* is smaller than 0.4 the response time tends to be stable.It indicates that prefetching files with file correlation degree lower than 0.4 is unlikely to benefit overall system performance.

5.3 Performance Evaluation of The FARMER-enabled Prefetching Algorithm

Aggressive prefetching algorithms can improve cache hit ratio and reduce response time by increasing the prefetching size, provided that prefetching accuracy is reasonably high. With poor prefetching accuracy, however, this ap-

proach can suffer from severe cache pollution and misprefetching penalty, making it ineffective and even counterproductive. The FARMER-enabled prefetching algorithm (FPA) improves prefetching accuracy significantly by eliminating prefetching candidates with low inter-files similarities.

Figure 7 shows that FPA has the highest prefetching accuracy under all traces when compared with Nexus and LRU. In particular, the cache hit ratio of FPA is 13% higher than that by Nexus in the HP trace. This improvement is the best among all traces (7.8% in INS and 3.1% in RES). The reason is that, in the HP trace, besides the basic information about requests (user id, process id, device id and so on), full file path information is also included, which enables FPA to more accurately mine and evaluate file correlations. However, in INS and RES, the fields of fid and dev id are used to identify the different location between the files. The INS and RES trace lack the file directory information that is critical in identifying locality and inter-file correlations effectively.

Trace	Prefetching Accuracy
FARMER	64.04%
Nexus	43.04%

Table 3: *Prefetching Accuracy for HP Trace*

We conduct our experiments on the HP trace to compare the prefetching accuracy. From Figure 4, experimental results show that about 65% of all predictions provided by FPA are correct. In contrast, Nexus' predictions are only about 43% correct. The higher prefetching accuracy of FPA translates into significantly reduced metadata access latency (i.e., average response time), as shown in Figure 8, where FPA can improve the average response time in metadata server over Nexus by up to 24% and over LRU by up to 35%.

5.4 Memory Overhead

Table 4 shows the overhead for FARMER processing the file request with different traces. The results show that additional memory footprint sizes for corresponding traces are no more than 100 MB. The reason is that, first, valid threshold *max strength* limit the size of *Correlator List*, thus FARMER only need to maintain a few members for each *Correlator List*. Second, only several additional entries such as file'IDs, *Semantic Distance*, file correlation degree, etc., are required to recorded for active files. Therefore, FARMER can effectively discover file correlations for different typical workloads.

Trace	LLNL	INS	RES	HP
Space (MB)	98.4	1.4	2.5	9.8

Table 4: *Space Overhead (max_strength is 0.4)*

6 Related Work

Data and file correlations can be exploited to improve the performance of distributed storage systems. We investigate previous works about block correlations and file correlations to provide the necessary background for our study reported in this paper. This section briefly discusses the most closely related and representative works in the literature on exploring correlations at the block or file level respectively.

Block correlations: Researches on block correlations has typically been conducted at the disk block abstraction level, obtaining more complex semantic patterns at the block level in storage systems.

C-Miner [26] uses data mining techniques to mine the frequent sequences from a set of short sequences, which in turn infers the correlations between blocks and presents approaches of block correlation-directed prefetching and data layout. C-Miner uses a *black-box* approach that is similar to FARMER's, although HUSt on which FARMER is prototyped provides an object-based interface to render it a *white-box* approach.

Sivathanu et al. proposed semantically-smart disk systems (SDS) [4] by using a *gray-box* approach. SDS exploits the on disk data structure information and categorizes data to transparently improve performance or enhance functionality beneath a standard block read/write interface. Similarly, Type-Safe Disks (TSD) [27] expose the block relationship by a specific interface (special system call functions) between the file system and the underlying block device. TSD uses block correlation information to enforce active constraints on data access.

File correlations: Previous studies on file correlations entail analyzing file access pattern or extracting semantic information.

File access patterns, which reflect user behaviors, can infer correlations among files. It is widely observed that access patterns follow previous patterns with a high probability. Based on this observation, several prefetching algorithms, such as Last Successor (LS), First Successor (FS) and Recent Popularity [30], predict files that are likely to follow recently accessed files. However, in distributed storage systems with multiple users, multiple programs and multiple hosts, the interactions among the different users, programs and hosts can render such predictions inaccurate. File access patterns, we believe, are not impertinent to file attributes (such as program id, user id, host id, etc.).

FARMER considers such file attributes as user, program and host to estimate the inter-file access relationship. Program-based Successors (PBS) and Program- and User-based Last Successor (PULS) [22] also identify the relation-

ships between file access patterns and the programs/users accessing them. PBS and PULS apply the LS/LnS algorithms under the Program- and User-based conditions. However, in addition to program and user information, we consider other file attributes that influence file correlations in FARMER, since different users or programs may also follow a similar file access pattern in distributed storage system.

Semantic Distance (SD) [9], Probability Graph[10] and Nexus [11] attempt to evaluate file access relationships by means of a weight-based graph. In the graph, each node represents a file and a back-link represents a file access sequence. The initial weight value of the weight-based graph is determined intuitively, and it is increased according to the request sequence. There are two potential problems for this approach. The first problem is that when multiple processes execute concurrently, the file access sequence will be interleaved by the scheduler of OS, which will reduce the accuracy of the collected statistics. The second problem, particularly related to Nexus, is that it attempts to decrease the response time by increasing the amount of prefetching, which reduces the prefetching accuracy and generates significant cache pollution.

While FARMER also evaluates file correlations to facilitate a prefetching algorithm, it differs from Nexus in that our approach considers both access sequence and semantic distance derived from file attributes such as user id, process id and other semantic information. Moreover, our prediction algorithm guarantees that successors that are not up to the mustard will not be prefetched.

An MIT team [12] develop a semantic file system to automatically extract the attribute information from files and index the key properties of file system objects. Gifford and Jouvelot provide associative attribute-based access to the content of information storage system with the help of file type specific transducers. A transducer's function is to extract the attributes of files from the upper level user-interface. Other semantic file systems such as SFS [28] and HAC [29] have been proposed to support both name-based and content-based access to file objects, allowing users to organize their files by content and present them with alternative views of data through the concept of semantic directories.

[13] shows that the statistical evidence of attribute association and provides useful hints to the file system in the form of file names and other attributes so that the file system can successfully predict many file properties from these hints.

7 Conclusion and Future Work

Integrating a semantic-based methodology into file access sequence is a novel and effective way to discover file correlations. In this paper, we introduce a File Access coRrelation Mining and Evaluation Reference (FARMER) Model for inferring the file correlations.

Simply using file access sequence (weight-based or not) can not avoid many incorrect predictions, especially in a multiple-user and multiple-process environment. Compared to the existing studies, FARMER judiciously combining file access sequence mining and semantic attribute mining to effectively mine and evaluate file correlations by leveraging *Vector Space Model (VSM)* technique.

We apply FARMER into an practical platform - HUSt [16] as a case study. Based on several typical distributed storage system traces, our experimental results show that FARMER is a useful and efficient tool to infer file correlations with reasonable overhead. More specifically, by comparing the effectiveness of considering different semantic attribute combinations, the result shows shows that which attribute combinations are more useful to exploit the maximum benefit of system.

In our proposed evaluation scheme, we consider two factors: semantic distance and file access frequency. The trade-off between them is controlled by a weight parameter. Experimental results indicate that the best performance is achieved when the weight of semantic distance is set to 0.7, implying that the semantic factor plays a more significant role in mining file correlations.

Nexus and PBS are the special cases of FARMER-enabled prefetching algorithm (FPA). If the weight value is 0, FARMER is reduced to Nexus; If only the process or user attribute factor is considered in our similarity computation, FARMER reduces to PBS or PULS. As shown in Figure 7 and Figure, FARMER is shown to improve the cache hit ratio of Nexus by up to about 30% while reducing average response time by up to 24%.

Our study has several limitations. First, even though FARMER is a useful tool that can be incorporated into general storage system, our evaluation of FARMER-enabled prefetching algorithm is based on our object-based system. We are in the process of implementing FARMER as a library to provide for other storage systems. Second, in this paper, we have shown some intuitive and statistical evidences to illustrate that various attributes have the impact on file correlations and compared the influence of different semantic attributes or attributes combinations. Furthermore, multiple regression can be used to learn more about association between file correlations and attributes.

8 Acknowledgments

This work is supported in part by the National Basic Research Program of China (973 Program) under Grant No. 2004CB318201, and the US NSF under Grant No. CCF-0621526. Additionally, we are especially grateful to Juan Wang and Yu Hua for valuable feedback, support and discussions. We appreciate Chen Chen for their help. We are

grateful to all the members of our Laboratory, for their continuous support.

References

[1] Bhadkamkar, M., Burnett, S., Liptak, J., Rangaswami, R., Hristidis, V.,*A Case for Self-Optimizing File Systems*, Carnegie Mellon University, Syracuse University, Florida International University

[2] A. Acharya, M. Uysal, and J. Saltz. *Active Disks*. In Proceedings of the 8th Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS VIII), San Jose, CA, October 1998.

[3] G. R. Ganger, J. D. Strunk, and A. J. Klosterman. *Self-* Storage: Brick-based Storage with Automated Administration*. Carnegie Mellon University Technical Report, CMU-CS-03-178, August 2003.

[4] M. Sivathanu, V. Prabhakaran, F. I. Popovici, T. E. Denehy, A. C. Arpaci-Dusseau, and R. H. Arpaci-Dusseau. *Semantically-Smart Disk Systems*. Proceedings of the USENIX Symposium on File and Storage Technologies, pages 73.88, March 2003.

[5] G. A. Gibson, D. F. Nagle, K. Amiri, J. Butler, F. W. Chang, H. Gobioff, C. Hardin, E. Riedel, D. Rochberg, and J. Zelenka. *A cost-effective, high-bandwidth storage architecture*. In the 8th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS), 1998.

[6] T. M. Kroeger and D. D. E. Long. *The case for efficient file access pattern modeling*. Hot Topics in Operating Systems (Rio Rico, Arizona), pages 14-19, 29-30 March 1999.

[7] Thomas Kroeger, Darrell D. E. Long, *Predicting file-system actions from prior events*, Proceedings of the Winter 1996 USENIX Technical Conference, January 1996, pages 319C328.

[8] Purvi Shah, Jehan-Franliois Paris, Ahmed Amer, Darrell D. E. Long, *Identifying Stable File Access Patterns*, Proceedings of the Twelfth NASA Goddard/Twenty First IEEE Conference on Mass Storage Systems and Technologies (MSST '04), April 2004.

[9] G. Kuenning. *Design of the SEER predictive caching scheme*. In Workshop on Mobile Computing Systems and Applications, 1994.

[10] J. Griffioen and R. Appleton. *Reducing file system latency using a predictive approach*. In In Proceedings of the 1994 Summer USENIX Conference, 1994.

[11] Peng Gu, Yifeng Zhu, Hong Jiang, Jun Wang: *Nexus: A Novel Weighted-Graph-Based Prefetching Algorithm for Metadata Servers in Petabyte-Scale Storage Systems*. CCGRID 2006: 409-416

[12] David K. Gifford, Pierre Jouvelot, Mark A. Sheldon, and James W. O'Toole, Jr. *Semantic file systems*. Programming Systems Research Group, MIT Laboratory for computer science.

[13] Daniel Ellard, Michael Mesnier, Eno Thereska, Gregory R. Ganger, Margo Seltzer. *Attribute-Based Prediction of File Properties*. Harvard Computer Science Group Technical Report TR-14-03, December 2003.

[14] Soules C A N,Ganger G R.*Toward automatic context-based attribute assignment for semantic file systems[R]*.CMUPDL-04-105,2004.

[15] Deepavali Bhagwat and Neoklis Polyzotis. *Searching a file system using inferred semantic links*. In Hypertext, pages 85-87, 2005.

[16] Lingfang Zeng, Ke Zhou, Zhan Shi, Dan Feng, Fang Wang, Changsheng Xie, Zhitang Li, Zhanwu Yu, Jianya Gong, Qiang Cao, Zhongying Niu, Lingjun Qin, Qun Liu, Yao Li, and Hong Jiang. *HUSt: A Heterogeneous Unified Storage System for GIS Grid*. HPC Storage Challenge of Supercomputing. Tampa, Florida, November 2006.

[17] LIU, G.Z.: *Semantic Vector Space Model: Implementation and Evaluation*. Journal of the American Society for Information Science 48(5) (1997), 395-417.

[18] Moses S. Charikar, *Similarity estimation techniques from rounding algorithms*, Proceedings of the thiry-fourth annual ACM symposium on Theory of computing, May 19-21, 2002, Montreal, Quebec, Canada.

[19] Wang, F., Xin, Q., Hong, B., Brandt, S. A., Miller, E. L., Long, D. D. E., and McLarty, T. T. *File system workload analysis for large scale scientific computing applications*. In Proceedings of the 21st IEEE / 12th NASA Goddard Conference on Mass Storage Systems and Technologies (College Park, MD, Apr. 2004), pp. 139C152.

[20] D. Roselli, J. R. Lorch, and T. E. Anderson, *A comparison of file system workloads*, in Proceedings of the 2000 USENIX Annual Technical Conference (USENIX-00). Berkeley, CA: USENIX Ass., June 18C23 2000, pp. 41C54.

[21] E. Riedel, M. Kallahalla, and R. Swaminathan, *A framework for evaluating storage system security*, in FAST, 2002, pp. 15C30.

[22] Tsozen Yeh, Darrell D. E. Long, Scott A. Brandt, *Using program and user information to improve file prediction performance*, Proceedings of the International Symposium on Performance Analysis of Systems and Software (ISPASS '01), November 2001, pages Proceedings.

[23] Semantic file systems, http://www.objs.com/survey/OFSExt.htm

[24] N. Agrawal, W. J. Bolosky, J.R. Douceur, and J.R. Lorch, *A Five-Year Study of File-System Metadata*, Proc. of 5th Conference on File and Storage Technologies (FAST07), Feb 2007.

[25] M. Sivathanu, L. N. Bairavasundaram, A. C. Arpaci-Dusseau, and R. H. Arpaci-Dusseau. *Life or Death at Block Level*. In Proceedings of the 6th Symposium on Operating Systems Design and Implementation (OSDI 04), pages 379C394, San Francisco, California, December 2004.

[26] Z. Li, Z. Chen, S. Srinivasan, and Y. Zhou. *C-Miner: Mining Block Correlations in Storage Systems*. Proceedings of the USENIX Conference on File and Storage Technologies (FAST), pages 173. 186, April 2004.

[27] SIVATHANU, G., SUNDARARAMAN, S., AND ZADOK, E. 2006. *Type-safe disks*. In Proceedings of the 7th Symposium on Operating Systems Design and Implementation. ACM SIGOPS, Seattle, WA, 15C28.

[28] D. K. Gifford, P. Jouvelot, M. A. Sheldon, and James W. OToole, Jr. *Semantic file systems.* In Proceedings of the 13th ACM Symposium on Operating Systems Principles (SOSP91), pages 16C25, Oct. 1991.

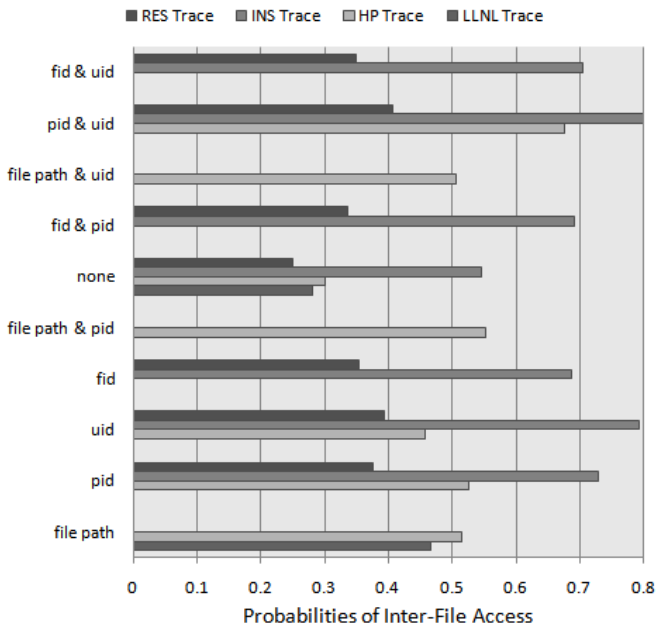

Figure 1: The probabilities of inter-file access quantify the influence on the association between file correlations and semantic attribute combinations (as indicated by the y-axis) for four typical distributed storage system traces.

[29] B. Gopal and U. Manber. *Integrating content-based access mechanisms with hierarchical file systems*. In Proceedings of the 3th USENIX Symposium on Operating Systems Design and Implementation (OSDI99), pages 265C278, 1999.

[30] Ahmed Amer, Darrell D. E. Long, Jehan-Franliois Paris, Randal Burns, *File access prediction with adjustable accuracy*, Proceedings of the International Performance Conference on Computers and Communication (IPCCC '02), April 2002.

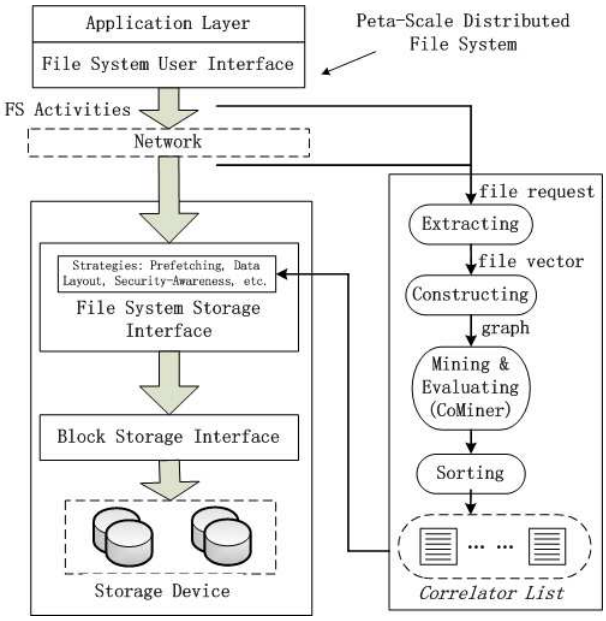

Figure 2: The FARMER Architecture

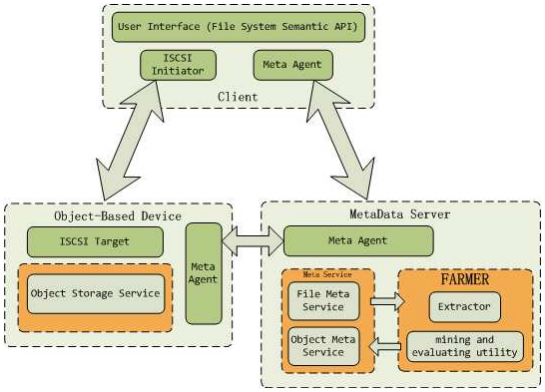

Figure 4: Architecture of HUSt

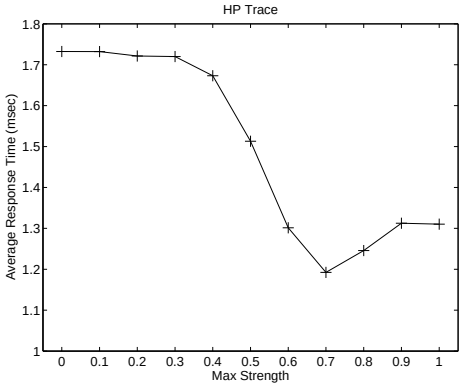

Figure 6: Impact of max strength

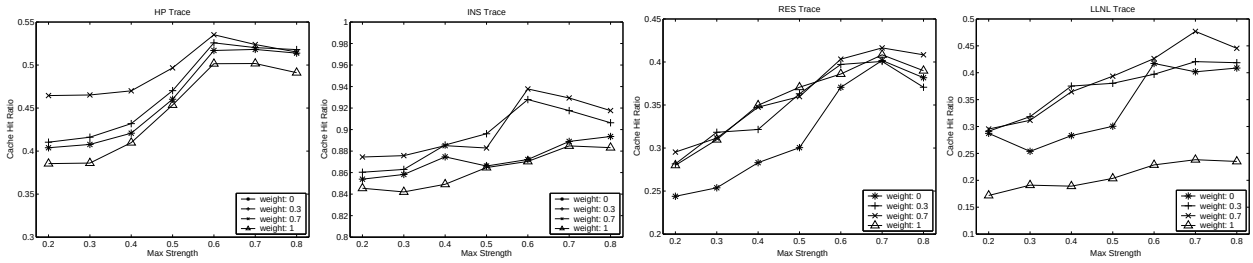

Figure 3: Impact of varying weight p for different traces

	HP Trace		**INS Trace**	**RES Trace**
Combination	**Hit Ratio**	**Combination**	**Hit Ratio**	**Hit Ratio**
{User}	47.8334%	{User}	93.3673%	38.3952%
{Process}	54.1105%	{Process}	93.3012%	37.5957%
{Host}	42.8946%	{Host}	89.0715%	35.7466%
{File Path}	53.5901%	{File ID}	86.3962%	36.7331%
{User, File path}	52.6727%	{User, File ID}	87.6923%	36.6741%
{Process, File path}	55.2138%	{Process, File ID}	87.2049%	35.3386%
{User, Process}	51.6512%	{User, Process}	93.0161%	40.6367%
{Host, process}	49.5810%	{Host, process}	90.8954%	40.6107%
{Host, User}	43.8305%	{Host, User}	91.0715%	41.6739%
{Host, File path}	48.3542%	{Host, File ID}	88.6923%	41.0377%
{Host, Process, File path}	48.9502%	{Host, Process, File ID}	90.3432%	36.7431%
{Host, User, File path}	47.6805%	{Host, User, File ID}	90.2269%	37.8957%
{User, Process, File path}	55.9857%	{User, Process, File ID}	93.2177%	41.9518%
{Host, Process, User}	47.5977%	{Host, Process, User}	92.7908%	41.3527%
{Host, User, Process, File path}	49.3087%	{Host, User, Process, File ID}	93.8839%	43.8533%

Figure 5: Cache Hit Ratios with different attribute conbinations

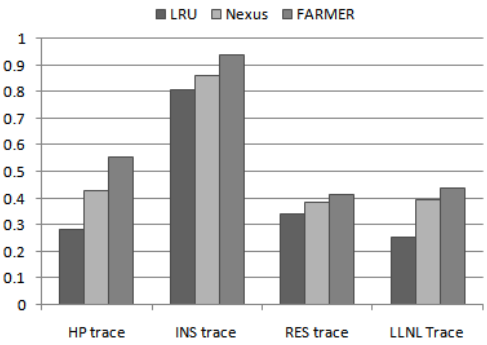

Figure 7: Cache hit ratio comparison

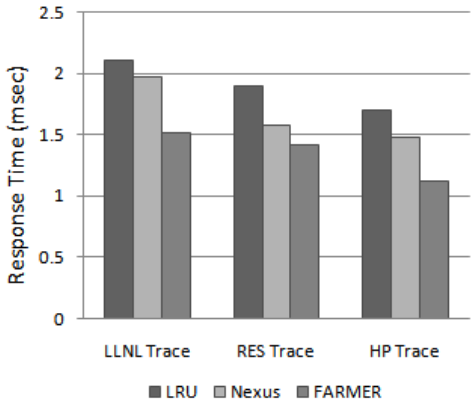

Figure 8: Response Time for LLNL, RES and HP Trace.