V. J. Sosa-Sosa, A. Barron, J. L. Gonzalez, J. Carretero and I. Lopez-Arevalo, "Improving Performance and Capacity Utilization in Cloud Storage for Content Delivery and Sharing Services," in *IEEE Transactions on Cloud Computing*.

DOI: 10.1109/TCC.2020.2968444

Improving Performance and Capacity Utilization in Cloud Storage for Content Delivery and Sharing Services

Victor J. Sosa-Sosa, Alfredo Barrón, Jose Luis González-Compean, Jesús Carretero, and Iván López-Arévalo

Alfredo Barron, Jose L. Gonzalez-Compean and Ivan Lopez-Arevalo are with CINVESTAV-Tamaulipas (Mexico). Jesus Carretero is with the UC3M (Spain). Victor J. Sosa-Sosa is on leave from CINVESTAV-Tamaulipas, visiting UC3M. Contact Email: jesus.carretero@uc3m.es

Abstract—Content delivery and sharing (CDS) is a popular and cost effective cloud-based service for organizations to deliver/share contents to/with end-users, partners and insider users. This type of service improves the data availability and I/O performance by producing and distributing replicas of shared contents. However, such a technique increases overhead on the storage/network resources. This paper introduces a threefold methodology to improve the trade-off between I/O performance and capacity utilization of cloud storage for CDS services. This methodology includes: i) Definition of a classification model for identifying types of users and contents by analyzing their consumption/ demand and sharing patterns, ii) Usage of the classification model for defining content availability and load balancing schemes, and iii) Integration of a dynamic availability scheme into a cloud-based CDS system. Our method was implemented on both a simulator and a real-world CDS service, supporting information sharing operations performed in a cloud storage. An experimental evaluation, conducted in a private cloud through simulation and emulation of workloads, showed the feasibility of this methodology in terms of storage capacity utilization, whereas the real-world implementation revealed the efficiency of applying a classification model to information sharing patterns in terms of I/O performance.

Index Terms—Cloud storage, Content Delivery and Sharing, Data Sharing, Classification Modelling, Availability.

1 INTRODUCTION

Cloud storage is a popular technology for organizations [1], [2] and end-users [3] to support information sharing, which is critical process for business continuity. Content delivery and sharing (CDS) is a cloud storage service that provides users with an effective and inexpensive mechanism to store and manage big data anytime and anywhere [4]. CDS infrastructures are typically deployed on clusters and/or the cloud, either virtual machines or virtual containers [5], [6]. In this type of environment, I/O speed and capacity storage consumption are critical metrics for both service providers and CDS end-users. Techniques of data distribution, load balancing and replication have been the solution for cloud storage providers to meet Service Level Agreement (SLA) contracted by the clients [7]. The replication techniques can be applied to the CDS in either static or dynamic manner. In the static strategy, the cloud storage system keeps the same replication factor during the life cycle of the systems. The replication factor (RF) defines the number of copies that will be distributed in the storage system. Popular systems, such as Amazon S3 [8] or HDFS [9], suggest as good practice to use RF = 3. Other systems have a predefined data placement technique, for example, a round robin policy, as it is implemented in Open Stack Swift [10]. The advantages of a predefined configuration are i) The parameter fine-tuning for replicated data flooding process is trivial as it is not necessary to constantly monitor and analyze the use of the storage system to adjust the RF and ii) The replica placement method is efficient with a low complexity as the patterns to deliver a given replica to an end-user are calculated in advance. Nevertheless, an important disadvantage of this strategy is not adapting the data placement to the workload created by the sharing patterns, which produces an inefficient data/replica distribution [11], affecting user experience and producing unnecessary storage consumption. In dynamic replication, configurations can be adjusted on-the-fly during the storage service life cycle [12]. However, the challenge is to find a good trade-off among the time spent in the replication process, the storage consumption, the network traffic caused by the distribution of replicas and the workload balancing among the nodes or storage resources that support the CDS system.

This paper introduces a methodology to improve I/O performance and capacity utilization of cloud storage for content delivery and sharing services. Since the main assets in this type of services are digital contents (mainly documents) shared by users in an organizational scenario, the methodology analyzes consumption/demand and sharing patterns to identify and classify content and users. This information is used as input for the decision engine, which is part of a replication system, to adaptively tune the replication factor and load balancing strategies, improving performance and consumption in the storage service. Figure 1 depicts the general steps follow in this methodology. In a first stage, classification modelling is applied to identify users and topics of contents according to the level of activity. Topic analytics and content-topics mapping are performed by analyzing contents (documents) or the metadata associated to them (multimedia). These tasks are achieved by using a graph-based method for topic extraction, which takes into account the co-occurrence of words (nouns and verbs) in sentences for generating a graph. Multimedia files includes standard metadata and manual annotations by content owners. An analysis of content producers and consumers is also carried out in this stage, as is shown in Figure 1. Sharing patterns allows to classify topics (high/medium/low demanded) and users (active/medium/passive). This classification is used as input (second stage) into a decision engine to dynamically establish the replication factor. This engine is integrated (third stage) in a dynamic availability scheme as part of a semantic CDS to provide information sharing operations in a private cloud storage. It is feasible, in a private cloud, to classify content suppliers according to content consumption/demand rates. The basic idea is that the CDS focuses its replication actions on active contents instead of using a general replication approach based on a static model. Our pro-active model is expected to reduce the number of actions (replications) to be performed by the CDS service, improving load balancing and storage utilization.

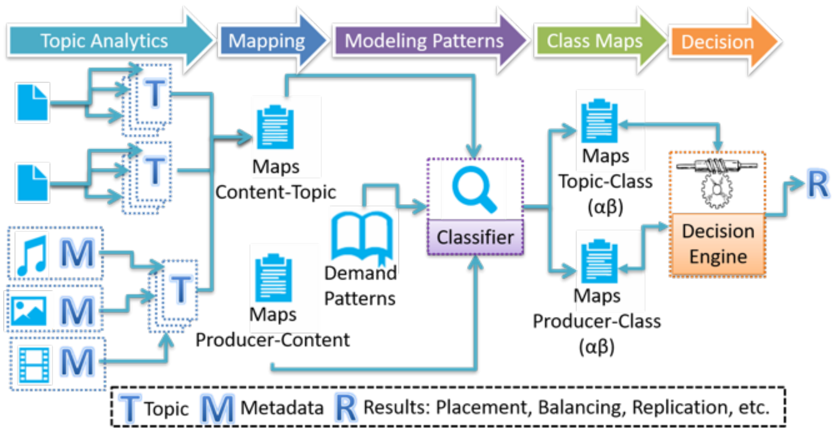

Fig. 1. Classification modelling based on demand patterns in cloud CDS services.

The main contributions of this paper are: a) Model for classification of users and content topics in cloud-based CDS systems considering their levels of activity (consumption/demand and sharing patterns), b) Efficient dynamic replication and data placement technique for CDS services based on this classification model, which improves the trade-off between I/O performance and capacity utilization of cloud storage when compared with the good practice approach, and c) Prototype of a Semantic CDS (S-CDS) system that implements our new proposals, showing application and performance results.

This paper is organized as follows: Section 1 introduces context and contributions of our proposal. Relevant related work is in Section 2. Section 3 presents a methodology to improve the trade-off between I/O performance and capacity utilization of cloud storage in CDS systems. An experimental evaluation of a S-CDS system and results are included in Sections 4 and 5 respectively. Finally, Section 6 summarizes the main conclusions and the future work.

2 RELATED WORK

Data replication and load balancing are effective techniques to address the availability and efficiency challenges when sharing information in a cloud-based Content Delivery and Sharing (CDS) systems [13], [14], [15], [12]. However, obtaining a good trade-off between storage consumption (replication) and user experience (performance) is not trivial [16], [17].

Some availability techniques are based on static methods that predefine number of replicas and data placement. For instance, 3 replicas is the default value in Amazon S3 [8], Swift, HDFS [9] and GFS [18], and a round robin policy is defined in Open Stack Swift [10]. Erasure Coding (EC) is a static redundancy strategy that offers similar levels of fault tolerance as replication, but with less storage space. This technique is used in Hadoop 3.0 [19] and Open Stack Swift[20]. Dynamic replication methods [21], [22] allows to adjust configurations on-the-fly during the storage service life cycle, based usually on monitoring data. Li et al. [21] proposed a dynamic data replication strategy based on incremental replication methods that can reduce storage cost, especially for temporal data or data with less reliability requirements. Wei et al. [22] presented a cost-effective dynamic replication management (CDRM) to find a minimal replica number for a given availability requirements using HDFS. Mansouri et al. [23] studied data replication strategies and defined a dynamic popularity aware replication strategy (DPRS) based on data access behavior. Static methods are easier to implement, but lack of flexibility, because they do not adapt the data placement to the workload produced by the sharing patterns. This situation generates an inefficient data/replica distribution [11], causing unnecessary storage consumption and affecting user experience.

Recent studies [24], [23], [25] have showed that information sharing patterns in CDS have similar behavior to those presented in social network. Kilianoti et al.[24] conducted experiments on a CDS simulation framework to compare different policies for content delivery, considering the information sharing patterns in online social networks. They realized the importance of this information to improve user experience. Hamrouni et al. [26] and Nine et al. [27] used datamining techniques as tools for discovering new meaningful knowledge to enhance replica management. Slimani et al. [28] proposed an improved replication strategy based on maximal frequent correlated pattern mining in data grids.

In-House caching, batching and deduplication solutions have been proposed to improve the cloud storage user experience [29], [30], [31], [32], [33], [34]. These solutions explore the improvement of the efficiency of synchronization clients of public cloud storage providers. They could be complemented by analyzing sharing patterns as it is proposed in our methodology and, at the same time, our methodology could improve user experience by incorporating this type of solution.

The main advantages of our proposal with respect to the related work are: i) It allows the creation of policies for providing cloud storage solution with data availability that improves storage utilization and performance of the information sharing for CDS services based on virtual machines and containers; ii) Most of the related work have been tested in simulated scenarios, while our methodology was tested in a prototype of a semantic CDS service for an organizational scenario; iii) To the best of our knowledge, this work would be the first approach that provides an integrated load balancing, availability and reliability scheme based on the classification of users and content (topics) activity, which dynamically adapt the replication factor and distribution of contents in a semantic CDS service to improve the trade-off between I/O performance and cloud storage consumption; and iv) A complementary simulator and emulator based on our methodology is available to evaluate state-of-the-art availability policies and load balancing methods at different scales. Some aspects that were not considered in our proposal and are available in the related work are: i) The use of synchronization techniques on the end-user side that can potentially improve user experience [32], [34]; and ii) The use of Erasure Coding (EC) techniques to provide data confidentiality and improve storage consumption [19], [20], [35].

3 A METHODOLOGY TO IMPROVE PERFORMANCE AND CAPACITY UTILIZATION OF CLOUD STORAGE FOR CDS SERVICES

This section describes a threefold methodology to improve the trade-off between I/O performance and capacity utilization of cloud storage for Content Delivery and Sharing (CDS) services. Its main three stages are described in next sections.

3.1 Classification modelling of users and topics of contents in CDS systems

CDS services have some similarities with social networks as the users adopt different roles, such as Supplier or Producer (users that upload and share contents), Consumer (users that download contents), and a combination of both. In this stage of our method, we focused on classifying contents grouped by supplier or by main topic, using consumption/demand patterns found in the logs of the CDS service. The classes and metrics used for this classification are explained in next sections.

3.1.1 Classification metrics

Two metrics were defined to classify content suppliers and content topics according to their level of activity: Volume and Density. Volume (V) represents the storage activity in the CDS system when serving a set of download requests during a period of time (workload). Density (D) refers to the average inter-arrival time of download requests during a workload. Calculations of these metrics are explained in Sections 3.1.4 and 3.1.5.

3.1.2 Classes of content suppliers

Content suppliers (S) are classified according to the consumption/demand patterns of their contents in the following classes:

1) **Gamma Content Supplier ($S\gamma$)**: A S_i content supplier becomes Gamma supplier if $V_{S_i} \leq AV_S$, i.e., the volume of bytes served, in download requests, of contents uploaded by supplier S_i (V_{S_i}) is less than or equal to the average volume of bytes served during the workload (AV_S). Content suppliers whose contents do not receive any download request will also be considered of this type.
2) **Beta Content Supplier ($S\beta$)**: A S_i content supplier becomes Beta supplier if the following restriction is reached: $S\beta = \{S_i | S_i \in S \wedge V_{S_i} > AV_S\}$.
3) **Alfa Content Supplier ($S\alpha$)**: A S_i content supplier becomes Alfa content supplier if the following condition is met: $S\alpha = \{S_i | S_i \in S\beta \wedge Di < D_S\}$, i.e., the average inter-arrival time of download requests (Di) for content uploaded by supplier S_i is less than the average inter-arrival time of download requests for content produced by all content suppliers (D_S).

3.1.3 Classes of content topics

Similarly to content suppliers, content topics are associated to a corresponding class, according to the demand patterns of the main topics found in the contents. The topic discovering is performed according to the type of files to be shared:

1) **Text files (documents)**. The topic is discovered analyzing the text of documents -articles, letters, reports, memorandums-. This task is achieved by using a graph-based method for topic extraction, which takes into account the co-occurrence of words (nouns and verbs) in sentences for generating a graph of the most relevant words. Each word corresponds to a node, and co-occurrence of words correspond to edges. Based on the Distributional Similarity approach [36], words co-occurred on a sentence tend to have the same or related meaning. Thus, a topic includes several words with related meaning. Words related to a topic are identified by clustering words in the graph by means of the Betweenness Centrality algorithm [37].
2) **Multimedia files**. In this case the identification of topics is by a direct way. Image, audio, and video files must contain (in addition to standard metadata), manual annotations by owners, defining characteristics of content (owner, supplier, title, short description, keywords, and up to 3 main topics). Terms used for keywords and main-topic fields can come, depending on the domain of files, from different vocabularies.

The set of topics is used in the availability schemes of the CDS systems for establishing the replication factor to be applied to each sharing information operation. In the context of content topics, the classes are defined as follows:

1) **Gamma Content Topic ($CT\gamma$)**: A CT_i content topic is considered Gamma if $V_{CT_i} \leq AV_{CT}$, i.e., the volume of bytes downloaded of contents with topic i (V_{CT_i}) is less than or equal to the average volume of bytes served in download requests for content of all content topics (AV_{CT}). Content topics whose contents do not receive any download request will be also considered of this type.
2) **Beta Content Topic ($CT\beta$)**: A CT_i content topic becomes Beta if $V_{CT_i} > AV_{CT}$.
3) **Alfa Content Topic ($CT\alpha$)**: A CT_i content topic becomes Alfa if CT_i is a Beta content topic and $D_{CT_i} < D_{CT}$. Since Alfa content topics ($CT\alpha$) are classified using only the Beta content topic set ($CT\beta$), the space of search and processing is reduced.

3.1.4 Volume and density calculation for content suppliers

The volume of a content supplier S_i (V_{S_i}) represents the storage consumption of the CDS system when serving download requests for the contents produced by S_i. It is defined as follows:

$$V_{S_i} = \sum_{j=1}^{|F_{S_i}|} \sum_{1}^{ND_{F_jS_i}} |F_jS_i| \qquad (1)$$

Where $|F_{S_i}|$ is the number of files uploaded by supplier S_i, $|F_{jS_i}|$ represents the size of a specific file, $F_j(1 \leq j \leq |F_{S_i}|)$, that was uploaded by supplier S_i, and $ND_{F_jS_i}$ is the number of downloads made to the F_{jS_i} file. The bytes transfer or volume (V_S) of the CDS system when serving the complete set of download requests is given by:

$$V_S = \sum_{i=1}^{|S|} V S_i \qquad (2)$$

Where $|S|$ is the number of suppliers whose content was downloaded at least once. The average volume of bytes transferred due to download requests in the CDS system is defined by:

$$AV_S = \frac{V_S}{|S|} \qquad (3)$$

Density of content suppliers (D_S) refers to the average inter-arrival time of download requests for contents produced by all content suppliers denoted by set S. Density for content of a specific supplier (S_i) is denoted by D_{S_i}. Content suppliers of class Alpha or Beta generate the most demanded content in a CDS. The density of every beta content supplier ($D_{S\beta i}$) is calculated as follows:

$$D_{S\beta_i} = \frac{\sum_{j=2}^{|WDS\beta_i|}(DATReq_j - DATReq_{j-1})}{|WDS\beta_i|} \quad (4)$$

Where $|WDS\beta_i|$ is the number of download requests that the storage system has received for content of supplier $S\beta_i$, and $DATReq_j$ is the inter-arrival time of a download request j for content of $S\beta_i$. Once obtained the density values of every beta provider, we apply the following formula to calculate density of the beta provider set ($D_{S\beta}$):

$$D_{S\beta} = \frac{\sum_{i=1}^{S\beta} DS\beta_i}{|S\beta|} \quad (5)$$

Where $|S\beta|$ is the number of beta suppliers. The density values of each beta supplier ($D_{S\beta_i}$) and the density of the complete set of beta suppliers ($D_{S\beta}$) are used to classify alfa content suppliers as follows: $S\alpha = \{i|i \in S\beta \wedge D_{S\beta_i} < D_{S\beta}\}$. Recall that $D_{S\beta_i}$ and $D_{S\beta}$ represent the average inter-arrival time of a download request for a specific beta supplier and for all beta suppliers respectively.

3.1.5 Volume and density calculation for content topics

The volume of a content topic CT_i denoted by (V_{CT_i}) represents the storage consumption of the CDS system when serving download requests for the content of topic CT_i, and it is defined as follows:

$$V_{CT_i} = \sum_{j=1}^{|F_{CT_i}|} \sum_{1}^{ND_{F_jCT_i}} |F_jCT_i| \quad (6)$$

Where $|F_{CT_i}|$ is the number of downloaded files of the content topic i (CT_i), $|F_{jCT_i}|$ represents the size of one specific file, Fj($1 \leq j \leq |F_{CT_i}|$), with CT_i topic, and $ND_{F_jCT_i}$ is the number of downloads made to the F_{jCT_i} file. The storage activity or volume (V_{CT}) of the CDS system when serving downloads requests for any content topic is given by:

$$V_{CT} = \sum_{i=1}^{|CT|} V\,CT_i \quad (7)$$

Where $|CT|$ is the number of topics which have contents/files that were downloaded at least once. The average storage service required by download requests to the CDS system is defined by:

$$AV_{CT} = \frac{V_{CT}}{|CT|} \quad (8)$$

Density of content topic, or D_{CT}, refers to the average inter-arrival time of download requests for contents of any topic and D_{CT_i} denotes the average inter-arrival time of download request for content of a specific topic. The following formula defines the density of a beta content topic i ($D_{CT\beta_i}$):

$$D_{CT\beta_i} = \frac{\sum_{j=2}^{|WDCT\beta_i|}(DATReq_j - DATReq_{j-1})}{|WDCT\beta_i|} \quad (9)$$

Where $|WDCT\beta_i|$ is the number of download requests that the storage system has received for content with topic $CT\beta_i$, and $DATReq_j$ is the inter-arrival time of a download request j for content of $CT\beta_i$. The following formula calculates the average density for the complete set of beta content topics ($D_{CT\beta}$):

$$D_{CT\beta} = \frac{\sum_{i=1}^{CT\beta} D_{CT\beta_i}}{|CT\beta|} \quad (10)$$

Where $|CT\beta|$ is the number of beta content topics. The density values of individual beta content topic ($D_{CT\beta_i}$) and density of all beta content topics ($D_{CT\beta}$) are used to define the alfa content topics set as follows: $CT\alpha = \{c|c \in CT\beta \wedge D_{CT_c} < D_{CT}\}$.

3.2 Usage of the classification model for content availability and load balancing schemes

The process of classification of content suppliers and content topics during the execution of a CDS service begins once volumes and densities of contents have been calculated. This process is executed by the CDS service periodically. In this phase of our methodology, a replication factor is assigned to each class of users (either alpha or beta), as the model does not considers actions for gamma users (low activity). Data replication is an important process for content availability and load balancing schemes, so our method uses the classification model for implementing adaptive data replication techniques based on content supplier and content topic classification.

3.2.1 Data replication based on content supplier classification

Content supplier classification allows storage system to adapt the replication factor (RF) based on content activity (Density and Volume). RF indicates the number of copies that the storage system has to generate for each uploaded content in order to distribute them in the n nodes available in the storage pool. As initial RF value, we have considered the good-practice reference with RF = 3 (three copies) in a pool of n = 5 storage nodes. RF will be adjusted depending on the classification of content suppliers. The RF values for the different types of content suppliers are: $RF\ S\gamma = 1$ (gamma suppliers), $2 \leq RF\ S\beta \leq 3$ (beta suppliers) and $4 \leq RF\ S\alpha \leq n$ (alfa suppliers). In alpha and beta suppliers, the final RF value is calculated by means of parameters tuning, carrying out a set of experiments comparing the impact on the storage system service when using tuned RF values versus using good practices RF values. Different RFs produce different impacts on storage resource determined by the type of content supplier and the number of nodes in the storage pool. To evaluate the content supplier impact (SI) in a storage service, we have assigned a value to SI that is in the range of $(0,1]$, according to the following equation:

$$SI = \frac{RF}{n} \tag{11}$$

Where RF is the replication factor for a content supplier and n is the number of nodes in a storage pool. For instance, for configurations $RF\ S\gamma = 1$, $2 \leq RF\ S\beta \leq 3$ and $4 \leq RF\ S\alpha \leq n$, in a pool of 5 nodes, gamma suppliers would have $SI = 0.2$, beta suppliers $SI = 0.4$ or $SI = 0.6$ and alfa suppliers $SI = 0.8$ or $SI = 1$. These values represent the portion of storage resources (in terms of storage nodes) used to offer content availability.

3.2.2 Data replication based on topic classification

CDS services manage content of different topics with different activity levels, which motivates us to offer a differentiated storage management. Classification of contents by topics is an appealing strategy for organizations that want to distribute or share corpus of documents (pdf, doc, ppt, html, xls, ppt, etc.) or contents that include a portion of text in them, for instance medical images such as DICOM, which include metadata. In this context, it is feasible to classify a representative dataset of the type of contents the storage services will manage. Figure 2 shows the visualization of the content topic extraction and clustering obtained from a set of 500 MedLine documents [38]. In this visualization small nodes represent documents (contents), stored in the CDS system, which are associated to big nodes that represent topics. This information is useful for decision makers to define the data placement and replication strategies based on content topics. This visualization tool is supported by the Gephi engine [39], which is an open source software for exploring networks.

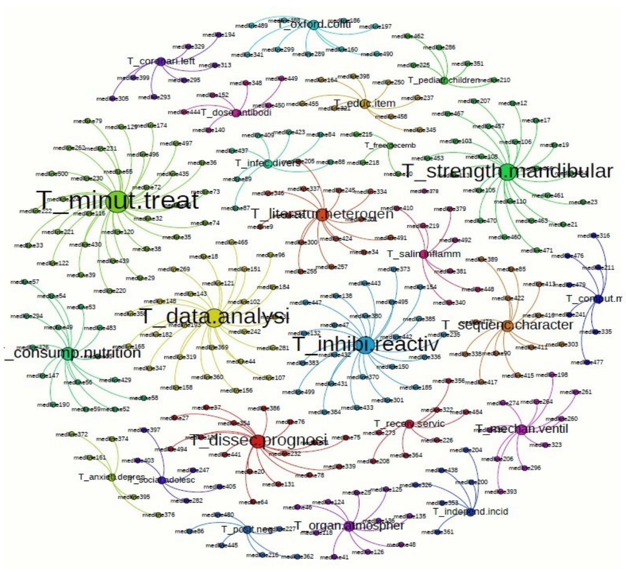

Fig. 2. Visualization of content topic extraction and clustering.

As happens in content supplier classification, the classes of content topics are used to adjust the replication factor (RF) in the storage service, offering a differentiated storage management based on content topic activity (Density and Volume). The principles of the configurations applied to the RF for alfa, beta and gamma content topics are similar to those explained for content supplier classification.

3.2.3 Load Balancing and data placement In a CDS, client processes (consumer or supplier) send upload or download requests to a Metadata instance, which in turn responds to the clients with a service authorization token and the address (URL) of the most suitable storage instance that will attend their requests. The client will PUT/GET content/file to/from the selected storage instance. When a storage instance receives the content from the client, the control is returned to the client process, and the storage instance begins the replication process (out of the critical path) depending on the replication factor (RF) configured. The selection of a storage instance is implemented using traditional random algorithms, such as *TwoChoices* [40], which randomly select two storage instances and choose the one with the lower utilization factor (UF) [6]. UF represents the portion of storage resources consumed in a storage instance with respect to the total storage capacity. The CDS services uses UF to achieve load balancing depending on the topics found in either documents or metadata associated to multimedia contents. As a result, the contents associated to the indexed/discovered topics will be distributed in a balanced manner avoiding bottlenecks for most downloaded contents. As it can be seen, the classification of topics and suppliers of contents can be used as criteria to make decisions. For instance, data placement and load balancing can be performed trying to distribute the contents associated to indexed topics over storage nodes in uniform manner. Moreover, this classification is useful for making decisions about availability, by choosing replication factors depending on the users and/or topic classification.

3.3 Integration of the classification model in a semantic CDS service

As a proof of concept, we deployed a Semantic Content Delivery and Sharing (S-CDS) service, an evolution of a content distribution and file sharing platform named SkyCDS [4], [35], which was implemented on a private cloud. We added our model to the Pub/Sub system of SkyCDS to classify users and content topics. The results of the classification model were used as input by the availability scheme of SkyCDS, which is used as an emulator of file sharing operations that enables us to test the efficiency of our model in terms of service times and storage consumption. The SCDS platform includes three main layers: Access, Processing and Data (see Figure 3). Suppliers and consumers are validated in the Access layer (authentication sub-layer). The processing layer includes Local Metadata, Semantic Engine, Global Metadata and Placement/Replication components. Local Metadata manages the information about all of the uploaded content (Supply Area) in each Storage Node (SN).

The Semantic Engine oversees extracting keywords and concepts from each uploaded file and defines a set of topics that are mapped to these files. This process also registers the maps in the Supply table of the Storage Node (SN). The Global Metadata component builds and manages an information cube unifying the Service (requests) and Supply tables of all SNs for the storage platform. The Placement and Replication component is in charge of supporting the content distribution process. Table 1 summarizes the characteristics of the instances that support the prototype. All the instances are Ubuntu 14.04 virtual machines created and deployed on an OpenStack [41] cloud manager.

TABLE 1
Cloud infrastructure for the content delivery and sharing system.

Instances	Type	VCPU	RAM	HDD	Software
5	Storage	2	2 GB	40 GB	
1	Metadata	2	2 GB	40 GB	
1	Client	2	2 GB	40 GB	

The storage instances were configured as a virtual unified storage pool that offers an operational interface to execute Put/Push (upload), and Get/Pull (download) operations. The central control of the storage instances is located in the Metadata instances, which are in charge of controlling the storage pool, supporting the data placement and replication processes. The configuration of these processes is based on the replication factor according to the content supplier or content topic popularity. Metadata also allows users to publish and subscribe to content, creating continuous workflows.

3.3.1 Data placement and replication

In the S-CDS, Metadata is responsible for making decisions on the usage of storage instances. The modular design of the S-CDS platform makes possible to change the Metadata by a new module, adding or modifying new functionalities that affect the behavior of storage instances, for example, adjusting the replication factor and workload distribution by using our classification model. Volume and Density metrics (see Section 3.1) are used in the classification process of S-CDS to determine if a content supplier or topic belongs to an Alpha, Beta or Gamma class. Initially all of them belongs to the Gamma class. After a warming phase, the S-CDS platform runs the classification process according to the parameters produced by the classification model for the availability scheme. Before any document is uploaded into the S-CDS system, it has to be processed in order to identify its topic. For this prototype, we have included in our prototype KeyGraph [42], a document classification tool that automatically identifies topics generating a thematic catalog, to allow the S-CDS platform to associate a topic to every document before being uploaded. Once every document has a topic assigned, the S-CDS platform calculates the Volume and Density metrics of each content topic (Section 3.1.5). Then, the system classifies every content as Alpha, Beta or Gamma class and dynamically adjusts the replication factor (RF) assigned to it in order to apply a differentiated service.

3.3.2 Prototype details

All the systems in the three layers of the S-CDS service were published as web microservices by using Apache and PostgreSQL. These microservices were accessed by using an API REST for PHP and Python. The authentication service in the access layer was developed by using a tokenization system developed in C language. In the processing layer, the metadata management service was developed in PHP, whereas the load balancing, data placement and replication mechanisms were developed in C language. The Semantic engine was developed in Java including a client of the metadata service to serve requests sent by this service. For the data layer, we developed a client of a multi-cloud storage service named SkyCDS [4] by using its PHP API REST, which was used for delivery and retrieval of contents for all the services in the prototype. All the services were encapsulated into both virtual machines and containers. The virtual machines were allocated by using OpenStack, whereas the containers were created by using Docker platform and deployed on the cloud by using the docker files.

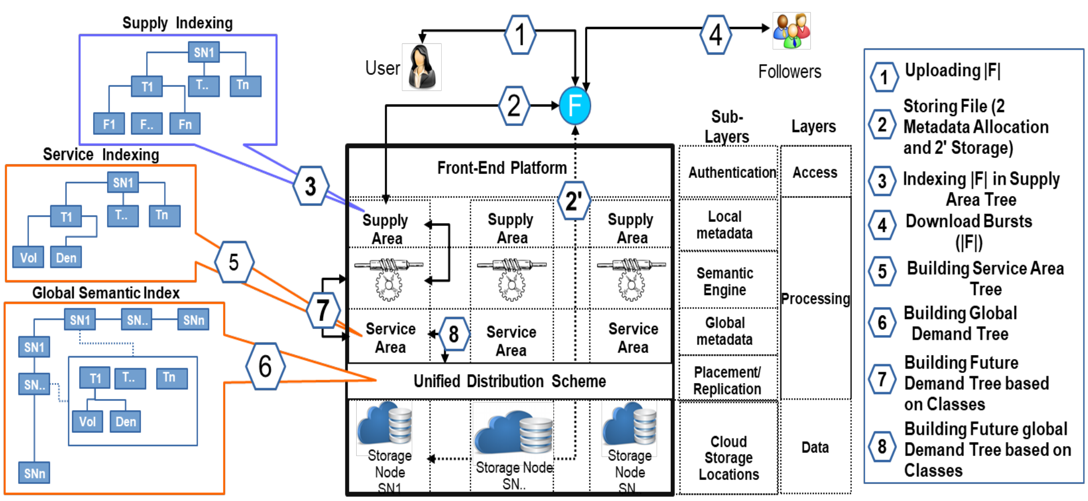

Fig. 3. Conceptual representation of S-CDS.

4 EXPERIMENTAL EVALUATION

The evaluation of the S-CDS platform was carried out in a controlled test scenario, using the storage cloud environment described in Table 1. Two extensive testbeds were built, one using the Medline dataset [38], which is a document corpus related to the medicine area, and another one using a synthetic workload generator that reproduces different user behaviors (content consumer and supplier).

4.1 Test configurations

The following test configurations were considered:

- **Norm or Static**: In this configuration, data placement is carried out using a traditional random scheme, such as *TwoChoices* [40] (two storage instances are random selected and the instance with the lowest workload will be chosen). Replication is executed in a pro-active way (once a storage instance receives the content, control returns back to client and the instance begins to send replicas to other instances) with a Replication Factor of 3 for Alpha, Beta and Gamma classes.

- **Reactive**: Data placement is carried out using a random scheme such as *TwoChoices*. Replication is executed in a reactive way, behaving similar to a cache service (when a download request is received by a storage instance that does not have a replica of the requested content, the storage instance will obtain the content and will keep a replica of it).
- **Proactive**: This configuration represents our replication proposal based on content supplier and content topic popularity. Data placement is carried out using also the *TwoChoices* strategy.

Proactive configurations use a flexible replication factor (RF) that can vary depending on the interest of storage system administrator and end users. The following are Proactive configurations used in our experiments deployed on a storage service pool of 5 nodes (n = 5):

- ***ProA3B1***: Proactive configuration interested in finding out a trade-off between storage consumption and storage service time with performance priority; Alpha class with RF = 3, Beta class with RF = 1, and Gamma class (or default class) with RF = 1.
- ***ProNorm***: Configuration that applies good practices (RF = 3) for Alpha and Beta classes (others with RF = 1), with interest in storage consumption.
- ***ProFlood***: It applies replica flooding in storage pool of n = 5 nodes with RF = 5 for Alpha and Beta classes (others with RF = 1).
- ***ProA5B3***: It applies flooding (RF = 5) for Alpha class and good practices (RF = 3) for Beta class, others RF = 1.
- ***ProA5B2***: Use flooding (RF = 5) for Alpha class and storage frugality for Beta class (RF = 2), others RF = 1.

The implementation of Proactive configurations varies according to the replication technique that is used, which could be based on popularity of content supplier or content topic. It is important to notice that for these experiments Proactive configurations in the storage service always implies the use of a RF = 1 in content suppliers or content topics that belong to the Gamma class.

TABLE 2
Proactive configurations.

Proactive configuration	Method	$\mathbf{RF}(\alpha,\beta,\gamma)$, $\mathbf{n = 5}$
ProA3B1	Good practices (Norm) for α users/topics	$RF(3,1,1)$
ProNorm	Norm for α & β classes	$RF(3,3,1)$
ProFlood	Flooding for α & β classes	$RF(5,5,1)$
ProA5B3	Flooding & Norm for α & β	$RF(5,3,1)$
ProA5B2	Flooding & Frugality	$RF(5,2,1)$

4.2 Metrics

.

During the experimentation process, the following metrics were collected from the S-CDS platform:

- **Response time (RT)**: It covers from the time a client sends a request (upload/download) into the S-CDS platform until the content is completely transferred. RT also includes the time spent by the client in obtaining, from the Metadata instance, the address (URL) of the storage node that will serve the client request.
- **Storage consumption (SC)**: Represents the amount of storage resources used by the storage service.
- **Throughput (Th)**: Volume of data (MB/s) that flows into the storage system.
- **Load balancing level (LBL)**: This metric represents the level of workload distribution among storage nodes in a period of time. The LBL of a storage instance x is given by:

$$LBL_x = CW_x - IW \qquad (12)$$

Where CW_x is the current workload in the storage instance x and IW represents the ideal workload distribution, which is given by:

$$IW = TSC/n \qquad (13)$$

Where TSC represents the total storage consumption and n the total number of storage instances. A LBL value close to cero would mean the storage instances are well-balanced.

4.3 Content delivery and sharing scenario

Our S-CDS service was deployed in the cloud infrastructure mentioned in Section 3.3. It allows identifying content suppliers and consumers, making it possible for users to publish and subscribe to content creating information workflows. The component that represents the client part of the S-CDS service (Launcher) was installed in a cloud instance. It was in charge of the workload generation processes, which were based on the behavior found in real content sharing systems
[43].

4.4 Workloads

In our S-CDS scenario, workload generation is made by a process named $Launcher$, which produces uploading and downloading requests emulating the behavior of real clients. Launcher was installed in the client instance as part of the S-CDS supplier infrastructure. The platform receives workload from Launcher and assumes it comes from real and valid users.

4.4.1 Content topics workload

The replication strategy based on the popularity of the topic of the content requires classifying textual content. It means that all of the files distributed and shared in the S-CDS platform include text as part of their contents, for example pdf, html, ppt files or images with metadata. Thus, for these experiments, we used the Medline dataset [38].

As we wanted to emulate real-world clients, content topic workload generation was based on the behavior found in real content sharing systems [43]. The launcher process uses its traffic generator (TG) component to generate 5 workloads for the S-CDS platform, each with 2530 traces. The inter-arrival time for requests in a S-CDS system follows a Poisson distribution (Launcher produces 10 requests per second in mean). The type of request follows a Zipf distribution, where 80% of operations are downloads and 20% are uploads. Content sizes were also defined by a Poisson distribution. Synthetic files of sizes of 5 MB in average are generated by the Content Generator (CG) component, more than 500 files per workload. Each synthetic file is associated with the textual content of a file taken from the Medline corpus, representing its metadata. This textual part of the content allows the classifier to tag it with the corresponding topic. As mentioned in Section 3, Volume and Density are two main criteria used to determine the popularity of a content topic and classify it in one of the following classes: Alpha, Beta and Gamma (from more to less popularity respectively). Figure 4 shows Volume (vertical axis) of the different topics extracted from contents in a dataset, which were tagged with numbers (horizontal axis) by the classification process. Topics whose volumes of contents are above the median value are classified as Beta class, the rest of contents are classified as Gamma class, which is the default value for any content.

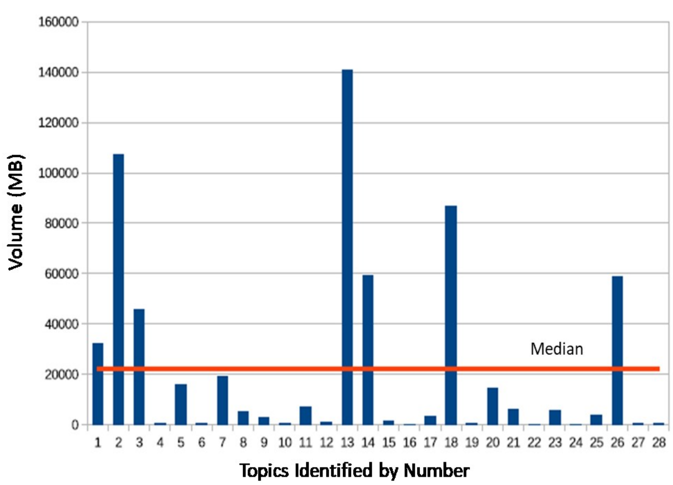

Fig. 4. Example of volume of 28 topics identified in documents uploaded into the S-CDS platform.

When the Beta topics have been identified by the semantic service, the density of this set of topics is analyzed to identify which of them can be classified as Alfa class topics. This classification is assigned to a topic when its median inter-arrival time for download operations is below the median value of all the Beta topics. Alpha topics are frequently demanded, which means these topics produce a higher density than the rest of analyzed topics. Figure 5 shows the inter-arrival time of topic consumption (vertical axis) found for all the topics classified as Beta. The median value is indicated with a horizontal line. Topics tagged with numbers 2, 14 and 26 have higher density than the rest of topics as their median consumption inter-arrival time is under the median of the topics classified as Beta. Therefore, these topics will be indexed as Alpha class.

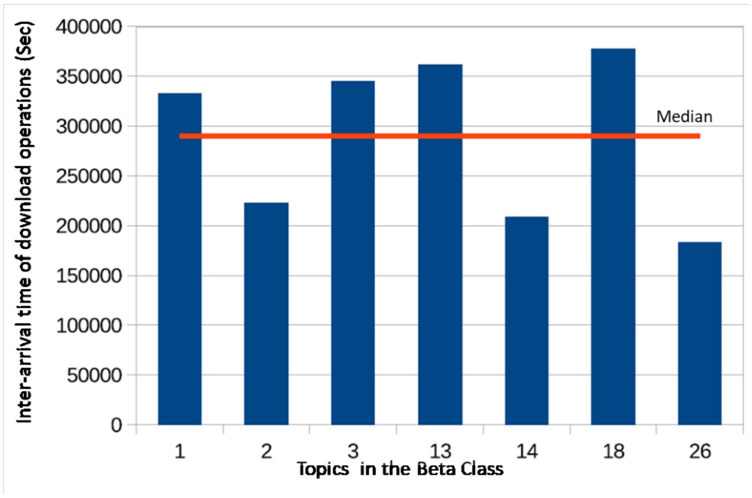

Fig. 5. Example of density of contents in the Beta class

5 RESULTS

This Section summarizes the relevant results obtained by the S-CDS platform when implementing the data replication and placement techniques described in Section 3.2. The test configurations used in our experimentation were described in Section 4.1.

5.1 Content supplier popularity approach

We begin the experimentation configuring the availability and load balancing schemes in the S-CDS service based on the popularity of the content suppliers. We evaluated the response time perceived by the users of the S-CDS system when the S-CDS platform applied different test configurations taking into account the popularity of the content suppliers. For this scenario, the Static/Norm configuration represents our baseline for comparison, as it implements the traditional, or good practices, replication configuration with RF = 3 and it does not change during the complete experimentation. The launcher process reproduced workloads as described in Section 4.4.

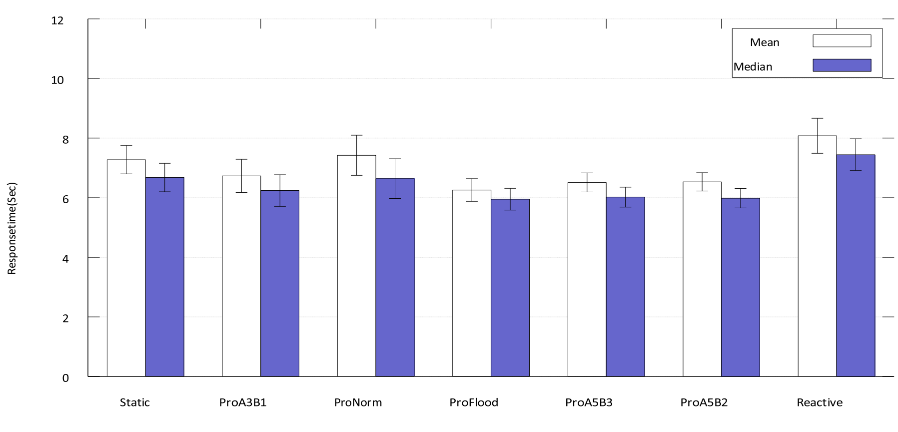

Fig. 6. Response time of the S-CDS platform when implementing replication based on content supplier popularity.

Figure 6 shows in vertical axis the median and mean response times in seconds of the S-CDS platform for the workload (download/upload operations), implementing replication based on content supplier popularity, following the test configurations described in Section 4.1 (horizontal axis). Most of the Proactive configurations (ProA3B1, ProFlood, ProA5B3 and ProA5B2) improved the Static/Norm configuration (baseline), except the ProNorm configuration, which use $RF = 3$ for Alpha and Beta content suppliers. Since Figure 6 shows response time median values obtained from the complete workload, considering download (80% of the workload) and upload (20% of the workload) operations, it seems that most of the replication configurations have a similar behavior when they are based on content supplier popularity. However, it is important to notice that some replication configuration such as ProFlood require more than twice storage space to provide a similar user experience.

Figure 7 shows a comparison among different replication policies, contrasting performance and storage consumption with the baseline configuration (represented with a value of zero in the vertical axis). As expected, the configurations that increase the number of replicas (ProFlood, ProA5B3, ProA5B2) have a positive effect in download operations and negative effect in upload operations (some of them, ProFlood, decreasing upload performance in more than 60%). When good practices ($RF = 3$) are applied only in alpha content suppliers (ProA3B1), performance improves in both upload and download operations. All of the Proactive configurations improved performance at different storage consumption. It is interesting to notice that when good practices ($RF = 3$) are applied to Alpha (ProA3B1) and Beta (ProNorm) content suppliers the service performance is improved, saving storage resources. ProA3B1 becomes the best trade-off between performance and storage consumption, ProNorm is a good deal when frugality is a priority, and ProA5B2 is a good option to improve performance with a moderate resource consumption. The significant performance improvement of upload operations produced by ProNorm and Reactive configurations does not produce performance improvement in download operations, which becomes an important impact to user experience, considering that download operations are majority.

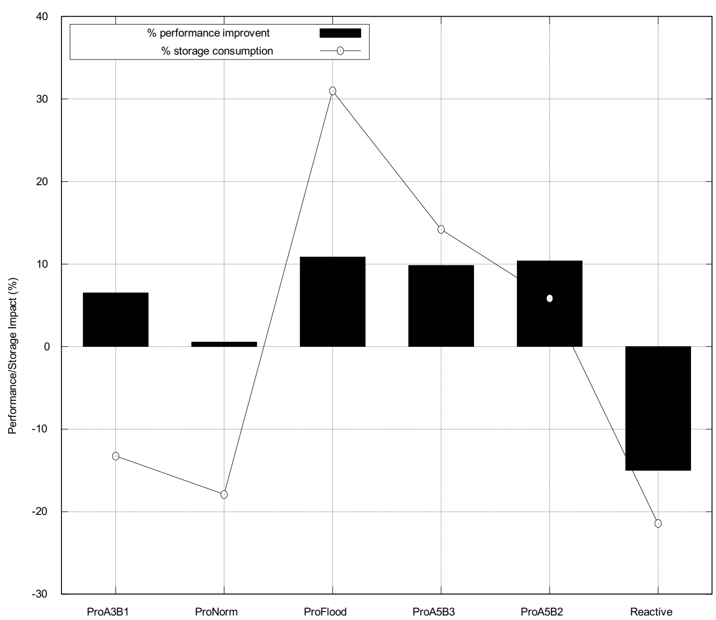

Fig. 7. Performance and storage impact with respect to the baseline considering content supplier popularity.

Figure 8 shows how replication policies based on content supplier popularity affected the load balancing in the storage nodes (in percentage terms). The ideal load balancing defined in Section 4.2 was used as baseline (value of zero in the vertical axis). It is worthy of note that Reactive and ProActive configurations produced load balancing very close to the ideal (i.e., very close to zero), improving the load balancing obtained by the good practice (Static, $RF = 3$) configuration.

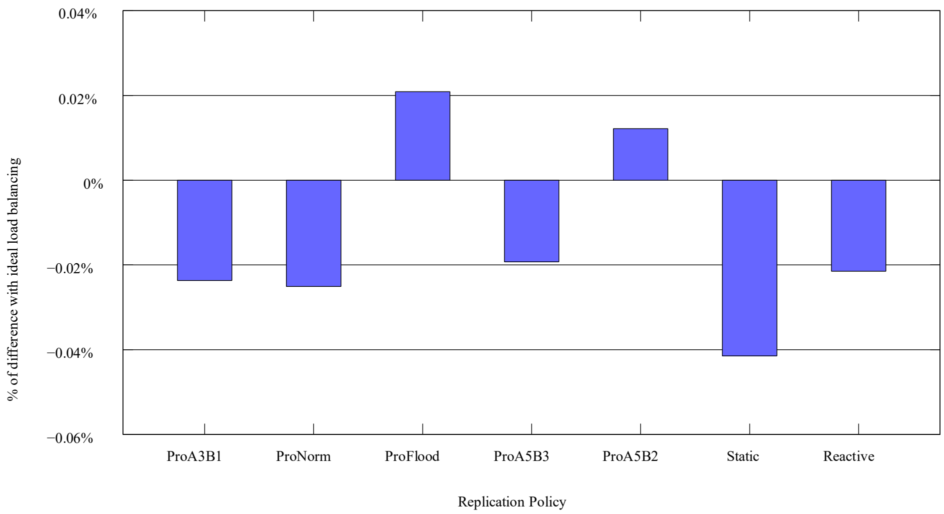

Fig. 8. Load balancing differences compared with ideal considering content supplier popularity.

5.2 Content topic popularity approach

This section presents results obtained from the evaluation of the prototype of the S-CDS platform implementing content replication based on the popularity of the content topic. Similar to the previous evaluation, in these experiments we used the metrics described in Section 4.2, the workloads prepared for topic classification (Section 4.4.1) and the test configurations (Section 4.1): Static/Norm, Reactive and Proactive, where Static/Norm or good practices configuration (with RF=3 for Alpha, Beta and Gamma topics) represents our baseline for comparison. These experiments were also carried out using Docker Virtual Containers (VC) instances, with the same configuration using for Virtual Machines (VMs), described in Table 1. Figure 9 shows the performance improvement of the SCDS platform, with respect to the baseline (Static configuration or good practice), implementing different replication policies based on popularity of content topics, considering upload and download operations separately and altogether (Global). We can see that most of ProActive configurations (except ProFlood on VMs) improved the good practice configuration. This demonstrates that a selective (Alpha, Beta and Gamma content topics) application of the replication factor can improve a generic application of the good practice, especially for the most popular topics (Alpha content topics, ProA3B1).

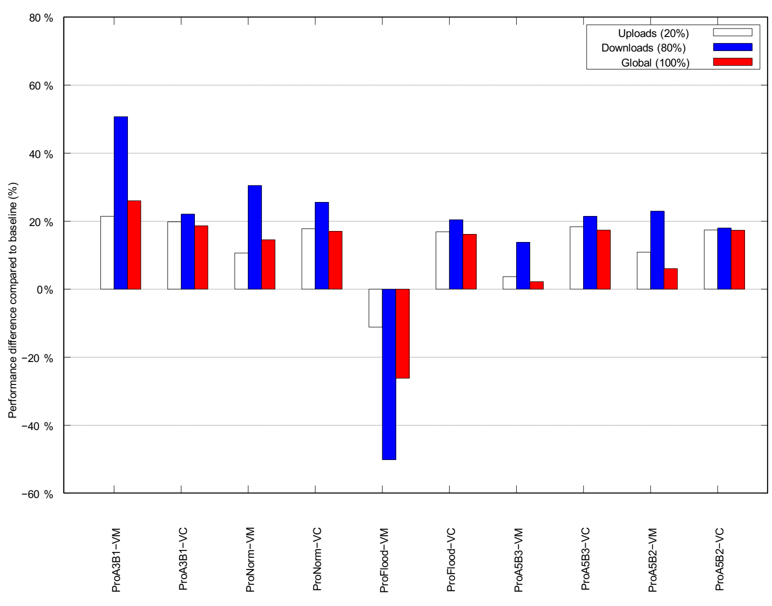

Fig. 9. Response time improvement w.r.t. baseline, considering replication based on content topic popularity.

Figure 10 contrasts performance improvement with storage consumption with respect to the baseline configuration, considering ProActive approaches (global median values). We can see that almost all ProActive configurations improve performance, except the ProFlood strategy implemented in VMs, which produces the worst of the two worlds: it affects performance negatively and requires a high storage consumption. The execution of this strategy on VCs showed better results than on VMs because the light weight virtualization offered by VCs. The best trade-off between storage consumption and performance is given by the ProA3B1 configuration, implemented in both VMs and VCs, which only generates replicas of the most popular content topics (Alpha topics).

5.3 Simulation and emulation to prepare large scale scenarios

As a complement of our S-CDS implementation, a trace-based simulator[3] and emulator was developed for cloud storage designers to conduct studies on large scale scenarios. In four phases, the designers can create a cloud storage solution based on the methodology proposed in this paper and even can deploy it by using the previously described S-CDS prototype.

. A simplified version of this simulator can be reached at http://disys0.tamps.cinvestav.mx:45000

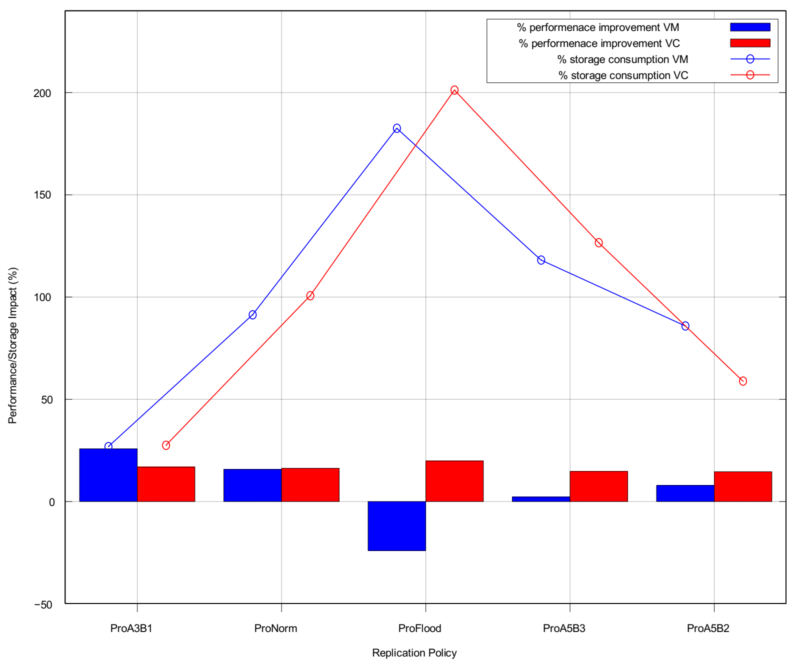

Fig. 10. Performance and storage impact with respect to the baseline (content topic popularity).

In the first phase, the designer chooses the number of nodes and features of these nodes for the simulator to create the configuration file of a cloud storage pool.

In the second one, the designer selects parameters in the probabilistic model to define the traffic and patterns of a file sharing scenario. This model considers parameters or user specifications to create pseudo-random events generated by using different random distributions. The events represent records that are stored in a *tracefile*, similar to those shown in Section 4.4. This trace includes the following fields: i) Request inter-arrival time, calculated by a traffic generator by using a Poisson distribution with mean inter-arrival as input parameter, and allowing to create bursty traffic; ii) The mean value of content/file size, which can be defined as a Poisson, Normal or Uniform distribution; and iii) The type of operation (Publish/Subscribe/Put/Get), calculated as a Pareto produced by random numbers following a Zipf distribution.

In the third phase, a training component of the S-CDS service creates a topic lexicon from a corpus chosen by the user according to data for sharing. At this point, the user can also calculate the α,β and γ for both users and topics by using both the lexicon and the trace file.

In the last part, the user launches the emulation of the data collected and calculated by the simulator, choosing a replication policy for α,β and γ classes and a load balancing method. The emulator returns the statistics of load balancing, storage consumption and a file of the topics demand/consumption to create a graph. At this point, the user could launch the configuration for an engine to deploy the cloud storage pool by using a cluster of virtual containers.

Although it is possible for users to provide the simulator with a real corpus and traces as input parameters of the S-CDS platform, this feature is not available in the online version of the simulator, which is not used for production purposes.

We carried out experiments with the simulator, defining a cloud infrastructure similar to that depicted in Table 1, but increasing the number of storage nodes. The simulator implements our classification model and allows users to modify configuration tests, workloads and replication policies. Test configurations and workloads defined in Sections 4.1 and 4.4 respectively were used in these experiments. Figure 11 shows the average MB per Node (vertical axis), in comparison with ideal workload per node, produced by proactive configurations when increasing the number of nodes in the cloud storage (horizontal axis). As expected, the more nodes in the cloud, the less the load per node is processed. This reveals that the load balancing of these configurations is preserved, improving performance (the performance graph is not shown here due to space reasons). The findings observed in these experiments were similar to those noticed in our S-CDS prototype.

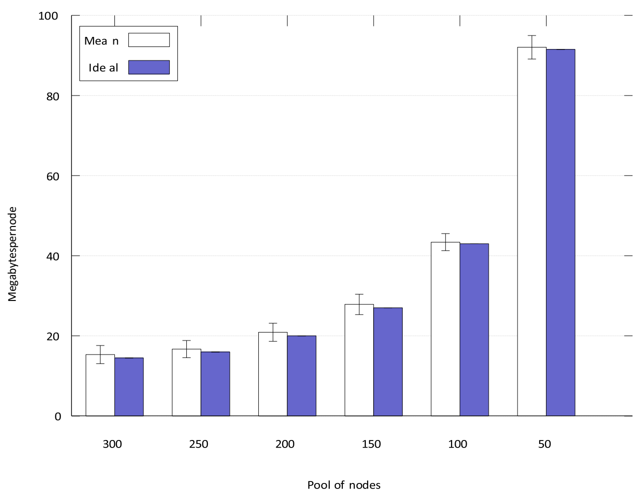

Fig. 11. Load balancing in a large scale scenario using a simulator that implements our methodology (content topic popularity).

6 CONCLUSIONS

This paper introduces a methodology to improve I/O performance and capacity utilization of cloud storage for Content Delivery and Sharing (CDS) services in an organizational scenario. Three main contributions were presented:
i) a classification model (defining Alfa, Beta and Gamma categories) to detect relevant users and content topics from their levels of activity (consumption/demand and sharing patterns); ii) a new availability and load balancing scheme for CDS storage services, whose decisions for replication and data placement are based on the classification made by the model; and iii) a prototype of a semantic CDS service implemented in a private cloud that integrates the availability and load balancing scheme. Results of the experiments showed the feasibility and efficiency of the methodology and provided encouraging insights about the importance of considering Volume and Density metrics as part of a new heuristic technique for improving performance and capacity utilization of CDS services in the cloud.

In future works, we plan to consider in our methodology the management of large and encrypted files, multicloud scenarios and synchronization techniques on the end-user side.

ACKNOWLEDGMENTS

This work was partially supported by the Spanish Ministry of Economy, Industry and Competitiveness under the grant TIN2016-79637-P ”Towards Unification of HPC and Big Data Paradigms”

REFERENCES

[1] D. Brame, “The best cloud storage and file sharing providers for business for 2019,” Sep. 27 2018. [Online]. Available: https://www.pcmag.com/roundup/350118/ the-best-cloud-storage-and-file-sharing-providers-for-busine

[2] Gartner, “Magic quadrant for public cloud storage services,” Jul. 31 2018.

[3] E. Bocchi, I. Drago, and M. Mellia, “Personal cloud storage benchmarks and comparison,” *IEEE Transactions on Cloud Computing*, vol. 5, no. 4, pp. 751–764, Oct. 2017.

[4] J. L. Gonzalez, J. C. Perez, V. J. Sosa-Sosa, L. M. Sanchez, and B. Bergua, “Skycds: A resilient content delivery service based on diversified cloud storage,” *Simulation Modelling Practice and Theory*, vol. 54, pp. 64–85, 2015.

[5] C. Pahl, A. Brogi, J. Soldani, and P. Jamshidi, “Cloud container technologies: a state-of-the-art review,” *IEEE Transactions on Cloud Computing*, p. 1, 2018.

[6] P. Morales-Ferreira, M. Santiago-Duran, C. Gaytan-Diaz, J. Gonzalez-Compean, V. J. Sosa-Sosa, and I. Lopez-Arevalo, “A data distribution service for cloud and containerized storage based on information dispersal,” in *Service-Oriented System Engineering (SOSE), 2018 IEEE Symposium on*. IEEE, 2018, pp.
86–95.

[7] H. Morshedlou and M. R. Meybodi, “Decreasing impact of sla violations:a proactive resource allocation approachfor cloud computing environments,” *IEEE Transactions on Cloud Computing*, vol. 2, no. 2, pp. 156–167, Apr. 2014.

[8] H. Yoon, A. Gavrilovska, K. Schwan, and J. Donahue, “Interactive use of cloud services: Amazon sqs and s3,” in *Proceedings of the 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (ccgrid 2012)*. IEEE Computer Society, 2012, pp. 523–530.

[9] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, “The hadoop distributed file system,” in *Proc. IEEE 26th Symp. Mass Storage Systems and Technologies (MSST)*, May 2010, pp. 1–10.

[10] J. Arnold, *Openstack swift: Using, administering, and developing for swift object storage*. ” O’Reilly Media, Inc.”, 2014.

[11] R. Honicky and E. L. Miller, “Replication under scalable hashing: A family of algorithms for scalable decentralized data distribution,” in *Parallel and Distributed Processing Symposium, 2004. Proceedings. 18th International*. IEEE, 2004, p. 96.

[12] J. Liu and H. Shen, “A low-cost multi-failure resilient replication scheme for high data availability in cloud storage,” in *Proc. IEEE 23rd Int. Conf. High Performance Computing (HiPC)*, Dec. 2016, pp. 242–251.

[13] J. Riley, J. Noss, W. Dillingham, J. Cuff, and I. M. Llorente, “A high-availability cloud for research computing,” *Computer*, vol. 50, no. 6, pp. 92–95, 2017.

[14] A. Pintilie, E. Apostol, and C. Dobre, “Efficient storage and replication solutions for healthcare applications,” in *Proc. IEEE Int. Black Sea Conf. Communications and Networking (BlackSeaCom)*, Jun. 2016, pp. 1–5.

[15] P. T. Endo, M. Rodrigues, G. E. Gonc¸alves, J. Kelner, D. H. Sadok, and C. Curescu, “High availability in clouds: systematic review and research challenges,” *Journal of Cloud Computing*, vol. 5, no. 1, p. 16, Oct 2016. [Online]. Available: https://doi.org/10.1186/s13677-016-0066-8

[16] J. M. Perez, F. Garcia, J. Carretero, A. Calderon, and L. M. Sanchez, “Data allocation and load balancing for heterogeneous cluster storage systems,” in *Proc. CCGrid 2003. 3rd IEEE/ACM Int. Symp. Cluster Computing and the Grid*, May 2003, pp. 718–723.

[17] E. Torres, G. Callou, G. Alves, J. Accioly, and H. Gustavo, “Storage services in private clouds: Analysis, performance and availability modeling,” in *Proc. and Cybernetics (SMC) 2016 IEEE Int. Conf. Systems, Man*, Oct. 2016, pp. 003 288–003 293.

[18] S. Ghemawat, H. Gobioff, and S.-T. Leung, “The google file system,” in *Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles*, ser. SOSP ’03. New York, NY, USA: ACM, 2003, pp. 29–43. [Online]. Available: http: //doi.acm.org/10.1145/945445.945450

[19] A. Chiniah and A. Mungur, “Dynamic erasure coding policy allocation (decpa) in hadoop 3.0,” in *Proc. 6th IEEE Int. Conf. Cyber Security and Cloud Computing (CSCloud)/ 2019 5th IEEE Int. Conf. Edge Computing and Scalable Cloud (EdgeCom)*, Jun. 2019, pp. 29–33.

[20] B. Kulkarni and V. Bhosale, “Efficient storage utilization using erasure codes in openstack cloud,” in *Proc. Int. Conf. Inventive Computation Technologies (ICICT)*, vol. 3, Aug. 2016, pp. 1–5.

[21] W. Li, Y. Yang, and D. Yuan, “A novel cost-effective dynamic data replication strategy for reliability in cloud data centres,” in *Proc. Autonomic and Secure Computing 2011 IEEE Ninth Int. Conf. Dependable*, Dec. 2011, pp. 496–502.

[22] Q. Wei, B. Veeravalli, B. Gong, L. Zeng, and D. Feng, “Cdrm: A cost-effective dynamic replication management scheme for cloud storage cluster,” in *Proc. IEEE Int. Conf. Cluster Computing*, Sep. 2010, pp. 188–196.

[23] N. Mansouri, M. K. Rafsanjani, and M. Javidi, “Dprs: A dynamic popularity aware replication strategy with parallel download scheme in cloud environments,” *Simulation Modelling Practice and Theory*, vol. 77, pp. 177 – 196, 2017. [Online]. Available: http:// www.sciencedirect.com/science/article/pii/S1569190X1730103X

[24] I. Kilanioti and G. A. Papadopoulos, “Content delivery simulations supported by social network-awareness,” *Simulation Modelling Practice and Theory*, vol. 71, pp. 114 –
133, 2017. [Online]. Available: http://www.sciencedirect.com/ science/article/pii/S1569190X16302696

[25] M. Mangili, F. Martignon, and A. Capone, “Performance analysis of content-centric and content-delivery networks with evolving object popularity,” *Computer Networks*, vol. 94, pp. 80 – 98, 2016. [Online]. Available: http://www.sciencedirect.com/ science/article/pii/S1389128615004521

[26] T. Hamrouni, S. Slimani, and F. B. Charrada, “A critical survey of data grid replication strategies based on data mining techniques,” *Procedia Computer Science*, vol. 51, pp. 2779 – 2788, 2015, international Conference On Computational Science, ICCS 2015. [Online]. Available: http://www.sciencedirect.com/ science/article/pii/S1877050915012429

[27] M. S. Q. Z. Nine, M. A. K. Azad, S. Abdullah, M. A. H. Monil, I. Zahan, A. B. Kader, and R. M. Rahman, “Application of association rule mining for replication in scientific data grid,” in *Proc. 8th Int. Conf. Electrical and Computer Engineering*, Dec. 2014, pp. 345–348.

[28] S. Slimani, T. Hamrouni, and F. B. Charrada, “New replication strategy based on maximal frequent correlated pattern mining for data grids,” in *Proc. Applications and Technologies 2014 15th Int. Conf. Parallel and Distributed Computing*, Dec. 2014, pp. 144–151.

[29] Z. Li, C. Jin, T. Xu, C. Wilson, Y. Liu, L. Cheng, Y. Liu, Y. Dai, and Z.-L. Zhang, “Towards network-level efficiency for cloud storage services,” in *Proceedings of the 2014 Conference on Internet Measurement Conference*. ACM, 2014, pp. 115–128.

[30] G. D. Gonc¸alves, I. Drago, A. V. Borges, A. P. Couto, and J. M. de Almeida, “Analysing costs and benefits of content sharing in cloud storage,” in *Proceedings of the 2016 workshop on Fostering Latin-American Research in Data Communication Networks*. ACM, 2016, pp. 43–45.

[31] Z. Li, C. Wilson, Z. Jiang, Y. Liu, B. Y. Zhao, C. Jin, Z.-L. Zhang, and Y. Dai, “Efficient batched synchronization in dropbox-like cloud storage services,” in *ACM/IFIP/USENIX International Conference on Distributed Systems Platforms and Open Distributed Processing*. Springer, 2013, pp. 307–327.

[32] G. Gonc¸alves, A. B. Vieira, I. Drago, A. P. C. Da Silva, and J. M. Almeida, “Cost-benefit tradeoffs of content sharing in personal cloud storage,” in *2017 IEEE 25th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*. IEEE, 2017, pp. 32–42.

[33] G. Gonc¸alves, I. Drago, A. P. C. Da Silva, A. B. Vieira, and J. M. Almeida, “The impact of content sharing on cloud storage bandwidth consumption,” *IEEE Internet Computing*, vol. 20, no. 4, pp. 26–35, 2016.

[34] T. T. Chekam, E. Zhai, Z. Li, Y. Cui, and K. Ren, “On the synchronization bottleneck of openstack swift-like cloud storage systems,” in *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*. IEEE, 2016, pp. 1–9.

[35] J. Gonzalez-Compean, V. Sosa-Sosa, A. Diaz-Perez, J. Carretero, and J. Yanez-Sierra, “Sacbe: A building block approach for constructing efficient and flexible end-to-end cloud storage,” *Journal of Systems and Software*, vol. 135, pp. 143 – 156, 2018. [Online]. Available: http://www.sciencedirect.com/ science/article/pii/S0164121217302212

[36] Z. Harris, “Distributional structure,” *WORD*, vol. 10, no. 2-3, pp. 146–162, 1954.

[37] J. Weeds and D. Weir, “Co-occurrence retrieval: a flexible framework for lexical distributional similarity,” *Computational Linguistics*, vol. 31, no. 4, pp. 439–475, 2005.

[38] “Download medline/pubmed data.” [Online]. Available: https:// www.nlm.nih.gov/databases/download/pubmed medline.html

[39] M. Bastian, S. Heymann, and M. Jacomy, “Gephi: An open source software for exploring and manipulating networks,” in *Proc. of the International AAAI Conference on Weblogs and Social Media*, 2009, pp. 1–2. [Online]. Available: http: //www.aaai.org/ocs/index.php/ICWSM/09/paper/view/154

[40] M. J. Luczak, C. McDiarmid *et al.*, “On the power of two choices: balls and bins in continuous time,” *The Annals of Applied Probability*, vol. 15, no. 3, pp. 1733–1764, 2005.

[41] “Openstack website,” online, 2018. [Online]. Available: https: //www.openstack.org/

[42] H. Sayyadi and L. Raschid, “A graph analytical approach for topic detection,” *ACM Transactions on Internet Technology (TOIT)*, vol. 13, no. 2, p. 4, 2013.

[43] H.-Y. Hsueh, J.-S. Hua, S.-M. Huang, and H. J. Will, “Resource sharing behavior in a socialized peer-to-peer internet environment,” in *Mobile Business, 2009. ICMB 2009. Eighth International Conference on*. IEEE, 2009, pp. 131–136.