Computing
https://doi.org/10.1007/s00607-024-01281-2

SPECIAL ISSUE ARTICLE

Cloud storage tier optimization through storage object classification

Akif Quddus Khan[1] · Mihhail Matskin[2] · Radu Prodan[3] · Christoph Bussler[4] · Dumitru Roman[5,6] · Ahmet Soylu[6]

Received: 9 December 2023 / Accepted: 14 March 2024

Abstract
Cloud storage adoption has increased over the years given the high demand for fast processing, low access latency, and ever-increasing amount of data being generated by, e.g., Internet of Things applications. In order to meet the users' demands and provide a cost-effective solution, cloud service providers offer tiered storage; however, keeping the data in one tier is not cost-effective. In this respect, cloud storage tier optimization involves aligning data storage needs with the most suitable and cost-effective storage tier, thus reducing costs while ensuring data availability and meeting performance requirements. Ideally, this process considers the trade-off between performance and cost, as different storage tiers offer different levels of performance and durability. It also encompasses data lifecycle management, where data is automatically moved between tiers based on access patterns, which in turn impacts the storage cost. In this respect, this article explores two novel classification approaches, rule-based and game theory-based, to optimize cloud storage cost by reassigning data between different storage tiers. Four distinct storage tiers are considered: premium, hot, cold, and archive. The viability and potential of the proposed approaches are demonstrated by comparing cost savings and analyzing the computational cost using both fully-synthetic and semi-synthetic datasets with static and dynamic access patterns. The results indicate that the proposed approaches have the potential to significantly reduce cloud storage cost, while being computationally feasible for practical applications. Both approaches are lightweight and industry- and platform-independent.

Keywords Storage tiers · Storage cost · Cloud computing · StaaS · Cloud storage · Cost optimization

Mathematics Subject Classification 68M14 · 68M11 · 68T09

Extended author information available on the last page of the article

1 Introduction

The volume of data generated, collected, processed, and used for advanced analytics through artificial intelligence (AI) and machine learning (ML) methodologies continues to expand drastically [43]. This expansion is propelled by the pervasive use of information and communication technologies (ICT), including applications such as social media, the Internet of Things (IoT), and sensor networks. For example, Heinrich from Lucid Motors [9] shares some estimates on the raw sensor data from autonomous vehicles (AV), ranging from 1.4 to 19 TB per hour. Similarly, according to Amend [2], Hyundai generates about 10 GB of data every second from its AV prototypes; while some of the data is stored on the vehicle, the rest is uploaded to the cloud. Cloud computing is pivotal in managing this data surge, offering the scalability needed for storage and computational resources. Furthermore, it delivers cost-effectiveness and an array of quality-of-service (QoS) attributes, including availability and security, via the storage-as-a-service (StaaS) paradigm [21]. According to a survey among record-keeping professionals [23], 86% of the respondents opt for cloud storage to save costs. Due to this, cloud computing, in general, and cloud storage, in particular, have experienced exponential growth in recent years [27, 28, 33, 34]. Organizations have increasingly embraced cloud services to meet their computing needs. According to Gartner, 85% of enterprises are expected to adopt a cloud-first approach by 2025 [32].

The use of cloud storage, i.e., StaaS, [21], instead of local storage, can provide more flexibility in terms of scalability, fault tolerance, and availability. Cloud storage systems (e.g., Amazon S3, Azure Blob Storage, and Google Cloud Storage) offer large storage capacity with high fault tolerance, addressing several big data-related storage concerns [41]. Leading cloud service providers (CSPs), such as Microsoft Azure, Google Cloud, and Amazon S3, offer four storage tier options and pricing policies tailored to their specific data storage and access requirements. Storage tiers categorize data based on access frequency, performance needs, and cost. The *premium tier* is for high-frequency and high-performance data. The *hot tier* is for data with frequent access patterns, such as daily or weekly, and requires fast access times. The *cold tier* is for infrequently accessed data such as backups or archives. The *archive tier* is for rarely accessed data with flexible latency requirements, offering the lowest storage cost but the highest network usage cost. Moving from premium to archive, storage cost decreases, while network usage cost increases. For example, for an application with many data access requests, the cost of the data stored in the premium tier is the cheapest, since the premium tier offers free data retrieval. However, for an application with storage objects with diverse access patterns, it is not cost-effective to keep the data stored in one tier all the time [15]. This presents an opportunity for users to optimize their StaaS cost. For example, Google Cloud Storage provides premium, hot, cold, and archive storage tiers. The pricing structure varies across these tiers, with the hot tier offering lower access prices but higher storage costs and the cold tier offering higher access prices but lower storage cost. This means that storing data objects with infrequent access in the cold tier can result in lower expenses compared to the hot tier. As a result, StaaS users can strategically migrate their data, i.e., perform storage tier optimization from the hot tier to the cold tier when access demands decrease, to reduce the overall cost.

Storage tier optimization organizes data into different tiers based on usage and performance requirements [36]. This can help improve storage performance and efficiency by ensuring that the most frequently accessed data is stored on the fastest tier, while less frequently accessed data can be stored on a slower, less expensive tier. There are many different ways to implement storage tiering. One common method is using a storage array with multiple tiers of storage media, such as high-performance flash storage, mid-range spinning disk drives, and low-cost nearline or offline storage. The storage array can then automatically move data between tiers based on its usage patterns [39]. Another common method of storage tiering is to use a software-defined storage solution [19]. These solutions typically provide a more flexible and scalable approach to storage tiering than traditional storage arrays. Software-defined storage solutions can also tier data across multiple physical storage locations, such as on-premise and cloud storage. Storage tier optimization can be complex, but it can offer significant performance, efficiency, and cost savings benefits. Frequent data migrations can increase expenses, undermining efforts to optimize the cost. Additionally, the cold storage tier typically imposes a minimum storage duration requirement. This means users may face extra charges if the data objects do not meet the required minimum storage duration (e.g., 30 days for Amazon S3) before being moved to the cold tier. This limitation can hinder the advantages of data migration, making cost optimization a complex task for cloud storage users. To navigate this challenge successfully, users must diligently leverage the pricing disparities between different storage tiers and judiciously plan their data migration strategies. Otherwise, even a small oversight can result in unexpected additional costs. By carefully planning and implementing a storage tiering strategy, organizations can improve the performance of their storage infrastructure and reduce their storage cost.

To this end, in this article, we focus on moving data between different tiers in a single region and explore storage tier optimization for cost-effective data storage using rule-based and game theory-based classification approaches. These two approaches consider four storage tiers instead of just two, do not require intense computing resources, and are platform-independent, lightweight, and fast. Three integral data characteristics are considered: size, access frequency, and age. For the rule-based approach, we propose a set of rules for calculating a score or priority score and define a threshold to classify each object stored in cloud storage into premium, hot, cold, or archive tiers. The approach also allows users to set the priority or weight of each data characteristic. It also has the ability to find the most suitable weights to achieve maximum cost reduction. The game theory-based approach, on the other hand, is an iterative approach that has the ability to take into account the feedback from previous iterations. To demonstrate the viability and potential of the proposed approaches, we use an evaluation strategy involving the generation of fully- and semi-synthetic datasets, encompassing various variations to provide a realistic representation of cloud storage usage. Subsequently, the proposed approaches are assessed against these datasets, showcasing their effectiveness in substantially reducing storage cost across static and dynamic access patterns. We also present a computational cost analysis to demonstrate the feasibility of the proposed approaches for practical applications in real-world cloud environments.

This article is an extension of our preliminary work on a rule-based classification approach on cloud storage tier optimization [12]; more specifically, it extends our earlier work by providing:

- a novel game theory-based approach as an alternative to the rule-based approach, which was previously introduced only at a conceptual level;
- an extensive and comparative evaluation of the rule-based and game theory-based approaches across various scenarios and multiple datasets;
- an evaluation of the computational cost of both approaches;
- a discussion of the results, including the strengths and limitations of the proposed approaches;
- a comprehensive set of notations and derivation of equations for the cost estimation; and,
- a more comprehensive overview of the related scientific literature and directions for future work.

The rest of the article is structured as follows. Section 2 overviews the cloud storage cost elements. Section 3 introduces the two proposed approaches (Sect. 3.1 for the rule-based classification approach and Sect. 3.2 for the game theory-based classification approach). Section 4 presents an evaluation strategy and the results. Section 5 presents an analysis of the computational cost, whereas Sect. 6 discusses the proposed approaches' results and limitations. Section 7 provides a summary and a discussion of the related work. Finally, Sect. 8 concludes the article and presents some directions for future work.

2 Cloud storage cost

The cost structure of cloud storage is based on a complex ecosystem. There are several pricing models, such as block pricing and pay-as-you-go, as well as various cost elements, both mandatory and optional. For example, network usage is a mandatory expense when storing data in the cloud, while data management and security costs are optional. For cloud storage cost, we consider the following five non-exhaustive main categories [13]: (1) data storage; (2) network usage; (3) transaction; (4) data retrieval; and (5) data replication/migration. To optimize different elements of cloud storage cost, it is important first to understand what they are and how they contribute to the total cost. Hence, in this section, we discuss each element separately. Some CSPs have a policy of a minimum storage duration for each storage tier that varies from 30 to 180 days, with the former applying to premium and hot tiers and the latter to cold and archive tiers. An early migration fee is charged if data is moved to another tier. We did not include the early migration fee for our cost estimations, but the evaluation is conducted for data stored for at least 90 days. Furthermore, we also developed mathematical formulas to calculate the total cost of the respective

Table 1 Cost of data storage by Google Cloud in a single region, Europe - Warsaw (europe-central2)

Cost element	Premium	Hot	Cold	Archive
Official term	Standard	Nearline	Coldline	Archive
Storage cost ($\GB\month)	0.023	0.013	0.006	0.0025
GET request ($ per 1000)	0.0004	0.001	0.01	0.05
PUT request ($ per 1000)	0.005	0.01	0.02	0.05
Data retrieval ($\GB)	0	0.01	0.02	0.05
Network usage ($\GB)	0.12	0.12	0.12	0.12[a]
Minimum duration (days)	None	30	90	365
Latency	Low[b]			
Durability	99.999999999%[c]			
Availability	Multi-region: > 99.99% Dual-regions: >99.99% Regions: 99.99%	99.95% 99.95% 99.9%	99.95% 99.95% 99.9%	99.95% 99.95% 99.9%

The cost of data storage is different for dual- and multi-region. Data collected on May 12, 2023

[a]Cost of data egress to worldwide destinations (excluding Asia & Australia)

[b]Time to first byte typically tens of milliseconds

[c]https://cloud.google.com/blog/products/storage-data-transfer/understanding-cloud-storage-11-9s-durability-target

services, which are further utilized in the proposed approaches and during evaluation. Table 1 shows the actual prices of different cost elements of cloud storage by using Google Cloud[1,2] as an example, while a comprehensive set of notations is listed in Table 2 to provide a clear and consistent reference framework throughout the article.

2.1 Storage cost

Storage cost refers to the cost of storing data in the cloud. It is charged on a per-GB-per-month basis. Since multiple tiers have unique characteristics, each storage tier also has its own pricing structure. It is important to note that the cost of storing data in each tier also depends on the amount of data being stored, as some CSPs offer block-rate pricing, i.e., the larger the amount of data, the lower the unit costs are [26]. For example, there is a certain cost for data between 0 and 50 TB, and then for some tiers, it might be cheaper for data over 50 TB. However, this article does not consider block-rate pricing when calculating cost estimates. The selection of storage tier affects the storage cost and other relevant costs, such as network usage and data retrieval. In the scope of this article, cost estimation and evaluation are performed based on the accumulated cost of the amount of data stored and the period of time

[1] https://cloud.google.com/storage/pricing.

[2] https://cloud.google.com/storage/docs/storage-classes.

Table 2 Summary of notations

Description	Notation	Example
Premium tier	p	
Hot tier	h	
Cold tier	c	
Archive tier	a	
Data storage cost (recall Table 1)	$S_{(x)}$	S_p, S_h, S_c, S_a
Network usage cost (recall Table 1)	$NW_{(x)}$	NW_p, NW_h, NW_c, NW_a
Data retrieval cost (recall Table 1)	$R_{(x)}$	R_p, R_h, R_c, R_a
Cost for GET requests (recall Table 1)	$G_{(x)}$	G_p, G_h, G_c, G_a
Cost for PUT requests (recall Table 1)	$P_{(x)}$	P_p, P_h, P_c, P_a
Total size of data	σ	
Size of storage object in GB	Z	
Age of storage object	A	
Object storage time in each tier (Subset of A)	$t(x)$	$t(p), t(h), t(c), t(a)$
R/W access frequency to an object in a time period	F_x	F_p, F_h, F_c, F_a
Weights	$W_{(x)}$	W_s, W_f, W_a
Object size score	α	
Object access frequency score	β	
Object age score	γ	
Priority score	λ	
Data storage cost in a specific tier	C_{S_x}	$C_{S_p}, C_{S_h}, C_{S_c}, C_{S_a}$
Total data storage cost for period A	$C_S(A)$	
Network usage cost in a specific tier	C_{NW_x}	$C_{NW_p}, C_{NW_h}, C_{NW_c}, C_{NW_a}$
Total network cost usage for period A	$C_{NW}(A)$	
Transaction cost in a storage tier	C_{T_x}	$C_{T_p}, C_{T_h}, C_{T_c}, C_{T_a}$
Total transaction cost for period A	$C_T(A)$	
Data retrieval cost in a specific storage tier	C_{R_x}	$C_{R_p}, C_{R_h}, C_{R_c}, C_{R_a}$
Total data retrieval cost for period A	$C_R(A)$	
Data migration cost	C_M	
Accumulated storage cost for object i in each tier	$C(i)_x$	$C(i)_p, C(i)_h, C(i)_c, C(i)_a$
Data storage cost matrix of object i in tier j	$CM(i,j)$	
Probability matrix for storing object i in tier j	$P(i,j)$	

it remains in each storage tier. Equation 1 is formulated accordingly to calculate the storage cost. Denoting C_{S_x} as the cost in Premium (p), Hot (h), Cold (c), or Archive (a) tiers for any given amount of time $t(x)$ (recall Table 2), the total storage cost can be calculated using Eq. 1:

$$C_{S_x} = \left(t(x) \cdot S_x \cdot Z\right), \text{where x} \in \{p, h, c, a\}. \tag{1}$$

Here, S_x is the data storage cost, and Z is the size of the storage object in GB. An object's total accumulated storage cost across different tiers for its entire age (A) can be calculated using Eq. 2:

$$C_S(A) = \sum_{x=p,h,c,a} \left(C_{S_x} \right). \tag{2}$$

2.2 Network usage cost

The quantity of data read from or sent between the buckets is known as network consumption or usage. The HTTP response headers reflect data transmitted by cloud storage through egress. Hence, network usage cost is defined as the bandwidth cost out of the cloud storage server. It is charged on a per-GB basis. In addition, network usage cost also varies based on the amount of data transferred, as it offers different slabs for different amounts of data. The higher the amount of data transferred, the cheaper the cost. Google Cloud offers two separate network tiers, each with its own pricing structure and quality of service characteristics. Microsoft Azure and Amazon S3 do not offer multiple network tiers; however, in all three CSPs, network usage cost varies based on the storage tier in which data is stored. Equations 3 and 4 are formulated to calculate the network usage cost for an object stored in each storage tier for a certain amount of time and the accumulated network usage cost for the whole lifespan of a storage object, respectively:

$$C_{NW_x} = \left(NW_x \cdot F_x \cdot Z \right), \text{where x} \in \{p, h, c, a\}. \tag{3}$$

$$C_{NW}(A) = \sum_{x=p,h,c,a} \left(C_{NW_x} \right). \tag{4}$$

Here, NW_x is the cost of network usage for tier x, F_x denotes the access frequency, and Z is the size of storage object in GB (recall Table 2).

2.3 Transaction cost

Transaction cost refers to managing, monitoring, and controlling a transaction when reading or writing data to cloud storage [29]. Regarding data storage, cloud providers charge for the amount of data transferred over the network and the number of operations it takes. Transaction cost deals with the number of operations. These costs are associated with managing, monitoring, and controlling a transaction when reading or writing data to cloud storage. Different types of transactions incur different costs. For example, READ and WRITE operations carry distinct charges determined by the number of requests made. DELETE and CANCEL requests, on the other hand, are free. The types of requests include PUT, COPY, POST, LIST, GET, and SELECT. In Google Cloud, transaction cost is referred to as “operation charges”, encompassing the expense incurred for all requests made to Google Cloud Storage. Since we are taking into account mainly the cost of GET requests, Eqs. 5 and 6 are formulated to calculate the transaction cost for an object stored in each storage tier for a certain amount of time and accumulated transaction cost for the whole lifespan of a storage object respectively in different storage tiers:

$$C_{T_x} = \left(G_x \cdot F_x \right), \text{where x} \in \{p, h, c, a\}. \tag{5}$$

$$C_T(A) = \sum_{x=p,h,c,a} \left(C_{T_x} \right) + P_x. \tag{6}$$

Here G_x denotes the cost for GET request whereas F_x denotes the access frequency, i.e., the number of times a storage object is accessed; similarly, P_x denotes the cost for PUT request (recall Table 2).

2.4 Data retrieval

Data retrieval fees refer to the charges incurred when retrieving or accessing data from a storage system or service. Data retrieval fees may apply when retrieving stored files or information in various cloud storage or object storage platforms. These fees are typically associated with the data transfer or bandwidth used during retrieval. However, it is also important to know that data retrieval cost is in addition to the cost of network usage and is charged on a per-GB basis. Data access frequency in this context is important when considering data retrieval's impact on cost and data availability. Like network usage, data retrieval cost also varies based on the tier in which the data is stored. Google Cloud charges no fee for data retrieval if the data is stored in the premium tier, making it a highly suitable choice for frequently accessed data; however, network usage cost still applies. Therefore, carefully considering this cost trade-off is necessary when moving data from the premium tier to other tiers. Equations 7 and 8 are used to calculate the cost of data retrieval in each storage tier for a certain amount of time and accumulated data retrieval cost for the whole lifespan of a storage object, respectively in different storage tiers:

$$C_{R_x} = \left(Z \cdot R_x \right), \text{where x} \in \{p, h, c, a\}. \tag{7}$$

$$C_R(A) = \sum_{x=p,h,c,a} \left(C_{R_x} \right). \tag{8}$$

Here, Z denotes the storage object size in GB, while R_x denotes the cost of data retrieval per GB of data (recall Table 2).

2.5 Migration cost

Different CSPs provide the capability to migrate data objects between tiers throughout their lifecycles, presenting a valuable opportunity for cost optimization. The migration process involves retrieving the complete object from the source tier and submitting a PUT request to the destination tier to inform it of the impending object. As such, the data migration operation is subject to expenses associated with data retrieval, calculated based on the object size in the source tier and expenses associated with the PUT request in the destination tier. Moreover, there is a penalty cost associated with the migration of storage objects from one tier to another if data is moved before a specific period of time. The time period varies for each specific storage tier, with a shorter minimum storage time requirement for premium and hot tiers and longer for cold and archive tiers. In this article, we are not factoring in the cost of migration, neither for cost estimation nor evaluation purposes. However, in the future, this cost can be calculated using Eq. 9. Here Z denotes the size of storage object in GB, C_{MN} denotes the cost of network usage, and C_T denotes the cost of a transaction (GET/PUT) (recall Table 2).

$$C_M = C_{WN} \cdot Z + C_T. \tag{9}$$

2.6 Total cost

In the context of evaluation, and specifically, while classification in the game theory-based approach, we rely on the mathematical expressions represented by Eqs. 1, 3, 5, and 7. These equations are the building blocks of our analysis, allowing us to assess the cost components with precision. They play an important role in the formulation of Eq. 10, which is further used to calculate the total cost of an object, denoted as i, stored within a particular storage tier, denoted as x.

$$C(i)_x = C_{S_x} + C_{WN_x} + C_{T_x} + C_{R_x}. \tag{10}$$

The equations merely reflect the complexity hidden within the cloud storage cost structure. The total cost encompasses multiple cost elements and unique costs associated with four different tiers. It is important to note that these equations estimate the cost that a user might incur when utilizing a cloud storage service, but these do not provide the exact cost. Additionally, they are for data stored in one region with a single redundancy model, excluding the cost of data migration between different regions. Furthermore, these equations are a fundamental component of the proposed methodology and pivotal in optimizing resource allocation and facilitating well-informed decision-making.

3 Cloud storage tier optimization

Cloud storage tier optimization involves classifying storage objects into different storage tiers to reduce cost. Several studies are dedicated to addressing this particular challenge, discussed in Sect. 7. In this section, we propose two novel approaches:

rule-based classification and game theory-based classification. Each of these approaches takes into account four distinct storage tiers. The following are general rules used to determine which characteristics are appropriate for each tier.

1. **Premium tier** should be used for data with the highest frequency of access, such as data accessed continuously or near-continuously, requiring the highest performance and durability levels. For example, mission-critical databases or high-performance computing workloads. The premium tier has the highest data storage cost but a lower network usage cost, making it suitable for data with a lower volume and higher access frequency.
2. **Hot tier** should be used for data with frequent access patterns, such as data that is accessed daily or weekly and requires fast access times. For example, this might include frequently accessed files, frequently used application data, or logs that require analysis regularly. The hot tier has a slightly lower cost of data storage than the premium tier, but it is still higher than the other tiers and slightly more expensive in network usage than the premium tier.
3. **Cold tier** should be used for data with infrequent or irregular access patterns, such as data accessed monthly, quarterly, etc. For example, backups, archives, or historical data that are rarely accessed but need to be kept for long periods for compliance or other reasons. This tier offers a cheaper cost for data storage, making it suitable for large amounts of data, but a higher network usage cost, only suitable for rarely accessed data.
4. **Archive tier** is designed for rarely accessed data with minimal retrieval requirements. It is typically used for long-term storage and compliance purposes. This tier is suitable for data with infrequent access patterns, such as annual or less frequent. It is also suitable for long-term backups (i.e., backups that must be kept for years), compared to the more short-term ones. In terms of cost, this tier offers the cheapest cost for data storage but the most expensive network usage cost. In summary, as we move towards the archive tier from the premium tier, storage cost decreases while network usage cost increases.

3.1 Rule-based classification approach

The term rule-based classification can refer to any classification scheme that uses IF-THEN rules for class prediction [37], as depicted in Fig. 1. The classification model can be deployed on a compute or serverless instance within the same zone/region as the storage instance. Object meta-data and access logs are fetched and then combined with user-defined weights to calculate the priority score, and based on that, each object is classified into one of the four tiers within the same storage instance. The proposed approach does not classify or move data between different regions.

In this approach, we define rules that assign each object to a storage tier based on specific criteria, such as the frequency of access, the data size, and the stored object's age. For example, we define a rule that assigns objects accessed frequently to a high-performance storage tier and those accessed less frequently to a lower-performance storage tier.

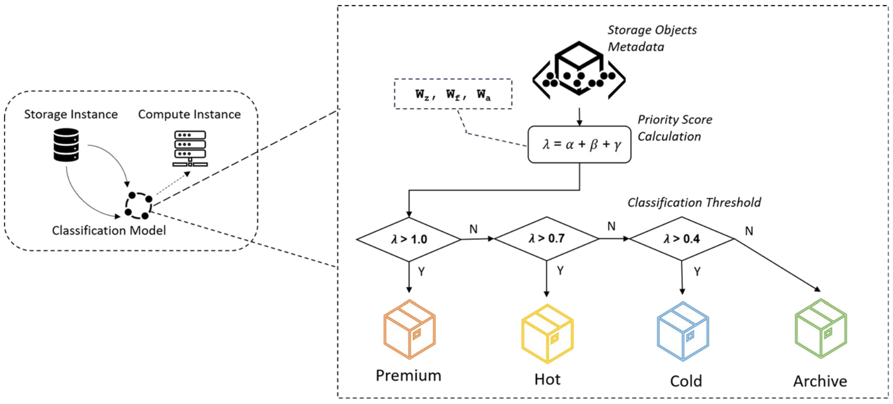


Fig. 1 The complete process of the proposed rule-based classification approach. The classification model can be deployed on a compute or serverless instance in the same zone with the storage server. Object meta-data is fetched and then combined with user-defined weights to calculate the priority score, and based on that, each object is classified into one of the four tiers

The process of the proposed approach is as follows. We first define the weights (W) for each factor (size (Z), access frequency (F), and age (A)) as W_z, W_f, and W_a, respectively (recall Table 2). Then, the data is defined as a list of dictionaries, where each dictionary represents an object and contains its size, access frequency, and age. Afterward, the priority score for each object is calculated using the defined weightings using Eq. 11 for size score (α), Eq. 12 for access frequency score (β), Eq. 13 for age score (γ) and Eq. 14 for calculating priority score (λ):

$$\alpha = W_z \cdot \log_{10}(Z). \tag{11}$$

$$\beta = W_f \cdot \log_{10}(F). \tag{12}$$

$$\gamma = W_a \cdot \begin{cases} \left(\frac{A}{12}\right) & \text{days} >= 100 \\ A & \text{days} < 100. \end{cases} \tag{13}$$

$$\lambda = \alpha + \beta + \gamma. \tag{14}$$

For the access frequency, the following are the nineteen possible windows: hourly (1, 2, 3, 4, 6, 8, 12), daily (1, 2, 3, 4), weekly (1, 2), monthly (1, 2, 3, 4, 6), and yearly. In this case, we have taken the frequency for a 90-day period and the whole period for which data was stored. The weight (W) of data indicates its priority or significance, allowing for varied importance levels across data objects, often determined by business criteria; for instance, vital data could bear a greater weight, directing it to higher-tier storage. Data size, access frequency, and age are pivotal determinants in storage choices, where larger values may entail increased storage cost. Applying the logarithm of these values, such as $\log_{10}(X)$, facilitates data normalization and mitigates the potential dominance of extreme values in classification.

This logarithmic transformation ensures a balanced scale for storage tiers, effectively accommodating a wide range of data sizes.

In this context (recall Table 2):

- Z represents the size of the storage object in Gigabytes (GB);
- F denotes the total number of R/W operations for an object in a specified period of time; and,
- A represents the age of the data in months.

Finally, the objects are divided into groups based on the available storage tiers by iterating over each object and checking if its priority score is greater than or equal to the threshold for each tier. If so, it is added to the corresponding group.

3.1.1 Priority score threshold

We set priority scores to classify each object into premium (1.0), hot (0.7), cold (0.4), or archive (0.1) tiers.

The selection of priority score thresholds for each storage tier aims to balance the trade-offs between data size, access frequency, and age. The premium tier, with a threshold of 1.0, represents the highest priority for critical and frequently accessed data. This tier ensures fast and reliable access to the most valuable information. The hot tier, set at 0.7, accommodates data with slightly lower priority but still significant access requirements. It provides a balance between performance and cost for frequently accessed data. With a threshold of 0.4, the cold tier caters to less frequently accessed data, offering cost-effective storage without compromising data availability. Lastly, archive tier 0.1 is a long-term storage solution for rarely accessed data, providing cost optimization while preserving data retention. These thresholds enable the effective allocation of data to the appropriate storage tiers based on their priority scores, ensuring optimal cost management while meeting the needs of data access and availability.

The selection of these specific thresholds, such as 0.7 for the hot tier, is based on the characteristics of the data and the need to balance several factors, such as data access frequency, storage object size, and storage duration. Data in the hot tier is accessed frequently; therefore, it must be readily available. A higher threshold means more data will be classified as "hot" and kept readily accessible, but if the object size is large, this comes at a higher storage cost. Another factor is the cost; lowering the threshold to, for example, 0.65 might include more data in the hot tier, hence increasing the cost if the volume is too high. The chosen threshold of 0.7 results from a cost-benefit analysis, ensuring that the most frequently accessed data is readily available without incurring excessive costs. However, these thresholds for data classification are not immutable. They can be adjusted based on changes in data access patterns, system performance, and cost considerations.

3.2 Game theory-based approach

Game theory [25] is the study of mathematical models of strategic interaction between rational decision-makers. It can analyze and optimize decision-making in various scenarios, including storage tier optimization. In the context of storage tier optimization, a game theory-based approach could be used to model the decision-making of various actors involved in the system, such as users, applications, and storage providers. For example, a game theory-based approach could be used to model the decision-making of users who access data stored in different storage tiers. By considering the trade-offs between access latency, data transfer cost, and storage cost, it is possible to identify the optimal storage tier for each object based on the users' access patterns. Similarly, a game theory-based approach could be used to model the decision-making of storage providers who must allocate resources to different storage tiers based on the demand from users and the cost of storage.

One possibility is using a multi-armed bandit problem variant [4, 5], where the agents are the arms and the storage tiers are the bandits. In this way, each agent maintains a probability distribution over the storage tiers and selects a storage tier to store an object based on the distribution. The distribution is updated based on the feedback from the system, which includes the storage cost and the retrieval time. The objective is to minimize the total cost while ensuring the retrieval time meets the service-level agreements. One possible implementation of this solution is the Thompson Sampling algorithm [35], which is a Bayesian approach to the multi-armed bandit problem. In this algorithm, each agent maintains a Beta distribution over the storage tiers, where the distribution parameters represent the number of successes and failures in selecting a storage tier. The agent selects a storage tier based on the highest sampled value from the distribution. The update of the distribution is done after the storage operation is completed, based on the feedback from the system. Specifically, if the storage cost is lower than the expected cost from the distribution, the Beta distribution parameters are updated to reflect success. Similarly, if the retrieval time is longer than the expected retrieval time from the distribution, the Beta distribution parameters are updated to reflect a failure.

Applying Bayesian game theory to the storage tier optimization problem involves modeling the decision-making process of multiple players (e.g., data objects, users, or applications) and their interactions. A general outline of the steps involved in deriving the formulas for Bayesian game theory is given in the following.

1. Define the players, i.e., to identify the relevant players in the game, such as data objects or applications that need to choose a storage tier. In this scenario, it's just the software application that is responsible for doing storage object classification.
2. Specify the strategies, i.e., to determine the possible strategies or choices that each player can make. The strategies are the different storage tiers available (i.e., premium, hot, cold, and archive).
3. Assign payoffs, i.e., define each player's payoffs or utility functions, which represent their preferences or objectives. The payoffs could be based on factors like cost, performance, durability, or access frequency, but in this case, we will choose

the cost. Assigning numerical values to the cost factor allows for quantifying the payoffs.

4. Specify beliefs, i.e., to determine each player's beliefs about the other players' choices and payoffs. These beliefs are based on historical data about the cost-effectiveness of previous decisions.

Based on the information above, we derived a formula for storage tier optimization using Bayesian game theory using the following variables (recall Table 2).

- Number of data objects N.
- Set of available storage tiers T.
- Cost matrix of storing data object i in storage tier j: $CM = \begin{bmatrix} C(1)_p & C(1)_h & C(1)_c & C(1)_a \\ C(2)_p & C(2)_h & C(2)_c & C(2)_a \\ C(3)_p & C(3)_h & C(3)_c & C(3)_a \\ \vdots & \vdots & \vdots & \vdots \\ C(N)_p & C(N)_h & C(N)_c & C(N)_a \end{bmatrix}$
- Probability that data object i is assigned to storage tier j based on historical data $P(i, j)$.

We define the objective as minimizing the total cost, considering the size, age, and access frequency of each data object, defined according to Eq. 15:

$$\sum_{i=1}^{N} \left(\sum_{j} P(i,j) \cdot CM(i,j) \right) \quad \forall j \in T. \tag{15}$$

- *Probability Matrix* P of shape $N \times 4$, where N is the number of data objects, and 4 represents the storage tiers. Initially, P is initialized with: $\frac{1}{4}$ for all elements.
- *Expected Cost Matrix* (EC) of the same shape as cost matrix C, representing the expected cost for each data object and storage tier. We calculate the EC for each data object i in each storage tier j based on the existing probabilities:

$$EC(i,j) = \sum_{k} \left(P(i,k) \times CM(i,k) \right) \quad \text{for all } k \in T \tag{16}$$

To achieve this, we propose the following steps.

1. Calculate the expected total cost or ETC for each storage tier j based on the expected costs of all data objects assigned to it:

$$ETC(j) = \sum_{i=1}^{N} \left(P(i,j) \times \mathrm{EC}(i,j) \right) \quad \text{for all } i \in N \tag{17}$$

2. Update the probabilities P based on the observed historical data and the cost associated with each storage tier in previous assignments using the following formula:

$$P(i,j) = \exp\left(- \sum_{j} |CM(i,j) - ETC(j)| \right) \quad \text{for all } j \in T \tag{18}$$

3. Calculate the minimum expected cost along with the corresponding storage tier:

$$\text{Cost\_i} = \sum_{k=1}^{T} \left(P(i,k) \times CM(i,j) \right) \text{for all } i \in N. \tag{19}$$

Using Eq. 19, the minimum cost ($\wedge$) is updated if the calculated value is less than the current value:

$$\wedge_i = \begin{cases} Cost\_i & \text{if } Cost\_i < \wedge_i \\ \wedge_i & \text{otherwise} \end{cases}. \tag{20}$$

Similarly, using the value acquired from Eq. 19, the appropriate storage tier ($Tier_i$) is selected, which is the one with the minimum cost:

$$Tier_i = \begin{cases} \text{j} & \text{if } Cost\_i < \wedge_i \\ \text{None} & \text{otherwise.} \end{cases} \tag{21}$$

4. Assign each data object i to the storage tier j that minimizes its expected cost:

$$P(i,j) = \begin{cases} 1 & \text{if } j = Tier_i \\ P(i,j) & \text{otherwise} \end{cases} \text{for all } i \in N \tag{22}$$

4 Evaluation

This section evaluates the two proposed approaches, assessing their efficiency, accuracy, and overall effectiveness. The presented evaluations validate the theoretical concepts on which algorithms are developed and provide a means of benchmarking, enabling comparisons with existing scenarios, i.e., when data is stored in a single storage tier for the whole duration. The comparison is conducted in two ways. Firstly, a cost estimation is calculated if the data remains in the same tier throughout. Secondly, the cost estimate is computed by considering data movement between different tiers based on the classification performed by the proposed approaches. Finally, the two proposed approaches are also compared against each other. In order to determine the viability and potential of the proposed approaches, a software tool has been developed to provide cost estimations based on the values obtained from Google Cloud storage—recall Table 1.

Due to limitations in acquiring a real dataset, our evaluations involve the generation of multiple datasets, both fully synthetic and semi-synthetic, covering

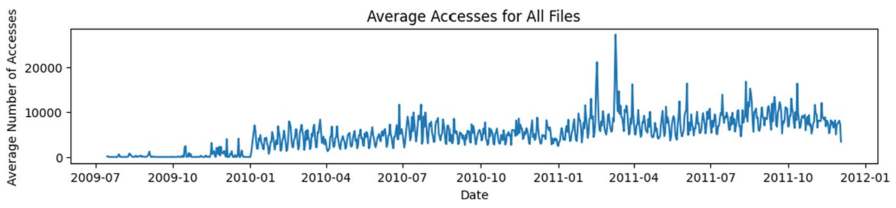


Fig. 2 Data access pattern over the whole time period; y-axis and x-axis represent the number of accesses and the date, respectively

the context of an IoT scenario as well, such as the third-party data preparation workflow used at Bosch [44], where data is generated from IoT sensors and further used for processing. We performed two detailed experiments. Experiment 1 includes fully synthetic datasets, as the evaluations are conducted on the dataset variations generated by increasing the size of storage objects within a semi-synthetic dataset. The comparison is presented with a static access pattern for the datasets, implying an assumption that the access frequency of storage objects remains constant. In Experiment 2, a dataset with dynamic access patterns is employed. This means that datasets from a website application are collected over a 180-days period, with half of the dataset utilized as input for classification approaches and the remaining half employed for evaluation. This setup allows for an even more realistic evaluation, considering the variability in access patterns over time.

These datasets are crucial in facilitating comprehensive evaluations across possible scenarios. By utilizing these datasets, we thoroughly assess the performance and robustness of the proposed approach under different conditions, thereby ensuring a more rigorous and reliable evaluation process. However, the utilization of small object sizes with a high frequency of access and large object sizes with low access frequency is not recommended for evaluation, as their placement is intuitively clear; small objects with high access should be placed in the premium tier, while large objects with low access frequency are better suited for the archive tier.

4.1 Experiment 1: static access pattern

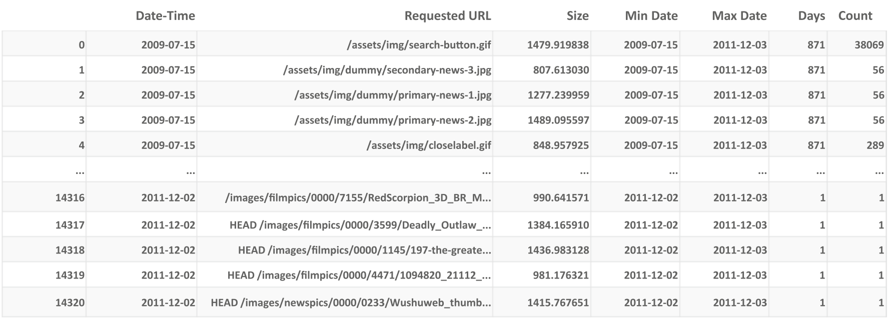
Three synthetic datasets were generated based on publicly available data on Kaggle. It is an access log of a software application deployed on the cloud for a period of almost 2.5 years. Figure 2 shows the access pattern based on the average number of accesses of all objects over the whole storage time period. Additionally, some of the key features of the dataset are as follows:

- total time of data storage: $A = 871$ days;
- total number of objects: $N = 14,321$;
- total number of GET Requests: $g(t) = 2,906,097$; and,
- total number of PUT Requests: $p(t) = 14,321$.

Table 3 Dataset variations: different dataset variations are created while maintaining the same access patterns

	Object size range	Total data size
Variation 1	50–100 MB	1052.45 GB
Variation 2	200–400 MB	4192.91 GB
Variation 3	800–1600 MB	16821.18 GB
Variation 4	3.2–6.4 GB	67215.39 GB

This involves (1) defining distinct size ranges for data storage objects and (2) using different total sizes for the dataset



	Date-Time	Requested URL	Size	Min Date	Max Date	Days	Count
0	2009-07-15	/assets/img/search-button.gif	1479.919838	2009-07-15	2011-12-03	871	38069
1	2009-07-15	/assets/img/dummy/secondary-news-3.jpg	807.613030	2009-07-15	2011-12-03	871	56
2	2009-07-15	/assets/img/dummy/primary-news-1.jpg	1277.239959	2009-07-15	2011-12-03	871	56
3	2009-07-15	/assets/img/dummy/primary-news-2.jpg	1489.095597	2009-07-15	2011-12-03	871	56
4	2009-07-15	/assets/img/closelabel.gif	848.957925	2009-07-15	2011-12-03	871	289
...	...	...	...	...	...	...	...
14316	2011-12-02	/images/filmpics/0000/7155/RedScorpion_3D_BR_M...	990.641571	2011-12-02	2011-12-03	1	1
14317	2011-12-02	HEAD /images/filmpics/0000/3599/Deadly_Outlaw_...	1384.165910	2011-12-02	2011-12-03	1	1
14318	2011-12-02	HEAD /images/filmpics/0000/1145/197-the-greate...	1436.983128	2011-12-02	2011-12-03	1	1
14319	2011-12-02	HEAD /images/filmpics/0000/4471/1094820_21112_...	981.176321	2011-12-02	2011-12-03	1	1
14320	2011-12-02	HEAD /images/newspics/0000/0233/Wushuweb_thumb...	1415.767651	2011-12-02	2011-12-03	1	1

Fig. 3 The dataset CSV file after the normalization process

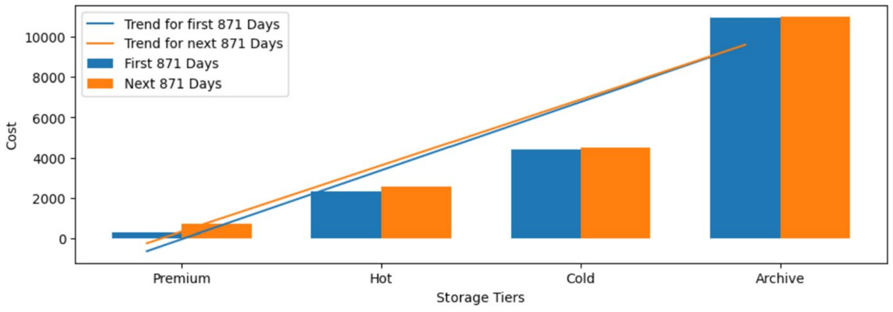


Fig. 4 Comparison of the cost of data storage for the first 871 days with each object having variable age vs. the cost of data storage if it is not moved between tiers for the next 871 days

4.1.1 Dataset preparation

Information regarding the size of the data objects was unavailable. Therefore, for the purposes of testing, we assigned a random size to each object and subsequently created a total of four variations for this dataset. The specific variations for the datasets can be found in Table 3, representing various access patterns associated with different data sizes. This approach offers several benefits, including continuing testing and analysis despite the initial lack of data size information. By assigning random sizes to data objects and creating four dataset variations, the testing process becomes more comprehensive, enabling the discovery of potential issues and patterns that

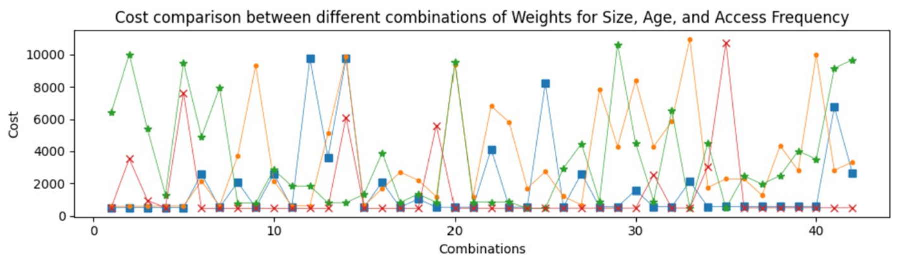


Fig. 5 Cost comparison between different combinations of weights for size, age, and access frequency. Cost in US Dollars is specified on the y-axis, whereas the combination number (#) is shown on the x-axis

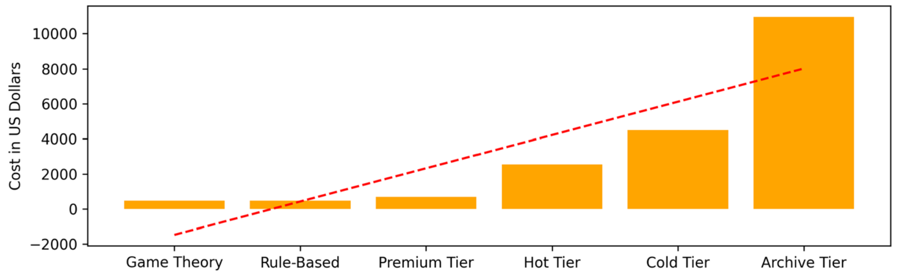


Fig. 6 Cost comparison between rule-based classification and game theory-based classification, and when data remains in a single storage tier for the entire duration

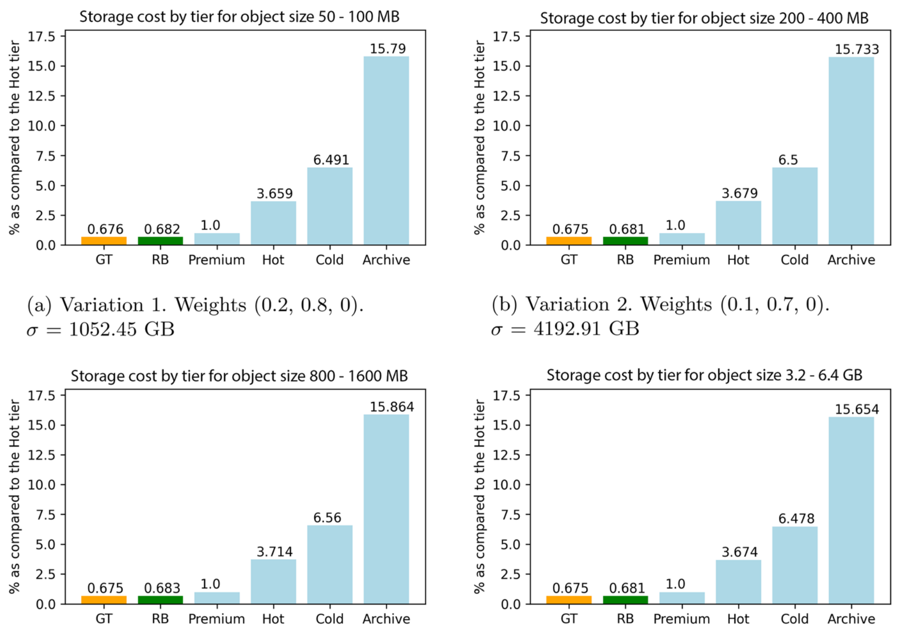
might not be apparent in a single dataset. This approach also seeks to mimic real-world scenarios with varying data sizes, enhancing the realism of the testing process. The dataset CSV file after the normalization process (i.e., converting access logs in the form of a text file into a structured CSV file) is shown in Fig. 3.

4.1.2 Cost comparison

Figure 4 compares the data storage cost if it remained static in one storage tier. The calculation is done keeping in view that the access pattern that objects will follow for the next 871 days will be similar to the first 871 days. Due to the high number of data access requests and free data retrieval for the premium tier, the cost of storing data in the premium tier is the cheapest. However, when calculating the cost of data storage for the next 871 days, the premium tier, although still cheaper than the rest of the tiers, shows the highest difference in the cost because of the low cost of data retrieval in the premium tier.

4.1.3 Weights

For the rule-based classification, weights play an important role. If 30% weight is set for size, 20% for access frequency, and 50% for the age of the data, the combination of weights would be (0.3, 0.2, 0.5). Generally, the sum of the total weights should be equal to 1. In that case, there are a total of 36 possible combinations of weights. By



(a) Variation 1. Weights (0.2, 0.8, 0). σ = 1052.45 GB

(b) Variation 2. Weights (0.1, 0.7, 0). σ = 4192.91 GB

(c) Variation 3. Weights (0.1, 0.7, 0). σ = 16,821.18 GB

(d) Variation 4. Weights (0.1, 0.6, 0). σ = 67,215.39 GB

Fig. 7 A comparison of cost estimation is made for the scenario where data remains static within each storage tier and when classified between different storage tiers using the game theory-based (GT) and rule-based (RB) classification approaches. Furthermore, comparisons are presented for four different variations of the dataset in (**a–d**), with the most suitable weights combination specified under each subfigure

removing the condition of the sum being equal to 1, we created a total of 286 combinations. Then, the priority score was calculated for each combination of weights, and subsequently, the cost was calculated. Out of 286, the cost calculation script returned 169 unique values for the cost. The comparison of cost with those combinations for the first variation of the dataset is shown in Fig. 5.

4.1.4 Results

Cost is calculated using the proposed rule-based and game theory-based classification, and a comparison is presented in Fig. 6. The effectiveness of the weights can vary according to the dataset's characteristics; hence, for dataset variation 1, the most suitable combination turned out to be size: 20%, access frequency: 80%, and age: 0%. It can be seen that with the proposed rule-based classification approach, the cost of data storage is \$473.39. In contrast, if the data is stored in the premium tier for the whole time, the total cost is \$694.35 (the cost of data migration is not included in this calculation). Additionally, the cost of the game theory-based approach is \$469 for this particular case; however, the computational time was less as the calculations didn't have to be done for multiple combinations of weights, which is explained in more detail in Sect. 5.

In Fig. 7, a comprehensive comparison of costs is depicted as the data size quadruples at each step (as indicated in Fig. 7a–d), all while maintaining the same access pattern. The rule-based approach achieves approximately 32% cost reduction, whereas the game theory-based approach achieves approximately 33%. Moreover, this figure offers insights into cost comparisons under two distinct scenarios: one where the data remains exclusively within a single tier throughout the entire duration and another where it dynamically migrates between different tiers, with migration strategies determined by game theory-based and rule-based approaches. The comparison is presented in percentages instead of exact dollar amounts. This approach is employed to provide a more relative and standardized perspective, with the cost of the premium tier serving as the pivotal benchmark for assessment. By expressing the costs in percentages, we facilitate a more meaningful understanding of how different tiers perform in the premium tier, offering a clearer picture of the cost dynamics as data sizes increase.

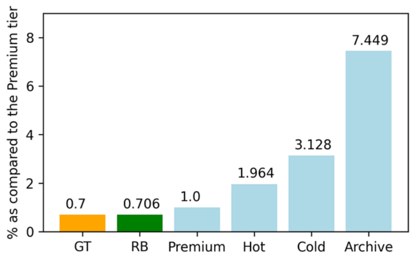
4.2 Experiment 2: dynamic access pattern

For the second dataset, we once again downloaded the access log for a web application, spanning a period of 90 days, and calculated its cost. We expanded our approach to conduct a more realistic evaluation and take advantage of the opportunity to obtain the most up-to-date web application access logs. In this case, cost evaluation is no longer solely based on the assumption of static access patterns. Instead, access logs were downloaded for a period of 180 days. The initial 90 days of data were used for classification, and the classified data was subsequently compared with the data from the following 90 days. This extended approach allows for a more comprehensive evaluation by incorporating real-world variations in access patterns, enhancing the accuracy of cost calculations, and providing valuable insights into the dynamics of web application usage. Some of the key features of the dataset are as follows:

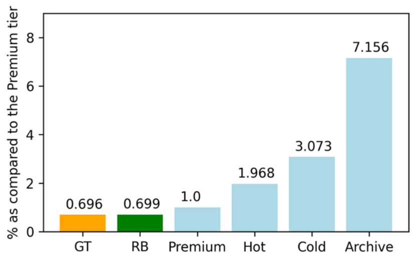
- total time of data storage: $A = 90$ days;
- total Size of the data: $\sigma = 6139.80$ GB;
- total number of objects $N = 149,603$;
- total number of GET requests: $g(t) = 2,847,838$; and,
- total number of PUT requests: $p(t) = 149,603$.

4.2.1 Results

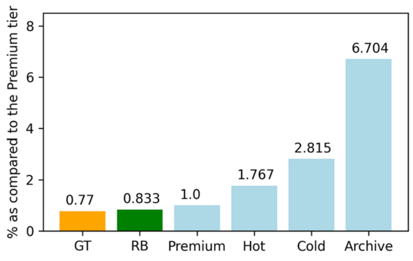
Promising results were achieved from the second dataset shown in Fig. 8. The first evaluation was performed on a dataset with larger data chunks stored for longer. However, this dataset reflects a scenario where relatively smaller data is stored for a shorter period of time, specifically 90 days. Figure 8a presents the cost comparison between



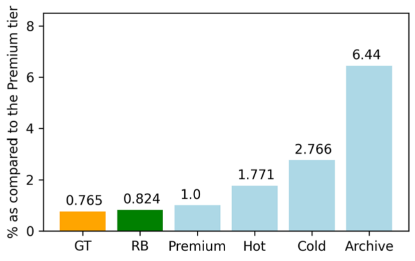
(a) Static access pattern.
Weights (0.0, 0.7, 0.2).
$\sigma =$ 6.0 TB



(b) Static access pattern.
Weights (0.0, 0.7, 0.2).
$\sigma =$ 61.39 TB



(c) Dynamic access pattern.

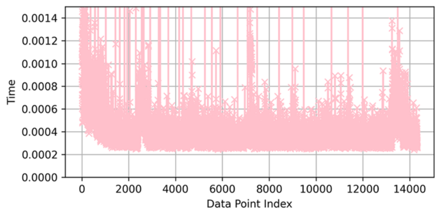


(d) Dynamic access pattern.

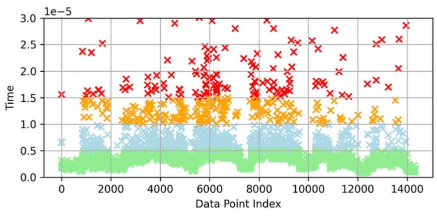
Fig. 8 A comparison of cost estimation is made for the scenario where data remains static within each storage tier and when classified between different storage tiers using the game theory-based (GT) and rule-based (RB) classification approaches. Furthermore, comparisons are presented for two different scenarios. In (**a, b**), the comparison when the access pattern remains static (i.e., the dataset follows a pattern in the future) is shown, whereas in (**c, d**), the comparison with the dynamic access pattern (i.e., dataset divided into two chunks, one for classification and one for testing) is shown

storage tiers before and after the data is classified using rule-based and game theory-based approaches.

To further enhance the comprehensiveness of the evaluation process, another variation of the dataset is created with a larger data volume but for the same storage period. The cost comparison is shown in Fig. 8b. This evaluation approach demonstrates a thorough exploration of diverse scenarios, including storing larger data chunks for an extended period and relatively smaller data for a shorter duration. Figure 8c and Fig. 8d show the comparison of the cost when the complete dataset is divided into two chunks, one used for classification while the second half is used for evaluation. This breadth of investigation allows for a well-rounded assessment of cost implications and performance variations, potentially leading to adaptable and optimized storage strategies.



(a) Aggregate computational cost of rule-based approach across all 286 combinations.



(b) Computational cost of rule-based approach for pre-set weight combinations.

Fig. 9 Comparative analysis of computational costs. In (**a**), the total computational cost of the rule-based approach across all 286 combinations is illustrated. In (**b**), insights into the computational cost of the rule-based approach when employing pre-set weight combinations are provided. These analyzes shed light on the computational efficiency of the rule-based approach under varying conditions and weight configurations

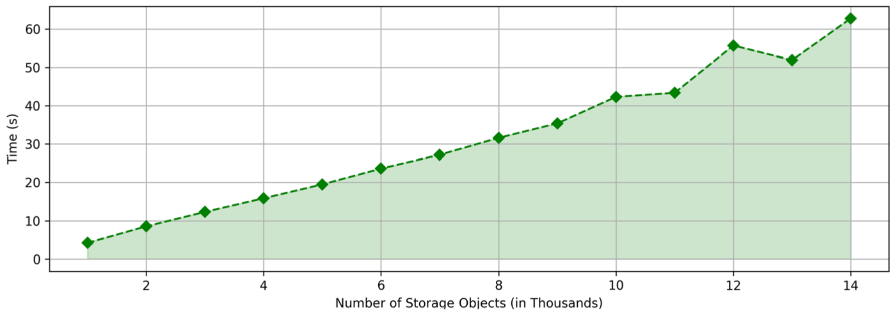


Fig. 10 Demonstration of computational time as a function of the number of storage objects in the Game theory-based classification approach

5 Computational cost

To calculate the execution time of the proposed algorithms, we examined the computational cost associated with the proposed rule-based and game theory-based classification approaches. As elaborated in Sect. 3.1, the rule-based approach calculates the priority score of the data objects. Our evaluation focuses on calculating the execution time of the rule-based classification approach for each object throughout its lifespan. The results are visually depicted in Fig. 9 in the form of a scatter graph showcasing the distribution of execution times for each data object. Figure 9a illustrates the total computational cost of the rule-based approach across all 286 combinations. Figure 9b provides insights into the computational cost of the rule-based approach when employing pre-set weight combinations.

Notably, our findings reveal that the computation time consistently falls within the millisecond range even when the priority score is calculated 286 times for each object, a performance characteristic that remains virtually imperceptible to the cloud infrastructure. This observation holds significance, especially in light of the typical latency of requests to the cloud, which tends to range from tens to hundreds of milliseconds. Hence, the proposed rule-based approach can be implemented without significantly impacting cloud storage performance while achieving substantial cost savings.

The game theory-based approach operates on a distinct principle. Instead of assessing individual objects by computing priority scores, it employs an iterative process to determine probabilities. This iterative nature implies that the computational cost increases with the number of objects involved. Figure 10 illustrates the relationship between the execution time, denoted on the y-axis in seconds, and the number of objects, represented in thousands on the x-axis. The graphical representation shows that the execution time correlates directly with the number of objects. It provides valuable insights into the scalability of the approach because, even though the execution time increases with the increase in the total number of storage objects, the change is not drastic. Even for big data applications and workflows with a large number of storage objects, the proposed approach can perform classification quickly while consuming very few compute resources.

Our evaluation provides insights into the performance characteristics of the rule-based and game theory-based classification approaches. The rule-based method showcases remarkable efficiency when the weights are pre-defined, consistently delivering computation times within the millisecond range for each object. This attribute positions the rule-based approach as an excellent choice for scenarios where weight combinations are fixed. Conversely, the game theory-based approach operates on a different principle, utilizing an iterative process to determine probabilities. While this methodology may result in increased computational cost as the number of objects rises, it provides flexibility and adaptability in scenarios where weights are dynamic or need frequent adjustments. Therefore, the choice between these approaches ultimately depends on the specific requirements and characteristics of the cloud storage scenario, with the rule-based approach excelling in static scenarios and the game theory-based approach offering versatility for dynamic weight assignments.

6 Discussion

The rule-based classification approach has shown promising results in reducing storage cost. According to the evaluation, the cost reduction is nearly 30–35%; even when factoring in the cost of data migration, the difference would be significant. However, it lacks the ability to consider feedback regarding each classification. There is a chance that the classification of a storage object may not be cost-effective, and to enhance the algorithm's performance, it is crucial to incorporate that information as feedback. To tackle this challenge, we proposed an approach that uses game theory to classify storage objects into different tiers. The game theory-based approach involves modeling the decision-making process of multiple players and their interactions and can anticipate their behavior and make more accurate predictions. It considers the access patterns and metadata of storage items and can optimize the storage tier selection in a multi-agent system. While the rule-based approach is simpler and more straightforward, the game theory-based approach is more complex but can potentially lead to a greater cost reduction. During our evaluations, the game theory-based approach yielded slightly lower costs but significantly reduced computational cost.

Rule-based and game theory-based approaches perform effectively across varying data sizes and make accurate classification decisions regardless of the data storage duration. Data with a high access frequency is best suited for storage in the premium tier, while data with a low access frequency can be stored in the cold or archive tiers. While this scenario is straightforward, the evaluations have demonstrated that the algorithms are suitable for simple cases and useful in more complex scenarios. These scenarios involve dynamic access patterns and the potential for significant fluctuations. In addition to the technical aspects, the rule-based and game theory-based classification approaches have practical implications in real-world scenarios. Both approaches are lightweight and industry- and platform-independent, making them suitable for various industries and applications where reducing storage cost is paramount. This adaptability means they can seamlessly integrate into various industry verticals, such as e-commerce, healthcare, logistics, etc. In these diverse sectors, the common challenge of reducing storage cost looms as data volumes surge.

While the proposed approaches exhibit promise and efficiency, it's important to note their limitations. First, these approaches do not factor in migration fees, which could yield even greater cost reductions if incorporated into cost estimation and decision-making processes. Second, block-rate pricing of storage and network usage costs could provide a more comprehensive understanding of the cost landscape to enhance evaluation accuracy. Finally, it's worth noting that both approaches excel when data is stored for relatively extended durations, typically at least 90 days. However, the potential for promising results even for data stored for shorter periods exists by incorporating the missing cost components, ensuring these approaches remain adaptable and effective across various timeframes. In conclusion, the rule-based classification approach demonstrates notable efficiency, particularly when predefined weights are in play. This advantage stems from its ability to bypass the exhaustive exploration of all conceivable weight combinations. Conversely, the execution time of the game theory-based approach exhibits a different dependency. It hinges on the total count of storage objects in consideration. While the rule-based approach excels in scenarios with established weight values, the game theory approach showcases its strengths in addressing datasets with varying sizes and complexities.

7 Related work

This section provides an overview of the related work on storage tier optimization. Existing work on storage tiering has primarily focused on two tiers: hot and cold. Hot data is frequently accessed and requires high-performance storage. Cold data is accessed infrequently and can be stored on lower-cost storage. Many earlier studies have focused on cutting costs in systems that use multiple clouds. Some of these studies have specifically looked at finding the most cost-effective ways to store data in different locations in multi-cloud environments, such as [11, 14, 16, 38]. Additionally, some studies aim to create tiered cloud storage that uses multiple clouds to suit different data types. Tiered storage is a popular topic in the physical layer, where different storage mediums like HDDs and SSDs are combined. This is done to examine how performance can be improved and reduce costs, such as in [8]. CSPs also offer storage tier migration services such as Google Autoclass [7], but it comes with additional management fee and enablement charge.

Liu et al. [18] proposed RLTiering, an auto-tiering system that uses deep reinforcement learning to place data in the most cost-effective tier in cloud storage. They also proposed a randomized online migration algorithm [17] for cost optimization. Similarly, Erradi et al. [6] proposed two online cost optimization algorithms for tiered cloud storage services. They are designed to minimize the overall cost of storage, while meeting users' QoS requirements. The first algorithm is a greedy algorithm that places data in the cheapest tier that meets the QoS requirements of users. The second reinforcement learning algorithm learns to place data in the most cost-effective tier over time. Alshawabkeh et al. [1] developed an automated and adaptive framework using efficient Markov chain correlation-based clustering to move active data to high-performance and inactive data to low-cost/high-capacity storage tiers. This framework can predict workload changes and group similar storage units, enhancing performance, reliability, and availability and reducing cost. On the contrary, we propose approaches to storage tiering that consider four storage tiers: premium, hot, cold, and archive.

Hsu et al. [10] proposed an approach to predict the data condition using learning-based techniques and then saved them in local or cloud storage. This could be used in cloud services with hot and cold tiers. However, it's difficult for cloud users or new businesses to get past access traces to train their learning-based techniques, as these traces are private in public clouds. At the same time, the predicted outcomes aren't always trustworthy, and running the process regularly might make their method not instantaneous. Blamey et al. [3] utilized hot and cold tiers to optimize the costs of keeping top-K documents in streaming workloads; however, their methods were specifically designed for streaming workloads and can not be generalized to other workloads.

Mansouri and Erradi [22], as well as Erradi and Mansouri [6], introduced a series of deterministic online algorithms to address cost reduction in this particular problem. However, access frequency, specifically the number of access requests, was not considered during their decision-making process. Our approach, however, considers three main factors when determining which tier to store an object size, age, and access frequency. Moreover, Zhang et al. [42] investigated how cloud providers can maximize their profits by using hot and cold storage tiers, but our research focuses on how cloud users can minimize their costs by using hot and cold storage tiers. Some scientific studies also introduce the multi-cloud setting that considers migrating data among multiple clouds for achieving cost-effective geo-distributed workloads [20, 30, 31]. In [24], Facebook developed a storage tier optimization approach and targeted two storage tiers. In addition, their proposed approach made decisions based on the characteristics of the whole bucket. Our approaches take decisions on objects rather than buckets, proposing a more flexible approach. In addition to that, they are generic, platform-, and industry-independent.

In summary, existing work on storage tiering has traditionally concentrated on two tiers: hot and cold. Hot data, frequently accessed and demanding high-performance storage, contrasts with cold data, which is seldom accessed and can be efficiently stored on more cost-effective storage solutions. Moreover, previous studies explored cost reduction strategies for multi-cloud systems, aiming to identify optimal data storage methods across diverse cloud locations. While tiered storage discussions typically revolve around physical mediums like HDDs and SSDs, seeking to enhance performance and minimize costs, this article introduces a novel perspective. It proposes innovative approaches that leverage rules and game theory to optimize storage tiering within cloud environments, a novel approach compared to previous research. These methods stand out for their lightweight nature and minimal computational overhead, rendering them exceptionally suitable for practical implementation in cloud storage environments.

8 Conclusions and future work

Maintaining data in a single tier continuously is ineffective and expensive, such as in the case of an IoT application, where data is generated from multiple sources and stored for processing. Storage tier optimization organizes data into different tiers based on its usage and performance requirements. This process helps reduce cost and improve storage performance and efficiency by ensuring that the most frequently accessed data is stored on the fastest media, while less frequently accessed data can be stored on slower, less expensive media. We explored two novel approaches: (1) a rule-based approach and (2) a game theory-based approach. The former examines object metadata and access patterns for storage tier optimization. The rule-based classification was demonstrated to be successful on a synthetic dataset and is straightforward and simple to use using $\lambda = \alpha + \beta + \gamma$ for priority score calculations. The latter approach is more complex, considers the dynamic nature of cloud storage pricing and user demand, and provides a flexible and adaptive solution to cloud storage tier optimization.

The proposed approaches can effectively minimize storage cost and provide a flexible cloud storage tier optimization solution. We demonstrated the viability and potential of the proposed approaches by evaluating them using fully- and semi-synthetic datasets, encompassing various variations to provide a realistic representation of cloud storage usage. The proposed approaches are assessed against these datasets, showcasing their effectiveness in substantially reducing storage cost across static and dynamic access patterns. Moreover, the proposed approaches are not platform- or industry-specific and are not very resource-intensive in computation. We presented a comprehensive computational cost analysis to demonstrate the practicality of the proposed approaches in a real cloud environment. They can, therefore, be considered appropriate for various software applications. Additionally, the findings indicate that while developing such an algorithm, it is crucial to consider the access patterns and metadata of storage items.

In the future, to make the estimations and comparisons more accurate and concise, there is a need for comprehensive mathematical modeling that not only correctly calculates the costs but also takes into account the following: (1) network usage cost based on block pricing; (2) data migration cost; and (3) penalty fees if an object is removed before the minimum time period specified for that particular tier. Moreover, we aim to conduct further evaluations for different Big Data and IoT workflows from different domains. These should be used to generate accurate and concise estimates and compare the cost of different storage options. In terms of practical applications, we intend to extend this research into a software tool that can be integrated with cloud storage systems, executing the classification of storage objects through cloud storage APIs. Furthermore, the future directions for the expansion of the proposed work include:

1. investigating the use of machine learning algorithms to improve the accuracy of the proposed approaches for storage tier optimization;
2. exploring the use of the proposed approaches in conjunction with other optimization techniques, such as data compression and deduplication [40], to further reduce storage cost; and,
3. evaluating the proposed approaches on real-world datasets to validate their effectiveness in practical cloud storage environments.

Acknowledgements We thank Nikolay Nikolov for the initial discussions around the topic of cloud cost optimization that helped to shape the direction of the work in this article.

Funding Open access funding provided by NTNU Norwegian University of Science and Technology (incl St. Olavs Hospital - Trondheim University Hospital). This work was partly funded through the EC-funded projects DataCloud (H2020 101016835), enRichMyData (HE 101070284), Graph-Massivizer (HE 101093202), UPCAST (HE 101093216), and INTEND (HE 101135576).

Data availibility No research data outside the submitted manuscript file.

Declarations

Ethical approval Not applicable.

References

1. Alshawabkeh M, Riska A, Sahin A, Awwad M (2012) Automated storage tiering using markov chain correlation based clustering. In: Proceedings of the 11th international conference on machine

learning and applications (ICMLA 2012). IEEE, vol 1, pp 392–397. https://doi.org/10.1109/ICMLA.2012.71

2. Amend JM (2018) Storage almost full: driverless cars create data crunch. https://www.wardsauto.com/technology/storage-almost-full-driverless-cars-create-data-crunch. Accessed 5 Dec 2023
3. Blamey B, Wrede F, Karlsson J, Hellander A, Toor S (2019) Adapting the secretary hiring problem for optimal hot-cold tier placement under top-K workloads. In: Proceedings of the 19th IEEE/ACM international symposium on cluster, cloud and grid computing (CCGRID 2019). IEEE, pp 576–583. https://doi.org/10.1109/CCGRID.2019.00074
4. Bubeck S, Cesa-Bianchi N et al (2012) Regret analysis of stochastic and nonstochastic multi-armed bandit problems. Found Trends Mach Learn 5(1):1–122. https://doi.org/10.1561/2200000024
5. Dzhoha A, Rozora I (2023) Multi-armed bandit problem with online clustering as side information. J Comput Appl Math 427:115132. https://doi.org/10.1016/j.cam.2023.115132
6. Erradi A, Mansouri Y (2020) Online cost optimization algorithms for tiered cloud storage services. J Syst Softw 160:110457. https://doi.org/10.1016/j.jss.2019.110457
7. Google: feedbackAutoclass (2024). https://cloud.google.com/storage/docs/autoclass. Accessed 9 Feb 2024
8. Guerra J, Pucha H, Glider J, Belluomini W, Rangaswami R (2011) Cost effective storage using extent based dynamic tiering. In: Proceedings of the 9th USENIX conference on file and storage technologies (FAST 11). USENIX Association
9. Heinrich S (2017) Flash memory in the emerging age of autonomy. https://www.flashmemorysummit.com/English/Collaterals/Proceedings/2017/Proceedings_Chrono_2017.html. Accessed 5 Dec 2023
10. Hsu YF, Irie R, Murata S, Matsuoka M (2018) A novel automated cloud storage tiering system through hot-cold data classification. In: Proceedings of the IEEE 11th international conference on cloud computing (CLOUD 2018). IEEE, pp 492–499. https://doi.org/10.1109/CLOUD.2018.00069
11. Ikken S, Renault E, Barkat A, Tari A, Kechad T (2017) Cost-efficient big intermediate data placement in a collaborative cloud storage environment. In: Proceedings of the IEEE 19th international conference on high performance computing and communications; IEEE 15th international conference on smart city; IEEE 3rd international conference on data science and systems (HPCC/SmartCity/DSS 2017). IEEE, pp 514–521. https://doi.org/10.1109/HPCC-SmartCity-DSS.2017.67
12. Khan AQ, Nikolov N, Matskin M, Prodan R, Bussler C, Roman D, Soylu A (2023) Towards cloud storage tier optimization with rule-based classification. In: Proceedings of the 10th IFIP WG 6.12 European conference on service-oriented and cloud computing (ESOCC 2023). LNCS. Springer, vol 14183, pp 205–216. https://doi.org/10.1007/978-3-031-46235-1_13
13. Khan AQ, Nikolov N, Matskin M, Prodan R, Song H, Roman D, Soylu A (2023) A taxonomy for cloud storage cost. In: The 15th international conference on management of digital ecosystems (MEDES 2023). CCIS. Springer, vol 2022, pp 317–330. https://doi.org/10.1007/978-3-031-51643-6_23
14. Khan AQ, Nikolov N, Matskin M, Prodan R, Roman D, Sahin B, Bussler C, Soylu A (2023) Smart data placement using storage-as-a-service model for big data pipelines. Sensors 23(2):564. https://doi.org/10.3390/s23020564
15. Krumm N, Hoffman N (2020) Practical estimation of cloud storage costs for clinical genomic data. Pract Lab Med 21:e00168. https://doi.org/10.1016/j.plabm.2020.e00168
16. Liu G, Shen H (2017) Minimum-cost cloud storage service across multiple cloud providers. IEEE/ACM Trans Netw 25(4):2498–2513. https://doi.org/10.1109/ICDCS.2016.36
17. Liu M, Pan L, Liu S (2021) Keep hot or go cold: a randomized online migration algorithm for cost optimization in STaaS clouds. IEEE Trans Netw Serv Manag 18(4):4563–4575. https://doi.org/10.1109/TNSM.2021.3096533
18. Liu M, Pan L, Liu S (2022) RLTiering: a cost-driven auto-tiering system for two-tier cloud storage using deep reinforcement learning. IEEE Trans Parallel Distrib Syst 34(2):73–90. https://doi.org/10.1109/TPDS.2022.3224865
19. Macedo R, Ja Paulo, Pereira J, Bessani A (2020) A survey and classification of software-defined storage systems. ACM Comput Surv. https://doi.org/10.1145/3385896
20. Mansouri Y, Toosi AN, Buyya R (2017) Cost optimization for dynamic replication and migration of data in cloud data centers. IEEE Trans Cloud Comput 7(3):705–718. https://doi.org/10.1109/TCC.2017.2659728
21. Mansouri Y, Toosi AN, Buyya R (2017) Data storage management in cloud environments: taxonomy, survey, and future directions. ACM Comput Surv. https://doi.org/10.1145/3136623
22. Mansouri Y, Erradi A (2018) Cost optimization algorithms for hot and cool tiers cloud storage services. In: Proceedings of the 11th international conference on cloud computing (CLOUD 2018). IEEE, pp 622–629. https://doi.org/10.1109/CLOUD.2018.00086

23. McLeod J, Gormly B (2018) Records storage in the cloud: are we modelling the cost? Arch Manuscr 46(2):174–192. https://doi.org/10.1080/01576895.2017.1409125
24. Muralidhar S, Lloyd W, Roy S, Hill C, Lin E, Liu W, Pan S, Shankar S, Sivakumar V, Tang L et al (2014) f4: Facebook's warm BLOB storage system. In: Proceedings of the 11th USENIX symposium on operating systems design and implementation. USENIX Association, pp 383–398
25. Myerson RB (1997) Game theory: analysis of conflict. Harvard University Press
26. Naldi M, Mastroeni L (2013) Cloud storage pricing: a comparison of current practices. In: Proceedings of the international workshop on hot topics in cloud services (HotTopiCS 2013). ACM, pp 27–34. https://doi.org/10.1145/2462307.2462315
27. Nikolov N, Dessalk YD, Khan AQ, Soylu A, Matskin M, Payberah AH, Roman D (2021) Conceptualization and scalable execution of big data workflows using domain-specific languages and software containers. Internet Things 16:100440. https://doi.org/10.1016/j.iot.2021.100440
28. Nikolov N, Solberg A, Prodan R, Soylu A, Matskin M, Roman D (2023) Container-based data pipelines on the computing continuum for remote patient monitoring. Computer 56(10):40–48. https://doi.org/10.1109/MC.2023.3285414
29. Nuseibeh H (2011) Adoption of cloud computing in organizations. In: Proceedings of the Americas conference on information systems (AMCIS 2011). AISeL
30. Oh K, Chandra A, Weissman J (2016) Wiera: towards flexible multi-tiered geo-distributed cloud storage instances. In: Proceedings of the 25th ACM international symposium on high-performance parallel and distributed computing (HPDC 2016). ACM, pp 165–176. https://doi.org/10.1145/2907294.2907322
31. Qiu X, Li H, Wu C, Li Z, Lau FC (2014) Cost-minimizing dynamic migration of content distribution services into hybrid clouds. IEEE Trans Parallel Distrib Syst 26(12):3330–3345. https://doi.org/10.1109/INFCOM.2012.6195655
32. Robinson K (2021) Why companies are flocking to the cloud more than ever. https://www.businessinsider.com/cloud-technology-trend-software-enterprise-2021-2. Accessed 5 Dec 2023
33. Roman D, Prodan R, Nikolov N, Soylu A, Matskin M, Marrella A, Kimovski D, Elvesæter B, Simonet-Boulogne A, Ledakis G, Song H, Leotta F, Kharlamov E (2022) Big data pipelines on the computing continuum: tapping the dark data. Computer 55(11):74–84. https://doi.org/10.1109/MC.2022.3154148
34. Rydning DRJGJ, Reinsel J, Gantz J (2018) The digitization of the world from edge to core. Technical report, International Data Corporation, Framingham
35. Thompson WR (1933) On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. Biometrika 25(3–4):285–294
36. Tier definitions and volume placement optimization (2022). https://www.ibm.com/docs/en/storage-insights?topic=SSQRB8/com.ibm.spectrum.si.doc/tpch_saas_r_volume_optimization_process.htm Accessed 5 Dec 2023
37. Tung AKH (2009) Rule-based classification. Springer, Boston, pp 2459–2462
38. Wang P, Zhao C, Liu W, Chen Z, Zhang Z (2020) Optimizing data placement for cost effective and high available multi-cloud storage. Comput Inf 39(1–2):51–82. https://doi.org/10.31577/cai_2020_1-2_51
39. What is a storage device hierarchy? (2021) https://www.ibm.com/docs/en/zos/2.2.0?topic=dfsmshsm-what-is-storage-device-hierarchy Accessed 5 Dec 2023
40. Xia W, Jiang H, Feng D, Douglis F, Shilane P, Hua Y, Fu M, Zhang Y, Zhou Y (2016) A comprehensive study of the past, present, and future of data deduplication. Proc IEEE 104(9):1681–1710. https://doi.org/10.1109/JPROC.2016.2571298
41. Yang C, Xu Y, Nebert D (2013) Redefining the possibility of digital Earth and geosciences with spatial cloud computing. Int J Digit Earth 6(4):297–312. https://doi.org/10.1080/17538947.2013.769783
42. Zhang Y, Ghosh A, Aggarwal V, Lan T (2018) Tiered cloud storage via two-stage, latency-aware bidding. IEEE Trans Netw Serv Manag 16(1):176–191. https://doi.org/10.1109/TNSM.2018.2875475
43. Zhou B, Nikolov N, Zheng Z, Luo X, Savkovic O, Roman D, Soylu A, Kharlamov E (2023) Scaling data science solutions with semantics and machine learning: Bosch case. In: Proceedings of the 22nd international semantic web conference (ISWC 2023). LNCS. Springer, vol 14266, pp 380–399. https://doi.org/10.1007/978-3-031-47243-5_21
44. Zhou B, Svetashova Y, Pychynski T, Baimuratov I, Soylu A, Kharlamov E (2020) SemFE: facilitating ML pipeline development with semantics. In: Proceedings of the 29th ACM international conference on information & knowledge management (CIKM 2020). ACM, pp 3489–3492. https://doi.org/10.1145/3340531.3417436

Authors and Affiliations

Akif Quddus Khan[1] · Mihhail Matskin[2] · Radu Prodan[3] · Christoph Bussler[4] · Dumitru Roman[5,6] · Ahmet Soylu[6]

✉ Akif Quddus Khan
akif.q.khan@ntnu.no

Mihhail Matskin
misha@kth.se

Radu Prodan
radu.prodan@aau.at

Christoph Bussler
chbussler@aol.com

Dumitru Roman
dumitru.roman@sintef.no

Ahmet Soylu
ahmet.soylu@oslomet.no

1 Norwegian University of Science and Technology, 2815 Gjøvik, Norway

2 KTH Royal Institute of Technology, Stockholm, Sweden

3 University of Klagenfurt, Klagenfurt, Austria

4 Robert Bosch LLC, Sunnyvale, CA, USA

5 SINTEF AS, Oslo, Norway

6 OsloMet – Oslo Metropolitan University, Oslo, Norway