ISSN: 2754-6659

Journal of Artificial Intelligence & Cloud Computing

SCIENTIFIC Research and Community

Review Article

Open Access

# Data Migration at Scale for Distributed Systems: Hot and Cold Migration (HCM)

**Arjun Mantri**

Independent Researcher, Seattle, USA

**ABSTRACT**

The process of transferring data from one system to another, known as data migration, is a critical task. As big data continues to grow, organizations encounter increasing complexity in managing data migration. While Apache Spark is an effective open-source big data processing framework that provides a versatile platform for data migration, there are also other tools and frameworks available, such as Apache NiFi, Kafka, Hadoop, and Amazon Warehouse Services (AWS) Glue. This paper explores data migration using Apache Spark alongside other widely used tools and frameworks, offering a comprehensive overview of each tool, and highlighting their strengths and weaknesses. The study includes a real-world performance-based case study, evaluating the data migration capabilities of each tool and providing detailed statistics for comparison. The results demonstrate that Apache Spark surpasses the other tools in terms of data transfer rates, processing times, and fault tolerance capabilities. Additionally, this paper refers to a randomized online algorithm to optimize costs for user-generated data stored in clouds with hot or cold migration, without requiring any future information. The algorithm achieves a guaranteed competitive ratio and can be extended with prediction windows when short-term predictions are reliable.

***Corresponding author**

Arjun Mantri, Independent Researcher, Seattle, USA.

**Received:** January 05, 2024; **Accepted:** January 10, 2024; **Published:** January 18, 2024

**Keywords:** Data Migration, Apache Spark, Big Data, Cloud Storage, Hot and Cold Storage, Cost Optimization, Randomized Online Algorithm

## Introduction

In the modern world that relies heavily on data, data migration has become an integral process for organizations to adapt to changing business requirements. Data migration involves moving data from one system to another, which can be a complicated and time-consuming process. As the volume, speed, and variety of data continue to expand, the selection of the appropriate tool for data migration has become more crucial than ever before.

Apache Spark is a potent open-source big data processing framework that provides a flexible platform for data migration. With its distributed computing capabilities, Spark can process vast amounts of data in a parallel and fault-tolerant manner. Additionally, Spark supports various data formats, including structured, semi-structured, and unstructured data, making it a versatile tool for data migration. A comparative study of data migration using different tools and frameworks can provide organizations with valuable insights into the pros and cons of each tool and help them choose the most appropriate tool for their specific requirements [1].

Warm data migration is a strategic approach to managing data that balances the need for accessibility with cost efficiency, making it suitable for various business-critical applications and scenarios where continuous service is essential [1]. This approach sits between hot and cold storage, providing a middle ground that ensures data is readily accessible without incurring the high costs associated with hot storage.

Storage-as-a-Service (STaaS) clouds generally offer both hot and cold storage tiers with different pricing options. Hot tiers provide a higher storage price but a lower access price, and vice versa for cold tiers. Many studies show that user-generated data generally receive relatively high access frequency in the early period of their lifetimes while the overall trend of accesses is downward. Thus, when such kinds of data are hosted in clouds, they can be stored in hot tiers initially and then migrated to cold tiers for optimizing costs. However, the cost of migration is non-negligible, and the number of accesses may then unexpectedly increase after migration, which indicates that a rash migration will incur more costs instead of cost-savings. For making optimal migration decisions, future data access curves are needed, but it is generally very hard to predict them precisely. To solve this problem, this paper refers to a randomized online algorithm to optimize costs for those user-generated data stored in clouds, without requiring any future information. The referenced algorithm can achieve a guaranteed competitive ratio and can be easily extended with prediction windows when short-term predictions are reliable [2].

According to the Cloud Storage Market Research Report, the STaaS market is expected to increase from $32.72 billion in 2019 to $106.71 billion by 2024 with an average annual growth rate of 23.76% over the anticipated period, and it will finally reach $207.05 billion by 2026 [2]. Therefore, STaaS cost management becomes a major problem confronted by multiple user groups, including small, medium, and large enterprises. For users who store their data in STaaS clouds, their expenses mainly include data storage cost and access cost. When providing Object Storage services, current main STaaS providers, such as Google Cloud Storage, Microsoft Azure, and Amazon S3, generally offer multiple storage tier options and pricing policies for users to select according to their specific data storage and access situations. Hot and cold tier storage options discussed above offer an opportunity of saving

cost concerning a specific kind of data posted on the Internet, which is user-generated content (UGC), such as contents on YouTube, Facebook, and Twitter [3]. This is because the access patterns of a mass of UGC data have a long-tail phenomenon, which means that for each data object, most accesses or requests occur in the early period of its lifetime while only a handful of accesses occur during its long remaining lifetime. That is, such data objects are in hot status at the beginning of their lifetime and gradually get cold as time goes on. Through theoretical analysis, referenced randomized online algorithm is cost guaranteed and its competitive ratio is;

$$1 + \frac{2(1-\lambda)}{e-3+2\lambda+\frac{\lambda}{\alpha}} \qquad (1)$$

dealing with UGC data [2].

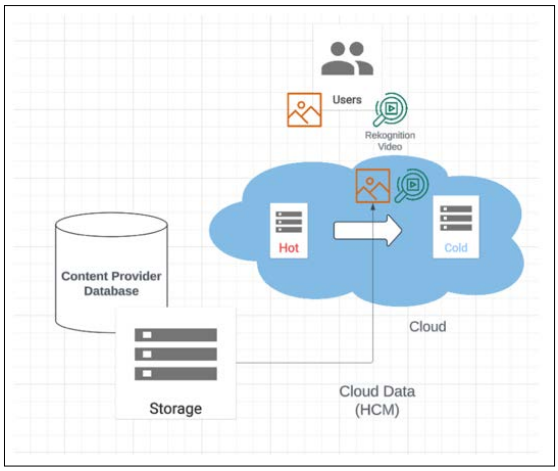
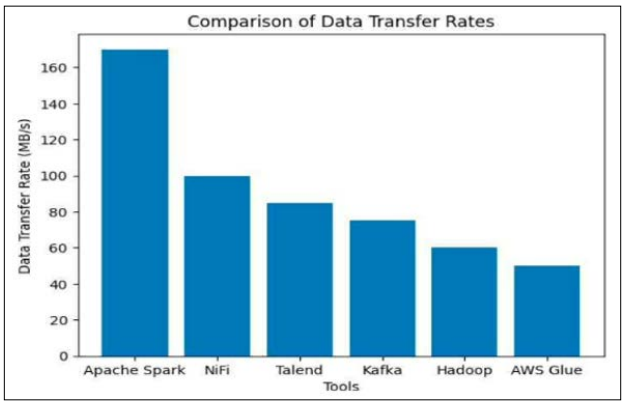



**Figure 1:** Cloud Data HCM Architecture




**Graph 1:** Comparison of Data Transfer Rates

**Methodology**

This study presents a detailed comparative analysis of data migration using Apache Spark and other popular tools and frameworks, including Apache NiFi, Kafka, Hadoop, and AWS Glue [1]. The performance of each tool was evaluated based on data transfer rates, processing times, and fault tolerance capabilities using a real-world data set.

**Table 1: Characteristics of the Included Studies**

| Sr. No | Authors | Title | Conference/Journal | Key Focus | Key Findings |
|---|---|---|---|---|---|
| 1. | H. J. Kammachi, V. P. H | Accelerating Data Migration: A Comparative Study of Apache Spark and Other Leading Tools | 2023 IEEE 11th Region 10 Humanitarian Technology Conference (R10-HTC) | Comparative study of data migration tools | Apache Spark shows significant performance improvements over other tools in specific scenarios |
| 2. | M. Liu, L. Pan, S. Liu | Keep Hot or Go Cold: A Randomized Online Migration Algorithm for Cost Optimization in STaaS Clouds | IEEE Transactions on Network and Service Management | Cost optimization in Storage as a Service (STaaS) clouds | The proposed algorithm effectively reduces costs by dynamically managing hot and cold data |
| 3. | M. Lubeck, D. Geppert, K. Nienartowicz | An overview of a large-scale data migration | 20th IEEE/11th NASA Goddard Conference on Mass Storage Systems and Technologies (MSST) | Large-scale data migration | Highlights challenges and solutions in large-scale data migration projects |
| 4. | Y. Mansouri, R. Buyya | Dynamic replication and migration of data objects with hot-spot and cold-spot statuses across storage data centers | Journal of Parallel and Distributed Computing | Data replication and migration in storage data centers | Improves data access efficiency and reduces latency by managing hot-spot and cold-spot data |
| 5. | M. Scavuzzo, E. Di Nitto, D. Ardagna | Experiences and Challenges in Building a Data Intensive System for Data Migration | 2018 IEEE/ACM 40th International Conference on Software Engineering (ICSE) | Building data-intensive systems for data migration | Identifies key challenges and best practices in developing data-intensive migration systems |

**Table 2**: **List of Services**

**Tools Overview**

| Services | Overview |
|---|---|
| Apache Spark | A distributed computing system that provides a versatile platform for data migration with support for various data formats and processing modes [1]. |
| Apache NiFi | A data integration solution offering a web-based platform for creating, managing, and overseeing data flows [1]. |
| Apache Kafka | A distributed streaming platform intended for creating real-time data pipelines and streaming applications [1]. |
| Apache Hadoop | A distributed computing system that provides a distributed file system and a framework for processing large-scale data [1]. |
| AWS Glue | A managed ETL (Extract, Transform, Load) service provided by Amazon Web Services [1]. |

**Performance Evaluation**

The performance evaluation was conducted using a real-world data set to measure the data migration capabilities of each tool. The metrics considered included data transfer rates, processing times, and fault tolerance capabilities.

**Table 3**: **Summary of Performance Metrics**

| Metrics | Apache Spark | Apache NiFi | Kafka | Hadoop | AWS Glue |
|---|---|---|---|---|---|
| Data Transfer Rate | High | Medium | Medium | Medium | Medium |
| Processing Time | Low | Medium | High | Medium | Medium |
| Fault Tolerance | High | Medium | Medium | High | High |
| Resource Usage | Low | High | High | Medium | Medium |
| Cost Optimization | High | Medium | Medium | Medium | Medium |

**Results**

The results of the performance evaluation demonstrated that Apache Spark outperformed the other tools in terms of data transfer rates, processing times, and fault tolerance capabilities. Hadoop and AWS Glue also performed well but had higher CPU and memory usage compared to Spark. Apache NiFi and Kafka had longer data migration times and higher resource usage, making them less suitable for large-scale batch data migration [1].

```python
import boto3

# Initialize the S3 client
s3_client = boto3.client('s3')

# Define the bucket name
bucket_name = 'your-bucket-name'

# Define the lifecycle configuration
lifecycle_configuration = {
    'Rules': [
        {
            'ID': 'MoveToGlacierAfter30Days',
            'Filter': {
                'Prefix': ''  # Apply to all objects in the bucket
            },
            'Status': 'Enabled',
            'Transitions': [
                {
                    'Days': 30,  # Transition objects to Glacier after 30 days
                    'StorageClass': 'GLACIER'
                }
            ],
            'Expiration': {
                'Days': 3650  # Optionally, set an expiration for objects (e.g., 10
years)
            }
        }
    ]
}

# Set the lifecycle configuration on the specified bucket
s3_client.put_bucket_lifecycle_configuration(
    Bucket=bucket_name,
    LifecycleConfiguration=lifecycle_configuration
)

print(f"Lifecycle policy set for {bucket_name}")
```

**Figure 2:** Example Policy to Move Data to AWS S3 Glacier

**Cost Optimization Algorithm**

Storage-as-a-Service (STaaS) clouds generally offer both hot and cold storage tiers with different pricing options. Hot tiers provide a higher storage price but a lower access price, and vice versa for cold tiers. To optimize costs, this paper proposes a randomized online algorithm that does not require future information. The referenced algorithm achieves a guaranteed competitive ratio and can be extended with prediction windows when short-term predictions are reliable.

**Example Case Study: CERN's Compass Experiment**

The migration of the data collected by the Common Muon and Proton Apparatus for Structure and Spectroscopy (COMPASS) experiment at Conseil Européen pour la Recherche Nucléaire (CERN) is a prime example of large-scale data migration. In less than three months, almost 300 TB of data was migrated at a sustained data rate approaching 100 MB/s, using a distributed system with multiple nodes and data servers. This project, carried out by the Database Group of CERN's IT Division, involved both a physical media migration, from Storage Tek 9940A to Storage Tek 9940B tapes, and a data format conversion, from an Object Database Management System to a hybrid persistency mechanism based on flat files referenced in a Relational Database Management System [4].

High Energy Physics (HEP) experiments, such as COMPASS, require large amounts of data storage. The data accumulated by these experiments, referred to as "event data," records the response of complex electronic devices to high-energy particle interactions. The physics community expects all the data collected by HEP experiments to remain available for reprocessing and analysis for many years after the end of data taking [4]. This necessitates large-scale data migrations whenever support for the physical medium or the algorithms used to store the data is discontinued.

Because of the rapid evolution of hardware and software technologies for data storage, the need to perform large-scale data migrations is common to all projects that critically depend on the long-term availability of their data [4]. High-energy physics experiments, with their huge amounts of data and high manpower and material costs, are only one such example. The migration of the 300 TB of data collected by the COMPASS experiment at CERN has been presented [4].

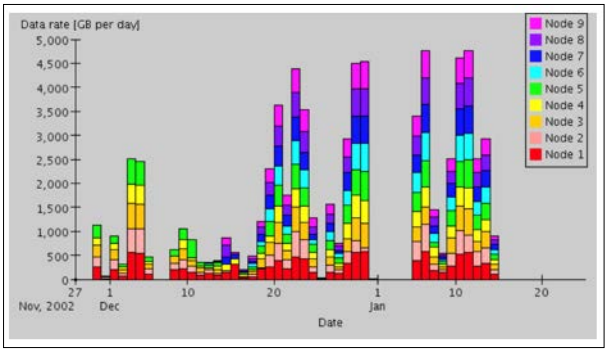
Recent analyses report that many sectors of our economy and society are more and more guided by data-driven decision processes (e.g., healthcare, public administrations, etc.). As such, Data Intensive (DI) applications are becoming more and more important and critical. They must be fault-tolerant, they should scale with the amount of data, and be able to elastically leverage additional resources as and when these last ones are provided. Moreover, they should be able to avoid data drops introduced in case of sudden overloads and should offer some Quality of Service (QoS) guarantees.

Ensuring all these properties is, per se, a challenge, but it becomes even more difficult for DI applications, given the large amount of data to be managed and the significant level of parallelism required for its components. Even if today some technological frameworks are available for the development of such applications (for instance, think of Spark, Storm, Flink), still lack solid software engineering approaches to support their development and, in particular, to ensure that they offer the required properties in terms of availability, throughput, and data loss.

This implies the need for highly skilled persons and the execution of experiments with large data sets and many resources, and, consequently, a significant amount of time and budget. To achieve the desired level of performance, fault tolerance, and recovery, it had to adopt a time-consuming, experiment-based approach, which, in our case, consisted of three iterations: the design and implementation of a Mediation Data Model capable of managing data extracted from different databases, the improvement of performance of our prototype when managing and transferring huge amounts of data; the introduction of fault-tolerant data extraction and management mechanisms, which are independent from the targeted databases [1,4].

**Table 4:** Lists the Number of Files and the Size of the Data Stored by the COMPASS Experiment in Each of its Many Independent "Periods" of Data Taking, as of Before the Start of the Migration

| Period | Raw Data (in GB) | Metadata (in GB) | #Events (x1000) |
|---|---|---|---|
| 2002 P1C | 10972.56 | 19.25 | 332.23 |
| 2002 P2A | 25572.87 | 40.69 | 764.87 |
| 2002 P2C | 19023.60 | 28.30 | 528.31 |
| 2002 P2D | 18689.93 | 24.36 | 455.49 |
| Total | 74258.96 | 112.60 | 2080.90 |




**Graph 2:** Data Throughput [GB per day]

## Conclusion

Choosing the right tool for data migration is critical to ensure the success of the migration process. Apache Spark emerged as the best solution for data migration based on its performance, scalability, and user-friendliness. Additionally, the referenced randomized online algorithm provides a cost-effective solution for managing user-generated data in cloud storage.

It is important but challenging for STaaS cloud users to make decisions on when to migrate data objects to cold storage tiers to optimize costs, as the overall trend of accesses is downward, while users do not know exact future access curves. A randomized online algorithm with two density functions can guide STaaS cloud users to make decisions on when to migrate objects to cold storage tiers without knowledge of their future access curves. Theoretically, this randomized online algorithm can achieve a guaranteed competitive ratio of:

$$1+\frac{2(1-\lambda)}{e-3+2\lambda+\frac{\lambda}{\alpha}}$$

This algorithm can be extended with a prediction window when short-term predictions are reliable.

The problem of optimizing the monetary cost spent on storage services when data-intensive applications with time-varying workloads are deployed across data stores with several storage classes was also studied. An optimal algorithm was proposed, but due to its high time complexity, a new heuristic solution formulated as a Set Covering problem with three policies was introduced. This solution takes advantage of pricing differences across cloud providers and the status of objects that change from hot-spot to cold-spot during their lifetime and vice versa. The evaluation results demonstrate that this solution is capable of reducing the cost of data storage management by approximately two times in some cases when compared to the widely used benchmark algorithm in which the data are stored in the closest data store to the users who access them [5].

## Future Work

Future work should explore new algorithms to optimize costs for objects with more than two statuses (e.g., cold, warm, and hot). Additionally, the home data center can be selected based on the mobility of the users, affecting the response time of the Gets/Puts issued by client data centers. Using a quorum-based model for data consistency provides stronger consistency semantics compared to eventual consistency. To determine the gap between the proposed optimal and heuristic algorithms in cost performance, it is required to compute the competitive ratio defined as the ratio between the worst incurred cost by the heuristic algorithm and the cost incurred by the optimal algorithm. This involves computing the upper-bound cost for the optimization of replicas migration and replica placement based on covered load volume.

## References

1. Kammachi HJ, Vishalakshi PH (2023) Accelerating Data Migration: A Comparative Study of Apache Spark and Other Leading Tools. IEEE 11th Region 10 Humanitarian Technology Conference (R10-HTC), Rajkot, India 790-794.
2. Liu M, Pan L, Liu S (2021) Keep Hot or Go Cold: A Randomized Online Migration Algorithm for Cost Optimization in STaaS Clouds. IEEE Transactions on Network and Service Management 18: 4563-4575.
3. Mansouri Y, Buyya R (2019) Dynamic replication and migration of data objects with hot-spot and cold-spot statuses across storage data centers. Journal of Parallel and Distributed Computing 126: 121-133.
4. Scavuzzo M, Di Nitto E, Ardagna D (2018) Experiences and Challenges in Building a Data Intensive System for Data Migration. 2018 IEEE/ACM 40th International Conference on Software Engineering (ICSE), Gothenburg, Sweden 93-93.
5. Lubeck M, Geppert D, Nienartowicz K (2003) An overview of a large-scale data migration. 20th IEEE/11th NASA Goddard Conference on Mass Storage Systems and Technologies, 2003. (MSST 2003). Proceedings. San Diego, CA, USA 49-55.