# Swarm: A federated cloud framework for large-scale variant analysis

(Bahmani A, Ferriter K, Krishnan V, Alavi A, Alavi A, Tsao PS, et al. 2021)

Candidate: Vali Florinel Craciun
Privacy in pangenomics, a.y. 2023-2024
Prof. Dr. Alexander Schönhuth

February 2024

**Abstract**

The exponential increase in genomic data production has brought substantial challenges in terms of analysis, necessitating a paradigm shift in our computational and storage capabilities. The sheer volume of data demands more extensive computational resources for processing and heightened storage capacity. Compounding this, genomic data is often generated by a multitude of entities scattered globally, resulting in a fragmented storage landscape. Swarm is a framework for federated computation that promotes minimal data motion and facilitates crosstalk between genomic datasets stored on various cloud platforms. This framework represents a cheap, fast and secure solution to extract and handle genomic data.

## 1 Introduction

Big data is fundamentally reshaping precision medicine and the landscape of information technology in the life sciences. The exponential growth of data in genomics research is transitioning the bottleneck from sequencing[1] costs to the demands of storage and computation [1]. To address these challenges, scalable, cheap and distributed computing is essential.

Cloud computing emerges as a crucial solution, offering elastic scalability and a flexible pay-as-you-go model. It empowers researchers to store vast amounts of data and conduct intensive computing [2], facilitating data sharing and collaboration. Given that, cloud computing represents a well-suited solution for large-scale genomic analysis [3–5].

[1]DNA sequencing is the process of determining the nucleic acid sequence – the order of nucleotides in DNA.

Currently, the genomic community employs multiple cloud service platforms for data hosting, computing, and sharing. For instance, the Encyclopedia of DNA Elements[6] relies on Amazon Web Services (AWS), while the Genome Aggregation Database[7] uses Google Cloud Platform (GCP). However, due to platform heterogeneity, joint analyses across cloud platforms are not optimized. Data motion across these platforms poses challenges, leading to high transfer costs, delays, and security and privacy risks.

With this said, that's where a framework like Swarm becomes valuable. Swarm includes an API leveraging a serverless computing model [8] to assess data motion needs, conduct in-situ computation whenever possible, and facilitate data transfer when necessary. Testing the framework in various research scenarios involving genomic variants demonstrates an optimized data crosstalk solution, significantly reducing the costs associated with data motion.

The federated computational framework of Swarm enables researchers with the same security protocols and data access rights to freely utilize services from different computational platforms without being restricted to specific features of the original data hosting platform. Minimizing data motion between cloud providers in Swarm also enhances the security and privacy of health data during transfer. Cloud providers, such as GCP [9] and AWS [10], encapsulate best practices supporting health privacy regulations, like the Health Insurance Portability and Accountability Act (HIPAA). Leveraging these logging, auditing, and monitoring best practices and using strong authentication and encryption mechanisms for data in transit would help reduce the security and privacy risks associated with multi-cloud orchestration.

## 2 Swarm framework

First of all, let's analyze how the Swarm framework works and which are its main components.

Swarm categorizes queries into two primary types, "StatQuery" and "DataQuery".

**StatQuery.** This part of the framework deals with queries that do not necessitate data motion and provides simple statistics such as matched record counts, averages etc.

**DataQuery.** On the other hand, the Data Query part manages queries involving the transfer of a record set to another computing platform for additional processing. Since in this scenario we need to deal with moving data, minimizing the amount of data moved is crucial for security reasons, especially to reduce the possible damages coming from in-transit attacks. To do so, Swarm assesses the size of compressed data returned from each platform, selects the smaller set, and transfers it to the computing platform hosting the larger dataset. Following the transformation of the small dataset into a temporary table in the new computing environment, Swarm merges the two datasets and delivers the final result to users.

While the default option is to move the smaller dataset, Swarm also offers

the flexibility to move the larger data, accommodating users' project-specific privacy and security requirements. In figure 1 a schematic representation of the framework is provided.

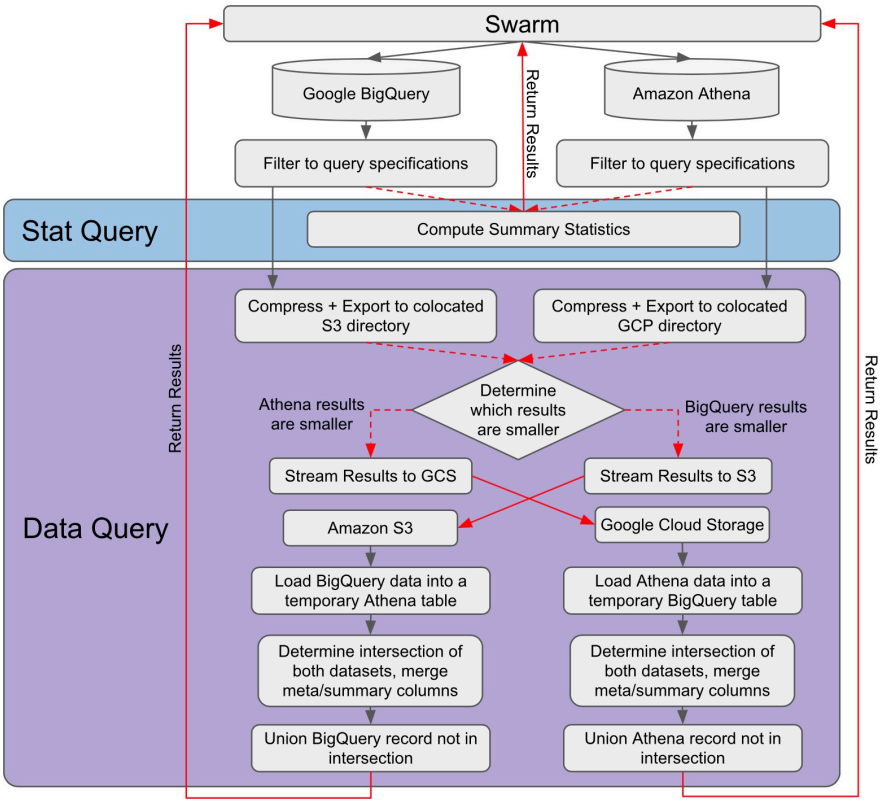


Figure 1: **Swarm framework**. The framework evaluates whether the required data can be retrieved separately from both datasets when given one or more queries. BigQuery and Athena were utilized in this instance, but as we'll demonstrate in the following sections, Swarm support multiple platforms. When data from different platforms doesn't need to be combined, StatQuery calculates the required statistics and provides the results. However, if joining datasets from different platforms is necessary to fulfill the query, DataQuery assists in this process. As depicted in the diagram, the initial step in Data Query involves compressing and comparing the necessary data from both platforms, selecting the smaller dataset for transfer to the platform hosting the larger tables to minimize data movement. Subsequently, all required operations such as joins and merges are executed, and the results are returned. Source: Bahmani A, Ferriter K, Krishnan V, Alavi A, Alavi A, Tsao PS, et al. (2021) Swarm: A federated cloud framework for large-scale variant analysis. PLoS Comput Biol 17(5): e1008977. https://doi. org/10.1371/journal.pcbi.1008977

# 3 Experiments setting

Swarm's performances were assessed through a series of experiments. Each experiment was conducted four times, and the average values along with standard deviations were calculated. To assess differences between pairs of experiments, F tests were employed to evaluate variances, while two-sample t-tests were utilized to compute P values. Within a group of experiments, such as runtimes with varying numbers of nodes in the Apache Presto runs, Anova tests were employed for calculating P values.

To assess the capabilities of Swarm, a set of queries were run. The queries were mainly about computing allele frequencies and functional annotations for certain variants, more about it in the next paragraphs. In doing so, multiple datasets were used [11], with the main one being the 1000 Genomes phase three, which regards data gathered during the third phase of the homonym project, explained in the next section.

## 3.1 The 1000 Genomes Project

The 1000 Genomes Project [12] is a groundbreaking international initiative aimed at creating a comprehensive catalog of human genetic variation. Launched in 2008, this collaborative effort brought together scientists from around the world with the ambitious goal of sequencing the genomes of a diverse group of individuals to better understand the spectrum of human genetic diversity.

The project's primary objective was to identify and characterize genetic variants present in at least 1% of the global population, providing a valuable resource for researchers studying the genetic basis of diseases, population history, and evolutionary processes. By capturing variations in the DNA sequences of thousands of individuals from various ethnic groups, the 1000 Genomes Project aimed to shed light on the genetic factors influencing health and disease susceptibility.

Over the course of several years, the project utilized advanced DNA sequencing technologies to analyze the genomes of around 2,504 individuals from 26 populations across Africa, the Americas, East Asia, Europe, and South Asia. This comprehensive dataset has significantly contributed to our understanding of human genetic variation, allowing researchers to identify common and rare genetic variants associated with various traits and diseases.

The 1000 Genomes Project has had a profound impact on genetics research, serving as a foundational resource for studies in population genetics, medical genetics, and personalized medicine. The data generated by the project is freely accessible to the scientific community, fostering collaboration and accelerating discoveries in the field of genomics.

## 3.2 Genomics background

This subsection provides a concise overview of the intended outcomes of the experiments' queries, emphasizing their biological meaning. It aims to cater to

individuals with limited knowledge in genomics.

As discussed in Section 3, the queries primarily center around computing allele frequencies and functional annotations for specific variants. Let's delve into the meaning of this concept.

### 3.2.1 What is a variant?

To begin with the basics, a gene is essentially a segment of DNA responsible for the manifestation of distinct traits, be it the color of eyes or susceptibility to a particular disease.

The emergence of specific traits is linked to mutations occurring at specific locations within a gene, commonly referred to as "variants." Figure 2 illustrates precisely this relationship, showcasing the connection between a particular variant and the specific trait associated with it. To be more precise, the rsIDs relates to a so-called SNP (single nucleotide polymorphism), which represents a variation in a single nucleotide (the building blocks of DNA) at a specific position in the genome (Figure 3 for graphical explanation).

| Description | rsID | Chr | Pos |
|---|---|---|---|
| Attention-deficit/hyperactivity disorder (ADHD) | rs671 | 12 | 112241766 |
| Blue Eye Color (BEC) | rs12913832 | 15 | 28365618 |
| Coronary Heart Disease (CHD) | rs1333049 | 9 | 22125503 |
| Lactose Intolerance | rs4988235 | 2 | 136608646 |

Figure 2: Variants used for testing Stat Queries. Source: Bahmani A, Ferriter K, Krishnan V, Alavi A, Alavi A, Tsao PS, et al. (2021) Swarm: A federated cloud framework for large-scale variant analysis. PLoS Comput Biol 17(5): e1008977. https://doi.org/10.1371/journal.pcbi.1008977

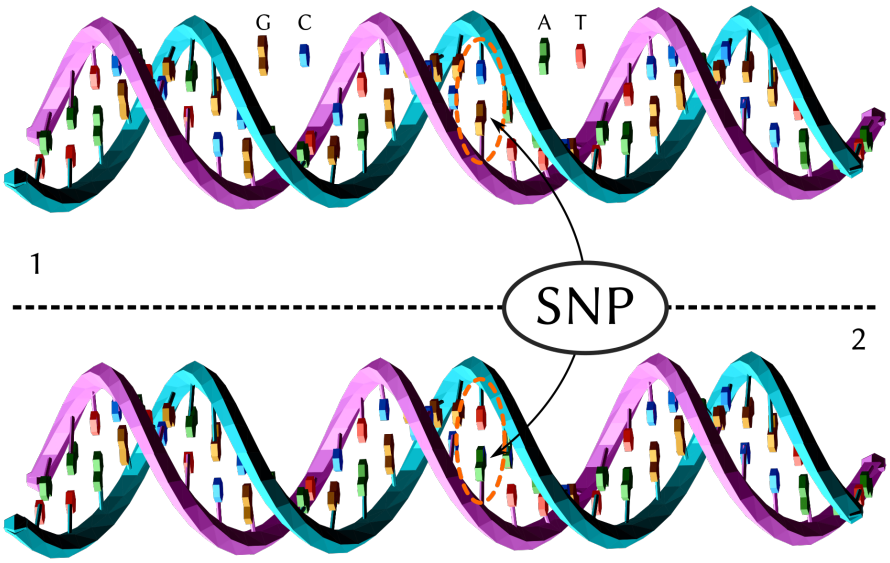


Figure 3: The upper DNA molecule differs from the lower DNA molecule at a single base-pair location. This an exapmle of G/A polymorphism, since the upper DNA contains the G base while the lower contains the A base because of the mutation. Source: SNP model by David Eccles (gringer), CC BY 4.0, https://commons.wikimedia.org/w/index.php?curid=2355125

Now that we know what a variant is, let's describe what computing allele frequencies and functional annotation of a variant means.

**Allele frequencies** Computing the allele[2] frequency for a specific SNP (or variant), such as rs671 of Figure 2, involves determining the proportion of a population that carries each of the possible alleles at that particular genetic locus. In poor words, understanding how common is a certain mutation for a specific spot in a gene.

The two common alleles at an SNP are often referred to as the "major allele" and the "minor allele." The allele frequency is usually expressed as a percentage and represents the proportion of chromosomes in the population that carry a specific allele.

For example, if in a population of 100 individuals, 60 individuals carry the "A" allele (major allele) and 40 carry the "G" allele (minor allele) at the rs671 locus, the allele frequencies would be:

Major allele (A): 60% (60 individuals out of 100) Minor allele (G): 40% (40 individuals out of 100).

Computing allele frequencies is essential for understanding the genetic diversity within populations and can be valuable in various fields, including popula-

[2] An allele, to simplify, can be considered as one of the possible mutations that may happen.

tion genetics, medical research, and forensic genetics. It provides insights into the distribution of genetic variations and helps researchers investigate associations between specific alleles and traits or diseases.

**Functional annotations of variants** Functional annotation of variants involves interpreting the potential functional consequences or effects of genetic variants identified in an individual's genome. When a person's DNA is sequenced, various genetic variations, such as single nucleotide polymorphisms (SNP, as explained before) or other types of mutations, are detected. Understanding the functional annotation of these genetic variants is crucial for deciphering their potential impact on gene function, protein structure, and overall biological processes.

# 4 Experiment 1 - Stat Query: Allele frequency computation across clouds

The first experiment was conducted by partitioning the 1000 Genomes samples and their genomic variants, storing half in GCP BigQuery and the remaining half in AWS Athena. Swarm was employed to compute variant allele frequencies in each cloud, and the results were subsequently merged to derive combined allele frequencies. To enhance performance, partitioning (4000 partitions over the start position) and clustering methods were implemented.

## 4.1 Results

Figure 4a illustrates the execution time with and without these optimizations, while Figure 4b presents the processed data volume.

For the variants listed in Figure 2, partitioning and clustering significantly improved runtimes. Swarm processed 0.15GB of data from BigQuery to calculate allele frequencies for half of the samples in the 1000 Genomes dataset. Considering the entire BigQuery table was 540GB, this represented a small fraction. In comparison to the alternative of transferring the dataset to the second cloud, Swarm performed in situ computation, reducing the egress cost by 99.98%[3]. In this specific instance, users could execute approximately 3,600 similar queries before reaching the break-even point. It's important to note that this experiment utilized only half of the 2,500 samples from the 1000 Genomes Project. Larger cohorts with hundreds of thousands or millions of samples stand to benefit significantly from systems like Swarm.

[3]BigQuery and Athena charges users on the amount of data processed. This huge cost reduction was possible due to the optimization techniques that made possible to traverse and process way less data than the whole table

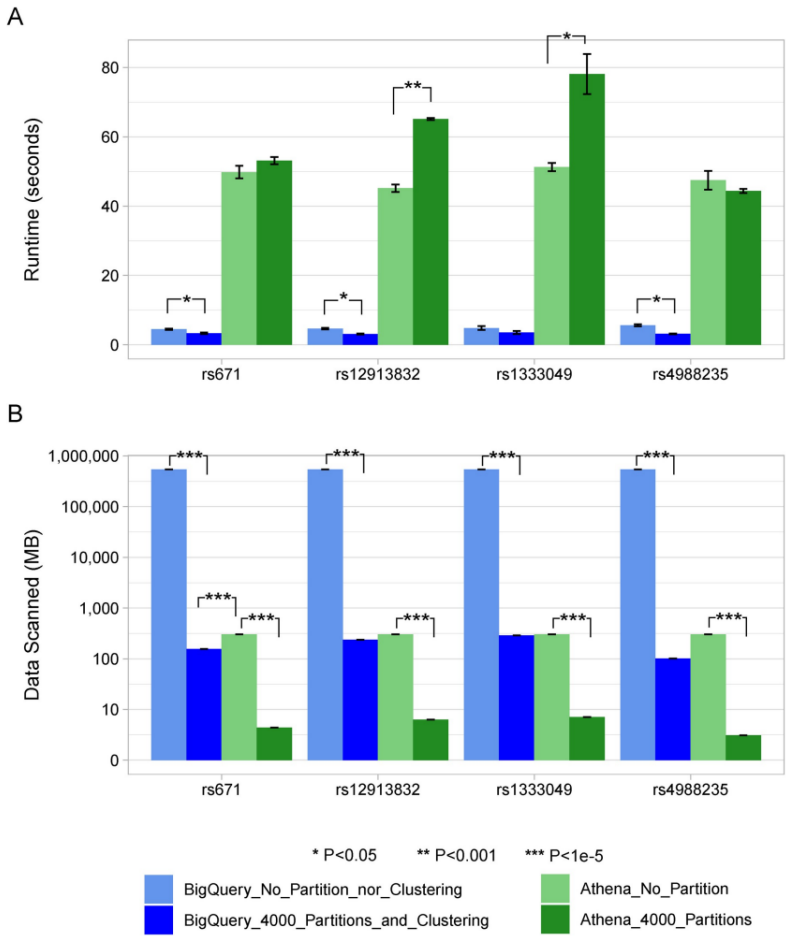


Figure 4: **Runtime and amount of data processed for computing allele frequency for an input set of rsIDs in BigQuery and Athena**. Average values and standard deviations were plotted. (A) depicts the average execution time in seconds. The light blue and light green bars represent configurations without any optimizations (i.e., the entire input data used as it was), and the dark blue and dark green bars represent configurations with optimizations (i.e., the input data was divided by partitioning or clustering); (B) shows the amount of data processed in megabytes, and the y-axis is logarithmic in scale. Significance differences between groups are indicated on top of the bars (two samples t-test). Note that for each rsID experiment, differences in runtimes between any BigQuery and Athena runs in (A) were highly significant ($P < 1e-5$), and for (B), differences within the BigQuery or Athena runs were also highly significant ($P < 1e-5$). Source: : Bahmani A, Ferriter K, Krishnan V, Alavi A, Alavi A, Tsao PS, et al. (2021) Swarm: A federated cloud framework for large-scale variant analysis. PLoS Comput Biol 17(5): e1008977. https://doi.org/10.1371/journal.pcbi.1008977

# 5 Experiment 2 - Data Query: Comprehensive functional annotation of variants or genes across clouds

In the second experiment, a scenario where variant calling results were stored on one computing platform (AWS Athena), while reference files for functional annotation were stored on another (GCP BigQuery), was simulated. The query aimed to annotate variants related to the TP53[4] and APC [5]gene. Swarm determined that the most efficient dataset to transfer was the extracted functional annotation records from BigQuery. These records, compressed and encrypted, were sent to AWS, stored in a temporary AWS Athena table and utilized for joint analysis with the functional reference table.

## 5.1 Results

Figure 5 illustrates Swarm's performance in annotating a set of genes. Here, Swarm moved annotation records overlapping with the input genes from BigQuery to Athena and conducted the annotation. The optimization strategies involving partitioning and clustering notably reduced runtime and minimized the amount of data scanned. In this instance, Swarm processed 99.4 MB in BigQuery and 13.5 MB in Athena. It compressed and transferred a 3.1 MB subset of the 233 GB annotation table from BigQuery to Athena, as opposed to copying the entire annotation table to Athena for computation. This approach resulted in moving only 0.001% of the full annotation table, showcasing the efficiency of Swarm in handling complex scenarios with substantial data while optimizing data transfer and processing.

[4]TP53 is the most common mutated gene associated with human cancer.

[5]APC gene mutation is associated with familial adenomatous polyposis, an inherited condition that affects the gastrointestinal tract.

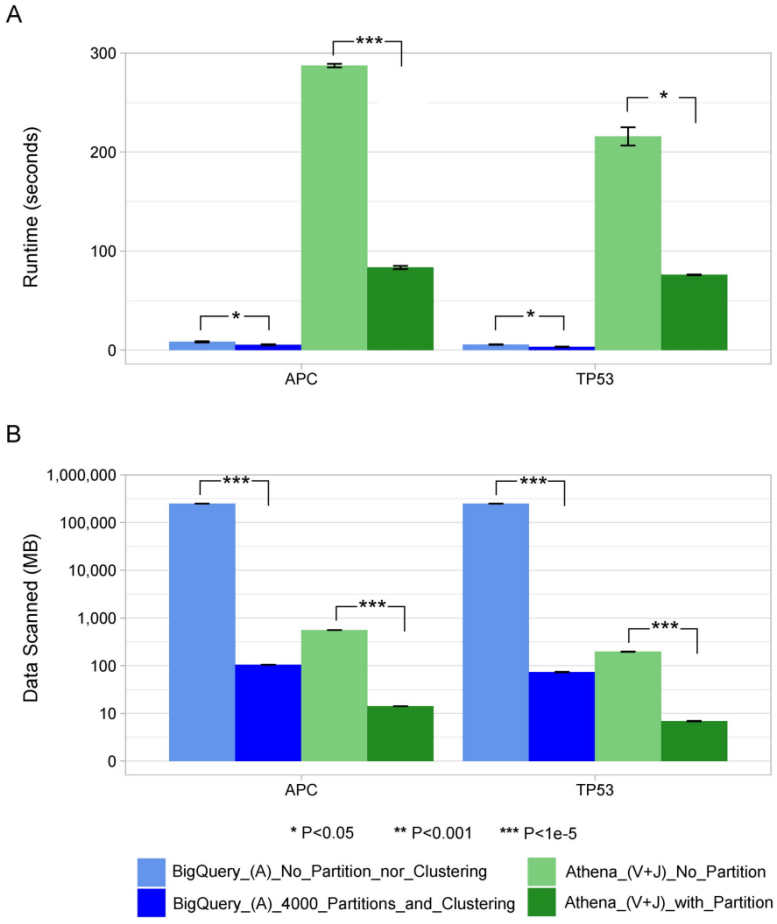


Figure 5: **Runtime and the amount of data processed for annotating an input set of genes.** A, V and J stand for Annotation records, Variant records and Join table operations, respectively. Average values and standard deviations were plotted. (A) depicts the execution time in seconds for the two input genes. In this experiment, the annotation table was in BigQuery and the variant table in Athena. Therefore, Swarm first found all the annotation records in BigQuery that overlapped with the input gene regions, compressed them and moved them to Athena. Then, on the Athena side, Swarm decompressed the overlapping annotation data and created a temporary table, which was eventually processed to join with the existing variant table. The light blue and light green represent the configurations without any optimizations by partitioning or clustering, and the dark blue and dark green represent the configurations with optimizations. (B) shows the amount of data processed in megabytes, and the y-axis is logarithmic in scale. Significance differences between groups are indicated on top of the bars (two samples t-test). Note that for (A), differences between any BigQuery and Athena groups were highly significant ($P < 1e-5$), and for (B), differences within the BigQuery or Athena groups were also highly significant ($P < 1e-5$). Source: : Bahmani A, Ferriter K, Krishnan V, Alavi A, Alavi A, Tsao PS, et al. (2021) Swarm: A federated cloud framework for large-scale variant analysis. PLoS Comput Biol 17(5): e1008977. https://doi. org/10.1371/journal.pcbi.1008977

# 6 Experiment 3 - Variant query in Apache Presto

In the third experiment, the cloud-agnostic capability of the Swarm platform was showcased. This demonstration involved evaluating query processing times for the rsID rs671, to compute allele frequencies. In this setting a DataProc cluster with different numbers of worker nodes (all n1-standard-4 machine types) was tried.

Google Cloud Dataproc is a fully-managed cloud service for running Apache Spark and Apache Hadoop clusters. It is an helpful program when dealing with such clusters since it offers features like:

- Cluster Deployment. Users specify the structure of the cluster and the type of nodes needed.
- Jobs submission. Users provide scripts (Python, Java, Scala...) in which tasks to be performed are specified.
- Resource management and optimization. DataProc split the task between (worker) nodes by means of a master node (cluster manager). The goal of the master node is to assign the tasks to worker nodes such that the resources are used in the most efficient way possible.

## 6.1 Results

Figure 6a presents the average runtime for querying rs671 for the cluster with varying numbers of nodes.

The rapid computation time can be attributed to the dedicated Presto experiments within these sets of nodes. After partition discovery, these nodes cache metadata, resulting in shorter execution times. In contrast, serverless systems like BigQuery and Athena assign a new set of nodes for each input query, lacking familiarity with the data structure, and consequently leading to increased query response times without caching. The Presto cluster exhibited significantly faster execution times than Amazon Athena.

Figure 6b illustrates the runtimes for searching for rs671 in a Dataproc cluster using partitioning and preemptible instances. In this example, the preemptible cluster included N worker nodes, with only two being non-preemptible instances (Non-PVM), while the rest were preemptible (N-2 PVMs).

A preemptible node is a type of virtual machine instance offered by cloud computing providers that can be preempted, or terminated, by the cloud provider with little notice (usually with a 30-second warning). This termination can occur if the resources are needed for other purposes or if there is excess demand for regular instances in the data center.

The execution time remained almost identical in the tested scenario, but Fig 6c demonstrates a substantial cost reduction when using preemptible instances as the number of worker nodes increased[6]. Dataproc utilizes preemptible in-

[6]Distributed systems like Apache Presto charges users based on how much time storage, memory and CPU are allocated.

stances as secondary workers to scale computation without scaling storage, as preemptible instances are unsuitable for HDFS storage [7].

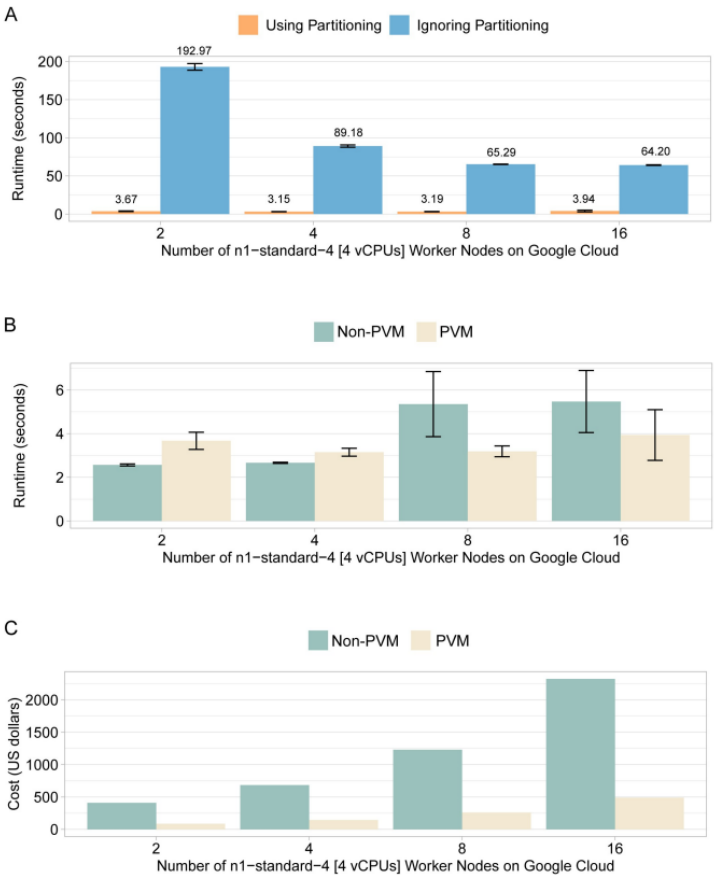


Figure 6: **Execution time for searching rs671 with different number of worker nodes for running Apache Presto on Dataproc.** (A) The average runtime using partitioning versus ignoring partitioning in Apache Presto. (B) The average runtime using preemptible (PVM) and non-preemptible (Non-PVM) instances. Average values and standard deviations were plotted. (C) The projected cost of reserving the dedicated nodes on GCP on a monthly basis. Monthly cost as of February 2021 https://cloud.google.com/compute/all-pricing. In (A), Differences between the paired groups of with or without partitioning were highly significant (two sample t-tests $P < 1e-5$). In (B), differences between the paired groups of Non-PVM and PVM, although not significant, had marginal P values close to 0.05. Source: : Bahmani A, Ferriter K, Krishnan V, Alavi A, Alavi A, Tsao PS, et al. (2021) Swarm: A federated cloud framework for large-scale variant analysis. PLoS Comput Biol 17(5): e1008977. https://doi. org/10.1371/journal.pcbi.1008977

[7] Hadoop Distributed File System (HDFS) is a distributed file system designed to store and manage very large data sets reliably and efficiently across a cluster of commodity hardware.

# 7 Experiment 4 - Variant query in MySQL

For the fourth experiment the whole 1000 Genome dataset was uploaded into MySQL 5.7, running the same queries of experiment 3. Differently than Presto, here just one n1-standard machine has been used, and various number of vCPU have been tried. The dataset was loaded as both CSV and Parquet file, and a comparison between MySQL and Presto was carried on CSV files.

## 7.1 Results

Figure 7a illustrates the performance of MySQL 5.7 in searching for rs671. In this experiment, MySQL exhibits efficiency on the instance equipped with 8 vCPUs and 30 GB memory, primarily attributed to the size of the table, which is 15.97 GB, fitting well within the main memory of the instance.

Figure 7b presents the performance of querying a table imported as CSV compared to the Parquet version on Apache Presto. The data scanned for Parquet was notably smaller at 1.27 GB compared to the CSV version, which amounted to 12.7 GB. This highlights the efficiency of Parquet as a columnar storage format, particularly well-suited for columnar distributed SQL query engines such as Apache Presto.

Overall, the MySQL experiment demonstrates vertical scaling (increasing the number of vCPUs per worker node), while the Presto experiment represents horizontal scaling (increasing the number of worker nodes). In a straightforward comparison, it becomes evident that MySQL running on a worker node with 8 vCPUs (CSV: 63.2625 seconds) is considerably slower than Apache Presto running on two worker nodes with a combined 8 vCPUs (Parquet: 12.205 seconds, CSV: 29.005 seconds).

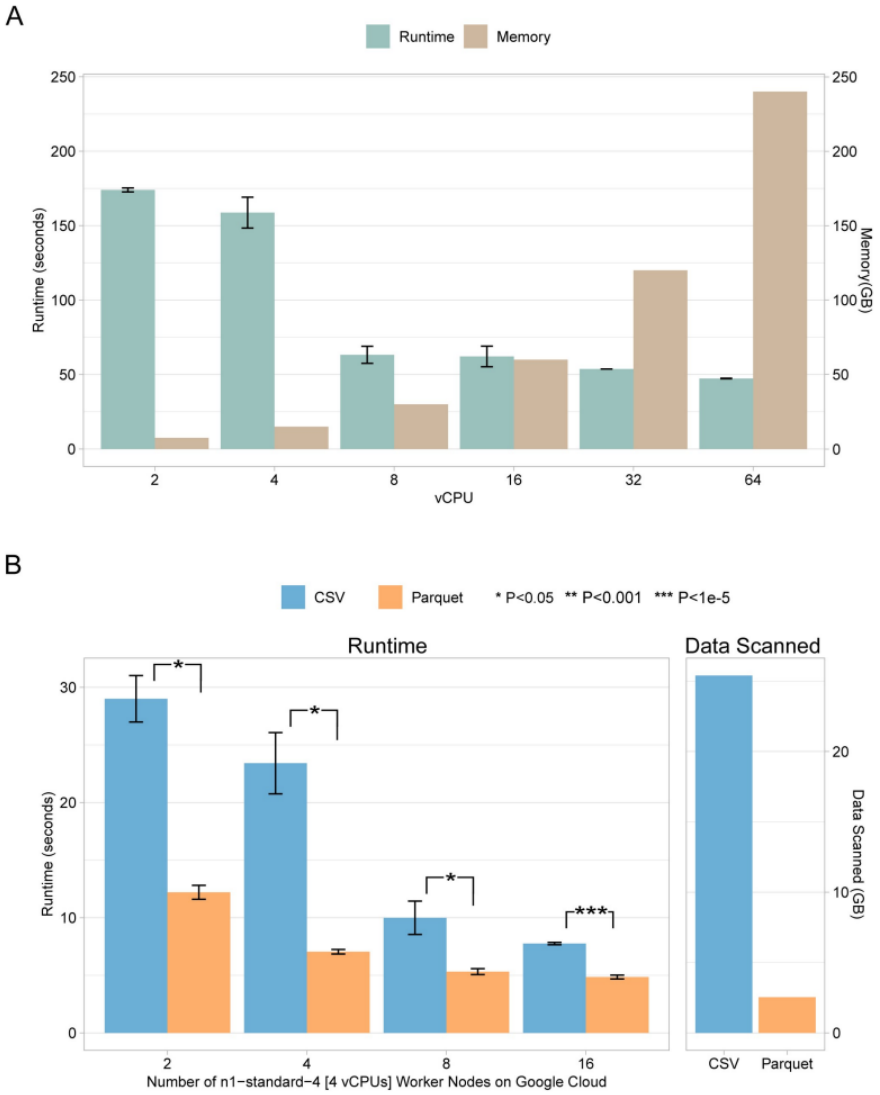


Figure 7: **Searching rs671 in the 1000 Genomes dataset loaded in (A) MySQL and (B) Apache Presto with different settings**, i.e., varying number of vCPUs and main memory sizes. Average values and standard deviations were plotted. For the Apache Presto runs in (B), runtimes between CSV input versus Parquet input were compared, and significant P values are indicated (two sample t-tests). In addition, anova tests indicated that the number of worker nodes had a significant impact on the runtimes ($P < 1e-5$). Source: Bahmani A, Ferriter K, Krishnan V, Alavi A, Alavi A, Tsao PS, et al. (2021) Swarm: A federated cloud framework for large-scale variant analysis. PLoS Comput Biol 17(5): e1008977. https://doi.org/10.1371/journal.pcbi.1008977

# 8 Conclusion

This paper introduced Swarm, a federated computation framework specifically designed to minimize data transfer and enhance communication among genomic datasets dispersed across diverse cloud platforms. Through four meticulously conducted experiments detailed in preceding sections, Swarm's key properties and innovations were showcased.

Experiment 1 illustrated the functionality of StatQuery and underscored how table optimization techniques (partitioning and clustering) significantly reduced query costs. Experiment 2 elucidates the workings of DataQuery and showcased Swarm's management of data motion, strategically minimizing it to enhance security. In Experiments 3 and 4, the cloud-agnostic nature of Swarm is demonstrated, with the first showing compatibility with distributed systems such as Apache Presto initially and later extending support to Relational Databases like MySQL in experiment 4. Notably, Experiment 3 highlights another pivotal feature of Swarm, the ability to interface seamlessly with additional services, such as integration with Google Dataproc for cluster deployment and management.

While Swarm exhibits exceptional performance in the conducted experiments, it is crucial to acknowledge certain limitations inherent in the underlying computing frameworks.

## 8.1 Limitations

Presently, BigQuery imposes a maximum limit of 4,000 partitions per partitioned table and 10,000 columns. AWS Athena, on the other hand, supports up to 20,000 partitions per partitioned table. BigQuery and Athena dynamically scale based on workloads. In contrast, Apache Presto operates on a Hadoop cluster, utilizing a different computing environment based on distributed architecture. Google Dataproc has been utilized, a managed cloud service supporting Apache Spark and Apache Hadoop, so system engineers are required to fine-tune clusters based on request volume, table size, and query complexity.

## 8.2 Further improvements

Additional privacy protection techniques can be seamlessly integrated into the proposed framework to be even less susceptible to man-in-the-middle attacks. Firstly, **Secure Multiparty Computation** (SMPC) [17] cryptographic protocols that enable multiple parties to collaboratively compute a function over their inputs while ensuring complete zero knowledge about those inputs. While these protocols preserve the privacy of inputs and processes among parties, addressing privacy concerns about the output is crucial.

Secondly, **Differential Privacy methods** [18] introduce noise to the data to simultaneously achieve high anonymization and utility preservation. The Swarm framework can be adapted to support differential privacy, K-anonymity, and K-isomorphism anonymization schemes.

Thirdly, **Homomorphic encryption** [19], another cryptographic protocol, protects data privacy by enabling computations over encrypted data without access to the secret key. However, the main bottleneck for such computation-heavy privacy-preserving techniques is latency.

Another enhancement could involve expanding the core code APIs to enhance flexibility, allowing the framework to accommodate data types beyond variants and variant annotation. This would enable more generic querying and functionality comparable to dataframes in Pandas or Spark libraries, allowing for the detection of columns and defining arbitrary table keys more broadly.

**References**


[1] Kahn SD. On the future of genomic data. Science. 2011; 331(6018):728–729. https://doi.org/10.1126/science.1197891 PMID: 21311016

[2] Langmead B, Nellore A. Cloud computing for genomic data analysis and collaboration. Nature Reviews Genetics. 2018; 19(4):208.https://doi.org/10.1038/nrg.2017.113 PMID: 29379135

[3] Bahmani A, Sibley A, Parsian M, Owzar K, Mueller F. SparkScore: Leveraging Apache Spark for Distributed Genomic Inference. IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), Chicago, IL, USA. 2016;435–442.

[4] Wall DP, Kudtarkar P, Fusaro VA, Pivovarov R, Patil P, Tonellato PJ. Cloud computing for comparative genomics. BMC Bioinformatics. 2010; 11(1):259.

[5] Pan C, McInnes G, Deflaux N, Snyder M, Bingham J, Datta S, et al. Cloud-based interactive analytics for terabytes of genomic variants data. Bioinformatics. 2017; 33(23):3709–3715. https://doi.org/10.1093/bioinformatics/btx468 PMID: 28961771

[6] The ENCODE Project Consortium., Moore JE., Purcaro MJ., Pratt HE, Epstein CB, Shoresh N et al. Expanded encyclopaedias of DNA elements in the human and mouse genomes. Nature. 2020; 583, 699–710. https://doi.org/10.1038/s41586-020-2493-4 PMID: 32728249

[7] Karczewski K. J., Francioli L. C., Tiao G., Cummings B. B., Alfo¨ldi J., Wang Q et al. The mutational constraint spectrum quantified from variation in 141,456 humans. Nature. 2020; 581, 434–443. https://doi.org/10.1038/s41586-020-2308-7PMID:32461654

[8] Ebert C, Gallardo G, Hernantes J, Serrano N. Devops. IEEE Software. 2016; 33(3):94–100.

[9] HIPAA Compliance on Google Cloud Platform: https://cloud.google.com/security/compliance/hipaa

[10] AWS HIPAA: https://aws.amazon.com/compliance/hipaa-compliance/

[11] Datasets:

dbNSFP 35a → https://sites.google.com/site/jpopgen/dbNSFP

1000Genomes → https://www.internationalgenome.org/data/

RegulomeDB → https://www.regulomedb.org/regulome-search/

ClinVar → https://ftp.ncbi.nlm.nih.gov/pub/clinvar/vcf_GRCh37/

UCSC gtex Eqtl Cluster →https://genome.ucsc.edu/gtex.html

gnomAD → https://gnomad.broadinstitute.org/downloads

Cosmic70 → https://annovar.openbioinformatics.org/en/latest/user-guide/filter/#cosmicannotations

Dann → https://annovar.openbioinformatics.org/en/latest/user-guide/filter/#dann-annotations

Eigen → https://annovar.openbioinformatics.org/en/latest/user-guide/filter/#eigen-scoreannotations

eQTL & Diseases → http://www.exsnp.org/Download

GTEx Analysis v7 eQTL → https://gtexportal.org/home/datasets

SNP151 → https://genome.ucsc.edu/cgi-bin/hgTrackUi?db=hg38&g=snp151

Wellderly → https://genomics.scripps.edu/browser/files/wellderly/vcf

[12] https://www.internationalgenome.org/

[13] Karki R, Pandya D, Elston RC, Ferlini C. Defining "mutation" and "polymorphism" in the era of personal genomics. BMC Med Genomics. 2015 Jul 15;8:37. doi: 10.1186/s12920-015-0115-z. PMID: 26173390; PMCID: PMC4502642.

[14] Hufeng Zhou, Theodore Arapoglou, Xihao Li, Zilin Li, Xiuwen Zheng, Jill Moore, Abhijith Asok, Sushant Kumar, Elizabeth E Blue, Steven Buyske, Nancy Cox, Adam Felsenfeld, Mark Gerstein, Eimear Kenny, Bingshan Li,

Tara Matise, Anthony Philippakis, Heidi L Rehm, Heidi J Sofia, Grace Snyder, NHGRI Genome Sequencing Program Variant Functional Annotation Working Group, Zhiping Weng, Benjamin Neale, Shamil R Sunyaev, Xihong Lin, FAVOR: functional annotation of variants online resource and annotator for variation across the human genome, Nucleic Acids Research, Volume 51, Issue D1, 6 January 2023, Pages D1300–D1311, https://doi.org/10.1093/nar/gkac966

[15] https://cloud.google.com/dataproc?hl=it

[16] https://cloud.google.com/kubernetes-engine/docs/how-to/preemptible-vms

[17] Secure Multiparty Computation (MPC) Yehuda Lindell Unbound Tech and Bar-Ilan University, https://eprint.iacr.org/2020/300, 2020

[18] Differential Privacy and Machine Learning: a Survey and Review Zhanglong Ji, Zachary C. Lipton, Charles Elkan, arXiv:1412.7584, 2014

[19] Complex Adaptive Systems, Publication 3 Cihan H. Dagli, Editor in Chief Conference Organized by Missouri University of Science and Technology, 2013- Baltimore, MD. Homomorphic Encryption. Monique Ogburna, Claude Turnerb , Pushkar Dahalc. Bowie State University, Department of Computer Science, 14000 Jericho Park Rd.,Bowie, MD 20715, United States