PGPregel: An End-to-End System for Privacy-Preserving Graph Processing in Geo-Distributed Data Centers

Amelie Chi Zhou
Shenzhen University
chi.zhou@szu.edu.cn

Ruibo Qiu
Shenzhen University
sz.qiu.ruibo@gmail.com

Thomas Lambert
Université de Lorraine, CNRS, Inria, LORIA
thomas.lambert@inria.fr

Tristan Allard
Univ. Rennes, CNRS, IRISA
tristan.allard@irisa.fr

Shadi Ibrahim
Inria, Univ. Rennes, CNRS, IRISA
shadi.ibrahim@inria.fr

Amr El Abbadi
UC Santa Barbara
amr@cs.ucsb.edu

ABSTRACT

Graph processing is a popular computing model for big data analytics. Emerging big data applications are often maintained in multiple geographically distributed (geo-distributed) data centers (DCs) to provide low-latency services to global users. Graph processing in geo-distributed DCs suffers from costly inter-DC data communications. Furthermore, due to increasing privacy concerns, geo-distribution imposes diverse, strict, and often asymmetric privacy regulations that constrain geo-distributed graph processing. Existing graph processing systems fail to address these two challenges. In this paper, we design and implement *PGPregel*, which is an end-to-end system that provides *privacy-preserving* graph processing in geo-distributed DCs with *low latency* and *high utility*. To ensure privacy, PGPregel smartly integrates Differential Privacy into graph processing systems with the help of two core techniques, namely *sampling* and *combiners*, to reduce the amount of inter-DC data transfer while preserving good accuracy of graph processing results. We implement our design in Giraph and evaluate it in real cloud DCs. Results show that PGPregel can preserve the privacy of graph data with low overhead and good accuracy.

ACM Reference Format:
Amelie Chi Zhou, Ruibo Qiu, Thomas Lambert, Tristan Allard, Shadi Ibrahim, and Amr El Abbadi. 2022. PGPregel: An End-to-End System for Privacy-Preserving Graph Processing in Geo-Distributed Data Centers. In *SoCC '22: ACM Symposium on Cloud Computing (SoCC '22), November 7–11, 2022, San Francisco, CA, USA.* ACM, New York, NY, USA, 17 pages. https://doi.org/10.1145/3542929.3563474

SoCC '22, November 7–11, 2022, San Francisco, CA, USA

ACM ISBN 978-1-4503-9414-7/22/11.
https://doi.org/10.1145/3542929.3563474

1 INTRODUCTION

Graph processing is a popular computing model to perform big data analytics for a wide range of applications. With the ever increasing sizes of data, many big data applications need to analyze large-scale graph data generated and maintained globally. For example, the social network graph at Facebook has over one trillion edges [14, 63]. In order to provide reliable and low-latency services to the users, Facebook has built multiple geographically distributed (geo-distributed) data centers (DCs) to maintain and manage those data. To run graph processing algorithms on top of such data, for example to study the importance of Facebook users using recommendation algorithms, coordination (e.g., data exchange) is needed among multiple geo-distributed DCs. Geo-distributed data communication leads to two challenges which make existing graph processing engines inefficient or even invalid.

First, due to the geo-distribution of graph data, data privacy has become an important concern that casts doubts on the utility of graph processing systems given both the high expectations on their performances and the diversity of the legal personal data protection frameworks around the world. Many countries and regions have strict laws and regulations to protect the privacy of personal data. For example, the states of the European Union (EU) enforces the General Data Protection Regulations (GDPR) [18] to protect personal data of individuals living in the EU. According to GDPR, which is considered as the most comprehensive privacy protection regulation to date, it is *permitted and legal* to transfer personal data out of EU only when the target country has "adequate" level of data protection [16]. However, different countries can have different views on the importance of data

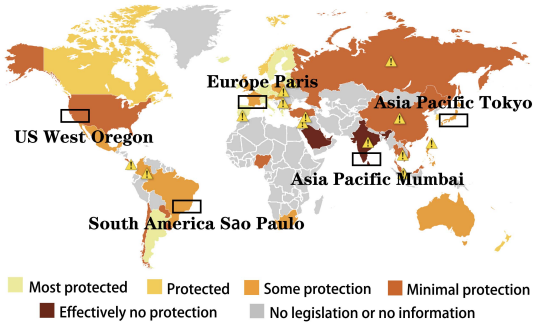


Figure 1 Privacy and data protection restrictions of 54 countries (Forrester's data privacy heatmap released in 2021 [28]).

privacy. Figure 1 shows the different levels of data privacy protection in 54 countries around the world. As a result, it is illegal for international companies to transfer its user data collected in the EU to the US DCs [17] without protection, which casts new challenge to geo-distributed data analytics.

Although the data transferred between DCs during geo-distributed graph processing may not contain identifiable information (e.g., for PageRank it consists of a sequence of rank values), it still needs to be protected in order to thwart the attacks, well-known in the privacy-preserving data publishing literature, that exploit aggregated information [24] (e.g., reconstruction attacks [20], membership inference attacks [36, 60]). Most existing graph processing systems for geo-distributed DCs have overlooked the privacy issue [37, 72]. Although there have been some privacy-preserving systems for applications such as stream processing [56] and machine learning [2], these systems cannot be easily adapted to graph applications which have different computation and communication patterns. On the other hand, although privacy-preserving graph publishing can be used to protect the privacy of individuals [19, 35, 38, 46, 47], there still lacks an end-to-end solution which can preserve graph data privacy during geo-distributed execution without modifying the applications.

Second, geo-distributed graph processing suffers from costly inter-DC data communication due to scarce wide area network (WAN) resources. Our measurements on real cloud DCs show that inter-DC network bandwidths can be lower than one tenth of the intra-DC network bandwidths (see Section 2). On the other hand, geo-distributed graph processing often leads to large amounts of inter-DC data transfers. For example, when running PageRank on the Twitter graph using the real geographical distribution of Twitter users [62], over 70% of the total communication are inter-DC. Thus, it is important to reduce inter-DC data transfer sizes to efficiently perform geo-distributed graph data analytics. More importantly, designing privacy-preserving techniques for geo-distributed graph processing should take into consideration the large amounts of inter-DC data communications to guarantee low latency for the applications.

Motivated by the above, our goal in this paper is to design *PGPregel*, a *privacy-preserving* graph processing system for geo-distributed DCs, hence, a Private Geo-distributed Pregel. Although the large body of protection-through-encryption techniques such as structured encryption [30, 64, 74], secure multi-party computation (MPC) [26, 31], (somewhat) fully homomorphic encryption (FHE) (e.g., [15]), or partially homomorphic encryption (PHE) [57] can be used to preserve data privacy, they usually result in high latency and bandwidth consumption [27], harming dramatically the performances of graph processing algorithms. *Differential Privacy (DP)* [23] protects the privacy of individuals by adding random noise to aggregated data. Satisfying it by perturbing the outgoing inter-DC messages would allow DCs to perform all computations over cleartext (perturbed) data, thus resulting in generic and lightweight graph applications without jeopardizing privacy guarantees. However, naively integrating DP into existing graph processing systems would add too much noise and harm the utility of graph processing results. Thus, the main technical problem addressed by this work is *how to practically integrate* ***Differential Privacy*** *into graph processing systems for* ***good utility*** *and* ***low latency***.

We address this problem using two techniques. First, we employ *sampling* at the vertex level to reduce the number of outgoing messages and thus reducing inter-DC data transfer and hence system latency. Sampling additionally benefits utility through its well-known amplification effect for DP [1, 6, 9, 39, 45, 66, 67]. We rigorously prove that our sampling technique can amplify the privacy budget and as a result improve the utility of our graph processing system. Second, we use *combiners* to combine outgoing messages at the DC level to further reduce inter-DC data transfer. Combiners also benefit DP in two aspects. For one thing, with a smaller number of combined messages, we can assign a larger privacy budget to each message and hence improve the accuracy of graph processing results. For another, aggregated messages (e.g., summed up) reduce the impact of the differentially private perturbation, further increasing the accuracy of results. We identify and make use of the multi-level and asymmetric feature of privacy constraints in geo-distributed DCs to perturb only the messages sent to DCs with lower privacy levels and consequently further improve system utility.

We focus on recommendation algorithms in this paper, which are an important type of graph applications usually involving global data distributions [25, 59]. We evaluate the effectiveness of PGPregel using four recommendation algorithms on real graphs and real user distributions [62]. Results show that PGPregel can preserve the privacy of graph data in

geo-distributed DCs with low overhead and good accuracy. To summarize, we make the following contributions.

- We present an end-to-end system named *PGPregel* specifically tailored for differentially private geo-distributed graph processing. To the best of our knowledge, this is the first study considering system latency, utility and privacy at the same time for geo-distributed graph processing.
- We present sampling and combiners techniques to make DP *practical* in our system. The two techniques can benefit both of our two goals: mitigating the impact of differentially private perturbation on system utility without incurring significant overhead.
- PGPregel is the first system that integrates privacy zones into large-scale graph processing, which is inspired from GDPR. Our extensive experiments clearly demonstrate the effectiveness and efficiency of PGPregel. We have integrated our design into Giraph and open sourced it at: https://github.com/PGPregel/PGPregel.

2 BACKGROUND AND MOTIVATION

2.1 Geo-Distributed Data Centers

We consider several DCs owned by a single entity (e.g., a social network provider) and distributed globally over distinct geographical areas possibly ruled by distinct privacy laws. We consider as a geographical area the areas that share the same legal right-to-privacy framework. Privacy laws can constrain many aspects regarding personal data, such as the collection, storage and processing of the data. In this work, we mainly focus on the aspect regarding data transfer across DCs [16]. *The privacy laws allow data transfers from a DC with lower (i.e., less strict) privacy protection level to DCs with higher protection levels, but not the other way around.*

The entity that owns the multiple DCs can perform global data analytical jobs that require coordination between the DCs. Data communication between geo-distributed DCs usually goes through the WAN, which has low bandwidth and high latency compared to network performance within a single DC. For example, our measurements on both Amazon EC2 and Windows Azure clouds show that the network bandwidth within a single DC can be 7-22x higher than that between geo-distributed DCs.

2.2 Graph Data Analytics

A number of graph processing systems such as Pregel [51] and PowerGraph [32] have been proposed to efficiently perform graph data analytics. Most of these systems follow the "think-as-a-vertex" philosophy and encode graph computation as vertex programs which run in parallel and communicate through edges. Vertices iteratively update their states according to the messages received from neighbors. Thus, efficient communication between vertices and their neighbors is important to performance of graph processing. In this paper, we focus on the Pregel [51] model, which has a clear abstraction based on *message passing* and thus can ease our discussion on data communication optimizations.

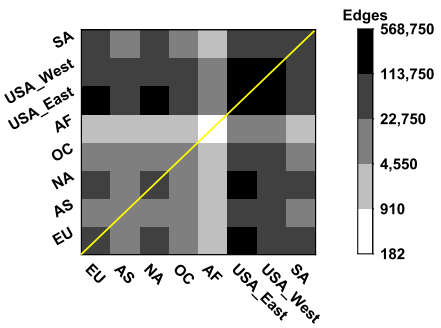


Figure 2 Number of edges between DCs in the Twitter graph, where vertices are located in eight different DCs. SA, AF, OC, NA, AS and EU represent DCs in South America, Africa, Oceania, North America, Asia and Europe, respectively.

With the global distribution of graph data sets, inter-DC data communications play an important role in geo-distributed graph processing. We study the distributions of high degree vertices ($> 10,000$ followers) in the Twitter graph using real geographic locations of users [41]. Specifically, we cluster user locations into eight DCs, including South America, USA West, USA East, Africa, Oceania, North America, Asia and Europe. Figure 2 shows the number of edges between different pairs of DCs, where the diagonal cells represent intra-DC edges and the rest represent inter-DC edges. The number of edges can reflect the amount of data transfer between DCs. We find that over 75% are inter-DC edges. This leads to two technical challenges to geo-distributed graph processing.

Challenge 1: Preserving privacy with good utility. As mentioned above, geo-distributed DCs usually have different rules and regulations on data privacy protection. This casts special constraints to inter-DC data transfer. Violations of data privacy laws and regulations can lead to huge fines [17]. Thus, it is important for future graph processing systems to preserve data privacy in geo-distributed DCs. One straightforward way to achieve this goal is to forbid any inter-DC data communication that violates privacy requirements. However, this can greatly harm the accuracy of graph processing results due to the large amount of inter-DC data communications. Another natural approach could be to encrypt messages before they leave their DCs. For example, (somewhat) fully homomorphic encryption (FHE) schemes [8, 13, 44] support (limited or no) computations over encrypted data. However, this does not work for our large-scale geo-distributed graph processing due to the following reasons. First, FHE is impractically expensive because all

computations including the ones performed within each DC have to execute over FHE data. A typical graph consists of up to over one billion edges and tenths of millions of vertices. As the search time on a small FHE dataset is in the order of hundreds of seconds [15], the overhead of FHE applied to graph processing is huge. Second, the bandwidth cost (aggravated by WAN) is unacceptable under FHE. FHE is known to suffer from large expansion factors (10^6 expansion factor is not rare [27]). Finally, applying FHE is not general enough because each algorithm should be specifically translated to a boolean circuit with a minimal multiplicative depth.

In this paper, we adopt differential privacy (DP) which allows computations over cleartext data while still satisfying sound privacy guarantees. A straightforward way of applying DP is to perform differentially private perturbation on the messages exchanged during graph processing. However, this leads to unacceptable accuracy loss because of the numerous inter-DC data communications. The challenge thus translates into making graph applications satisfy differential privacy without thwarting the quality of their results. Compared to encryption-based methods, the overhead of our DP-based approach is negligible, leading to generic and lightweight distributed algorithms, at the cost of an approximation in results.

Challenge 2: Improving graph processing performance. Due to the limited inter-DC network bandwidth, the large amount of inter-DC data communications in geo-distributed graph processing have become the major performance bottleneck. To optimize system performance, one straightforward idea is to avoid inter-DC data communications as much as possible, which however can impact the accuracy of graph processing results and thus contradicts the utility optimization goal. Some studies perform message rerouting to make usage of high bandwidth links in geo-distributed DCs to improve system performance [77]. However, this may not work when considering privacy constraints. Due to the complicated and irregular data communication patterns in graph processing, it is non-trivial to improve graph processing performance considering both privacy and utility.

3 PROBLEM OVERVIEW

In this work, we consider the problem of executing targeted applications over a large graph partitioned on multiple DCs located in distinct geographical areas. The goal is to satisfy data privacy requirements while keeping low latency and high utility. In the following, we formally define the problem.

3.1 Data Model

Consider processing a dataset partitioned across multiple geo-distributed DCs and the full dataset is a single directed and unweighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is a set of vertices and $\mathcal{E}$ is a set of edges. The application partitions the graph into *k graph partitions* $\mathcal{G}_i = (\mathcal{V}_i, \mathcal{E}_i)$, where $\mathcal{V}_i \subset \mathcal{V}$, $\mathcal{E}_i \subset \mathcal{E}$, $\mathcal{V}_i \cap \mathcal{V}_j = \emptyset$ and $\mathcal{E}_i \cap \mathcal{E}_j = \emptyset$ if $i \neq j$. Inter-DC edges connect two nodes belonging to different partitions. We consider in that case that a (directed) inter-DC edge belongs to the set of edges of the partition of the starting node: $(u, v) \in \mathcal{E}_i$ if $u \in \mathcal{V}_i$. For simplicity and without loss of generality, we assume in the following that there is a single DC per geographical area and we consider that each DC holds a single partition of $\mathcal{G}$.

3.2 Targeted Graph Applications

In this work, we target four graph algorithms that are widely adopted in recommendation systems, including PageRank [55], HITS [40], SALSA [42] and ALS [73].

PageRank was originally proposed by Google to measure the importance of different webpages according to the link relationship between the pages. At each graph processing iteration, each vertex updates its importance (i.e., rank value) using the rank values of its neighbors from the last iteration.

HITS is another algorithm used to rank webpages for a topic search. Given a topic search, the set of highly relevant webpages are called *Authorities*. Webpages not very relevant but pointing to many related authorities are called *Hubs*. The algorithm strives to obtain authority and hub scores for each vertex. In each iteration, the two scores are updated in a mutually recursive manner. We normalize the scores to $[1, 2]$ instead of $[0, 1]$ as in the original algorithm [40] to achieve faster convergence while keeping the same ranking.

SALSA has been shown to be one of the most effective link analysis algorithms [54] and has been applied in social network recommendations [34]. Similar to HITS, SALSA divides webpages into hubs and authorities and constructs a bipartite graph by putting hubs on one side and authorities on the other. It has two main differences from HITS. First, the authority/hub score of each vertex in SALSA is determined by the authority/hub scores of other vertices, while HITS uses "mutual enforcement" to update those values. Second, SALSA updates the scores similar to PageRank, by performing two independent random walks (i.e., a hub walk and an authority walk) on the neighborhood graph.

ALS is a popular matrix factorization algorithm that tries to make item recommendations to users given sparse user-item ratings. Essentially, ALS decomposes a rating matrix into the product of two lower dimensional matrices to capture the potential factors of users and items. The two matrices are alternatively updated in multiple iterations.

For simplicity, we overload the terminology when the semantics is clear from the context: PageRank may refer to the complete PageRank algorithm or the function applied at each vertex, and the same holds for HITS, SALSA and ALS.

3.3 Quality Measures

Low latency and high utility are our two main goals and we measure the optimization quality of the two goals as follows.

Latency measure. As inter-DC data communication is the main performance bottleneck in geo-distributed graph processing, we use the WAN usage consumed during graph processing to measure system latency. To obtain this value, we simply aggregate the size of every message transmitted between different DCs. The size of a message is determined by the data contained in the message and can be estimated using TCP data packets.

Utility measure. We quantify the impact of our techniques on the utility of the targeted graph applications based on a general error-quantification measure, i.e., the average relative error (denoted *ARE*), and set-based measures specifically related to recommendation systems, i.e., the precision and recall of top-k values retrieved. Note that for a top-k list, the number of false positives is equal to the number of false negatives, and consequently the precision and the recall are equal. As a result we only measure the precision below.

We measure *ARE* by comparing the graph processing results of our system with those from Pregel (denoted as baseline). That is, $ARE = \frac{\sum_{v \in \mathcal{V}} |(m_v - b_v)/b_v|}{|\mathcal{V}|}$, where b_v is the baseline value of vertex v and m_v is the result of vertex v obtained using our system. For precision, we sort the baseline results and the results of our system separately, and compute the true positives (denoted *TP*) and the false positives (denoted *FP*). Precision is then defined as $P = \frac{TP}{TP+FP}$.

3.4 Privacy

PGPregel adapts specifically to the existing diversity of the privacy protection levels that must be enforced by geo-distributed DCs. In a nutshell, it protects the structural information of the sub-graph of any DC that communicates to a less trusted DC. PGPregel satisfies *ϵ-edge differential privacy* [35], a variant of differential privacy [21] dedicated to graph data, by allowing each DC to inject noise into intermediate results before they are transmitted. Loosely speaking, ϵ-edge differential privacy guarantees that no single edge, e.g., social link between two individuals, significantly impacts the protected information communicated by a DC to a less trusted DC.

3.4.1 Threat model. Despite belonging to a single organization, the DCs are physically located in distinct geographical areas. This results in various and possibly asymmetric privacy requirements across DCs (see Figure 1). We capture this diversity using discretized and totally ordered privacy levels[1]. When the privacy level of DC_i is lower than (or equal to) that of DC_j, it represents the real-life situation where the geographical area of DC_j has privacy requirements compatible with those required by the area of DC_i. DC_j is thus considered by DC_i to be *fully trusted*. Otherwise, when the privacy level of DC_i is strictly higher than that of DC_j, then DC_j is considered by DC_i to be *untrusted*. More particularly, DC_i considers DC_j to be *honest-but-curious* in that it does not deviate from the algorithm but it may try to infer any information that can be inferred for jeopardizing the privacy guarantees (see Section 3.4.3). Note that since privacy levels are totally ordered, the trust relationship is transitive.

3.4.2 Information Protected. PGPregel aims to protect structural information, i.e., vertices and edges, of local, intra-DC, subgraphs. We make the following two usual security assumptions. First, the edges that connect two subgraphs stored at two distinct DCs, i.e., inter-DC edges, are already known to the two connected DCs. We focus thus on the local, intra-DC, structural information. Second, each DC protects its subgraph with appropriate security measures (e.g., access control, encryption at rest). We focus thus on the information communicated from one DC to another during the execution of graph applications. This information consists of the intermediate results of the graph application being executed. Since this depends directly on the structural information of the local subgraph, it must be protected accordingly in order to avoid adversarial inferences and resulting privacy breaches.

3.4.3 Edge Differential Privacy. To preserve data privacy, we adopt the well-known ϵ-edge-DP model [35] which essentially aims at hiding the presence/absence of any single edge in a graph (e.g., a social relationship in the context of social networks). This model protects against attacks that aim at inferring the existence of an edge in the sub-graph of the communicating DC by exploiting the information communicated across DCs (e.g., membership inference attacks [60]). These are the usual protection guarantees of pure differential privacy applied to the edges of the graph. We consider that two graphs $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$ are *neighbors* if $\mathcal{G}$ is $\mathcal{G}'$ with one edge difference, i.e., $\mathcal{V} = \mathcal{V}'$ and $|\mathcal{E} - \mathcal{E}'| = 1$.

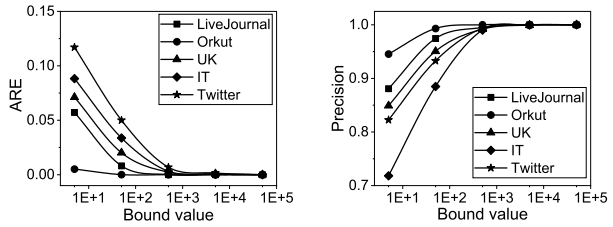
Definition 1 (ϵ-edge-DP [35]). *Let $\mathcal{G}_1$ and $\mathcal{G}_2$ be two neighboring graphs. Let* `f` *be a randomized function and* `Im(f)` *be its image (all possible outputs of* `f`*). If for all $\mathcal{S} \subset$* `Im(f)`*,*

$$Pr[\texttt{f}(\mathcal{G}_1) \in \mathcal{S}] \leq e^{\epsilon} Pr[\texttt{f}(\mathcal{G}_2) \in \mathcal{S}]$$

then `f` *is said to satisfy ϵ-edge-DP. ϵ is called privacy budget. The smaller the budget, the better the privacy.*

In this paper, `f` $\in \{PageRank, HITS, SALSA, ALS\}$. In order to satisfy ϵ-edge-DP, it is possible to perturb their output based on the Laplace mechanism [22] parameterized with ϵ and their respective *sensitivities*. The sensitivity quantifies the maximum impact of the presence/absence of any single edge on the result of the function. We calculate below the

[1] PGPregel can be extended with more general trust models (e.g., a matrix specifying the trust relationship between each pair of DCs).



(a) ARE of bounded PageRank **(b) P of bounded PageRank**

Figure 3 Impact of upper bound on the accuracy of PageRank.

global sensitivities of the targeted applications. Denote $\mathcal{G}_1$ and $\mathcal{G}_2$ as two neighboring graph partitions.

- **PageRank.** For PageRank, the largest possible difference of the rank value between any pair of neighboring graph partitions is upper-bounded by the difference between the maximum and the minimum rank values, namely max $\| PageRank(\mathcal{G}_1) - PageRank(\mathcal{G}_2) \|_1 < (pr_{max} - pr_{min})$, where pr_{max} (resp. pr_{min}) is the maximum (resp. minimum) rank value. In theory, the maximum rank value is unbounded, leading to an infinite global sensitivity. However in practice, a reasonable maximum upper bound can be set [12, 48, 69]. We experimentally show in Figure 3 using Pregel and five real-world graphs that choosing reasonable upper bounds leads to a bounded sensitivity without suffering from a dramatic utility loss (detailed experimental setup can be found in the evaluation section).
- **HITS.** Since the results of HITS are normalized, the largest possible difference between any pair of neighboring graph partitions is lower-bounded by 1 and upper-bounded by 2, namely max $\|HITS(\mathcal{G}_1) - HITS(\mathcal{G}_2)\|_1 \leq 1$.
- **SALSA.** Similar to HITS, the results of SALSA are naturally bounded between 0 and 1. Thus, we have max $\|SALSA(\mathcal{G}_1) - SALSA(\mathcal{G}_2)\|_1 \leq 1$.
- **ALS.** The results of ALS contain one matrix of users and one matrix of items. We adopt the unbounded sensitivity analyzed in existing study [29] for the two sets of results.

Given a privacy budget ϵ_i and the global sensitivity max $\| \mathtt{f}(\mathcal{G}_1) - \mathtt{f}(\mathcal{G}_2) \|_1$, satisfying ϵ_i-edge-DP requires perturbing the output of the function by adding random noise sampled from the Laplace distribution $\mathtt{Lap}(\max \|\mathtt{f}(\mathcal{G}_1) - \mathtt{f}(\mathcal{G}_2)\|_1 / \epsilon_i)$ to the results. Finally, the self-composition properties of ϵ-DP [53] result in a consumption of the privacy budget that is 1) linearly increasing with the number of perturbed outputs when the function takes as inputs overlapping graphs (i.e., sequential composition) or 2) set to the maximum privacy budget used when the function takes as inputs disjoint graphs (i.e., parallel composition).

4 DESIGN DETAILS OF PGPREGEL

PGPregel is designed to achieve differentially private graph processing in geo-distributed DCs with low latency and high utility. To achieve this goal, we propose two optimizations including *sampling* and *combiners* to improve the accuracy of differentially-private graph computations while reducing the inter-DC data communication size. Figure 4 gives an overview of our PGPregel model. Although most of our discussions focus on Pregel, the design ideas are general and can be useful for other graph execution models [32].

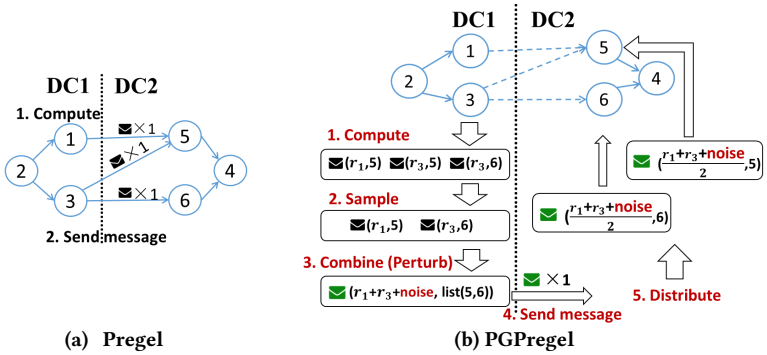
4.1 Privacy Budget Allocation

To preserve DP, a privacy budget ϵ is given to the entire graph application. The budget is further allocated to each inter-DC message for differentially private perturbation. Recall that, the larger the budget, the lower the noise. According to the composition property of DP, we first distribute ϵ evenly among multiple graph processing iterations. Second, we evenly distribute the budget of iteration i (denoted ϵ_i) to each DC. Finally, we further distribute ϵ_i to each outgoing message that needs protection. For PageRank, HITS and SALSA, the sequential composition property is satisfied among the messages and thus we evenly distribute ϵ_i among them. For ALS where parallel composition is satisfied among messages, the budget allocated to each of them equals to ϵ_i.

4.2 Sampling Technique

In large-scale graph processing, it is not always necessary to transfer every message between vertices to obtain good accuracy. For example, in PageRank, the rank value of a vertex is updated using the rank of its neighbors. Thus, dropping a few messages from low-degree neighbors does not significantly reduce the accuracy. Our experiments using PageRank and LiveJournal graph on five DCs (detailed setup in Section 6) show that, using random sampling to reduce the number of inter-DC messages can greatly reduce the WAN usage with good accuracy. For example, Figure 5 shows that reducing the sampling rate from 1.0 to 0.5 results in precision losses close to 30% and a WAN usage gain close to 50%, under the setup of the figure. Further, as we will prove later, sampling the edges amplifies the privacy budget and leads to satisfying ϵ-edge-DP with less perturbation. Thus, it can benefit both our low-latency and high-utility goals.

Given a graph application, we sample its inter-DC data communications with a sampling probability p. Users can empirically decide or smartly learn the best sampling probability according to the trade-off between graph processing accuracy and WAN usage. Although sampling only inter-DC messages may introduce skewness to graph processing results, Figure 2 shows that inter-DC messages are dominating in real geo-distributed graph applications. As a result, sampling only inter-DC messages does not affect much the accuracy. What's more, the sampling technique can help improve the accuracy of graph processing with its *amplification effect* on the privacy budget.



(a) Pregel (b) PGPregel

Figure 4 System overview. A message contains the content and the list of receivers. For example, (r_1,5) means the message is sent to vertex 5 with a value of r_1.

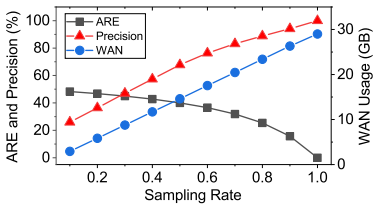


Figure 5 Impact of sampling on the quality of graph processing results using PageRank on LiveJournal graph.

The amplification effect of sampling on DP was first sketched and proved for the case when the privacy budget is 1 [61]. To study the amplification effect in more general cases, we prove Lemma 1, which expands the privacy budget ϵ from 1 to an arbitrary value. In the following, A is a random algorithm which takes graphs as input.

Definition 2 (Algorithm A_p). *Let $p \in]0,1[$ and A be a random function that satisfies ϵ-DP and $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph. We define $\mathcal{E}_\mathcal{T}$ as a subset of $\mathcal{E}$ obtained by sampling each element of $\mathcal{E}$ independently with probability p (for all $x \in \mathcal{E}$, $Pr(x \in \mathcal{E}_\mathcal{T}) = p$)). If $\mathcal{G}_\mathcal{T} = (\mathcal{V}, \mathcal{E}_\mathcal{T})$ then we define $A_p(\mathcal{G}) = A(\mathcal{G}_\mathcal{T})$.*

Lemma 1 (Amplification via sampling). *If A satisfies ϵ-DP, then for any $p \in]0,1[$, A_p satisfies $p(e^\epsilon - 1)$-DP.*

Proof. In the following, let us denote as δ the function that samples any subset with independent sampling probability p for any element of the input set (using the notation from Definition 2, $\mathcal{E}_\mathcal{T} = \delta(\mathcal{E})$). For a given graph $\mathcal{G}$, we denote as $\mathcal{G}_\delta$ its subgraph after the sampling of its edge set $A_p(\mathcal{G}) = A(\mathcal{G}_\delta)$). Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$ be two adjacent graphs. We assume here that $\mathcal{E} = \mathcal{E}' \cup \{x\}$.

Let now $\mathcal{E}'_\mathcal{T}$ be any subset of $\mathcal{E}'$ ($\mathcal{E}'_\mathcal{T}$ is a possible output of $\delta(\mathcal{E}')$). Let us now denote $\mathcal{E}_\mathcal{T} = \mathcal{E}'_\mathcal{T} \cup \{x\}$. Because the sampling of each element is independent, we have $Pr(\delta(\mathcal{E}) = \mathcal{E}_\mathcal{T}) = p \cdot Pr(\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T})$ and $Pr(\delta(\mathcal{E}) = \mathcal{E}'_\mathcal{T}) = (1-p) \cdot Pr(\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T})$. Thus,

$$\begin{aligned} Pr(\delta(\mathcal{E}) = \delta(\mathcal{E}')|\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T}) &= \frac{Pr(\delta(\mathcal{E}) = \mathcal{E}'_\mathcal{T} \& \delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T})}{Pr(\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T})} \\ &= \frac{(1-p) \cdot Pr(\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T})^2}{Pr(\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T})} \\ &= (1-p) \cdot Pr(\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T}). \end{aligned}$$

Similarly, $Pr(\delta(\mathcal{E}) \neq \delta(\mathcal{E}')|\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T}) = p \cdot Pr(\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T})$. Note that when the sampling event "$\delta(\mathcal{E}) \neq \delta(\mathcal{E}')$" occurs, under the assumption that $\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T}$, we know that the two induced subgraphs $\mathcal{G}_\delta$ and $\mathcal{G}'_\delta$ are adjacent (because the only possible difference is the edge x being part of $\delta(\mathcal{E})$).

Let us now consider any $S \subset Im(A)$. By definition, $A_p(\mathcal{G}) \in S$ if and only if $A(\mathcal{G}_\delta) \in S$. The same applies for $A_p(\mathcal{G}')$ and $A(\mathcal{G}'_\delta)$. Thus, $Pr(A_p(\mathcal{G}) \in S|\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T}) = Pr(A(\mathcal{G}_\delta) \in S)|\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T})$.

As mentioned above, if $\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T}$, then either $\mathcal{G}_\delta = \mathcal{G}'_\delta$ (with probability $(1-p)$), or $\mathcal{G}_\delta$ and $\mathcal{G}'_\delta$ are adjacent (with probability p). Thus,

$$\begin{aligned} &Pr(A(\mathcal{G}_\delta) \in S|\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T}) \\ &\quad = (1-p) \cdot Pr(A(\mathcal{G}_\delta) \in S|\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T}|\mathcal{G}_\delta = \mathcal{G}'_\delta) \\ &\quad\quad + p \cdot Pr(A(\mathcal{G}_\delta) \in S|\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T}|\mathcal{G}_\delta \neq \mathcal{G}'_\delta) \\ &\quad \leq (1-p) \cdot Pr(A(\mathcal{G}'_\delta) \in S)|\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T}) \\ &\quad\quad + pe^\epsilon \cdot Pr(A(\mathcal{G}'_\delta) \in S)|\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T}) \\ &\quad \leq (1 + p(e^\epsilon - 1)) \cdot Pr(A(\mathcal{G}'_\delta) \in S)|\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T}) \\ &\quad \leq e^{p(e^\epsilon - 1)} \cdot Pr(A(\mathcal{G}'_\delta) \in S)|\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T}). \end{aligned}$$

as $1 + t \leq e^t$ for any t.

Thus we have $Pr(A_p(\mathcal{G}) \in S|\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T}) \leq e^{p(e^\epsilon - 1)} \cdot Pr(A_p(\mathcal{G}') \in S|\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T})$. From it we can conclude

$$\begin{aligned} Pr(A_p(\mathcal{G}) \in S) &= \sum_{\mathcal{E}'_\mathcal{T} \subset \mathcal{E}'} Pr(A_p(\mathcal{G}) \in S \bigcap \delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T}) \\ &= \sum_{\mathcal{E}'_\mathcal{T} \subset \mathcal{E}'} Pr(A_p(\mathcal{G}) \in S|\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T}) \times Pr(\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T}) \\ &\leq \sum_{\mathcal{E}'_\mathcal{T} \subset \mathcal{E}'} e^{p(e^\epsilon - 1)} \cdot Pr(A_p(\mathcal{G}') \in S|\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T}) \\ &\quad \times Pr(\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T}) \\ &\leq e^{p(e^\epsilon - 1)} \cdot \sum_{\mathcal{E}'_\mathcal{T} \subset \mathcal{E}'} Pr(A_p(\mathcal{G}') \in S|\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T}) \\ &\quad \times Pr(\delta(\mathcal{E}') = \mathcal{E}'_\mathcal{T}) \\ &\leq e^{p(e^\epsilon - 1)} \cdot Pr(A_p(\mathcal{G}') \in S) \end{aligned}$$

which proves that A_p satisfies $p(e^\epsilon - 1)$-DP when $\mathcal{E} = \mathcal{E}' \cup \{x\}$. In the other case, the same proof applies, as the probability of $\mathcal{G}_\delta$ and $\mathcal{G}_{\delta'}$ being equals or adjacent are unchanged (but this time under the assumption $\delta(\mathcal{E}) = \mathcal{E}_T$). □

We prove Lemma 2, which shows that the amplification effect of the sampling technique allows us to use a larger budget to generate noises while preserving the same level of differential privacy. The proof is straightforward.

Lemma 2. *For any $k > 0$, if A satisfies $\ln(k+1)$-DP and $\epsilon \in]0, k[$, then $A_{\epsilon/k}$ satisfies ϵ-DP.*

This is to say, if we sample the inter-DC messages with probability $p = \epsilon/k$, the original ϵ-DP will still be satisfied even if we use a bigger privacy budget, $\ln(k+1)$, for the sampled messages. Using sampling together with DP results in adverse impacts on quality: reducing the sampling rate both decreases quality (see, e.g., Figure 5) and increases it (lower noise magnitude – see Lemma 2). As a result, the quality of a given algorithm does not necessarily vary uniformly with the decrease of the sampling rate. We use empirical approaches to learn the best sampling rate for different applications (see Section 6.6). Our guideline is that most applications can benefit from sampling. Even when sampling is disabled, PGPregel still outperforms existing solutions.

4.3 Combiners

Similar to Pregel, we adopt combiners at each DC to combine outgoing messages. The difference is that, combiners in PGPregel are defined for pairs of DCs that are communicating. For example, if DC A sends messages to two other DCs B and C, then we deploy two combiners at DC A, each responsible for combining inter-DC messages sent to DCs B and C separately. The function of each combiner can be defined by users according to the computation of graph algorithms. For example, as shown in Figure 4b, the combine operation for PageRank is to 1) sum up the rank values of all combined messages and 2) combine the receivers of all messages into a list. On receiving a combined message, the target DC distributes the message to each receiver in the list. The distribute operation is also application-dependent. For example, for PageRank, the combined message is evenly distributed to each receiver.

The main motivation of introducing combiners in PGPregel is to enlarge the privacy budget allocated to inter-DC messages and hence improve system utility. We define *perturbed combiners* which only combine messages that need privacy protection. According to Section 4.1, we set one perturbed combiner for each pair of communicating DCs to have the best benefit from the available privacy budget.

We can also combine inter-DC messages that do not need protection using *non-perturbed combiners* to reduce WAN usage and hence system latency. However, doing so comes with the cost of accuracy loss. To minimize the accuracy loss introduced by combined data distribution, messages with close values are assigned to the same combiner. We study the impact of combine operation in Section 6.8, which shows that setting the number of combiners to a relatively small number can achieve both good accuracy and low latency. Thus, by default, we set the number of non-perturbed combiners to one between each pair of communicating DCs.

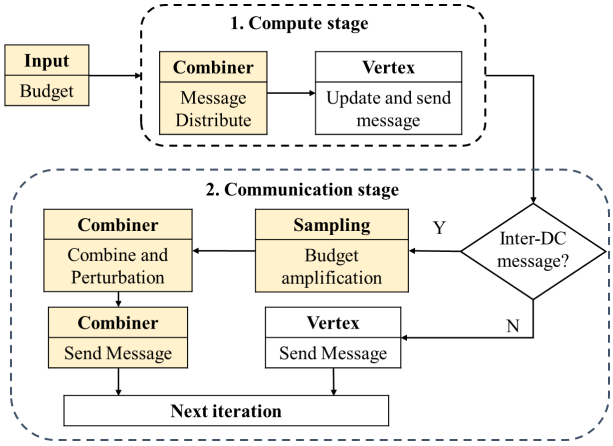


Figure 6 Execution flow of PGPregel in one iteration. Shaded modules are added/modified for PGPregel.

5 IMPLEMENTATION

We integrate PGPregel into Apache Giraph [4], an open-sourced implementation of the Pregel model. Giraph adopts the Hadoop execution engine to run distributed graph computations using multiple workers. Each worker is assigned with one partition of the graph and runs two stages in each iteration, namely the compute stage and communication stage. Since Giraph is not privacy-aware, we made three changes when implementing PGPregel: we 1) implemented DP related parameters and functions; 2) overwrote the combiner module; and 3) implemented a sampling module. Figure 6 shows the execution flow of a PGPregel worker in one iteration. In our experiments, we deploy one worker for each DC. Users can also deploy multiple workers in one DC for better parallelism, in which case a leader has to be chosen for each DC to run the combiner and sampling modules. In the following, we introduce details of the three changes.

DP parameters and functions. PGPregel takes a privacy *budget* as input, which is then allocated to each iteration and each worker using a *budgetAlloc* function before application runs (see Section 4.1). A *noiseGenerator* is implemented to sample noises from a Laplace distribution parameterized by the allocated budget and global sensitivities of applications (see Section 3.4.3).

Table 1 Social and web graphs for PageRank, HITS and SALSA

Real Graph	Vertices	Edges	α_{in}	α_{out}
LiveJournal (LJ)	4,847,571	68,993,773	2.26	2.51
Orkut (OT)	3,072,441	117,185,083	2.83	1.93
uk-2005 (UK)	39,454,746	936,364,282	1.65	2.98
it-2004 (IT)	41,290,682	1,150,725,436	1.58	2.82
Twitter (TW)	41,652,230	1,468,365,182	1.80	2.02

Sampling module. In the compute stage, vertices update their values and generate messages to be sent out to their neighbors. If a message is an inter-DC message, it first goes through the *sampling* module which decides whether the message will be passed on or not. We also implement a *budgetAmplification* function to calculate the amplified privacy budget according to the sampling rate (see Section 4.3). The amplified budget and sampled messages are then passed to the combiner module for combine and perturbation.

Combiner module. We overwrote the *combiner* module of Giraph for each worker to combine perturbed and non-perturbed inter-DC messages separately. We redesigned the *message* data structure for a combined message, which includes a list of recipients in the target DC. A *distribute* function is implemented for the worker in the target DC to distribute the received message onto each recipient (i.e., a vertex) in the list (see Section 4.3).

The DP, sampling and combiner modules are treated as building blocks to PGPregel. When adopted for different graph applications, these modules can be adaptively modified accordingly. For example, for graph algorithms sending scalar values, the DP model is the standard model and when graph algorithms send set-valued data, we adopt DP models specifically designed for this data type [10]. To simplify our discussion, in this paper, we focus on graph algorithms sending scalar-valued data, including a single scalar value (e.g., PageRank) or a vector of scalar values (e.g., ALS). The sampling probability and number of combiners are also application-dependent. To improve the usability of PGPregel, it is our future work to design automated parameter tuning for different modules.

6 EVALUATION

We perform an in-depth evaluation of the performance of PGPregel and analyse the quality of the results given various graph applications and real-life large-scale graph data, under various privacy settings. Our experiments focus on the impact of integrating differential privacy on the execution of graph applications, and the main factors influencing both quality and performance. We run experiments on a physical cluster and, to enable flexible evaluations, we implemented a simulator (in about 3000 lines Of code).

Table 2 User-Movie ratings for ALS

Real Ratings	Users	Movies	Ratings
MovieLens-1M	6,040	3,706	1,000,209
MovieLens-10M	71,567	10,681	10,000,054

6.1 Dataset

The four graph applications that we target (see Section 3.2) can be classified into two types. First, PageRank, HITS and SALSA are centrality algorithms which rely on random walk to rank the relative importance of a node in a graph. Thus we adopt large real-world social and web graphs for their evaluations as shown in Table 1. Second, ALS is a collaborative filtering algorithm that usually works on user-item datasets. Thus we adopt the widely used MovieLens dataset for its evaluation as shown in Table 2. We distribute the graphs on geo-distributed DCs based on the real geographical distribution of Twitter users [62]. Specifically, we first select five leading countries that have the largest numbers of Twitter users, including United States, Japan, India, Brazil and France. We then build a geo-distributed platform by selecting one Amazon EC2 cloud region in each of the five countries, including US West Oregon (USW), Asia Pacific Tokyo (TKY), Asia Pacific Mumbai (MUB), South America Sao Paulo (SPA) and Europe Paris (EUR). According to the proportions of Twitter users located in the five countries, we distribute 43%, 31%, 11%, 10%, and 5% of graph vertices to the USW, TKY, MUB, SPA and EUR DCs, respectively, for all graphs. Vertices are randomly mapped to different DCs. The privacy level of each region is derived from Figure 1, which are 2, 3, 1, 3 and 3, respectively, for USW, TKY, MUB, SPA and EUR. We adopt real network bandwidths measured between real cloud regions to evaluate graph processing performance.

6.2 Comparison Alternative Solutions

We compare PGPregel to the following solutions, where the first four are compared for all applications and the last one is a DP-based comparison specialized for ALS. To the best of our knowledge, no DP-based solutions have been proposed for the other three applications.

- **Pregel** [51] is the baseline comparison which does not consider data privacy issue. We use it as a baseline for accuracy and WAN usage measurements.
- **Pregel-DP** is Pregel integrated with standard DP techniques to ensure privacy. Specifically, Pregel-DP adopts the same budget allocation method as PGPregel and it perturbs every cross-DC message. As we are the first to preserve DP in geo-distributed graph processing, we implement Pregel-DP as the state-of-the-art privacy-preserving graph engine to show the effectiveness of PGPregel.
- **Monarch** [37] is designed for geo-distributed graph processing which uses local computation and global communication to reduce WAN usage and improve system efficiency.

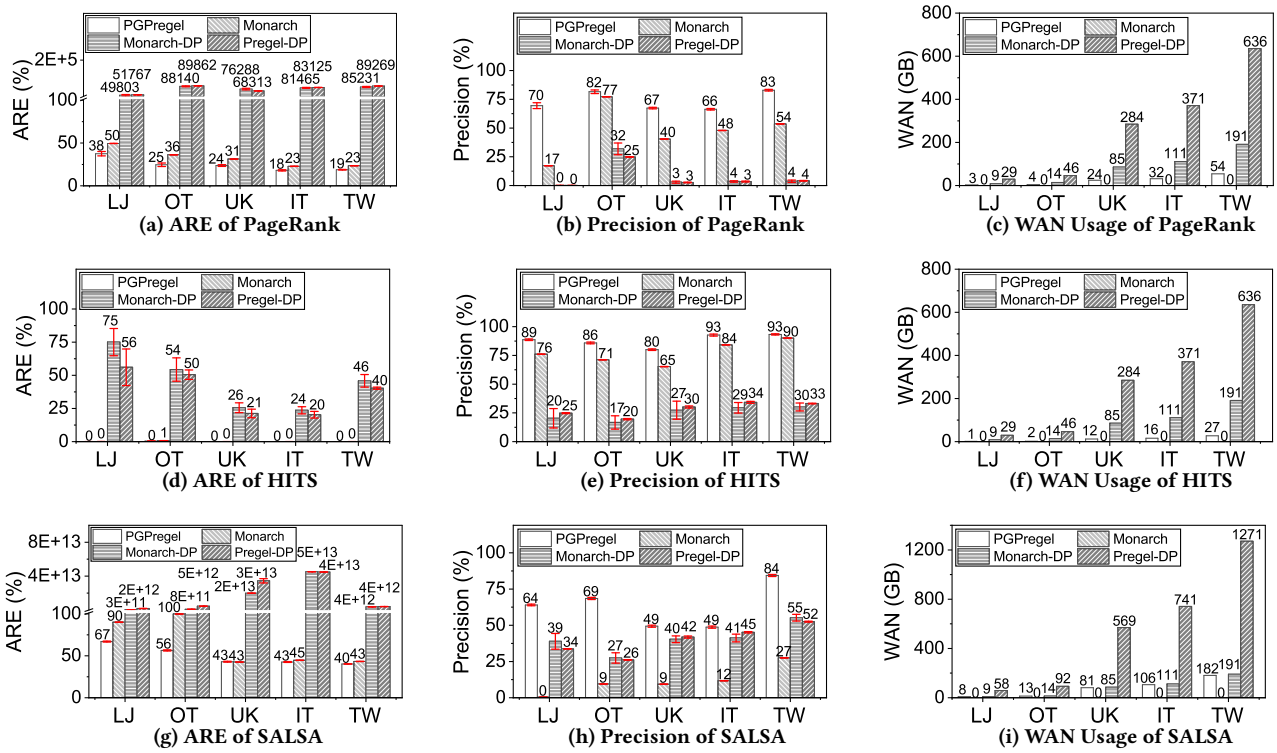


Figure 7 End-to-end evaluation results of compared systems on five real-world graphs and three applications. Error bars show the standard deviation of accuracy results.

Monarch does not consider privacy issue and we use only local computations (i.e., zero inter-DC data transfer) in the implementation to guarantee data privacy.

- **Monarch-DP** [37] implements Monarch with global communication and adopts standard DP to preserve privacy in the same way as Pregel-DP does. We implement Monarch-DP using the Pregel graph computation model instead of GAS model for fair comparison in this paper.
- **PALS** [29] is a DP-based solution specifically designed for ALS and we adopt it as the state-of-the-art comparison to show the effectiveness of PGPregel on preserving DP for graph applications while achieving good accuracy. Root Mean Square Error (RMSE) is often used to measure the prediction accuracy of ALS algorithms, thus we adopt RMSE as the utility measure for ALS application.

For all comparisons, we run the applications with the same fixed number of iterations. We set the number to 20 for PageRank and 10 for the other applications, according to their convergence speed when running on Pregel.

6.3 End-to-End Evaluations

We run a physical cluster of six nodes to emulate the geo-distributed environment, where one node acts as the coordinator of Giraph and the other five are workers each representing a cloud DC. During graph processing, messages are transferred across the five workers and we control the network bandwidths between nodes using network traces measured from Amazon EC2. Below we present the end-to-end performance (i.e., WAN usage) and accuracy (i.e., ARE, precision, RMSE) results of compared methods for the two types of applications separately. The two types of applications differ in datasets and communication/computation features. Thus we believe the results can demonstrate good generality of PGPregel for different recommendation-based applications.

6.3.1 Overall Results of PageRank, HITS and SALSA. Figure 7 shows how PGPregel compares with Pregel-DP, Monarch and Monarch-DP with respect to the quality measures on the five real-world graphs for PageRank, HITS and SALSA. We set the sampling rate to 30%, 60% and 100% for HITS, PageRank and SALSA, respectively. The privacy budget is 1 and the top-k size is 2%. We have the following observations.

First, looking at the accuracy, PGPregel obtains the lowest average relative error (ARE) and the highest precision for all applications on all graphs. The ARE of PGPregel is 99%-100%, 1%-74% and 99%-100% lower than Pregel-DP, Monarch and Monarch-DP, respectively, and the precision of PGPregel is 8%-1225x, 4%-154x and 18%-1345x higher than Pregel-DP, Monarch and Monarch-DP, respectively. Taking a closer look

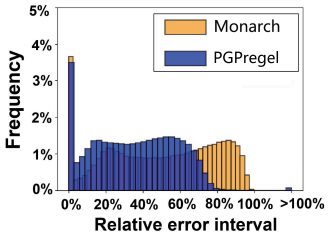


Figure 8 Relative error distribution of a vertex in LiveJournal using PageRank algorithm.

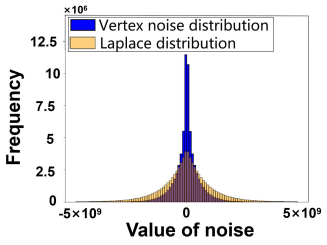
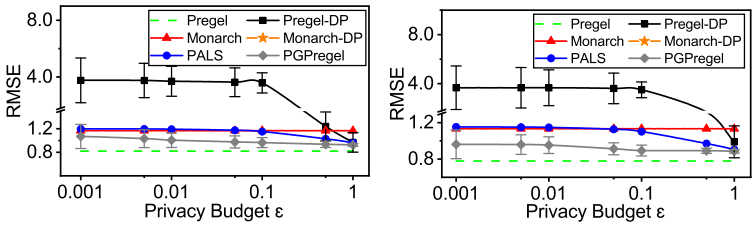
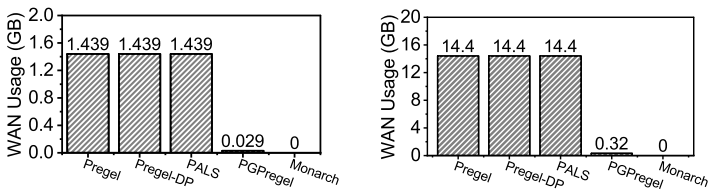


Figure 9 Vertex noise distribution in the first iteration of SALSA on LiveJournal.



(a) MovieLens-1M **(b) MovieLens-10M**

Figure 10 End-to-end accuracy results of compared systems for ALS. Error bars represent standard deviation.



(a) MovieLens-1M **(b) MovieLens-10M**

Figure 11 End-to-end WAN usage results of ALS application.

at the results, Figure 8 shows the relative error distributions of a vertex in the LiveJournal graph when running PageRank using Monarch and PGPregel. The results demonstrate that our proposed techniques are effective in improving the accuracy of geo-distributed graph processing while satisfying differential privacy.

Second, Pregel-DP and Monarch-DP perform the worst among the comparisons, with extremely large error and low precision in most cases. This is mainly due to the large amount of noises added to inter-DC data communications during graph processing. The precision of Pregel-DP and Monarch-DP are relatively high for SALSA, which is due to the fact that an inter-DC message in SALSA requires adding twice the Laplace noise before being used to update the value of a target vertex. That is, each update in SALSA needs two random walks - one forward and one backward. For example, to update the hub/authority score on the left-hand side of the bipartite graph, the random walk will always end up back on the left-hand side. This results in a noise cancellation effect which helped to improve the accuracy of Pregel-DP and Monarch-DP. To demonstrate this effect, we show in Figure 9 the distribution of vertex noise in the first iteration of SALSA using LiveJournal graph. The noise added to each inter-DC message is sampled from the Laplace distribution shown in the figure. Clearly, the noise added to the vertices in one SALSA iteration can lead to smaller error compared to the noise sampled directly from the Laplace distribution.

Third, by avoiding all inter-DC communications, Monarch can reach good accuracy for some applications such as HITS. Monarch achieves very low ARE and high precision on HITS, which are 0-1% and 65%–90%, respectively. The reason that ARE is very low is that we normalize the hub and authority scores of a vertex to [1,2) by adding 1 to a small value in [0,1) calculated using all messages received by the vertex. Thus, the impact of missing messages on the ARE of the scores becomes less significant. However, Monarch performs poorly on the other two applications, especially on SALSA. This is again due to the fact that one vertex update in SALSA requires two message passing to get the desired data, which is more vulnerable to message dropping as in Monarch.

When looking at the WAN usage, PGPregel achieves lower WAN usage than Pregel-DP at all times. Specifically, PGPregel reduces the WAN usage by 86%-96% and 52%-86% compared to Pregel-DP and Monarch-DP, respectively. Although Monarch-DP has much lower WAN usage compared to Pregel-DP thanks to its local computation design, it still consumes much more WAN usage (thus worse performance) compared to PGPregel. Although Monarch leads to zero inter-DC communication, it has poor generality in obtaining good graph processing accuracy. In contrast, PGPregel is able to obtain good accuracy on all three applications with very low inter-DC data communication overhead.

Lastly, PGPregel adds some extra overhead to the end-to-end latency of geo-distributed graph processing. Evaluations using PageRank and the LiveJournal graph show that PGPregel spends on average 248ms in each iteration for message sampling, combination and distribution, which is lower than 3% of the end-to-end graph processing latency per iteration using PGPregel.

6.3.2 Overall Results of ALS. Figures 10 and 11 show the accuracy and WAN usage results of compared algorithms for the ALS application. We set the sampling probability of PGPregel to 30%. The privacy budget varies from 0.001, 0.01, 0.1 to 1. We have the following observations.

First, PGPregel obtains the lowest RMSE results among all privacy-preserving comparisons. When the privacy budget is 1, all DP-based solutions, namely Pregel-DP, Monarch-DP, PALS and PGPregel, obtain equally good accuracy results. With the decrease of the budget, the RMSE result of Pregel-DP and Monarch-DP both increase rapidly. PGPregel is able

to obtain better accuracy results compared to PALS, the DP-based solution specifically tailored for ALS, more obvious at low privacy budgets.

Second, similar to the first three applications, PGPregel can greatly reduce the WAN usage and hence system latency for ALS compared to the other algorithms except Monarch. For example, PGPregel reduces the WAN usage by 93%-98% compared to the other three DP-based solutions. Although the WAN usage of PGPregel is slightly higher than Monarch, it obtains much lower RMSE.

We make the following takeaways from the above results:

- Compared to both naive (e.g., Pregel-DP) and optimized (e.g., PALS) DP solutions, PGPregel obtains better accuracy, performance and generality;
- Compared to studies using privacy regulations as data movement constraints (e.g., Monarch), PGPregel obtains much better accuracy;
- Adapting existing geo-distributed graph processing system to preserve privacy is a non-trivial task (e.g., Monarch-DP).

In the following, we perform sensitivity studies to show the effectiveness of PGPregel. For simplicity yet without loss of generality, we only use PageRank, HITS and SALSA which adopt the same quality measures. Results of Monarch-DP are not shown since it performs similarly to Pregel-DP.

6.4 Effectiveness under Varying Top-k Sizes

As precision is more relevant measure for recommendation-based applications, we report the precision obtained by the compared systems under different top-k sizes. Specifically, we vary k from 1%, 2%, 5%, 10% to 100% of the number of vertices using Twitter graph and show the results in Figure 12. We have the following observations.

First, PGPregel obtains the highest precision among the compared solutions for all k values and all applications. This demonstrates the effectiveness of PGPregel regardless of the value of k. Second, graph processing precision increases as k increases for all three comparisons. This is because when we increase the value of k, the vertices that were not in the top-k list of the baseline results may have a chance to get in the longer list. Hence the precision of the results measured using the larger k value will increase. By default, k is 2%.

6.5 Impact of Privacy Budget ϵ

The degree of privacy protection is related to ϵ. The smaller the ϵ, the higher the protection and the larger the noise. Specifying the desired accuracy is application-dependent, similar to specifying the privacy budget. PGPregel helps users to trade-off accuracy and privacy requirements. We study the accuracy of PGPregel under different ϵ using Twitter graph, so as to learn whether PGPregel can support higher privacy protection. We vary ϵ from 0.001, 0.005, 0.01, 0.05, 0.1, 0.5 to 1. We set ϵ to 1 by default. Figure 13 shows the accuracy results of the comparisons using simulator. We have the following observations.

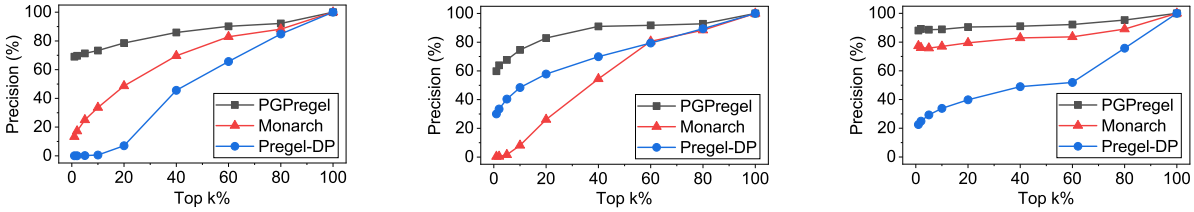
First, PGPregel obtains higher precision and lower ARE when ϵ increases. This is expected as a smaller privacy budget leads to more noise added during geo-distributed graph processing. Second, PGPregel can achieve better accuracy than state-of-the-art comparisons even at a very low budget. For example, for PageRank and SALSA, PGPregel obtains higher precision compared to Monarch and Pregel-DP under all budgets. This means PGPregel can guarantee very tight differential privacy levels while improving graph processing accuracy. Third, PGPregel shows different degrees of sensitivity on the ϵ parameter for different applications. For example, the accuracy of PGPregel decreases abruptly when ϵ is lower than 0.01 for PageRank, and less so for SALSA and HITS. This is partly due to the different global sensitivities of the applications. Both SALSA and HITS have global sensitivity of 1 while PageRank has a larger bounded sensitivity. Thus, PageRank is injected with larger amounts of noise compared to the other two applications and is more vulnerable to small privacy budgets. The reason that HITS is less sensitive to changes in privacy budgets compared to SALSA is due to the normalization operation in HITS which bounds the noise to the range of (0,1).

6.6 Impact of Sampling Rate

The sampling technique can both positively and adversely affect the accuracy of PGPregel due to the budget amplification effect and the loss of useful inter-DC information. We study its impact by varying the sampling rate from 0, 10%, 20%, ..., to 100% using Twitter graph. Figure 14 shows the obtained results. Note that, when the sampling rate is 0, PGPregel is equivalent to Monarch since there is no inter-DC communication. We have the following observations.

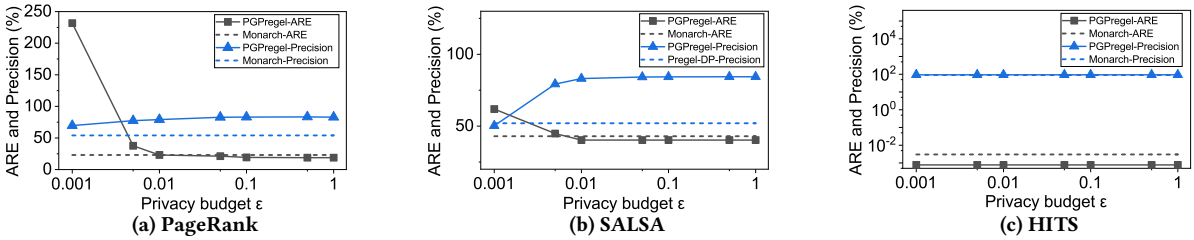
First, the graph processing precision obtained by PGPregel decreases along with the decrease in sampling rate. This is expected as a higher sampling rate maintains more useful information and hence leads to higher precision. The ARE results slightly increase with the decrease of the sampling rate. This is mainly due to the budget amplification effect which helps to diminish the adverse impact of message dropping on the accuracy of graph processing results.

Second, PGPregel shows different degrees of sensitivity dependence on sampling rate for different applications. For example, the precision of PGPregel remains high with respect to HITS (above 90%) even with decreasing sampling rates. This is consistent with the good precision results of Monarch for HITS as shown in Figure 7. Note that, the WAN usage is almost linearly related to the sampling rate. Thus, we prefer choosing a small sampling rate for HITS to achieve good accuracy and low WAN usage at the same time. PageRank is less sensitive to the sampling rate compared to SALSA. Thus,



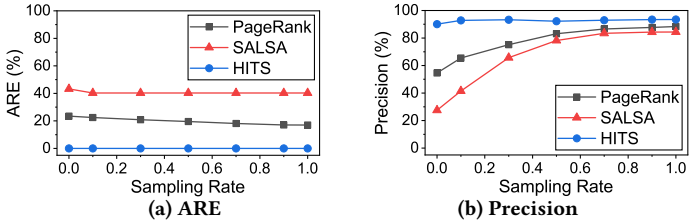
(a) PageRank (b) SALSA (c) HITS

Figure 12 Evaluating the effectiveness of PGPregel under different top-k sizes using Twitter graph.



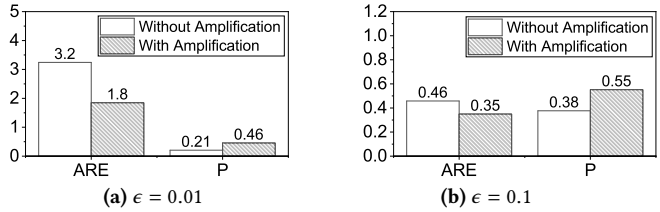
(a) PageRank (b) SALSA (c) HITS

Figure 13 Sensitivity study on the privacy budget ϵ using Twitter graph. Dashed lines represent the ARE and precision results of Monarch and Pregel-DP (whichever performs better).



(a) ARE (b) Precision

Figure 14 Sensitivity study on sampling rate using Twitter.



(a) $\epsilon = 0.01$ (b) $\epsilon = 0.1$

Figure 15 Impact of budget amplification using Livejournal.

we can choose a smaller sampling rate for PageRank than that for SALSA. By default, we set the sampling rate to 30%, 60% and 100% for HITS, PageRank and SALSA, respectively, for a good balance between system utility and latency.

6.7 Impact of Budget Amplification

Budget amplification can effectively reduce the accuracy loss caused by DP perturbation. To clearly show the positive effects brought by budget amplification, we compared the accuracy of using sampling with and without budget amplification. We perform experiments using simulator and set the privacy budget to 0.01 and 0.1 for Livejournal graph and PageRank algorithm. Figure 15 shows the obtained results. We have the following observations.

First, the budget amplification effect is useful on improving the accuracy of graph processing results under different privacy budgets. Specifically, when ϵ is 0.01 and 0.1, the budget amplification reduces the ARE by 44% and 24%, respectively, and improves the precision by 119% and 45%, respectively. Second, the budget amplification effect is more useful on improving the accuracy of graph processing results when the privacy budget is lower. This shows that PGPregel can achieve good utility even under strict privacy protection requirements.

6.8 Impact of Combiners

To study the impact of combiners on accuracy, we vary the number of non-perturbed combiners between each pair of DCs from 1 to the number of total messages. Privacy budget is set extremely large to only show the impact of combiners. Figure 16 shows the results of the study using PageRank algorithm and LiveJournal graph. The accuracy of processing results improves when the number of combiners increases from 1 to 15, while the WAN usage remains almost the same. When the number of combiners is 15, the ARE is almost zero and the precision is close to 100%. When the number of combiners further increases, the accuracy does not change much while the WAN usage increases significantly. Similar observations are also obtained on the other three applications. This shows that when the number of combiners is small, we can achieve good performance (i.e., low WAN usage) without sacrificing too much accuracy.

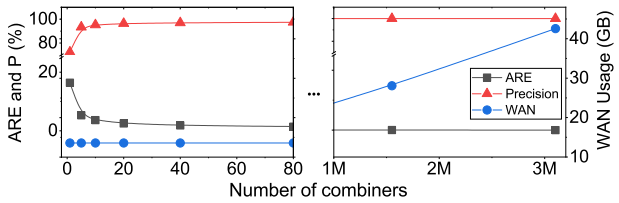


Figure 16 Impact of non-perturbed combiners on graph processing results using PageRank and LiveJournal.

7 RELATED WORK

7.1 Geo-Distributed Graph Processing

There have been many graph processing systems [3, 11, 33, 70, 71, 76] proposed for scalable and efficient graph executions in single DCs. Recently, there have been some studies focusing on the cost and performance optimizations for graph processing in geo-distributed DCs via graph partitioning [43, 77, 78]. However, such studies require vertex migration to perform graph partitioning, which introduces large inter-DC data movement and may violate privacy regulations when used in geo-distributed environments.

A few graph execution models have been proposed specifically for geo-distributed DCs [5, 37, 49, 72]. For example, Monarch [37] proposed to use sampling technique and incremental computation to reduce the data exchanged across geo-distributed DCs. HSP [49] is extended from the BSP model by performing synchronization in a two-level hierarchy to get a lower WAN bandwidth usage and faster convergence. GeoGraph [72] is a universal framework to support efficient geo-distributed graph query processing based on clustering DCs and meta-graph. Compressed data direct computing [75] is another promising technique to improve the efficiency of large-scale graph processing. None of the studies mentioned above has considered the privacy issue in geo-distributed DCs, which is crucial to big data applications [17].

7.2 Privacy-Preserving Graph Processing

The privacy issue has attracted great attention from different applications. Vulimiri e. al [65] considered privacy regulation as constraints and restricted data movement in case of regulation violation. Quoc et. al [56] proposed a privacy-preserving stream analytics system on distributed users' private dataset. Agarwal et. al [2] proposed a privacy-preserving model for distributed machine learning based on DP. However, these systems cannot be easily adapted to graph applications which have different computation and communication patterns. Privacy-preserving graph publishing techniques have been studied to preserve the utility of graphs while preserving data privacy [19, 35, 38, 46, 47]. Although such studies can protect graph structures, there still lacks an end-to-end solution which can preserve graph data privacy during geo-distributed execution without modifying the applications.

Achieving privacy preservation during graph processing is non-trivial. Although existing techniques such as data encryption [30, 64, 74] and secure multi-party computation [26, 31] can be used to preserve data privacy, they are known for high latency. Differential privacy (DP) [23] is a general and lightweight technique for privacy-preserving and has been explored by existing studies [7, 50, 52, 68] in collaborative filtering recommendation systems to protect personal data privacy. Sealfon [58] uses DP when calculating the shortest path of graphs, where the graph topology is public and the private information consists only of edge weights. However, these studies are proposed for a specific application only. To the best of our knowledge, there is no universal DP solution that can adapt to different graph applications.

8 CONCLUSION

Graph processing in geo-distributed DCs faces two new challenges, including the performance issue due to scarce network resource in wide area and the privacy issue due to asymmetric privacy regulations in different geographical regions. To address the challenges, we propose *PGPregel*, an end-to-end system for privacy-preserving graph processing in geo-distributed DCs with *high utility* and *low latency*. PGPregel adopts differential privacy (DP) to preserve privacy and incorporates two techniques including sampling and combiners to reduce the impact of DP perturbation hence improving the accuracy of graph processing results. The two techniques are also useful for reducing inter-DC data communication and hence reducing system latency. PGPregel takes the first step towards making a *practical* DP-based system for a type of important graph applications. Compared to existing DP-based solutions which are mostly specialized for one application [29], PGPregel shows much better generality. Evaluations using real-world graphs and real cloud traces have demonstrated the effectiveness of PGPregel.

ACKNOWLEDGEMENT

This work is supported by the National Natural Science Foundation of China 62172282, Guangdong Natural Science Foundation (2022A1515010122, 2019A1515012053), Guangdong Provincial Key Laboratory of Popular High Performance Computers, Shenzhen Science and Technology Foundation JCYJ20210324093212034, the Tencent "Rhinoceros Birds" - Scientific Research Foundation for Young Teachers of Shenzhen University, and the ANR KerStream project (ANR-16-CE25-0014-01). We thank our shepherd Ahmed Eldawy and all the reviewers for their insightful comments.

REFERENCES

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. 308–318.

[2] Naman Agarwal, Ananda Theertha Suresh, Felix Yu, Sanjiv Kumar, and H. Brendan McMahan. 2018. CpSGD: Communication-Efficient and Differentially-Private Distributed SGD. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems* (Montréal, Canada) *(NIPS'18)*. Curran Associates Inc., Red Hook, NY, USA, 7575–7586.

[3] Zhiyuan Ai, Mingxing Zhang, Yongwei Wu, Xuehai Qian, Kang Chen, and Weimin Zheng. 2017. Squeezing out all the value of loaded data: An out-of-core graph processing system with reduced disk i/o. In *2017 USENIX Annual Technical Conference (USENIX ATC 17)*. 125–137.

[4] Apache. 2020. Apache Giraph. https://giraph.apache.org/.

[5] Kaustubh Beedkar, Jorge-Arnulfo Quiané-Ruiz, and Volker Markl. 2021. Compliant Geo-distributed Query Processing. In *Proceedings of the 2021 International Conference on Management of Data*. 181–193.

[6] Amos Beimel, Shiva Prasad Kasiviswanathan, and Kobbi Nissim. 2010. Bounds on the sample complexity for private learning and private data release. In *Theory of Cryptography Conference*. Springer, 437–454.

[7] Arnaud Berlioz, Arik Friedman, Mohamed Ali Kaafar, Roksana Boreli, and Shlomo Berkovsky. 2015. Applying differential privacy to matrix factorization. In *Proceedings of the 9th ACM Conference on Recommender Systems*. 107–114.

[8] Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. 2020. Candidate iO from homomorphic encryption schemes. (2020).

[9] Mark Bun, Cynthia Dwork, Guy N Rothblum, and Thomas Steinke. 2018. Composable and versatile privacy via truncated cdp. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*. 74–86.

[10] Rui Chen, Noman Mohammed, Benjamin CM Fung, Bipin C Desai, and Li Xiong. 2011. Publishing set-valued data via differential privacy. *Proceedings of the VLDB Endowment* 4, 11 (2011), 1087–1098.

[11] Rong Chen, Jiaxin Shi, Yanzhe Chen, Binyu Zang, Haibing Guan, and Haibo Chen. 2019. Powerlyra: Differentiated graph computation and partitioning on skewed graphs. *ACM Transactions on Parallel Computing (TOPC)* 5, 3 (2019), 1–39.

[12] Alice Cheng and Eric Friedman. 2006. Manipulability of PageRank under sybil strategies.

[13] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. 2020. TFHE: fast fully homomorphic encryption over the torus. *Journal of Cryptology* 33, 1 (2020), 34–91.

[14] Avery Ching, Sergey Edunov, Maja Kabiljo, Dionysios Logothetis, and Sambavi Muthukrishnan. 2015. One trillion edges: Graph processing at facebook-scale. *Proceedings of the VLDB Endowment* 8, 12 (2015), 1804–1815.

[15] Seung Geol Choi, Dana Dachman-Soled, S. Dov Gordon, Linsheng Liu, and Arkady Yerukhimovich. 2021. Compressed Oblivious Encoding for Homomorphically Encrypted Search. *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security* (2021).

[16] European Commission. 2016. Article 45 GDPR: transfers on the basis of an adequacy decision. https://bit.ly/3qPD9cT.

[17] Federal Trade Commission. 2019. FTC Imposes $5 Billion Penalty and Sweeping New Privacy Restrictions on Facebook. https://bit.ly/3eRUwnP.

[18] Amol Deshpande and Ashwin Machanavajjhala. 2018. PRIVACY CHALLENGES IN THE POST-GDPR WORLD: A DATA MANAGEMENT PERSPECTIVE. https://bit.ly/3zuiyP7.

[19] Xiaofeng Ding, Cui Wang, Kim-Kwang Raymond Choo, and Hai Jin. 2019. A novel privacy preserving framework for large scale graph data publishing. *IEEE transactions on knowledge and data engineering* 33, 2 (2019), 331–343.

[20] Irit Dinur and Kobbi Nissim. 2003. Revealing information while preserving privacy. In *PODS '03*.

[21] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*. Springer, 265–284.

[22] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*. Springer, 265–284.

[23] Cynthia Dwork and Aaron Roth. 2014. The Algorithmic Foundations of Differential Privacy. *Found. Trends Theor. Comput. Sci.* 9 (2014), 211–407.

[24] Cynthia Dwork, Adam D. Smith, Thomas Steinke, and Jonathan Ullman. 2017. Exposed! A Survey of Attacks on Private Data. *Social Science Research Network* 4 (2017), 61–84.

[25] Chantat Eksombatchai, Pranav Jindal, Jerry Zitao Liu, Yuchen Liu, Rahul Sharma, Charles Sugnet, Mark Ulrich, and Jure Leskovec. 2018. Pixie: A system for recommending 3+ billion items to 200+ million users in real-time. In *Proceedings of the 2018 world wide web conference*. 1775–1784.

[26] David E. Evans, Vladimir Kolesnikov, and Mike Rosulek. 2018. A Pragmatic Introduction to Secure Multi-Party Computation. *Found. Trends Priv. Secur.* 2 (2018), 70–246.

[27] Caroline Fontaine. 2019. Fully Homomorphic Encryption Implementation Progresses and Challenges. http://www.lsv.fr/~fontaine/CFontaine-FIC-2019-01-22.pdf.

[28] Forrester Research. 2021. Forrester's Global Map of Privacy Rights and Regulations. https://bit.ly/3NUQjje.

[29] Arik Friedman, Shlomo Berkovsky, and Mohamed Ali Kaafar. 2016. A differential privacy framework for matrix factorization recommender systems. *User Modeling and User-Adapted Interaction* 26, 5 (2016), 425–458.

[30] Esha Ghosh, Seny Kamara, and Roberto Tamassia. 2021. Efficient Graph Encryption Scheme for Shortest Path Queries. In *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security*. 516–525.

[31] Oded Goldreich. 2004. Foundations of Cryptography: Volume 2, Basic Applications.

[32] Joseph E Gonzalez, Yucheng Low, Haijie Gu, Danny Bickson, and Carlos Guestrin. 2012. Powergraph: Distributed graph-parallel computation on natural graphs. In *10th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 12)*. 17–30.

[33] Samuel Grossman, Heiner Litz, and Christos Kozyrakis. 2018. Making pull-based graph processing performant. *ACM SIGPLAN Notices* 53, 1 (2018), 246–260.

[34] Pankaj Gupta, Ashish Goel, Jimmy Lin, Aneesh Sharma, Dong Wang, and Reza Zadeh. 2013. Wtf: The who to follow service at twitter. In *Proceedings of the 22nd international conference on World Wide Web*. 505–514.

[35] Michael Hay, Chao Li, Gerome Miklau, and David Jensen. 2009. Accurate estimation of the degree distribution of private networks. In *2009 Ninth IEEE International Conference on Data Mining*. IEEE, 169–178.

[36] Nils Homer, Szabolcs Szelinger, Margot Redman, David Duggan, Waibhav Tembe, Jill Muehling, John V. Pearson, Dietrich A. Stephan, Stanley F. Nelson, and David W. Craig. 2008. Resolving Individuals Contributing Trace Amounts of DNA to Highly Complex Mixtures Using High-Density SNP Genotyping Microarrays. *PLoS Genetics* 4 (2008).

[37] Anand Padmanabha Iyer, Aurojit Panda, Mosharaf Chowdhury, Aditya Akella, Scott Shenker, and Ion Stoica. 2018. Monarch: gaining command

on geo-distributed graph analytics. In *10th {USENIX} Workshop on Hot Topics in Cloud Computing (HotCloud 18)*.

[38] Zach Jorgensen, Ting Yu, and Graham Cormode. 2016. Publishing Attributed Social Graphs with Formal Privacy Guarantees. *Proceedings of the 2016 International Conference on Management of Data* (2016).

[39] Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. 2011. What can we learn privately? *SIAM J. Comput.* 40, 3 (2011), 793–826.

[40] Jon M Kleinberg et al. 1998. Authoritative sources in a hyperlinked environment.. In *SODA*, Vol. 98. Citeseer, 668–677.

[41] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. 2010. What is Twitter, a social network or a news media?. In *WWW '10*. 591–600.

[42] Ronny Lempel and Shlomo Moran. 2001. SALSA: the stochastic approach for link-structure analysis. *ACM Transactions on Information Systems (TOIS)* 19, 2 (2001), 131–160.

[43] He Li, Hang Yuan, Jianbin Huang, Jiangtao Cui, Xiaoke Ma, Senzhang Wang, Jaesoo Yoo, and S Yu Philip. 2021. Group Reassignment for Dynamic Edge Partitioning. *IEEE Transactions on Parallel and Distributed Systems* 32, 10 (2021), 2477–2490.

[44] Jing Li, Xiaohui Kuang, Shujie Lin, Xu Ma, and Yi Tang. 2020. Privacy preservation for machine learning training and classification based on homomorphic encryption schemes. *Information Sciences* 526 (2020), 166–179.

[45] Ninghui Li, Wahbeh Qardaji, and Dong Su. 2012. On sampling, anonymization, and differential privacy or, k-anonymization meets differential privacy. In *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*. 32–33.

[46] Qiongxiu Li, Jaron Skovsted Gundersen, Richard Heusdens, and Mads Græsbøll Christensen. 2021. Privacy-Preserving Distributed Processing: Metrics, Bounds and Algorithms. *IEEE Transactions on Information Forensics and Security* 16 (2021), 2090–2103.

[47] Xiang-Yang Li, Chunhong Zhang, Taeho Jung, Jianwei Qian, and Linlin Chen. 2016. Graph-based privacy-preserving data publication. In *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*. IEEE, 1–9.

[48] Qi Liu, Biao Xiang, Nicholas Jing Yuan, Enhong Chen, Hui Xiong, Yi Zheng, and Yu Yang. 2017. An influence propagation view of pagerank. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 11, 3 (2017), 1–30.

[49] Shuhao Liu, Li Chen, Baochun Li, and Aiden Carnegie. 2018. A hierarchical synchronous parallel model for wide-area graph analytics. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*. IEEE, 531–539.

[50] Ashwin Machanavajjhala, Aleksandra Korolova, and Atish Das Sarma. 2011. Personalized social recommendations-accurate or private? *arXiv preprint arXiv:1105.4254* (2011).

[51] Grzegorz Malewicz, Matthew H Austern, Aart JC Bik, James C Dehnert, Ilan Horn, Naty Leiser, and Grzegorz Czajkowski. 2010. Pregel: a system for large-scale graph processing. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*. 135–146.

[52] Frank McSherry and Ilya Mironov. 2009. Differentially private recommender systems: Building privacy into the netflix prize contenders. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. 627–636.

[53] Frank D. McSherry. 2009. Privacy Integrated Queries: An Extensible Platform for Privacy-preserving Data Analysis. In *Proc. of ACM SIGMOD* (Providence, Rhode Island, USA). 12 pages. http://doi.acm.org/10.1145/1559845.1559850

[54] Marc A Najork. 2007. Comparing the effectiveness of HITS and SALSA. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*. 157–164.

[55] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. *The PageRank citation ranking: Bringing order to the web.* Technical Report. Stanford InfoLab.

[56] Do Le Quoc, Martin Beck, Pramod Bhatotia, Ruichuan Chen, Christof Fetzer, and Thorsten Strufe. 2017. PrivApprox: Privacy-Preserving Stream Analytics. In *2017 USENIX Annual Technical Conference (USENIX ATC 17)*. USENIX Association, Santa Clara, CA, 659–672. https://bit.ly/3pRMaD4

[57] Alex Sangers, Maran van Heesch, Thomas Attema, Thijs Veugen, Mark Wiggerman, Jan Veldsink, Oscar Bloemen, and Daniël Worm. 2019. Secure Multiparty PageRank Algorithm for Collaborative Fraud Detection. In *Financial Cryptography and Data Security*. Springer International Publishing, 605–623.

[58] Adam Sealfon. 2016. Shortest paths and distances with differential privacy. In *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*. 29–41.

[59] Aneesh Sharma, Jerry Jiang, Praveen Bommannavar, Brian Larson, and Jimmy Lin. 2016. GraphJet: Real-time content recommendations at Twitter. *Proceedings of the VLDB Endowment* 9, 13 (2016), 1281–1292.

[60] R. Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership Inference Attacks Against Machine Learning Models. *2017 IEEE Symposium on Security and Privacy (SP)* (2017), 3–18.

[61] Adam Smith. 2009. Differential privacy and the secrecy of the sample. https://bit.ly/3nbYU5T.

[62] Statista. 2021. *Leading countries based on number of Twitter users as of July 2021.* https://bit.ly/32IM2NC.

[63] Pamela Vagata and Kevin Wilfong. 2014. *Scaling the Facebook data warehouse to 300 PB.* https://bit.ly/3qPEG2D.

[64] Alexander Viand, Patrick Jattke, and Anwar Hithnawi. 2021. SoK: Fully Homomorphic Encryption Compilers. *2021 IEEE Symposium on Security and Privacy (SP)* (2021), 1092–1108.

[65] Ashish Vulimiri, Carlo Curino, P. Brighten Godfrey, Thomas Jungblut, Jitu Padhye, and George Varghese. 2015. Global Analytics in the Face of Bandwidth and Regulatory Constraints. In *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*. USENIX Association, Oakland, CA, 323–336.

[66] Yu-Xiang Wang, Borja Balle, and Shiva Prasad Kasiviswanathan. 2019. Subsampled rényi differential privacy and analytical moments accountant. In *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, 1226–1235.

[67] Yu-Xiang Wang, Stephen Fienberg, and Alex Smola. 2015. Privacy for free: Posterior sampling and stochastic gradient monte carlo. In *International Conference on Machine Learning*. PMLR, 2493–2502.

[68] Zhengzheng Xian, Qiliang Li, Gai Li, and Lei Li. 2017. New collaborative filtering algorithms based on SVD++ and differential privacy. *Mathematical Problems in Engineering* 2017 (2017).

[69] Biao Xiang, Qi Liu, Enhong Chen, Hui Xiong, Yi Zheng, and Yu Yang. 2013. Pagerank with priors: An influence propagation perspective. In *Twenty-Third International Joint Conference on Artificial Intelligence*.

[70] Wenlei Xie, Guozhang Wang, David Bindel, Alan Demers, and Johannes Gehrke. 2013. Fast iterative graph computation with block updates. *Proceedings of the VLDB Endowment* 6, 14 (2013), 2014–2025.

[71] Xiating Xie, Xingda Wei, Rong Chen, and Haibo Chen. 2019. Pragh: Locality-preserving graph traversal with split live migration. In *2019 USENIX Annual Technical Conference (USENIX ATC 19)*. 723–738.

[72] Ye Yuan, Delong Ma, Zhenyu Wen, Yuliang Ma, Guoren Wang, and Lei Chen. 2020. Efficient Graph Query Processing over Geo-Distributed Datacenters. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 619–628.

[73] Dave Zachariah, Martin Sundin, Magnus Jansson, and Saikat Chatterjee. 2012. Alternating least-squares for low-rank matrix reconstruction. *IEEE Signal Processing Letters* 19, 4 (2012), 231–234.

[74] Can Zhang, Liehuang Zhu, Chang Xu, Kashif Sharif, Chuan Zhang, and Ximeng Liu. 2020. PGAS: Privacy-preserving graph encryption for accurate constrained shortest distance queries. *Information Sciences* 506 (2020), 325–345.

[75] Feng Zhang, Jidong Zhai, Xipeng Shen, Onur Mutlu, and Xiaoyong Du. 2022. POCLib: A high-performance framework for enabling near orthogonal processing on compression. *IEEE Transactions on Parallel and Distributed Systems* 33, 2 (2022), 459–475.

[76] Mingxing Zhang, Youwei Zhuo, Chao Wang, Mingyu Gao, Yongwei Wu, Kang Chen, Christos Kozyrakis, and Xuehai Qian. 2018. GraphP: Reducing communication for PIM-based graph processing with efficient data partition. In *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 544–557.

[77] Amelie Chi Zhou, Shadi Ibrahim, and Bingsheng He. 2017. On achieving efficient data transfer for graph processing in geo-distributed datacenters. In *2017 IEEE 37th international conference on distributed computing systems (ICDCS)*. IEEE, 1397–1407.

[78] Amelie Chi Zhou, Bingkun Shen, Yao Xiao, Shadi Ibrahim, and Bingsheng He. 2019. Cost-aware partitioning for efficient large graph processing in geo-distributed datacenters. *IEEE Transactions on Parallel and Distributed Systems* 31, 7 (2019), 1707–1723.