Sampling Based Range Partition Methods for Big Data Analytics

Milan Vojnović
Microsoft Research
milanv@microsoft.com

Fei Xu
Microsoft Corporation
feixu@microsoft.com

Jingren Zhou
Microsoft Corporation
jrzhou@microsoft.com

March 2012

Technical Report
MSR-TR-2012-18

Microsoft Research
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
http://www.research.microsoft.com

Abstract– Big Data Analytics requires partitioning datasets into thousands of partitions according to a specific set of keys so that different machines can process different partitions in parallel. Range partition is one of the ways to partition the data that is needed whenever global ordering is required. It partitions the data according to a pre-defined set of exclusive and continuous ranges that covers the entire domain of the partition key. Providing high-quality (approximately equal-sized) partitions is a key problem for the big data analytics because the job latency is determined by the most loaded node. This problem is especially challenging because typically no statistics about the key distribution over machines for an input dataset is available at the beginning of a range partition. The system needs to find a way to determine the partition boundaries that is both cost-effective and accurate. This paper presents a weighted-sampling based approach, implemented in Cosmos–the cloud infrastructure for big data analytics used by Microsoft Online Service Division. The approach has been used by many jobs daily and was found to be both efficient and providing desired partition quality.

1. INTRODUCTION

Big data analytics has been gaining a momentum and growing in importance over the last few years [5, 6, 10, 12, 15, 1]. This involves large-scale computations that use as input data of volume that is often in the order of terabytes or petabytes and are run in production data centers involving tens of thousands of machines. A key factor to make such computations efficient is to partition the data evenly across different machines, so as to fully exploit parallelism in computation where individual data partitions are processed in parallel on different machines. However, we cannot always partition the data arbitrarily. Different operators impose different restrictions on the data partitioning. For example, a GROUP BY query requires all the records with the same key to reside in the same node. ORDER BY requires the output file to be fully sorted, which means the data needs to be range partitioned. In distributed data analytics, three partition methods are commonly used (1) random, (2) hash partition and (3) range partition. Different operators may impose different restrictions and thus only some of the partitions are applicable. Section 2 provides a detailed discussion between commonly used operators and the three partition methods.

Compared to the range partition, the other two methods are relatively easier to implement since they do not need to worry about the relationship between different key values. On the other hand, range partition requires a set of key-ranges to be pre-defined. It is especially difficult to determine the range partition boundaries in a time and communication cost efficient manner for massive scale data that is input to a computation. This input data may be a result of an intermediate step in a computation and may be of an arbitrary format. In such cases, we may not have statistics about the data (to be partitioned), such as the number of distinct keys or the total number of keys. Furthermore, the input data may be skewed and the statistics such as the number of records per machine or the number of records per machine per key would typically be a priori unknown, making the problem of data partitioning even more difficult.

A typical architecture for solving the range partition problem is to retrieve some statistics from each site that contains the input data by a coordinator node. The coordinator then uses the statistic to determine the partition boundaries. In general, effective range partitioning is crucial for many scenarios in a distributed environment and how to choose partition boundaries is a challenge. Sampling based methods for range partition are in particular appealing in comparison with alternative approaches (e.g. using quantile summaries) in view of their simplicity and efficiency (see the discussion in related work section). For example, Terasort [13], which won the data sorting challenge in 2008 [11], used a customized sampling based partitioner with standard Mapreduce sort running on Hadoop [5, 6]. The importance of selecting an appropriate sample for range partition was articulated and experimentally demonstrated in [14].

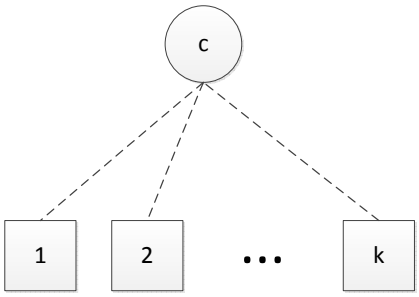

Figure 1: System consists of a coordinator node and k sites. Each site has access to a portion of input data.

We consider a standard distributed system model that consists of k sites and a coordinator node (see Figure 1). The coordinator is a designated node that needs to compute the partition boundaries using as input some statistic about the dataset. A site may refer to a task running on a machine in a distributed cluster of machines, which reads a portion of input data. Tasks are usually assigned to machines respecting data locality, so that typically, a task is assigned to a machine that has a proximal access to data. Hence, our goal is to make efficient use of the communication bandwidth between the coordinator node and sites. Notice that the range partition problem bears an intrinsic accuracy-cost trade-off - we would like to take a large number of samples in order to accurately determine the boundaries of ranges; at the same time, we want to limit the number of samples in order to save communication bandwidth. The question that we study in this paper is as follows:

> *Q) How samples should be taken from a set of distributed sites such that the data range partition meets a prescribed accuracy guarantee?*

We define the range partition problem studied in this paper more specifically as follows. Suppose the input data is a multiset Ω of $n \geq 1$ data items that admit a total order, that is for every pair of data items $a, b \in \Omega$ we have either $a \leq b$ or $a > b$. This accommodates as a special case a dataset where each item a corresponds to a real value in Ω, and note that we allow for values to be non unique. The input data is partitioned across k sites such that each site $j \in [k] = \{1, 2, \ldots, k\}$ is associated with a multiset of data

items Ω_j and we have $\Omega = \uplus_{j\in[k]}\Omega_j$. Given an input parameter $\vec{p} = (p_1, p_2, \ldots, p_m)$ such that $0 \leq p_i \leq 1$, for every $i \in [m]$ and $\sum_{i\in[m]} p_i = 1$, the goal is to partition data in m ranges such that the number of data items mapped to a range i is approximately $p_i n$. Our primary interest in this paper is to produce a balanced range partition, so that $p_i = 1/m$, for every $i \in [m]$, as this is the most important case for big data analytics applications.[1] A range partition of Ω according to $\vec{p}$ is said to be ϵ-accurate, if for given $0 \leq \epsilon < 1$, the number of data items that are mapped to range i, denoted as Q_i, satisfies $Q_i \leq (1+\epsilon)p_i n$, for every range $i \in [m]$. This is a standard objective for range partition [2]. A sampling based range partition amounts to collecting a sample of data items from Ω by the coordinator node, and then computing the boundaries of ranges using a statistic derived from this sample. The challenge lies in choosing a sample size that is large enough so that the range partitioning meets the accuracy guarantee with a prescribed probability of success.

In this paper we characterize the required sample size to guarantee the aforementioned quality of the data range partition. We consider a simple sampling scheme which assumes a provision to take a random sample of a given sample size from a distributed dataset. We first characterize the sufficient sample size for the simple sampling scheme without worrying how to take such a sample (assuming this is enabled by an underlying system component). Previous work assumes that each record in the dataset (to be partitioned) has a unique key. We provide results that allow non-unique key values. Taking a random sample from a distributed dataset is a nontrivial task in absence of statistics such as the number of data items per site. TeraSort uses a simple approximate approach by taking an equal number of samples from each site independently. This approximation is acceptable if the input among different sites are approximately equally-sized, though they still did not answer an important question on how many samples to take from each site so as to guarantee a prescribed partition accuracy. This approach fails in many day-to-day jobs running in Cosmos where the distribution of the number of data items across sites is commonly imbalanced. We discuss several common scenarios that cause this imbalance and provide empirical evidence that such imbalances are indeed common patterns in practice, using some typical datasets that are processed in Cosmos. We then introduce a more sophisticated weighted sampling scheme that accounts for the imbalance of the number of data items across sites. This sampling scheme takes an equal number of samples from each site, but then merges these samples into a summary sample in such a way that ensures a site gets a proportional representation in this summary sample with respect to the number of data items accessed through this site. We then mathematically analyze how many samples are needed using this approach in order to satisfy a prescribed accuracy guarantee.

1.1 Summary of Our Results

Our contributions can be summarized in the following points:

1) Simple Sampling Scheme: We studied a simple sampling scheme and characterized a sufficient sample size for this sampling scheme to guarantee a given accuracy with a prescribed probability where the input data could have multiple records with the same partition key value. Previous research work was restricted to the case of unique key values [2].

2) Weighted Sampling Scheme: We provide and analyze a practical weighted sampling scheme that allows input data to have repeated partition key values and allows the input data to have arbitrary distribution of the number of data items per site, given that the simple sampling scheme is difficult to be implemented in practice. We characterize the sufficient sample size to guarantee a given accuracy with a prescribed probability. This analysis reveals what is the sufficient statistics needed about the dataset to determine the required sample size. In particular, we find that a key statistic is a measure of imbalance of the number of data items across sites.

3) Data Analysis: We provide a data analysis of some typical datasets processed in a production data center of a major online service provider to evaluate the extent of data imbalance across sites, and provide simulation results to evaluate the efficiency of the weighted sampling scheme.

1.2 Outline of the Paper

The paper is structured as follows. In Section 2 we first overview the architecture of Cosmos and then discuss the relationship between the commonly used operators and data partitioning methods. In Section 3, we first introduce the simple sampling scheme, and then provide the sufficient sample size analysis. In Section 4, we first discuss why the simple sampling scheme is not practical. We then discuss the approach used in TeraSort and point out that this is also not practical because the input data is not always balanced across different sites for day-to-day jobs. We then discuss several typical scenarios that cause imbalance of the data items across sites, and then introduce our weighted sampling scheme. We finally provide analysis results on the required sample size to meet a prescribed accuracy guarantee for data partition with a given probability using the weighted sampling scheme. Section 5 provides data analysis results and evaluation of the required sample size for different values of input parameters and data inputs. Related work is discussed in Section 6 and in Section 7 we conclude. All the proofs are deferred to Appendix (some to the Appendix of the companion online technical report [18]).

2. DATA PARTITIONING IN BIG DATA ANALYTICS

In this section, we first discuss how a typical big data analytics platform looks like by giving a brief description of Cosmos, the cloud infrastructure for big data analytics used in Microsoft Online Service Division. We then discuss the SCOPE language, a high-level SQL-like descriptive language used by Cosmos to for big data analytics. Then we discuss different operators used in SCOPE and their relationship with three well-known partition methods. Finally, we briefly discuss the infrastructure described in the intro-

[1] Our analysis allows for arbitrary range partition, which may be of general interest. This would support environments where data has to be range-partitioned across heterogeneous computing machines or when computational complexity of an underlying task is dependent on the section of the input data range.

duction section, which is used for the range partition.

2.1 Cosmos and SCOPE

Cosmos is a large-scale, distributed storage and computing platform developed by Microsoft, running on tens of thousands of commodity servers. Cosmos provides a cost-effective infrastructure to reliably store massive amounts of data, and process and analyze the data in an efficient, robust and scalable fashion.

Cosmos has three layers: (1) Cosmos storage layer, (2) Cosmos execution layer and (3) SCOPE layer. Cosmos storage layer is a distributed storage system that is designed and optimized for storing petabytes append-only sequential files. Each file is composed by a sequence of extents. Each extent is typically a few hundred megabytes in size. Extents are compressed, and replicated to different machines for cost efficiency and reliability.

Cosmos execution layer uses Dryad [10] to execute a job. A job is modeled as a directed acyclic graph (DAG). Each vertex in the graph is a computation unit and each edge represents a data flow. The job is executed by the Job Manager, which is a single coordinator for all the vertices of the job. Job manager constructs the job graph and schedules the vertices (computation units) to different machines in the data center. It also monitors the running vertices, and rerun the vertices when failure happens. We refer the interested reader to the Dryad paper [10] for details.

SCOPE is the official (and only) language for users within Microsoft to submit analytical jobs to Cosmos. Each day, thousands of scope jobs are running over petabytes of data. SCOPE is a SQL-like descriptive scripting language. But it is also very extensible to allow arbitrary customized C# types and functions. A SCOPE script is a sequence of SCOPE statements. Most SCOPE statements take one or more rowsets as input and generates one output rowset. The most commonly used statement is the SELECT statement, which basically implements the SQL SELECT statement. It supports filtering, projection, all sorts of joins, aggregates, group by and order by. Sub-queries are not supported. SCOPE also provides three statements, PROCESS, REDUCE, and COMBINE, that allows users to write customized C# code to manipulate the rowset transformation. PROCESS is a row-wise independent operator that is equivalent to Map in MapReduce. REDUCE is the same as the Reduce in MapReduce. Combine is equivalent to a customized join. We refer the interested reader to the SCOPE paper [1] for details.

2.2 Data Processing Operators and Data Partitioning

Like other data analytics platforms, Big Data Analytics needs similar operators: projection, filtering, join, group by, sorting, and etc. Given the input data volume, it is important for Big Data Analytics platform to be able to partition input data into thousands of partitions, so that each partition can be executed in parallel. However, data cannot be partitioned arbitrarily as different operators impose different restrictions on how the system can partition the data. We review some basic restrictions.

We first look at the MapReduce programming model. Map is basically an operator that applies to each row independently. We call this type of operator a row-wise independent operator. Row-wise independent operators allow the system to process each row independently and possibly in parallel. Therefore, regardless how we partition the data, it would not break the semantic of row-wise independent operators. Reduce, however, is different. It is required that all the rows that has the same key value must be presented and processed together for a Reduce command. Therefore, our partition scheme is constrained. We cannot put rows with the same key into different partitions. We call such an operator a key-wise independent operator.

In SCOPE, it is clear that operators such as filtering and projection are row-wise independent operators. Join and group by are key-wise independent operators. Order by is different. Order by imposes a global ordering according to some ordering key. But no single machine can sort all the data, even using disk-based sorting, in a reasonable amount of time. Thus, we require data to be partitioned so that these partitions have an ordering: all records in one partition must all be either smaller or larger than any record in another partition.

As mentioned in Section 1, random partition, hash partition and range partition are three commonly used partition methods in big data analytics. Random partition randomly partitions the data into partitions without any limitations. Hash partition first needs to hash the record partition key, and then map the hash value of the key to a partition. There are multiple ways for this mapping. A typical approach is round-robin, that is, mod the hash-key with the number of partitions, the result is the partition id (0 is the first index). For example, if hash key is 38 and we need 5 partitions. This record is mapped to partition 3. It is important to understand that hash-partition provides key-wise independent guarantee, because if records have the same key value, they must have the same hash value. Thus, they will be mapped to the same partition. However, hash partition does not guarantee ordering among the partitions.

Range partition is to partition the data according to a prescribed ranges. Range partition provides both key-wise independent guarantee and partition-wise ordering. Therefore, anything that could be implemented using hash partition can also be implemented using range partition. In a big data analytical platform, if global sorting is a required feature, then range partition is required to be implemented, but hash partition is not required.

3. SIMPLE SAMPLING-BASED RANGE PARTITION

This section first introduces the simple sampling scheme. Then we discuss the sufficient sample size for this approach.

3.1 Simple Sampling Approach

In practice, the sample size is typically arbitrary fixed and thus does not necessarily meet an a priori specified guarantee. For example, in Terasort [13], the sampler used 100,000 keys to determine the reduce boundaries, and it was observed that the distribution between reduces was hardly perfect and would have benefited from more samples. It is thus important to understand how many samples should be taken from a dataset to meet an a priori specified accuracy guarantee. We will consider the following simple sampling

scheme.

Simple Sampling Scheme

Input: partition relative sizes $\vec{p} = (p_1, p_2, \ldots, p_m)$
Parameter $t :=$ number of samples

1. SAMPLE: Coordinator collects a set S of t random samples from Ω
2. PARTITION: Coordinator determines boundaries of ranges using data samples S

This is a natural and standard random sampling scheme that was considered in prior work as early as in [2]. The random sample is assumed to be either with or without replacement where the latter could be realized by a reservoir sampling scheme [17], if the data is stored on one site. The challenge lies in setting the sample size large enough such that the desired data range partition meets a desired accuracy guarantee for given relative partition sizes $\vec{p}$ and input data which may consist of data items with repeated data values.

3.2 Approximation Guarantee

We recall that given as input, relative partition sizes $\vec{p} = (p_1, p_2, \ldots, p_m)$ and the relative accuracy parameter $\epsilon > 0$, a range partition of a dataset Ω is said to be ϵ-accurate, if denoting with Q_i the number of data items of Ω that fall in range i, we have $Q_i \leq (1+\epsilon)p_i n$, for every range $i \in [m]$. We would like to guarantee the latter property to hold with a probability at least $1-\delta$, where $\delta \in (0, 1]$ is an input parameter. We will denote with π the smallest relative partition size, i.e. $\pi = \min_{i \in [m]} p_i$. Notice that for the case of a balanced partition, $\pi = 1/m$.

Let us also introduce some additional notation. Let $h(a)$ denote the frequency of a data item $a \in \Omega$, i.e. $h(a) = \frac{|\{b \in \Omega | b = a\}|}{|\Omega|}$. We define ϕ to be an upper bound on the largest frequency of a data item, i.e.

$$h(a) \leq \phi, \text{ for every } a \in \Omega.$$

Notice that if all data items are distinct, then $\phi = 1/n$.

3.3 Sufficient Sample Size for Simple Sampling Scheme

We characterize the sufficient sample size for the simple sampling scheme as follows.

THEOREM 1. *Suppose that sampling is random either with or without replacement. Suppose that $\epsilon > \phi/\pi$ and that the sample size t satisfies*

$$t \geq 2\frac{1-\pi}{\pi\epsilon^2}\frac{(1+\phi/\pi)(1-\phi/(1-\pi))}{(1-\phi/(\epsilon\pi))^2} \cdot \text{polylog}(1/\delta, m, n)$$

where $\text{polylog}(1/\delta, m, n) = \log\left(\frac{1}{\delta}\right) + \log(m) + \log(n)$.[2]

Then, the range partition is ϵ-approximate with probability at least $1-\delta$.

[2]Hereinafter, for simplicity of exposition, we omit a factor in the sufficient sample size which is $1 + O(\epsilon)$.

Proof is provided in Appendix A. Note that the theorem tells us that in order to derive a sufficient sample size, we need the following information about the input dataset: (1) an upper bound on the total number of data items and (2) an upper bound on the frequency of the most frequent data item. For the case of a balanced range partition, we have the following corollary.

COROLLARY 1. *For a balanced range partition over $m > 1$ ranges, we have that an ϵ-approximate range partition can be computed with probability at least $1-\delta$, provided that $\epsilon > \phi m$ and the sample size t is such that*

$$t \geq \frac{2m}{\epsilon^2}\frac{1+\phi m}{(1-\phi m/\epsilon)^2} \cdot \text{polylog}(1/\delta, m, n).$$

These results are established by using the framework of large deviations and provide a tight characterization of the probability of error exponent (the factor that multiplies the polylog term in the above expressions).. For the special case of a balanced partition into m ranges and $\phi m = o(1)$,[3] the bound in the corollary boils down to

$$t \geq \frac{2m}{\epsilon^2} \cdot \text{polylog}(1/\delta, m, n).$$

The result of this type for a balanced range partition was first asserted in [2] and later extended to arbitrary range partition in [16]. The new information in the above results is in allowing for non unique data values which affects the sample size through the parameter ϕ. The communication complexity of the sampling scheme is $\tilde{O}(m/\epsilon^2)$ where the scaling $\Theta(1/\epsilon^2)$ is a folklore known to hold for sampling based schemes. We further discuss related work later in Section 6.

Remark In Appendix A, we provide a less explicit but tighter characterization of the sufficient sample size than that in Theorem 1.

4. WEIGHTED SAMPLING-BASED RANGE PARTITION

In this section, we first discuss why simple sampling scheme is not practical in applications. We then introduce the weighted sampling scheme, and provide the sufficient sample size analysis for this scheme.

4.1 Why is Simple Sampling Scheme not Practical?

The key assumption of the simple sampling scheme is that there is a provisioning to take a random sample from the dataset. This is a nontrivial task in distributed environments where the input data is distributed across several sites. Any one-pass algorithm requires to merge the samples taken from each site into a statistically correct random sample, with or without replacement. The merge operation is costly. One needs to take a random sample from a hypo-geometric distribution or multinomial distribution for sampling without replacement and with replacement, respectively. This limits size of merged sample to be the minimum size of the samples from each site. That is, most of the

[3]This holds, for example, when the data values are unique so that $\phi = O(1/n)$ and $m/n = o(1)$, i.e. the number of partitions is much smaller than the total number of data items.

samples are discarded in order to make a statistically correct random sample. Other approaches are no longer one-pass. Typical approaches need at least two passes. In the first pass, each site sends the number of records from each site to the coordinator. The coordinator then can determine how many samples is needed from each site by using a hypergeometric distribution or multinomial distribution, depending whether sampling without replacement or replacement is used. These samples can then be simply merged. A two-pass algorithm is generally too costly in this environment.

TeraSort [13] uses an approximate approach by taking equal-size random samples from each site, and the merges these samples at the coordinator directly. This approach is reasonable for TeraSort since each site has the same input data volume, even though they do not study the sufficient sample-size problem in their approach, and just picked some arbitrary number as the sample size. However, this approach is not practical for day-to-day big analytical jobs in practice. This is because the data input sizes across sites is imbalanced and there may be a lack of a prior information about this imbalance. In the next section, we evaluate origins of such data input size imbalance and then present a sampling scheme that accounts for this imbalance.

4.2 What may Cause Input Sizes Imbalance?

We discuss several common data processing operations that may cause imbalance of data input sizes across sites. This, together with data analysis in Section 5, serves as a motivation to consider a more sophisticated sampling scheme in the section following the present one. We consider the following common data processing operations.

JOIN. Consider the following basic join operation:

```
SELECT
FROM A INNER JOIN B ON A.KEY==B.KEY
ORDER BY COL;
```

This requires the system to co-partition A and B on A.KEY and B.KEY, respectively. For intermediate results, our system writes the output to the local volume of the same machine. Then, the system needs to partition the join result (using range-partition) on COL. Even though we can guarantee that A and B are evenly partitioned, there is no guarantee that the join result of each node would contain approximately the same number of records. In fact, the number of records on different nodes can be very different. This is one real example where we need to partition a data input with imbalanced sizes across machines. In distributed computing environments, there are even other fundamental reasons why a balanced input does not guarantee a balanced output, for example, in these environments it is typical to write intermediate output to local volumes directly, which are then used as input for subsequent computations. Again, there is no reason that such intermediate result sets would be of roughly the same sizes across different nodes.

REDUCE. Consider a reduce command that reduces on an integer key. What this reducer does is to replicate copies of rows it sees according to the key value. For example, if key=2 then all records are replicated twice. If key=100, all records are recorded 1000 times. Now if the number of keys at each node is identical, the output data may be different, depending only on the key values.

Lookup Table. Consider that we have a static lookup table. Each node does the following: for any given input row, look at column X, if column X is in the lookup table, then the row is returned. Otherwise, the row is filtered out. If the input was range partitioned (or hash partitioned) on column X, then there is a good chance that many nodes return very few rows and a few nodes return many rows.

UNPIVOT. Consider the following example of the unpivot operation, which is widely used in processing of graph data:

```
Column 1  Column 2
1         2, 3
2         3, 9, 8, 13
...
```

The output of the unpivot operation is (1,2), (1,3), (2,3), (2,9), (2,8), (2,13), $\cdots$. If the data range partition is with respect to Column 1, then the output sizes may well be imbalanced because of the variable number of elements in Column 2.

The above common data processing operations suggest that in practice input data sizes across sites may well be imbalanced, and later in Section 5 we provide an evidence of this using some common datasets from an operational data center.

4.3 Weighted Sampling Approach

In this section, we introduce a weighted sampling scheme that accounts for imbalance of data input sizes across sites. The sampling scheme is designed to create a summary sample such that a site with a larger number of data items is represented with a larger number of elements in this summary sample. This is done with the aim to reduce the bias due to imbalance of the data input sizes across different sites.

Weighted Sampling Scheme

Input: partition relative sizes $\vec{p} = (p_1, p_2, \ldots, p_m)$
Parameter $t :=$ total number of samples to take

1. SAMPLE: Coordinator node requests t_1 samples from each site. Each site i reports to the coordinator a multiset S_i that is a random sample of t/k data items from Ω_i and the total number of items $n_i = |\Omega_i|$.
2. MERGE: Coordinator node constructs a summary sample S of data items by initializing $S = \emptyset$ and then adding n_i copies of each item in S_i for every site i.
3. PARTITION: Determine a set of keys that partition the summary sample S into m ranges with each range i containing a fraction p_i of data items in S.

Notice that the data input size accessed through each site is used only after the sample is collected. This means that the scheme can be implemented by making one pass through the data - each site may take samples by using a sampling scheme (e.g. reservoir sampling [17]) that makes one pass through the data, and once this pass is completed, the site knows the total number of data items accessed through this

site. Using the above weighted sampling scheme, a summary sample is constructed such that each site is represented by a number of sampled items that is proportional to the number of data items accessed by this site. Note that is to mitigate the bias due to the imbalance of data input sizes across sites, but it does not necessarily completely remove it.[4]. Therefore, we need to study what the sufficient sample size is in order to guarantee accuracy of range partition.

4.4 Sufficient Sample Size for Weighted Sampling Scheme

We next characterize the sufficient sample size for the weighted sampling scheme. We shall see that a single parameter that quantifies the imbalance of data sizes across sites plays a key role in determining the sufficient sample size, which we introduce in the following definition.

DEFINITION 1 (INPUT SIZES IMBALANCE). *Given a number of sites $k \geq 1$ and the number of data items per site specified by the vector $\vec{n} = (n_1, n_2, \ldots, n_k)$, we define the input imbalance statistic as follows*

$$\alpha(\vec{n}) = k^2\sigma(\vec{n})^2/4.$$

where $\sigma(\vec{n})^2$ is the variance parameter given by $\sigma(\vec{n})^2 = \frac{1}{k}\sum_{i=1}^{k}\left(\frac{n_i}{n}\right)^2 - (\frac{1}{k}\sum_{i=1}^{k}\frac{n_i}{n})^2$.

Notice that $\alpha(\vec{n}) = [k\sum_{i=1}^{k}(n_i/n)^2 - 1]/4$. For the perfectly balanced number of data items per site, i.e. $\vec{n} = (n/k, n/k, \ldots, n/k)$, we have $\alpha(\vec{n}) = 0$. The other extreme is when all data items reside exclusively at one site, i.e. $n_i = n$, for some $i \in [k]$, and in this case $\alpha(\vec{n}) = (k-1)/4$.

The following is the main theorem of this paper.

THEOREM 2. *Assume $\phi < \pi\epsilon$. Then, the sufficient sample size t to guarantee ϵ-accurate range partition with probability at least $1-\delta$ is*

$$t \geq \frac{2(1-\pi+\alpha(\vec{n}))}{\pi\epsilon^2(1-\phi/(\pi\epsilon))^2} \cdot \varrho(1/\delta, m, n, k)$$

where $\varrho(1/\delta, m, n, k) = \log(1/\delta) + \log(m) + k\log(n)$.

Proof is provided in Appendix. The proof is based on a careful analysis of the probability of error exponent and extracting the most essential terms. The theorem tells us that in addition to the sufficient statistics identified for the simple sampling scheme, the additional statistic that is needed to determine a sufficient sample size is the input sizes imbalance statistic $\alpha(\vec{n})$. In Appendix B.3, we provide a normal approximation that provides the same result as in the theorem with $\varrho = \log(m/\delta)$, which we suggest to use in applications.

We next derive a corollary that characterizes sufficient sample size for data input whose sizes across sites are constant factor imbalanced, a simple concept that we introduce in the following definition.

DEFINITION 2 (CONSTANT FACTOR IMBALANCE). *A distribution data items over sites is said to be ρ-constant factor imbalanced if the number of data items n_i at each site i is such that $n_i \leq \rho n/k$.*

[4]Under a uniform random sampling scheme, the number of samples taken across sites $(t_1, t_2, \ldots, t_k)$ is a multivariate random variable that has a multinomial distribution with parameter $(t, \vec{n}/n)$. Moreover, the weighted sampling scheme replicates samples which may introduce correlations.

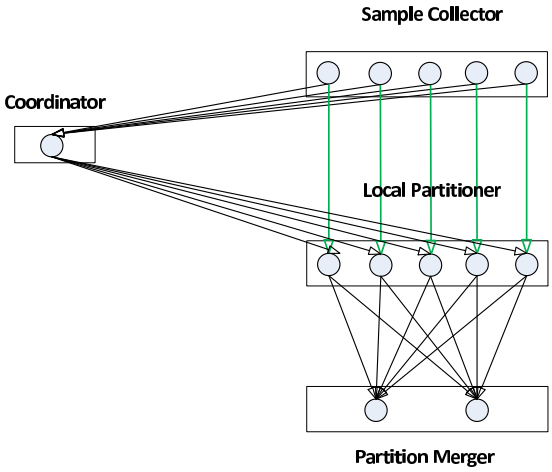

Figure 2: Weighted Sampling algorithm implementation in SCOPE.

In other words, the maximum number of data items per site is at most a factor ρ of the mean number of data items per site. We observe that for data input sizes $\vec{n}$ that are ρ-constant factor imbalanced, we have $\alpha(\vec{n}) = (\rho^2 - 1)/4$, which yields the following corollary.

COROLLARY 2. *Assume $\phi < \pi\epsilon$ and that the distribution of data items over sites is ρ-constant factor imbalanced. Then, the sufficient sample size to guarantee ϵ-accurate partition with probability at least $1-\delta$ is*

$$t \geq \frac{2(1-\pi) + \frac{1}{2}(\rho^2-1)}{\pi\epsilon^2(1-\phi/(\pi\epsilon))^2} \cdot \varrho(1/\delta, m, n, k).$$

The corollary tells us that for any distribution of data over sites such that the maximum number of data items at a site is at most a constant factor of the mean number of data items per site, the required sample size increases only for a constant factor compared with the case of perfectly balanced data input.

The above results provide an upper bound on the communication complexity. Notice that the algorithm uses a single round of communication between the coordinator node and sites, where in the forward direction $\tilde{O}(k)$ information is transferred, it remains only to asses the information in the backward direction, from the sites to the coordinator. For a balanced range partition, the total communication is $\tilde{O}(m(1+\alpha(\vec{n}))/\epsilon^2)$, and we note that in case of a constant factor balanced input sizes, the communication complexity is $\tilde{O}(m/\epsilon^2)$. This is again is inline with the folklore that sampling based schemes require $\Omega(1/\epsilon^2)$ amount of communication.

4.5 How is Weighted Sampling Range Partition Implemented in SCOPE?

Dryad uses a DAG as its execution graph, where each vertex is a computation unit and each edge is a data flow. SCOPE generates a few specific vertices and edges for the range partition in the execution graph. Figure 2 describes the graph. We describe the algorithm as follows:

1. Each site that contains the input data takes a fixed-size sample, described as the sample collector in the figure.

The site sends the total number of records together with the samples to a coordinator, discussed below. Notice that these vertices output two data channels, one for the samples to the coordinator, and one for the local partitioner–the computation vertices that do the local partitioning according to boundaries from the coordinator.

2. The coordinator collects all the samples according to the weighted-sampling scheme, and computes the range boundaries. The boundaries are sent to the local partitioners.

3. Each of the local partitioner partitions its local data according to the boundaries sent by the coordinator.

4. Finally, each downstream partition merger (one per partition) reads its portion of the data from every local partitioner and aggregates the inputs to continue its computation.

There are a few things that are worth mentioning. First, the job manager is quite intelligent in that it could figure out that sample collectors and local partitioners are reading the same data. Therefore, each corresponding local partitioner will be scheduled to the same machine as each sample collector to minimize the overhead. Second, the aggregation at the coordinator side to collects all the samples is done by building an aggregation tree to improve the efficiency and reduce the cross-pod network traffic. Third, when the coordinator sends back the partition boundaries to each local partitioner, the job manager is smart enough to only let the coordinator communicate to one vertex per pod; other vertices in the same pod will get the data from the one that communicates to the coordinator. This way, we reduce the cross-pod network traffic. Finally, the local partitioner writes to a single file even though it partitions the data into multiple partitions. The partition mergers are able to figure out what offsets they need to read from each local partitioner using the information passed as part of the metadata to build the execution graph. This way, we reduce some overhead to the distributed file system, especially for the metadata service.

5. DATA ANALYSIS

In this section we evaluate the extent of data input sizes imbalance in some real-world datasets and compare the sufficient sample size with empirical estimates. For our analysis, we use some typical datasets that are processed in Cosmos in daily job processing. We also use some synthetic data to further evaluate tightness of the sample size characterizations.

We consider a set of datasets that we retrieved by executing SCOPE queries on some typical datasets over particular intervals of time, so as to keep the data sizes manageable for the purpose of our analysis but still being representative of typical datasets. The datasets are diverse with respect to the total number of records, the raw byte sizes and the number of sites through which the data is accessed. These datasets are also diverse with respect to the application domain; they pertain to some major online services supported and processed in the Cosmos platform on a day-to-day basis. A summary of the datasets is provided in Table 5.

Table 1: Datasets used.

Dataset	Records	Raw bytes	Sites
DataSet-1	62M	150G	262
DataSet-2	37M	25G	80
DataSet-3	13M	0.26G	1
DataSet-4	7M	1.2T	301
DataSet-5	106M	7T	5652

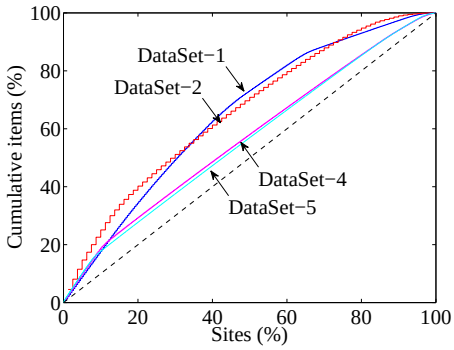

Figure 3: Distribution of the number of data items across sites.

5.1 Data Input Sizes Imbalance

We first evaluate the imbalance of the number of data items across sites. In Figure 3, for a given dataset, we present the percentage of data items covered by a given percentage of sites where sites are sorted in decreasing order with respect to the number of data items they are associated with. These representation clearly shows that the distribution of the number of data items across sites is imbalanced. If the data input sizes were perfectly balanced, then this would correspond to the dashed line in Figure 3. We observe that data input sizes may be significantly imbalanced.

We now examine the data input sizes imbalance, the statistic, which in Section 4.4 we found to play a key role in determining the sufficient sample size for the weighted sampling scheme. In Figure 4, we present this statistic for various datasets that we study and observe that there exist cases for which this parameter can be in excess of 10%.

5.2 Sufficient Sample Size

We demonstrate correctness and accuracy of the weighted sampling scheme using both real and synthetic dataset. The use of real data enables us to demonstrate performance of the algorithm in an operational system. We also use synthetic data to demonstrate accuracy and tightness of the sufficient sample size as this allows us to have control over data input sizes imbalance.

We first demonstrate the accuracy of the range partition using Dataset-1 as input data to our implementation of the weighted sampling scheme in Cosmos. This range partition was run for the number of ranges determined such that there are 100,000 of records per range and fixing the probability of error δ to 0.1, for various choices of the input parameter ϵ. In Figure 5, we show the sufficient sample size and the observed relative error versus the input parameter ϵ. These results demonstrate how closely the observed relative

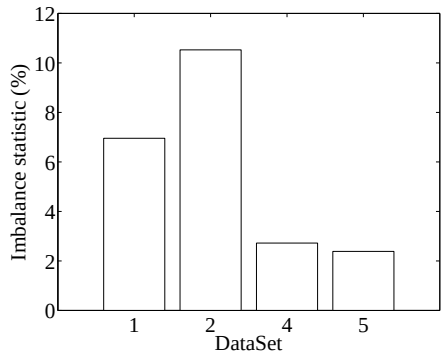

Figure 4: The data input sizes imbalance statistic for different datasets.

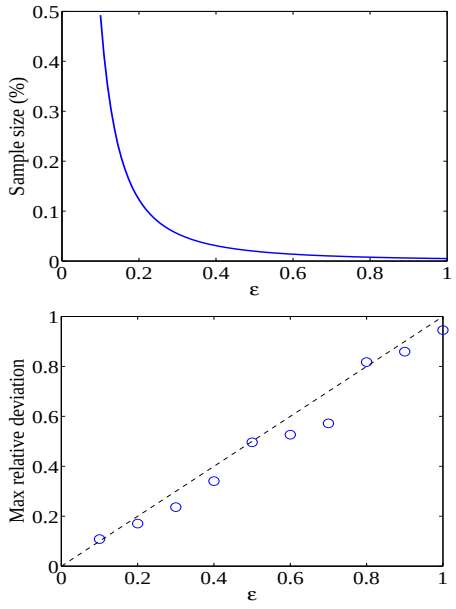

Figure 5: Range partition of DataSet-1: (top) sample size vs. ϵ and (bottom) relative error vs. ϵ.

error matches the input parameter ϵ and how this can be achieved by sampling only a small percentage of the input data records.

We further demonstrate the accuracy of the weighted sampling range partition using a synthetic input dataset in a simulator, which allows us to control the data input sizes imbalance. The input dataset is assumed to be distributed across sites so that there is a unique site that may access a larger number of data items than any other site, and all other sites access an equal portion of the input dataset. Concretely, without loss of generality, we assume that data input sizes are such that $n_1 \geq n_2 = \cdots = n_k = an_1$, where $a \geq 1$ is an input parameter that we vary. The case $a = 1$ corresponds to a balanced input dataset. Given the total data input size n, we have $n_1 = na/(a + k - 1)$ and $n_i = n/(a + k - 1)$, for $i = 2, 3, \ldots, k$. Specifically, we consider the case the input data of $n = 100,000$ data items, partitioned across $k = 4$ sites and balanced range partition in $m = 5$ ranges with the relative accuracy $\epsilon = 1/10$. The input data items are assumed to be of distinct values and assigned uniformly at random across sites subject to given data input sizes across sites. In Figure 6 we present estimated probability of error for given sample size along with 95% confidence intervals for varying values of parameter a. We observe that the probability of error clearly exibits an exponential decrease with the sample size. In the figures, we also show our analytical sample sizes, derived in Theorem 2, for $\varrho = \log(1/\delta)$ and (dashed line) and $\varrho = \log(m/\delta)$ (solid line). These results demonstrate the accuracy and tightness of the sufficient sample size characterization.

6. RELATED WORK

Big data analytics becomes an important area over last few years. MapReduce [5, 6] introduced by researchers allows users to process massive amount of data using two simple primitives: Map and Reduce. Hadoop is an open source implementation of MapReduce. Dryad [10] is a more flexible approach that allows users to describe the computation as a data flow DAG, where each vertex is a computation unit and an edge indicates a data flow. These systems provide automatic parallelism and automatic fault tolerance and failure recovery mechanisms. Pig [12] and Hive [15] are two high-level descriptive languages on Hadoop for big data analytics. SCOPE [1] is a SQL-like descriptive language on Cosmos for big data analytics. These languages make big data analytics easy to perform for end users while maintaining the scalability and robustness of the underlining system.

One of the early references related to our work is that of random sampling for histogram construction by Chaudhuri et al [2] which may be seen as simple sampling scheme for determining a balanced range partition. Our analysis of the simple sampling scheme yields similar type of bounds for sufficient sample size, but allows for non-unique data values and arbitrary range partition.

The range partition problem may be seen as essentially equivalent to the problem of computing quantiles. For the quantile computation there are two versions of the problem (1) one-shot, where the output of the computation is required at the end of processing of an input stream and (2) continuous, where the output is required at each instant while passing through an input stream of data. Our work in on the one-shot version of the problem. Recent work on continuous quantile tracking [9, 19, 3] suggests that a practical algorithm may be derived for balanced range partition in m ranges with expected total communication cost $\tilde{O}(m\sqrt{k}/\epsilon)$. Note, however, that the algorithms that we consider are different as they require only a single round of communication between the coordinator node and sites.

An alternative method for computing quantiles is to use a histogram-based approach where each sites compute quantile summaries from their locally accessed data, which are then merged to create an aggregate quantile summary. By results of Greenwald and Khanna [7, 8], we know that using this approach we could compute a balanced range partition in m ranges with total communication cost $\tilde{O}(mk/\epsilon)$. A drawback of this approach is that it assumes that the largest frequency of an item locally at each site is at most 2ϵ where ϵ is the relative accuracy parameter. While this is a reasonable assumption for the case of unique data values, it may be a restrictive assumption in case of non-unique values and where the data items are distributed across sites such that

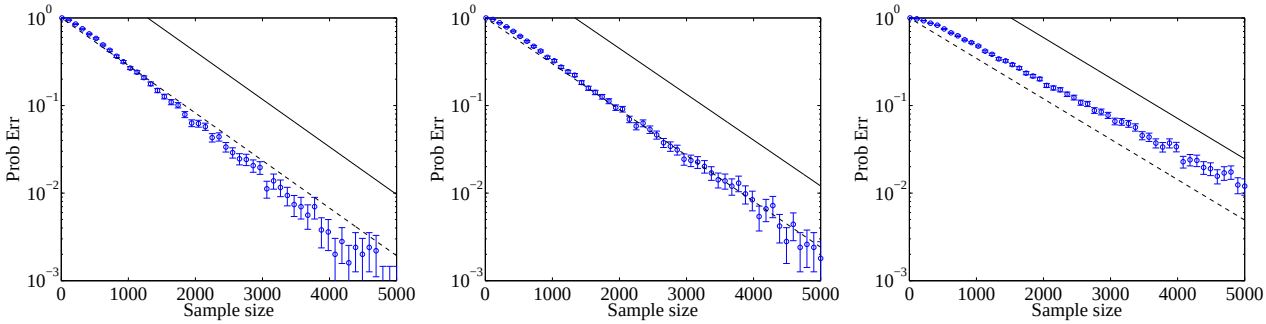

Figure 6: Probability of error vs. sample size: (left) $a = 1$, (middle) 2, (right) 4.

for some sites the local frequency of an item is large while the frequency of each item in the entire dataset is actually small. Another drawback is that the total communication cost would be larger whenever the number of sites k is sufficiently larger than $1/\epsilon$, which may well be the case in big data analytics environments. Finally, the approach based on quantile summaries is more computationally expensive as each site needs to create a quantile summary which requires sorting the locally accessed data.

7. CONCLUSION

We have analyzed several sampling based methods for range partition of data for big data analytics processing tasks. We introduced a simple weighted sampling scheme that accounts for data input sizes imbalance, which we demonstrated to be prevalent in real production data centers. Our analysis provides tight estimates for the sufficient sample size to guarantee a prescribed accuracy of range partition. The weighted sampling scheme presented in this paper has been used daily in the Cosmos environment, and has proved itself to provide a desired accuracy at a low cost.

There are several interesting directions for future work. First, one may investigate how to design practical algorithms that are communication cost efficient for various types of queries and allowing for multiple rounds of communication. Furthermore, one may study how to integrate with a query optimizer so that the most effective sampling method is chosen given the prior knowledge that the optimizer knows while the query is computed.

Acknowledgments

We would like to thank Wei Lin and Bing Shi for parts of basic system infrastructure that they developed and we used in our implementation and all the SCOPE team members for their support to this project.

8. REFERENCES

[1] R. Chaiken, B. Jenkins, P.-A. Larson, B. Ramsey, D. Shakib, S. Weaver, and J. Zhou. Scope: easy and efficient parallel processing of massive data sets. *Proc. of VLDB '08*, 1:1265–1276, August 2008.

[2] S. Chaudhuri, R. Motwani, and V. Narasayya. Random sampling for histogram construction: how much is enough? In *Proc. of ACM SIGMOD*, volume 27, pages 436–447, June 1998.

[3] G. Cormode, M. Garofalakis, S. Mutukrishnan, and R. Rastogi. Holistic aggregates in a networked world: Distributed tracking of approximate quantiles. In *Proc. of SIGMOD*, June 2005.

[4] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley-Interscience, 2 edition, 2006.

[5] J. Dean and S. Ghemawat. Mapreduce: simplified data processing on large clusters. In *Proc. of OSDI '04*, pages 10–10, 2004.

[6] J. Dean and S. Ghemawat. Mapreduce: simplified data processing on large clusters. *Commun. ACM*, 51:107–113, January 2008.

[7] M. B. Greenwald and S. Khanna. Space-efficient online computation of quantile summaries. In *Proc. of SIGMOD*, pages 58–66, 2001.

[8] M. B. Greenwald and S. Khanna. Power-conserving computation of order-statistics over sensor networks. In *Proc. of PODS*, 2004.

[9] Z. Huang, K. Yi, and Q. Zhang. Randomized algorithms for tracking distributed count, frequencies, and ranks. In *arXiv:1108.3413v1*, Aug 2011.

[10] M. Isard, M. Budiu, Y. Yu, A. Birrell, and D. Fetterly. Dryad: distributed data-parallel programs from sequential building blocks. In *Proc. of ACM SIGOPS/EuroSys '07*, pages 59–72.

[11] C. Nyberg, M. Shah, and N. Govindaraju. Softbenchmark. http://softbenchmark.org, 2011.

[12] C. Olston, B. Reed, U. Srivastava, R. Kumar, and A. Tomkins. Pig latin: a not-so-foreign language for data processing. In *Proc. of ACM SIGMOD '08*, pages 1099–1110, 2008.

[13] O. O'Malley. Terabyte sort on apache hadoop. http://sortbenchmark.org/YahooHadoop.pdf, May 2008.

[14] O. O'Malley and A. C. Murthy. Winning a 60 second dash with a yellow elephant. http://sortbenchmark.org/Yahoo2009.pdf, 2009.

[15] A. Thusoo, J. Sarma, N. Jain, Z. Shao, P. Chakka, N. Zhang, S. Antony, H. Liu, and R. Murthy. Hive - a petabyte scale data warehouse using hadoop. In *Proc. of IEEE ICDE*, pages 996 –1005, march 2010.

[16] D. Vasudevan and M. Vojnovic. Random sampling for data intensive computations. Technical Report MSR-TR-2009-08, Microsoft Research, April 2010.

[17] J. S. Vitter. Random sampling with a reservoir. *ACM Trans. on Mathematical Software*, 11:37–57, March 1985.

[18] M. Vojnovic, F. Xu, and J. Zhou. Sampling based range partition methods for big data analytics. Technical Report MSR-TR-2012-18, Microsoft Research, Februrary 2012.

[19] K. Yi and Q. Zhang. Optimal tracking of distributed heavy hitters and quantiles. In *Proc. of PODS*, June 2009.

APPENDIX

A. SIMPLE SAMPLING SCHEME

We consider range partition of the dataset Ω with the following relative range sizes $\vec{p} = (p_1, p_2, \ldots, p_m)$. Let S be a sample of t samples obtained by uniform random sampling from Ω without replacement. The boundaries $\vec{b} = (b_1, b_2, \ldots, b_{m-1})$ of the ranges are determined using the sample S such that range 1 contains items $a \in \Omega$ such that $a \leq b_1$, range i contains items $a \in \Omega$ such that $b_{i-1} < a \leq b_i$, for $i \in \{2, 3, \ldots, m-1\}$, and range m contains items $a \in \Omega$ such that $a > b_{m-1}$. Let μ_i be the fraction of samples in S that fall in range $i \in [m]$, and let $\vec{\mu} = (\mu_1, \mu_2, \ldots, \mu_m)$. The range boundaries $\vec{b}$ are determined such that the following inequalities hold

$$p_i - \phi \leq \mu_i \leq p_i + \phi, \text{ for every } i \in [m]$$

where recall ϕ is an upper bound on the frequency of the most frequent data item.

Let q_i denote the fraction of data items in Ω that fall in range $i \in [m]$. Notice that the sampling based scheme uses μ_i as an estimate for q_i, $i \in [m]$. We shall use the notation $T_i = (b_{i-1}, b_i, \mu_i)$, for $i \in [m]$.

We define the cumulative distribution function F associated with the dataset Ω as follows

$$F(a) = \frac{|\{b \in \Omega \mid b \leq a\}|}{|\Omega|}, \text{ for } a \in \Omega.$$

Recall that the range partition is said to be ϵ-accurate with probability at least $1 - \delta$, if the following holds:

$$\xi_t := \mathbf{P}[\cup_{i=1}^m \{q_i > (1+\epsilon)p_i\} \mid S] \leq \delta. \quad (1)$$

Notice that we have

$$\begin{aligned} & \mathbf{P}[\cup_{i=1}^m \{q_i > (1+\epsilon)p_i\} \mid S] \\ = \; & \mathbf{P}[\cup_{i=1}^m \{q_i > (1+\epsilon)p_i\} \mid (\vec{b}, \vec{\mu})] \\ \leq \; & \sum_{i=1}^m \mathbf{P}[q_i > (1+\epsilon)p_i \mid T_i] \end{aligned}$$

where the first equality holds because under uniform random sampling, given a set of samples, the probability of the given event depends only on the width of the range and the fraction of samples in S that fall in the given range, and the last inequality is by the union bound.

Note under T_i, q_i is a random variable that takes values from the set $Q(T_i)$ defined as follows

$$Q(T_i) = \{\frac{t\mu_i}{n}, \frac{t\mu_i + 1}{n}, \ldots, F(b_i) - F(b_{i-1})\}.$$

Let us define $f_{n,t}(x, y)$ as the probability that a uniform random sample of size t, drawn without replacement from a pool of n items, draws ty items from a designated subset of nx items. This probability is equal to

$$f_{n,t}(x, y) = \frac{\binom{nx}{ty}\binom{n(1-x)}{t(1-y)}}{\binom{n}{t}}. \quad (2)$$

Note that

$$\begin{aligned} & \mathbf{P}[q_i > (1+\epsilon)p_i \mid T_i] \\ = \; & \sum_{q_i \in Q(T_i): q_i \geq (1+\epsilon)p_i} f_{n,t}(q_i, \mu_i) \\ \leq \; & \psi(T_i) \cdot f_{n,t}(q_i^*, \mu_i). \end{aligned}$$

where q_i^* is such that $q_i^* \geq (1+\epsilon)p_i$ and

$$f_{n,t}(q_i^*, \mu_i) = \max_{q \in [0,1]: q \geq (1+\epsilon)p_i} f_{n,t}(q, \mu_i)$$

and

$$\psi(T_i) = \frac{\sum_{q \in Q(T_i): q \geq (1+\epsilon)p_i} f_{n,t}(q, \mu_i)}{f_{n,t}(q_i^*, \mu_i)}.$$

From (2), using a simple calculus, we observe $\log(f_{n,t}(x, y)) = -\varphi_{n,t}(x, y)$ with

$$\varphi_{n,t}(x, y) = n\left[\frac{t}{n} D(y||x) + \left(1 - \frac{t}{n}\right) D\left(\frac{x - \frac{t}{n}y}{1 - \frac{t}{n}} || x\right)\right]$$

where $D(x||y)$ is the Kullback-Lieber divergence between two Bernoulli distributions with parameters x and y in $[0, 1]$.

Let us define $\mathcal{A}_i = \{(q, \mu) \in [0,1]^2 : q > (1+\epsilon)p_i, \mu \leq p_i + \phi\}$. We then observe

$$\begin{aligned} & \log \mathbf{P}[q_i > (1+\epsilon)p_i \mid T_i] \\ \leq \; & \log(\psi(T_i)) - \min_{(\mu_i, q_i) \in \mathcal{A}_i} \varphi_{n,t}(\mu_i, q_i). \end{aligned}$$

The inequality (1) holds provided that for every $i \in [m]$ and $(q_i, \mu_i) \in \mathcal{A}_i$, the following holds

$$\varphi_{n,t}(\mu_i, q_i) \geq \log\left(\frac{1}{\delta}\right) + \log(m) + \max_{j \in [m]} \log(\psi(T_j)).$$

Since

$$\varphi_{n,t}(\mu_i, q_i) = -\log(f_{n,t}(\mu_i, q_i)) \geq t D(q_i||\mu_i) \quad (3)$$

it suffices that

$$\begin{aligned} t \; & \geq \; \max_{i \in [m]} \max_{(q_i, \mu_i) \in \mathcal{A}_i} \frac{1}{D(q_i||\mu_i)} \cdot \varrho \\ & \geq \; \frac{1}{\min_{i \in [m]} D((1+\epsilon)p_i||p_i + \phi)} \cdot \varrho \\ & = \; \max_{i \in [m]} \frac{2(p_i + \phi)(1 - p_i - \phi)}{(\epsilon p_i - \phi)^2} [1 + O(\epsilon)] \cdot \varrho \\ & = \; \frac{2(\pi + \phi)(1 - \pi - \phi)}{(\epsilon\pi - \phi)^2} [1 + O(\epsilon)] \cdot \varrho \end{aligned}$$

where $\varrho = \log\left(\frac{1}{\delta}\right) + \log(m) + \max_{i \in [m]} \log(\psi(T_i))$.

It remains only to note that for every given T_i, $\log(\psi(T_i)) \leq \log(|Q(T_i)|) \leq \log(n)$, for every $i \in [m]$.

A.1 A Tighter Sufficient Sample Size

In the preceding section, we used a crude bound for the term $\max_{i \in [m]} \log(\psi(T_i))$. We now describe how one may derive a tighter bound which may be of interest in applications.

For every given value $\theta \geq 0$, let us define the set $B_\theta(T_i)$ as follows

$$B_\theta(T_i) = \{q \in Q(T_i) \mid D(q||\mu_i) - D(q_i^*||\mu_i) \geq \theta\}.$$

Then, note

$$\psi(T_i) \leq |Q(T_i) \setminus B_\theta(T_i)| + |B_\theta(T_i)|e^{-\theta t}.$$

Now, by convexity of the Kullback-Lieber divergence, note that we have

$$D(q||\mu_i) - D(q_i^*||\mu_i) \geq \alpha_i(q - q_i^*)$$

where

$$\alpha_i = \log\left(\frac{q_i^*(1-\mu_i)}{(1-q_i^*)\mu_i}\right).$$

We will use this for $q_i^* = (1+\epsilon)p_i$ and $\mu_i = p_i + \phi$ as these are the values that minimize the error exponent (see preceding section).

We observe that for the set defined as $\tilde{B}_\theta(T_i) := \{q \in Q(T_i) \mid \alpha_i(q - q_i^*) \geq \theta\}$, we have $\tilde{B}_\theta(T_i) \subseteq B_\theta(T_i)$. Thus,

$$\psi(T_i) \leq |Q(T_i) \setminus \tilde{B}_\theta(T_i)| + |\tilde{B}_\theta(T_i)|e^{-\theta t}.$$

Furthermore, note that

$$|Q(T_i) \setminus \tilde{B}_\theta(T_i)| \leq \frac{\theta}{\alpha_i} n \text{ and } |\tilde{B}_\theta(T_i)| \leq n.$$

Therefore,

$$\psi(T_i) \leq n(\frac{\theta}{\alpha_i} + e^{-\theta t}).$$

By minimizing the right-hand side over $\theta \geq 0$, we obtain

$$\psi(T_i) \leq n\frac{\log(\alpha_i t) + 1}{\alpha_i t}.$$

Therefore,

$$\max_{i\in[m]} \log(\psi(T_i)) \leq \log(n) + \max_{i\in[m]} \log\left(\frac{\log(\alpha_i t) + 1}{\alpha_i t}\right).$$

Combining this with the analysis in the preceding section, we obtain the following sufficient condition for the sample size to guarantee ϵ-accuracy with probabity at least $1 - \delta$,

$$Dt \geq \log(1/\delta) + \log(m) + \log(n) - \min_{i\in[m]} \log\left(\frac{\alpha_i t}{\log(\alpha_i t) + 1}\right)$$

where $D = \min_{i\in[m]} D((1+\epsilon)p_i||p_i + \phi)$.

B. WEIGHTED SAMPLING SCHEME

We denote with ν_j the portion of data items that reside at site j, i.e. $\nu_j = |\Omega_j|/|\Omega|$, for $j \in [k]$, and, without of loss of generality, we assume that sites are enumerated in decreasing order of μ_j, i.e.

$$1 \geq \nu_1 \geq \nu_2 \geq \cdots \geq \nu_k > 0.$$

We consider the range partition for given relative range sizes $\vec{p} = (p_1, p_2, \ldots, p_m)$. Let $R_i(\vec{b})$ denote the set of data items from Ω that fall in range i according to the range boundaries $\vec{b}$, which are computed using the sample of data items $\tilde{S}$ that are input to the weighted sampling scheme. Let q_i be the fraction of data items from Ω that fall in range i. Let $q_{i,j}$ denote the fraction of data items in Ω that are in range i and reside at site j, i.e. $q_{i,j} = |\{a \in \Omega \mid a \in \Omega_j, a \in R_i\}|/|\Omega|$, for $i \in [m]$ and $j \in [k]$. Similarly, let $\mu_{i,j}$ denote the fraction of samples in $\tilde{S}$ that are in range i and are from site j, i.e. $\mu_{i,j} = |\{a \in \tilde{S} \mid a \in R_i, a \in \Omega_j\}|/|\tilde{S}|$, for $i \in [m]$ and $j \in [k]$. Notice that under weighted sampling scheme, the range boundaries are chosen such that the following holds

$$p_i - \phi \leq \sum_{j=1}^{k} \nu_j \mu_{i,j} \leq p_i + \phi, \text{ for every } i \in [m].$$

Now, the error probability ξ_t is given as follows:

$$\xi_t = \mathbf{P}[\cup_{i=1}^m \{q_i > (1+\epsilon)p_i\} \mid S].$$

Using the union bound, we have

$$\begin{aligned} \xi_t &\leq \sum_{i=1}^{m} \mathbf{P}[q_i > (1+\epsilon)p_i \mid S] \\ &\leq m \max_{i\in[m]} \mathbf{P}[q_i > (1+\epsilon)p_i \mid S]. \end{aligned} \tag{4}$$

We fix an arbitrary range i and consider the probability of the error event $\{q_i > (1+\epsilon)p_i\}$. Let $\vec{\mu}_i = (\mu_{i,1}, \mu_{i,2}, \ldots, \mu_{i,m})$, for every $i \in [m]$.

Note that the following holds for every range $i \in [m]$,

$$\mathbf{P}[q_i > (1+\epsilon)p_i \mid S] = \mathbf{P}[\sum_{j=1}^{k} \nu_j q_{i,j} > (1+\epsilon)p_i \mid T_i]$$

where we define $T_i = (b_{i-1}, b_i, \vec{\mu}_i)$.

Let us denote with F_j the cumulative distribution function of data items that reside at site j, which is defined as follows

$$F_j(a) = \frac{|\{b \in \Omega_j \mid b \leq a\}|}{|\Omega_j|}, \ a \in \Omega_j, \ j \in [k].$$

Notice that under T_i, $q_{i,j}$ is a random variable that takes values on the set $Q_j(T_i)$ defined by

$$Q_j(T_i) = \{\frac{t\mu_{i,j}}{n}, \frac{t\mu_{i,j}+1}{n}, \ldots, \frac{n_j(F_j(b_i) - F_j(b_{i-1}))}{n}\}.$$

Let $Q(T_i) = \times_{j\in[k]} Q_j(T_i)$. We observe

$$\begin{aligned} &\mathbf{P}[\sum_{j=1}^{k} \nu_j q_{i,j} > (1+\epsilon)p_i \mid T_i] \\ &= \sum_{\vec{q}_i \in Q(T_i): \sum_{j=1}^k \nu_j q_{i,j} > (1+\epsilon)p_i} \ \prod_{j\in[k]} f_{n,t}(q_{i,j}, \mu_{i,j}) \end{aligned}$$

where $f_{n,t}(x,y) = \frac{\binom{nx}{ty}\binom{n(1-x)}{t(1-y)}}{\binom{n}{t}}$. From (3), we have

$$\prod_{j\in[k]} f_{n,t}(q_{i,j}, \mu_{i,j}) \leq \exp(-t \sum_{j\in[k]} D(q_{i,j}||\mu_{i,j})).$$

Therefore,

$$\begin{aligned} &\mathbf{P}[\sum_{j=1}^{k} \nu_j q_{i,j} > (1+\epsilon)p_i \mid T_i] &(5)\\ &\leq \psi(T_i) \cdot \exp(-t \sum_{j\in[k]} D(q_{i,j}^*||\mu_{i,j})) &(6) \end{aligned}$$

where $\vec{q}_i^*$ maximizes $\exp(-t\sum_{j\in[k]} D(q_{i,j}^*||\mu_{i,j}))$ over $\vec{q}_i \in [0,1]^k$ such that $\sum_{j=1}^k \nu_j q_{i,j} > (1+\epsilon)p_i$, and

$$\psi(T_i) = \sum_{\vec{q}_i \in \tilde{Q}(T_i)} e^{-t\sum_{j\in[k]}[D(q_{i,j}||\mu_{i,j}) - D(q_{i,j}^*||\mu_{i,j})]}$$

where $\tilde{Q}(T_i) = \{\vec{q}_i \in Q(T_i) \mid \sum_{j=1}^k \nu_j q_{i,j} > (1+\epsilon)p_i\}$.

B.1 Analysis of the Error Exponent

Let D_i be the optimal value of the following optimization problem:

$$\begin{aligned} \text{minimize} \quad & \sum_{j=1}^{k} D(q_{i,j}||\mu_{i,j}) \\ \text{over} \quad & 0 \le q_{i,j} \le 1,\ j \in [k] \\ & 0 \le \mu_{i,j} \le 1,\ j \in [k] \\ \text{subject to} \quad & \sum_{j=1}^{k} \nu_j q_{i,j} \ge (1+\epsilon) p_i \\ & \sum_{j=1}^{k} \nu_j \mu_{i,j} \le p_i + \phi. \end{aligned}$$

This is a convex optimization problem and has a unique solution. The following result provides a tight characterization of the error exponent and follows by Sanov's theorem on the deviation of empirical measures [4].

THEOREM 3. *Assume $\phi < \pi\epsilon$ and let $\Delta = \min_i D_i$. The error probability ξ_t satisfies*

$$-\frac{1}{t}\log(\xi_t) \sim \Delta,\ \textit{large } t.$$

We will next provide a tight characterization of the error exponent Δ and on the way, we will characterize the optimal solution $\vec{q}_i$ and $\vec{\mu}_i$. This will enable us to identify what the worst-case distribution of data over sites is with respect to the error exponent.

Since it suffices to consider an arbitrary range $i \in [m]$, in order to simplify the notation, we will omit the subscript i and, thus, write q_j and μ_j in lieu of $q_{i,j}$ and $\mu_{i,j}$, respectively. Furthermore, we simplify the notation by denoting $A = p_i(1+\epsilon)$ and $B = p_i + \phi$. With our new notation, the optimization problem can be rewritten as follows

$$\begin{aligned} \text{minimize} \quad & \sum_{j=1}^{k} D(q_j||\mu_j) \\ \text{over} \quad & 0 \le q_j \le 1,\ j \in [k] \\ & 0 \le \mu_j \le 1,\ j \in [k] \\ \text{subject to} \quad & \sum_{j=1}^{k} \nu_j q_j \ge A \qquad (7) \\ & \sum_{j=1}^{k} \nu_j \mu_j \le B. \qquad (8) \end{aligned}$$

We solve this problem by the method of Lagrange multipliers. Let $\alpha \ge 0$ and $\beta \ge 0$ be the Lagrange multipliers associated with the constraints (7) and (8), respectively, and let $\gamma_j \ge 0$ and $\delta_j \ge 0$ (resp. $\hat{\gamma}_j \ge 0$ and $\hat{\delta}_j \ge 0$) denote the Lagrange multipliers associated with constraints $0 \le q_j$ and $q_j \le 1$ (resp. $0 \le \mu_j$ and $\mu_j \le 1$), respectively.

The optimal solution is such that for some positive valued α, β, γ_j, δ_j, $\hat{\gamma}_j$ and $\hat{\delta}_j$, $j \in [k]$, the following holds for every $j \in [k]$,

$$\begin{aligned} \frac{\mu_j - q_j}{\mu_j(1-\mu_j)} &= \alpha\nu_j + \gamma_j - \delta_j \qquad (9) \\ \frac{q_j}{1-q_j} &= \frac{\mu_j}{1-\mu_j} 2^{-\beta\nu_j + \hat{\gamma}_j - \hat{\delta}_j}. \qquad (10) \end{aligned}$$

In addition, the following complementarity slackness conditions hold, for every $j \in [k]$,

$$\begin{aligned} \alpha(\sum_{j=1}^{k} \nu_j q_j - A) &= 0 \\ \beta(B - \sum_{j=1}^{k} \nu_j \mu_j) &= 0 \\ \gamma_j q_j &= 0 \\ \hat{\gamma}_j \mu_j &= 0 \\ \delta_j(1-q_j) &= 0 \\ \hat{\delta}_j(1-\mu_j) &= 0. \end{aligned}$$

The following lemma provides a characterization of the optimal solution and is easy to conclude from the conditions above, so the proof is omitted.

LEMMA 1. *For every $j \in [k]$, $q_j = 0$ if and only if $\mu_j = 0$ and similarly $\mu_j = 1$ if and only if $\mu_j = 1$. Otherwise, if $0 < q_j < 1$, then*

$$\frac{\mu_j - q_j}{\mu_j(1-\mu_j)} = \alpha\nu_j \textit{ and } \frac{q_j}{1-q_j} = \frac{\mu_j}{1-\mu_j} 2^{-\beta\nu_j}.$$

From (9) and (10), we have that for every $j \in [k]$ such that $0 < q_j < 1$, it holds

$$\mu_j = 1 - \frac{1}{\alpha\nu_j} + \frac{1}{2^{\beta\nu_j} - 1}$$

and

$$q_j = \alpha\nu_j \frac{2^{\beta\nu_j}}{(2^{\beta\nu_j}-1)^2} - \frac{1}{2^{\beta\nu_j}-1}.$$

We next note another important property of the optimum solution.

LEMMA 2. *There exists an integer $\kappa \in [k]$ such that and $q_j, \mu_j > 0$, for $j \in [\kappa]$, and $q_j = \mu_j = 0$, otherwise.*

The lemma tells us that the worst case is to have a strictly positive fraction of the range in a site only if the fraction of data items residing at the site is large enough.

PROOF OF THE LEMMA. From (9) and (10), note that $q_j > 0$ iff

$$\alpha\nu_j > 1 - 2^{-\beta\nu_j}.$$

It is easily noted that this inequality holds iff $\nu_j > \theta$, for some $\theta > 0$. Thus, if the inequality holds for some ν_i, then it holds for every $\nu_j \ge \nu_i$. This completes the proof. □

We next characterize the optimal values of the Lagrange multipliers α and β and the value of the threshold κ. It is easy to establish that conditions $\sum_{j=1}^{k} \nu_j \mu_j = A$ and $\sum_{j=1}^{k} \nu_j q_j = B$ can be, respectively, written as follows:

$$\frac{1}{\kappa}\sum_{j=1}^{\kappa} \frac{\nu_j 2^{\beta\nu_j}}{2^{\alpha\nu_j}-1} - \frac{1}{\alpha} = \frac{A}{\kappa} \qquad (11)$$

and

$$\alpha\frac{1}{\kappa}\sum_{j=1}^{\kappa} \frac{\nu_j^2 2^{\beta\nu_j}}{(2^{\beta\nu_j}-1)^2} - \frac{1}{\kappa}\sum_{j=1}^{\kappa} \frac{\nu_j}{2^{\beta\nu_j}-1} = \frac{B}{\kappa}. \qquad (12)$$

LEMMA 3. *The threshold κ is equal to the largest integer $i \in [k]$ such that $q_i > 0$, i.e. the largest integer $i \in [k]$ such that*

$$\frac{2^{\beta\nu_i}}{2^{\beta\nu_i}-1}\nu_i > \frac{1}{i}\left(\sum_{j=1}^{i}\frac{2^{\beta\nu_j}}{2^{\beta\nu_j}-1}\nu_j - A\right).$$

PROOF OF THE LEMMA. The lemma follows by Lemma 2 and noting that $q_i > 0$ is equivalent to $1-\alpha\nu_i < 2^{-\beta\nu_i}$ and then using the identities (11) and (12) to derive the assertion above. □

The above characterizations of the optimal solution enables us to identify the optimal error exponent which is stated in the following theorem.

THEOREM 4. *Given dual optimal values $\alpha > 0$ and $\beta > 0$, the optimal error exponent is given by*

$$\Delta = \frac{\kappa}{k}\left(\frac{1}{\kappa}\sum_{j=1}^{\kappa}\log\left(\frac{\alpha\nu_j}{2^{\beta\nu_j}-1}\right) + \frac{\sum_{j=1}^{\kappa}\nu_j - B}{\kappa}\beta\right).$$

The last theorem gives us an explicit characterization of the error exponent as a function of the dual variables α and β. We provide a more explicit characterization of the error exponent that holds for the case of small ϵ parameter in the following.

THEOREM 5. *Assume $\phi < \pi\epsilon$. The error exponent satisfies*

$$\Delta = \frac{\kappa}{k}\frac{\epsilon^2\pi^2(1-\phi/(\pi\epsilon))^2}{2\pi(\sum_{j=1}^{\kappa}\nu_j - \pi) + \frac{1}{2}\kappa^2\sigma_\kappa^2} + o(\epsilon^2).$$

The theorem reveals that for the case of small ϵ parameter, the key parameter for the error exponent is the variance of the distribution of the number of data items over sites. The value of the parameter κ is given by Lemma 2 and is noteworthy that for the case of small β, κ is the largest $i \in [k]$ such that the following holds

$$\nu_i > \frac{1}{i}\left(\sum_{j=1}^{i}\nu_j - 2A\right) + O(\beta).$$

Indeed, this follows by combining (2) with the following fact $\frac{\nu_j\beta 2^{\beta\nu_j}}{2^{\beta\nu_j}-1} = 1 + \frac{1}{2}\beta\nu_j + O(\beta^2)$.

Remark For the case of perfectly balanced number of data items per site, i.e. $\nu_j = 1/k$, for each site $j \in [k]$, we have $\kappa = k$, $\sigma_\kappa^2 = 0$ and

$$\Delta = \frac{\epsilon^2\pi(1-\phi/(\pi\epsilon))^2}{2(1-\pi)} + o(\epsilon^2)$$

which corresponds to the error exponent of sampling from a dataset residing in a single site or multiple sites but where the sample is taken by sampling without replacement across the entire dataset (i.e. our simple sampling scheme).

PROOF OF THEOREM 5. We make use of the following property of the Kullback-Lieber divergence, for every $j \in [k]$ such that $0 < q_j < 1$,

$$D(q_j||\mu_j) = \frac{(q_j-\mu_j)^2}{2\mu_j(1-\mu_j)} + O((q_j-\mu_j)^3).$$

Combining with (9), we have

$$D(q_j||\mu_j) = \frac{\alpha}{2}\nu_j(q_j-\mu_j) + O(\alpha^3).$$

We observe

$$\frac{1}{k}\sum_{j=1}^{\kappa}D(q_j||\mu_j) = \frac{\alpha}{2k}(A-B) + O(\alpha^3). \tag{13}$$

LEMMA 4. *The following equality holds, for some $v \geq 0$,*

$$\alpha = \kappa\frac{A-B}{A(\sum_{j=1}^{\kappa}\nu_j - A) + \kappa^2 v}.$$

Furthermore, for the small α case, it holds $v = \frac{1}{4}\sigma_\kappa^2 + o(1)$ where

$$\sigma_\kappa^2 = \frac{1}{\kappa}\sum_{j=1}^{\kappa}\nu_j^2 - (\frac{1}{\kappa}\sum_{j=1}^{\kappa}\nu_j)^2.$$

The lemma is established by the following arguments. Subtracting (7) and (8), we obtain

$$\alpha\left[\left(\sum_{j=1}^{\kappa}\nu_j + 2\sum_{j=1}^{\kappa}\frac{\nu_j}{2^{\beta\nu_j}-1}\right)\frac{1}{\alpha} - \right.$$
$$\left. -\frac{\kappa}{\alpha^2} - \sum_{j=1}^{\kappa}\frac{\nu_j^2 2^{\beta\nu_j}}{(2^{\beta\nu_j}-1)^2}\right] = A - B.$$

By plugging α and some elementary calculus, it can be showed that the last equality is equivalent to

$$\frac{\alpha}{\kappa}\left[a(\sum_{j=1}^{\kappa}\nu_j - a) + \kappa^2 v(\nu,\alpha,\beta)\right] = A - B$$

where $v(\nu,\alpha,\beta) = C_1 - C_2$ and

$$C_1 = \frac{1}{\kappa}\sum_{j=1}^{\kappa}\nu_j\frac{1}{\kappa}\sum_{j=1}^{\kappa}\frac{\nu_j}{2^{\beta\nu_j}-1} - \frac{1}{\kappa}\sum_{j=1}^{\kappa}\frac{\nu_j^2}{2^{\beta\nu_j}-1}$$

$$C_2 = \frac{1}{\kappa}\sum_{j=1}^{\kappa}\left(\frac{\nu_j}{2^{\beta\nu_j}-1}\right)^2 - \left(\frac{1}{\kappa}\sum_{j=1}^{\kappa}\frac{\nu_j}{2^{\beta\nu_j}-1}\right)^2.$$

It can be readily checked that $C_1 = \frac{1}{2}\sigma_\kappa^2 + O(\beta)$ and $C_2 = \frac{1}{4}\sigma_\kappa^2 + O(\beta)$ which completes the proof of the lemma.

Using the last lemma and (13), we obtain

$$\frac{1}{k}\sum_{j=1}^{\kappa}D(q_j||\mu_j) = \frac{\kappa}{2k}\frac{(A-B)^2}{A(\sum_{j=1}^{\kappa}\nu_j - A) + \kappa^2 v} + O(\alpha^3).$$

This completes the proof of Theorem 5. □

While the previous characterization provides a tight estimate of the error exponent for small value of parameter ϵ, it depends on the threshold parameter κ which requires knowing the distribution of the number of data items over sites. It is of interest to obtain an estimate of the error exponent that requires fewer parameters about the underlying distribution of data over sites, which we present in the following corollary.

COROLLARY 3. *Assume $\phi < \pi\epsilon$. Then the error exponent satisfies*

$$\Delta \geq \frac{\epsilon^2\pi(1-\phi/(\pi\epsilon))^2}{2\pi(1-\pi) + \frac{1}{2}k^2\sigma_k^2} + o(\epsilon^2).$$

PROOF OF COROLLARY 3. It suffices to show that for every set of values $\nu_1 \geq \nu_2 \geq \ldots \geq \nu_k > 0$ and $0 \leq a \leq 1$, it holds

$$\frac{\kappa}{2k}\frac{(A-B)^2}{a(\sum_{j=1}^{\kappa}\nu_j - A) + \frac{1}{4}\kappa^2\sigma_\kappa^2} \geq \frac{1}{2}\frac{(A-B)^2}{a(1-a)+\frac{1}{4}k^2\sigma_k^2}.$$

This relation is indeed equivalent to

$$\frac{1}{\kappa}a(\sum_{j=1}^{\kappa}\nu_j - A) + \frac{1}{4}\kappa\sigma_\kappa^2 \leq \frac{1}{k}A(1-A) + \frac{1}{4}k\sigma_k^2.$$

For the case $\kappa = k$, we have that the last relation holds with equality. It thus remains to consider the case $\kappa < k$ which we do in the following by induction over $i = [k-\kappa]$.

Base case $i = 1$. By definition of κ and the assumption $\kappa < k$, we have

$$\nu_k \leq \frac{1}{k}(1-2A)$$

as $k = \kappa$, otherwise. We need to show that

$$\frac{1}{k-1}A(\sum_{j=1}^{k-1}\nu_j - A) + \frac{1}{4}(k-1)\sigma_{k-1}^2 \leq \frac{1}{k}A(1-A) + \frac{1}{4}k\sigma_k^2.$$

After some elementary algebra, we can show that this is equivalent to:

$$4A(\frac{1-A}{k} - \nu_k) + k\nu_k(\frac{2}{k} - \nu_k) - \frac{1}{k} \leq 0.$$

The left-hand side is increasing with ν_k over $[0,(1-2A)/k]$, hence the inequality holds if it holds for $\nu_k = (1-2A)/k$. Now, it is easy to check that for $\nu_k = (1-2A)/k$, the inequality holds with equality, which proves the base case.

Induction step $1 < i < k-\kappa$. We will show that

$$\begin{array}{l}\frac{1}{k-i-1}A(\sum_{j=1}^{k-i-1}\nu_j - A) + \frac{1}{4}(k-i-1)\sigma_{k-i-1}^2 \\ \leq \frac{1}{k-i}A(\sum_{j=1}^{k-i}\nu_j - A) + \frac{1}{4}(k-i)\sigma_{k-i}^2.\end{array} \quad (14)$$

By the induction hypothesis, the right-hand side is less than or equal to $\frac{1}{k}A(1-A)+\frac{1}{4}\sigma_k^2$, and thus provided that the last above inequality holds, we have that our desired condition holds for $k-i-1$. Notice that

$$\nu_{k-i} \leq \frac{1}{k-i}\left(\sum_{j=1}^{k-i}\nu_j - 2A\right)$$

as, otherwise, we will have $k-i = \kappa$.

By similar steps as for the base case, we can show that (14) is equivalent to

$$4A(u-A-(m-i)\nu_{k-i}) - (k-i)^2\nu_{k-i}^2 + 2(k-i)u\nu_{k-i} - u^2 \leq 0$$

where $u = \sum_{j=1}^{k-i}\nu_j$. The left-hand side is increasing with ν_{k-i} over $[0, \frac{u-2A}{k-i}]$ and achieves 0 for $\nu_{k-i} = \frac{u-2A}{k-i}$. This completes the proof. □

B.2 Proof of Theorem 2

From (6) and the notation in the preceding section, we have for every $i \in [k]$,

$$\mathbf{P}[q_i > (1+\epsilon)p_i \mid T_i] \leq \psi(T_i)\cdot\exp(-t\Delta).$$

Therefore, combining with (4), we have that a sufficient sample size is

$$t \geq \frac{1}{\Delta}[\log(1/\delta) + \log(m) + \max_{i\in[m]}\log(\psi(T_i))]$$

Notice that, for every $i \in [m]$,

$$\begin{array}{rcl}\log(\psi(T_i)) & \leq & \log(|Q(T_i)|) \\ & = & \log(\prod_{j\in[k]}|Q_j(T_i)|) \\ & \leq & \log(\prod_{j\in[k]} n_j) \\ & \leq & k\log(n).\end{array}$$

The last two relations, combined with Corollary 3, provide a proof of Theorem 2.

B.3 A Normal Approximation

The result in Theorem 2 provides a sufficient sample size to guarantee ϵ-approximate range partition with probability at least $1-\delta$. This sufficient sample size is derived by using a large deviations approach which results in the factor $\varrho(1/\delta, m, n, k)$ that is logarithmic in $1/\delta$ and the number of partitions m, and is linear in $k\log(n)$. We will now show that using a normal approximation we obtain the same remaining factor as in Theorem 2 but this normal approximation suggests a smaller factor ϱ that is independent of k.

We observe that the following inequalities hold, for every range $i \in [m]$,

$$\begin{array}{rl} & \mathbf{P}[\sum_{j=1}^{k}\nu_j q_{i,j} > (1+\epsilon)p_i \mid S] \\ = & \mathbf{P}[\sum_{j=1}^{k}\nu_j q_{i,j} - \sum_{j=1}^{k}\nu_j\mu_{i,j} > (1+\epsilon)p_i - \sum_{j=1}^{k}\nu_j\mu_{i,j} \mid S] \\ \leq & \mathbf{P}[\sum_{j=1}^{k}\nu_j q_{i,j} - \sum_{j=1}^{k}\nu_j\mu_{i,j} > \epsilon p_i - \phi \mid S]\end{array}$$

where the last inequality holds because $\sum_{j=1}^{k}\nu_j\mu_{i,j} \leq p_i + \phi$.

Now, we may use the normal approximation for the random variable $\sum_{j=1}^{k}\nu_j q_{i,j} - \sum_{j=1}^{k}\nu_j\mu_{i,j}$ with mean zero and variance $\sum_{j=1}^{k}\nu_j^2\mu_{i,j}(1-\mu_{i,j})/(t/k)$. The above bound on the probability of error thus may be approximated by

$$\bar{\Phi}\left(\frac{\epsilon p_i - \phi}{\sqrt{\sum_{j=1}^{k}\nu_j^2\mu_{i,j}(1-\mu_{i,j})}}\sqrt{t/k}\right)$$

where $\Phi(x)$ is the cumulative distribution function of a normal random variable and $\bar{\Phi}(x) = 1-\Phi(x)$. Now let $M = \{(\mu_{i,1},\ldots,\mu_{i,k}) \in [0,1]^k \mid \sum_{j=1}^{k}\nu_j\mu_{i,j} = 1\}$ and note that

for every $\vec{\mu}_i = (\mu_{i,1}, \ldots, \mu_{i,k}) \in M$,

$$\begin{aligned}
& \bar{\Phi}\left(\frac{\epsilon p_i - \phi}{\sqrt{\sum_{j=1}^{k} \nu_j^2 \mu_{i,j}(1-\mu_{i,j})}}\sqrt{t/k}\right) \\
\leq \; & \max_{\vec{\mu}_i \in M} \bar{\Phi}\left(\frac{\epsilon p_i - \phi}{\sqrt{\sum_{j=1}^{k} \nu_j^2 \mu_{i,j}(1-\mu_{i,j})}}\sqrt{t/k}\right) \\
= \; & \bar{\Phi}\left(\frac{\epsilon p_i - \phi}{\sqrt{\max_{\vec{\mu}_i \in M} \sum_{j=1}^{k} \nu_j^2 \mu_{i,j}(1-\mu_{i,j})}}\sqrt{t/k}\right) \\
= \; & \bar{\Phi}\left(\frac{\epsilon p_i - \phi}{\sqrt{(p_i+\phi)(1-(p_i+\phi)) + \alpha(\vec{n})}}\sqrt{t}\right)
\end{aligned}$$

Requiring that the last expression is less than or equal to δ/m yields

$$t \geq \frac{(p_i+\phi)(1-(p_i+\phi)) + \alpha(\vec{n})}{(\epsilon p_i - \phi)^2} \bar{\Phi}^{-1}(\delta/m)^2.$$

Since $\bar{\Phi}(x) \leq \exp(-\frac{x^2}{2})$, for large enough x ($x \geq 1/\sqrt{2\pi}$ suffices), we have $\bar{\Phi}^{-1}(x)^2 \leq 2\log(1/x)$. Using this yields a sufficient condition for the last above inequality, which reads as

$$t \geq \frac{2[(p_i+\phi)(1-(p_i+\phi)) + \alpha(\vec{n})]}{(\epsilon p_i - \phi)^2} \log(m/\delta).$$