Author: Bashar Al-Rawi - basharal@

Motivation

Current histogram metric type has inherent inefficiencies due to the way it stores and interprets bucket counts leading to overhead in series cardinality no matter what the metric distribution looks like since it requires providing all bucket ranges for each sample at scrape time. In many cases, it's not needed to do so, but due to the nature of storing cumulative values and relying on "le" labels for upper bounds, it's not possible to optimize histogram_quantile queries. In this document, we would like to propose a new format that is more efficient without breaking changes to existing behavior.

Goals

- Using less buckets at scrape time.
- Using histogram_quantile transparently without any breaking changes.
- Backward compatibility. Existing metrics should work as is without any behavioral changes.
- No regression in performance. New format should be at least as good as the existing one in all cases.

New Format

Instead of creating new metrics with xxx_bucket{"**le**"="BB"}=ZZ, create metrics in the following format xxx_bucket{"**range**"="AA,BB"}=CC where these placeholders are defined as follows:
BB: upper-bound for the bucket (inclusive)
AA: lower-bound (exclusive)
ZZ: cumulative count for values in this bucket and all buckets whose upper bound is < BB.
CC: count (not cumulative) for values in this bucket range only.

histogram_quantile Changes

Using the new format, we can compute quantiles similar to the existing method by using each bucket lower-bound and upper-bounds. By having a lower-bound and counts per bucket (not cumulative counts), we can allow gaps in bucket ranges without losing accuracy. Imagine the following input:

```
xxx_bucket{“range”=”0,1”}=4
xxx_bucket{“range”=”5,10”}=8
xxx_bucket{“range”=”10,+Inf”}=10
```

If we didn’t have **5** as a lower-bound in the previous in the second sample and instead used the previous format via **le**, then we would inherently lose accuracy during linear interpolation since the range will be from (1-10] unless the user also provided another sample to cover the gap without increasing the cumulative count, which is what’s required today to fill the gaps (All ranges must be specified).

Using counts and not cumulative counts in the new format as opposed to the old format allows aggregating the counts across various ranges even with gaps. This is done by computing the cumulative counts after sorting the buckets by their upper-bounds (and coalescing overlapping ones) as opposed to having them be specified at scrape time.

With the new format, scraped resources don’t have to report all the buckets (including le=”+Inf”) all the time to avoid incorrect results.

histogram_quantile can be updated to recognize the new format in addition to the existing one. If it finds samples with the new format, then it will use the new method to compute quantiles based on the bucket range counts as opposed to the old one. By having both formats implemented, we can support both without breaking existing behavior data format.

Scenarios

The new format should be as efficient as the existing one in all cases. The main cases where it outperforms the existing format significantly are highlighted below.

Sparse Metrics

Imagine having a metric that has a wide range of values, but each sample may span very few buckets depending on other labels applied. In other words, at scrape time, you may never need to use more than a couple buckets for a metric depending on the other label values. For example, imagine you have a metric that reports RPC latencies per RPC method. There could

be a huge variance between different RPC methods in terms of latency. Some are more real-time and might take a few milliseconds, whereas others might take many seconds. In the old format, different histogram metrics will need to be defined per RPC method to get sufficient accuracy since each has its own distribution. Otherwise, by using a single histogram that covers a good range of the values, many series will be created with mostly duplicate values since a single RPC method might only span a few ranges, but due to using cumulative values, all buckets must be reported at scraping time with mostly duplicate cumulative counts leading to substantial cardinality in series. In the new format, there is no need to report any values for any bucket whose count is 0. Therefore, it's possible to use a single histogram metric that spans the entire range, even with many buckets, since most buckets will be empty depending on the RPC method used.

It should be noted that the bucket ranges still need to be statically defined, even with the new format. The main advantage it provides is that it can prune many buckets that would otherwise need to be reported every single time.

Implementation

The following commit shows an implementation for histogram_quantile using the new format. Please, take a look.