**Per-element Quantile Estimation**

**INTRODUCTION**

Quantiles are of particular interest, as they often capture the user's utility. For example, if a video connection has high tail latency, the perceived quality will suffer, even if the average and median latencies are low.

**OBJECTIVE**

We consider the problem of approximating the per-item quantiles. Elements in our stream are (ID, latency) tuples and we wish to track the latency quantiles for each ID.

**ALGORITHMS**

We consider tracking the quantiles for the heavy hitters which are often considered particularly important, without knowing them beforehand.

- **Sampling**
- **Sketching**
- **SQUAD**: combines sampling and sketching while improving the asymptotic space complexity.

**SAMPLING**

Selects a uniform random sample of a given size from an input stream of unknown size without replacement.

\[ \text{Quantile}(x): \text{Apply Quantile algorithms on samples with ID } x \]

**SKETCHING**

Allocates a separate quantile sketch to track the latency quantiles of each potential heavy hitter. Since we do not know the IDs of the heavy hitters beforehand, we use a space-saving instance where each entry, in addition to its counter and ID fields, has a quantile sketch instance.

\[ \text{Quantile}(x): \text{If } x \text{ has an allocated SS entry, its quantile instance is then queried.} \]

**SQUAD**

SQUAD, a hybrid algorithm, that combines sampling and sketching. Intuitively, the sampling helps us capture the behavior of the latencies experienced by an item before it was allocated with an SS entry and a quantile sketch.

\[ \text{Quantile}(x): \text{we use both the sample and the SS counter.} \]

**RESULTS**

The results show improvements in both error and memory usage compared to other algorithms.