Background:
Advancements in storage technologies led to decreased costs of storing vast amounts of data. While speed increases start to fail Moore’s law, storage seems to be keeping up. Before storage became cheap, organizations stored only essential, very limited amounts of data. That could be compared to storing snapshots of the timeline. Today, instead of storing snapshots, organizations store whole timelines in order to better understand the trends in the data. Analyzing such large datasets is an expensive computational task. Many datasets are too large to fit into the main memory of a single machine and their analysis requires frequent access to disk resulting in high latency. Even determining the number of distinct elements in the dataset becomes very expensive. That resulted in the development of algorithms that utilize probabilistic counting to determine the number of distinct elements in a dataset.

Overview:
Suppose you have a large dataset that can’t fit in memory. The dataset contains duplicate entries and you want to determine how many duplicate entries there are in the dataset. Data is not sorted. Sorting and counting would be inefficient since you can’t sort the data in main memory. You have to resort to estimating how many distinct elements are in the dataset. This is a well-known problem with numerous applications such as database query planning and optimizations and counting distinct species in a population.
Flajoret and Martin introduced the early algorithms in 1985. They estimated the number of distinct elements based on a single pass through the whole dataset and a small additional storage. “The algorithms are based on a statistical observations based on bits of hashed value of records”\(^1\). Further refinements were introduced by Durand and Flajoret in the paper “LogLog counting of large cardinalities” and “HyperLogLog: The analysis of a near optimal cardinality estimation algorithm” by Flajoret et al. Several other algorithms such as Adaptive Sampling are proven to perform well. The common characteristic of all afore mentioned algorithms is that they scan the whole dataset once to produce the estimation. However, there doesn’t seem to be a good general algorithm that can produce an estimation of the number of unique elements by scanning only a subset of data (several shards in the database).

**Problem statement:**

Even with working implementations of cardinality estimation algorithms such as HyperLogLog or Adaptive Sampling dealing with very high volumes of data is still expensive (scanning the whole set). Further optimization of the algorithms would be to provide the estimation based on only the subset of the data. Such formulation of the problem would have two parts: (1) Estimate the number of distinct events in the subset of data using one of the existing algorithms and (2) estimate the number of distinct events in the whole dataset based on the estimation on a subset. Doing that with high precision is a difficult problem. I had troubles finding papers describing such approaches and successfully solving this problem seems rather ambitious for a semester project. In order to simplify the problem I will investigate how the characteristics of the dataset influence the precision and performance of algorithms and try to solve the problem with utilization of heuristics that hold only for such type of dataset. For example: Data coming from a clickstream has some special characteristics, such as that owner of sequence of actions always one user etc.

\(^1\) Accessed September 21, 2015.  
I will then try to extend the heuristics to the bigger problem of estimating number of unique elements based on subset of data.

In my work I am planning to use a framework that is developed by Interana (a company in which I worked as Engineering Intern). The company agreed to allow me to use the framework for testing and benchmarking purposes, as long as I don’t disclose the implementation details of the framework. The framework has already implemented solutions for cardinality estimation and brute force counting and the results of those solutions will be taken as a reference point. Furthermore, Interana has many different datasets with billions of events, mostly coming from clickstream and user facing applications. Those datasets will be valuable in determining the effect of data characteristics on the performance of the algorithms.

**Goals:**

I am going to test the performance of existing algorithms (HyperLogLog and Adaptive Sampling) on specific datasets, benchmark them and understand the patterns in performance of different algorithms on different types of datasets.

I will try to extend the solution by scanning only the subset of data and estimating based on the subset and check how accurate the results are from that approach.

**Deliverables:**

- Report about the performance of cardinality estimation algorithms (HyperLogLog, Adaptive Sampling etc.) for data of different characteristics
- Benchmarks of algorithm performances for such datasets
- Source code of any developed sample based estimation algorithms

**Bibliography:**

(1) Flajolet, Martin. Probabilistic Counting Algorithms for Data Base Applications, IBM Development Laboratory, 1985