6 Sketches

6.1 Definition

Sketches are a useful "summary" of a set \( S \). More formally,

**Definition**: A sketch is a couple \( < \sigma, \tau > \) where, for any sets \( S_1, S_2 \), the following property is verified: \( \sigma(S_1 \cup S_2) = \tau(\sigma(S_1), \sigma(S_2)) \).

**A few examples where sketches are useful** Sketches are very useful for map-reduce jobs. For instance if we want to to the job

**MAP**: \( < a, b > \)
\( \)emit \( < a, b > \)

**REDUCE**: \( < a, \{b_1, \ldots, b_K\} > \)
\( \)emit \( < a, f(\{b_1, \ldots, b_K\}) > \)

Using a sketches allows us to use a combiner and diminish the shuffle size.

Now the reduce phase is:

**REDUCE**: \( < a, (\sigma(S_1), \ldots, \sigma(S_J)) > \)
\( \)emit \( < a, f(\sigma(S_1), \ldots, \sigma(S_J)) > \)

Sketches are also useful in a streaming environment. Suppose we need to compute at time \( t \) on the whole stream. We would like to compute \( f(\{a_1, \ldots, a_t\}) \). With sketches we can easily do

**Initialization**: \( \sigma(\{\}) \)
**At time \( t \)**: update \( \tau(\sigma(S_{t-1}), \sigma(a_t)) \)

6.2 Desirable qualities

To be useful, a sketch should have the following properties:

- Computing sigma should be linear
- We should be able to estimate \( f \) efficiently and accurately from \( \sigma \)
• \( \sigma \) should be small
• \( \tau \) should be efficient

6.3 A few reminders

**Linearity of the Expectation**
If \( X_1, X_2 \) are random variables then \( E[X_1 + X_2] = E[X_1] + E[X_2] \)

**Markov inequality**
If \( X \) is a positive or null random variable then \( \Pr(X > cE[X]) < \frac{1}{c} \)

**Chernoff bounds**
If \( S = \sum_{i=1}^{k} X_i \) where \( X_i \) are independent random variables drawn from a Bernouilli distribution, then
\[
\Pr(S < (1 - \delta)\mu) \leq e^{-\frac{\delta^2 \mu}{2}}
\]
And
\[
\Pr(S > (1 + \delta)\mu) \geq \left( \frac{e^{\delta}}{(1 + \delta)^{1+\delta}} \right)^\mu
\]

We can see in the second inequality that as \( \delta \to 0 \) the right hand side is equivalent to \( e^{-\frac{\delta^2 \mu}{2}} \)

6.4 First example of a Sketch

We would like to find a sketch for the function : \( f(S) = \text{uniform sample from } S \).

**Attempt 1** \( \sigma(S) = \text{uniform sample from } S \) and \( \tau(\sigma(S_1), \sigma(S_2)) \) a random choice from \( \sigma(S_1), \sigma(S_2) \)
That doesn’t work if the size of the sets are different.

**Attempt 2** To counter the flaw of the first attempt, we can try to modify \( \tau \) and \( \sigma \) by keeping the size of the sample in memory :
\[
\sigma(S) = \langle \text{uniform sample of } S, |S| \rangle
\]
And \( \tau(\langle v_1, s_1 \rangle, \langle v_2, s_2 \rangle) \) gives \( \langle v_1, s_1 \rangle \) with \( i = \frac{s_1}{s_1 + s_2} \) and \( \langle v_2, s_2 \rangle \) with \( i = \frac{s_2}{s_1 + s_2} \)

This method doesn’t work if the sets \( \sigma(S_1), \sigma(S_2) \) aren’t disjoint. To give a proper solution, we need to define the notion of a Consistent random Hash function.
**Consistent random Hash function**

A Consistent random Hash function \( h(x) \) verifies

- \( h(x) \) follows a uniform law on \([0, 1]\)
- \( h(x) \) and \( h(y) \) are independent for any \( x \neq y \)
- \( h(x) \) returns the same value each time for a given \( x \)

For instance, in any modern programming language, one could write:

```python
def h(x):
    srand(x)
    return rand()
```

We are now ready to give the solution

**Solution**

Let \( h \) be a Consistent random Hash function. We define

\[
\sigma(S) = \arg\min_{x \in S} h(x)
\]

and

\[
\tau(\sigma_1, \sigma_2) = \arg\min_{x \in \sigma_1, \sigma_2} h(x)
\]

This two functions define a sketch and give an accurate response to our problem.

### 6.5 Second example of a sketch

In this case we consider a streaming example \( a_1, \ldots, a_t \) and try to evaluate the frequency of each item in the stream \( f_t(a) = |\{i \leq t | a_i = a\}| \) and we also define the \( k^{th} \) frequency moment

\[
F_k(t) = \sum_{a \in V} (f_t(a))^k \quad k > 0
\]

and \( F_0(t) \) as the number of distinct values in the stream seen at time \( t \).