Lecture 17

1 Overview and Motivation

In this lecture we introduce the model of sketching, its relation to streaming as well as some examples of industrial applications. We will also consider two sketching algorithms – min-wise hashing, and random hyper-plane – and an application for establishing lower bounds.

2 Sketching

In lecture 9 we showed that any streaming algorithm gives us a one-round communication protocol. Sketching, also known as simultaneous communication model, can also be cast as a communication complexity problem. In the simultaneous complexity model, Alice and Bob both send messages to a third party, a Referee, and the goal is for the Referee to report the output. In this model, we can assume that both Alice and Bob have access to common randomness and they can generate the same random number independent of each other.

Almost all of the streaming algorithms that we have seen can be used for sketching:

- $L_0, L_1, L_2, L_k$ norm estimation.
- The heavy hitters, sparse approximation, and compressed sensing problems
- Geometric MST and matching (these can be reduced to norm estimation)
- Core-sets. Note that here we do not rely on randomness.
- Sparse certificates.

For example, in the heavy hitters problem we can think of Alice and Bob both having some part of the input, and the goal is for the Referee to compute the heavy hitters of the combined input.

Similarly, norm estimation, sparse approximation, and compressed sensing can be used for sketching because of their linearity. On the other hand, core-sets and sparse certificates can be used for sketching because of their composability.

In fact, the only problem that we have seen, for which streaming algorithm does not directly lead to sketching, seems to be the graph matching problem.

3 Industrial Applications

3.1 Word Frequency Vectors

One use of sketching is for text documents. Consider a representation in which we map documents to frequency vectors where the coordinates are the unique words in the document. The problem of storing these vectors is difficult, so applications that use this representation can benefit a lot from sketching.
3.2 Map-Reduce

Google’s Map-Reduce is another such application. Although, this is not a very clear model, it can be interpreted as a subroutine that works on data elements that are essentially key, value pairs. In the first step of the processing (the ”map” step), the keys are aggregated together. After that, in the ”reduce” step for each key the subroutine computes the ”sum” of the values for that key. The second step can compute any commutative and associative function, i.e. one that does not depend on the order.

This model is parallelizable, because the total value does not depend on the order and a lot of the operations can be executed in parallel.

The relation to sketching is that we can think of each key as a sketch and then, reducing corresponds to composition of sketches. k-means has been implemented using Map-Reduce.

3.3 Min-wise Hashing

In many applications, the vector are very sparse, e.g. a high-dimensional problem with very few 1’s, and it is easier to think of them as sets. Note that here, we focus on the 0-1 vectors case, such as word frequency vectors that just mark whether the word occurs in the document and not how many time it occurs.

**Definition 1** (Jaccard coefficient). For two sets \( A \) and \( B \), the Jaccard coefficient is defined as

\[
J(A, B) = \frac{|A \cap B|}{|A \cup B|}.
\]

Note that if \( A = B \), \( J(A, B) = 1 \) and if \( A \) and \( B \) are disjoint, \( J(A, B) = 0 \). Otherwise the value is between 0 and 1. The Jaccard coefficient is a measure of distance, but is not itself a distance function, because it has higher values for more similar sets. However, \( 1 - J(A, B) \) is a metric.

The natural question is to compute sketches of sets, such that \( J(A, B) \) is preserved. This could be done by using earlier sketches for Hamming metric (i.e., \( L_0 \) norm). This is because Jaccard coefficient is naturally related to Hamming distance. Using the identity \( 2|A \cap B| = |A| + |B| - |A \oplus B| \) we can rewrite \( J(A, B) \) in terms of \( |A \oplus B|, |A|, \) and \( |B| \). Here \( |A \oplus B| \) is precisely the Hamming distance between \( A \) and \( B \). However, the resulting sketching method is quite messy and suboptimal.

A much better method was proposed in [1, 3, 2]. Consider a mapping \( g(A) = \min_{a \in A} h(a) \) where \( h \) is a random permutation of the elements in the universe.

**Claim 1.** \( Pr[g(A) = g(B)] = J(A, B) \), where \( g(A) \) and \( g(B) \) are computed over the same permutation \( h \).

**Proof.** Consider the minimum element \( \min(h(A) \cup h(B)) \). If the minimum element is in \( A \) or \( B \) then the equality \( g(A) = g(B) \) does not hold. Therefore, the minimum must be in \( A \cap B \) and the probability of this happening is exactly \( J(A, B) \). □

The above method provides an “atomic” sketch, which can be further improved by repetition. Let \( Sk(A) = [g_1(A), g_2(A), \ldots, g_t(A)] \) be the sketch of \( A \). Using Chernoff bounds, we can set \( t = C \frac{\log(1/P)}{J(A, B)^2} \) and with probability at least \( 1 - P \) get:

\[
1 - \frac{||Sk(A) - Sk(B)||_0}{t} = J(A, B)(1 \pm \epsilon)
\]

\( ||Sk(A) - Sk(B)||_0 \) is the hamming distance between the two sketches and \( 1-||Sk(A) - Sk(B)||_0/t \) is the fraction of positions that are equal.
It is a little unfortunate that $J(A, B)$ appears in the expression for $t$. However, there is no way around it: if $t$ was independent of $J(A, B)$, then we could tell if $J(A, B) = 0$ or not, i.e., detect if $A \cap B$ is empty. This cannot be done using sketches of small size, as per Lecture 10.

Finally, observe that the aforementioned sketching method has very strong property: the probability that two atomic sketches are equal is dependent on the similarity between the sketched objects. Such sketching methods are often referred to as Locality Sensitive Hashing, and have other applications, such as approximate nearest neighbor search.

Industrial relevance: Alta Vista used min-wise hashing on a large database of web pages [3]. It is a pretty popular technique by now.

### 3.4 Random Hyperplane

People often normalize the frequency vectors of documents. In such situations, we have two unit vectors $u$ and $v$ in $\mathbb{R}^m$. We will use a different metric than what we are used to – the angular distance.

**Definition 2** (Angular distance). The angular distance between two vectors (in our case unit vectors), $A(u, v)$ is equal to the angle between $u$ and $v$. We take the smaller angle, so that $A(u, v) \leq \pi$.

To sketch vectors in a way that estimates $A(u, v)$, we use the following method [5]: choose a random unit vector $r$ and define $s(u) = \text{sign}(u \cdot r)$.

**Claim 2.** The probability that $s(u \cdot r) \neq s(v \cdot r)$ is $A(u, v)/\pi$.

**Proof.** Let $P$ be a two-dimensional plane spanned by vectors $u$ and $v$. Decompose $r = r' + r''$, where $r' \in P$ and $r''$ is orthogonal to $P$. Clearly, $u \cdot r = u \cdot r'$, so the value of $r''$ does not influence the values of $u \cdot r$ and $v \cdot r$. By symmetry, we can assume that $u$ and $v$ are unit vectors in $\mathbb{R}^2$, and that $r'$ lives in $\mathbb{R}^2$ as well. In fact, we can as well consider only the case when $r'$ is a unit vector as well, since the sign is invariant under scaling of $r'$ by some positive factor. Overall, we can assume that $r'$ is chosen uniformly at random from a unit circle.\(^1\)

Consider the two dot products $u \cdot r'$ and $v \cdot r'$. The two dot products have different sign when the vector $r'$ separates $u$ and $v$ on the circle, that is it lies within the smaller angle formed between $u$ and $v$. The probability of this happening is exactly $A(u, v)/\pi$. (Note that taking the complement of $r'$ changes the sign of both dot products, so we only need to evaluate the probability over a region of angle $\pi$ and not the entire circle)

As above, we define the sketch as $Sk(u) = [s_1(u), s_2(u), \ldots, s_t(u)]$ and by setting $t = C \frac{\log(1/P)}{A(u,v)\epsilon^2}$ we have that with probability $1 - P$ the following holds:

$$||Sk(u) - Sk(v)||_0/t = A(u, v)(1 \pm \epsilon)/\pi$$

The idea of sketching using random hyperplanes was suggested in [4], and was implemented and used to detect near duplicates in web pages. See [6] for an overview.

\(^1\)This argument is intuitive, but informal. To make it formal, we choose $r$ from the $n$-dimensional normal distribution $N^m(0, 1)$. By spherical symmetry property of normal distribution, we know that $r'$ has normal distribution. If we rotate the plane $P$ so that it coincides with $\mathbb{R}^2$, $u$ and $v$ are still unit vectors, while $r'$ has $N^2(0, 1)$ distribution. Now, instead of considering $r'$, we can consider $r'' = r'/||r'||_2$, since $\text{sign}(u \cdot r') = \text{sign}(u \cdot r'')$. Happily, $r''$ is uniformly distributed over a two-dimensional unit circle.
4 Application to Lower Bounds

Recall the Gap Dot Product (GDP) problem mentioned earlier: Suppose Alice has a vector \( u \in \mathbb{R}^m \) with \( \|u\|_2 = 1 \) and Bob has a vector \( v \in \mathbb{R}^m \) with \( \|v\|_2 = 1 \). The goal is for them to compute whether \( u \cdot v = 0 \) (in which case they return 0) or \( u \cdot v \geq \Delta \) (in which case they return 1).

\[ \Delta = \frac{1}{(m/2)^{1/2}} \]

is the gap parameter.

This is a one-round communication complexity question and we proved the following theorem, stating that the lower bound for it is linear in \( m \) or quadratic in \( \Delta \):

**Theorem 3.** The randomized one-round communication complexity for Gap Dot Product is \( \Omega(m) \).

We can relate this to the random hyperplane sketching method discussed earlier. Let the sketch length be \( t = C/\Delta^2 \). For large enough \( C > 1 \) we can ensure that:

- If \( u \cdot v = 0 \) then \( A(u, v) = \pi/2 \) and so \( \|Sk(u) - Sk(v)\|_0/t = 1/2 \pm \Delta/8 \)

- If \( u \cdot v \geq \Delta \) then \( A(u, v) < \pi/2 - c\Delta \) and so \( \|Sk(u) - Sk(v)\|_0/t = 1/2 - \Delta/\pi \pm \Delta/8 \). Note that \( c = c(\Delta) \approx 1 \) for small values of \( \Delta \), because \( u \cdot v = \cos(A(u, v)) \) and at \( \cos(\pi/2) = 0 \), the cosine has a derivative of -1.

This reduces GDP to the problem of Hamming distance estimation. Given that GDP requires \( \Omega(m) \) communication, we can conclude that the Hamming distance problem also requires \( \Omega(m) \) communication.

**Corollary 4.** The randomized one-round communication complexity of \((1 + \Delta/8)^{-}\)-approximating the Hamming distance between vectors \( x, y \in \{-1, 1\}^{O(1/\Delta^2)} \) is \( \Omega(1/\Delta^2) \).

**Corollary 5.** \((1 + \Delta)^{-}\)-approximating the \( L_0 \) norm in a stream under insertions and deletions requires \( \Omega(1/\Delta^2) \) space.

In fact, one can even show that the above Corollary holds for insertions-only streams. For more, see [8, 7].

**References**


