COMP 480/580 — Probabilistic Algorithms and Data Structure Mar 26, 2018

Lecture 17

Lecturer: Anshumali Shrivastava *Scribe By: Mengnan Zhao*

1 Sampling in high-dimensional space and curse of dimensionality

There is an exponential increase in sampling cost with adding dimensions of the space. For example, we want to estimate the area of a in Fig. 1, by sampling random points from A and compute the ratio of number of points from a to that of A. However, the ratio $\frac{a}{A} \to 0$ as dimensionality of space goes to infinity. The sampling gets inefficient as dimensionality grows since we could hardly sample enough data points from a. Sampling is one domain which curse of dimensionality happens, other domains include machine learning, optimization, etc. Instead,

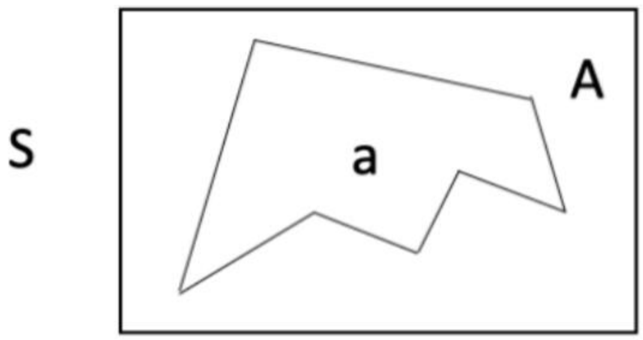

Figure 1: Estimate of volume of a in low-dimensional space is easy. But the sampling becomes inefficient as dimensionality grows.

we could sample in this way: Sample N points $x_1, x_2, \cdots, x_N$ from A. Denote $I_i = 1$ if $X_i \in a$. $E[I_i] = \frac{a}{A} = (\frac{1}{factor})^d$. And we could see that as d increases, this value decreases exponentially and goes to 0.

Lemma 1 *Let $I_1, I_2, \cdots, I_N$ be i.i.d indicator random variables with $\mu = E[I_i] = \frac{a}{A}$, then $Pr[\frac{1}{N}\sum I_i - \mu \geq \epsilon\mu] \leq \delta$ if $N \geq \frac{3\ln\frac{2}{\delta}}{\epsilon^2\mu}$.*

Therefore, to have given failure probability, the number of samples required is $\mathcal{O}(\frac{A}{a}) = \mathcal{O}(factor^d)$.

Consider the example in Fig.2, where we want to estimate the volume of K (in high-dimensional space). We construct a series of balls $B_0, B_1, \cdots, B_r$. Specifically, B_0 is entirely within K and B_r contains K.

$$vol(K) = vol(K \cap B_r) = \frac{vol(K \cap B_r)}{vol(K \cap B_{r-1})}\frac{vol(K \cap B_{r-1})}{vol(K \cap B_{r-2})} \cdots \frac{vol(K \cap B_1)}{vol(K \cap B_0)} vol(K \cap B_0)$$

The LHS of the above equation is hard to estimate, however the RHS is easy to estimate since each ratio term could be estimated with sampling. To prevent each ratio term to be too small due to curse of dimensionality, the radius of B_i cannott be too much larger than B_{i-1}. Specifically, we require $rad_{B_i} = (1+\frac{1}{d})rad_{B_{i-1}}$. As a result, the number of balls cannot be arbitrarily small. Here we require $\mathcal{O}(d^4)$ balls, which is polynomial in dimension, compared to $\mathcal{O}(factor^d)$, which is exponential in dimension, in previous case.

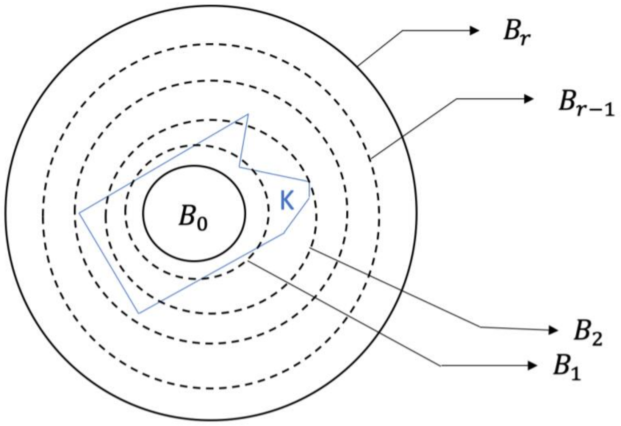

Figure 2: Estimate of volume of K in high dimensional space

2 Disjunctive normal forms (DNF) counting

Disjunctive normal forms (DNF) is a conjunction of clauses, where each clause is a disjunction of literals of x or $\bar{x}$. An example is $(x_1 \wedge x_2 \wedge x_3) \vee (x_4 \wedge x_5 \wedge \bar{x_2})$ The problem is to count the number of assignments to the variables to satisfy the formula.

Note that as long as one clause is true, the whole statement is true. In other words, the formula is unsatisfied if and only if each clause is false.

In general, a counting problem can be descrebed as follows: Given the universe U and set $S \subset U$, determine $|S|$. In DNF counting, we can think of U as the set of all possible assignments and hence $|U| = 2^n$, where n is the number of variables. The problem can then be translated into estimating the ratio $\frac{|S|}{|U|}$, given that we could deterine the size of U.

theorem 1 *DNF is sharp-P complete.*

This indicates that DNF counting is much harder than decision problem, whereas the case is opposite for CNF problem.

Denote X_i as the indicator random variable of the i-th iteration and $X = \sum_{i=1}^{M} X_i$. $E[X] = \sum_{i=1}^{M} E[X_i] = \sum_{i=1}^{M} \frac{|S|}{2^n} = M * \frac{|S|}{2^n}$. Therefore we can use the above estimator for $|S|$. However, this sampling procedure is usually quite inefficient sampling since the number of assignments is usually quite small compared with 2^n.

2.1 Importance Sampling

In order to improve the previous algorithm, we need to increase the ratio $\frac{|S|}{|U|}$ by decreasing the size of U. We could rewrite the DNF as:

$$DNF = C_1 \vee C_2 \vee \cdots C_m$$

Algorithm 1 Randomized Monte Carlo

```
Input: DNF
Output: Number of assignments that could satisfy the given DNF
1: $count \leftarrow 0$
2: for $i = 1 : M$ do
3:     random assignment
4:     if DNF is satisfied then
5:         $count++$
6:     end if
7: end for
8: return $\frac{count}{M} 2^n$
```

where C_i is the i-th clause. Let SC_i to be the set of all assignments that satisfies C_i. We can write $SC_i = (i, a_1), (i, a_2), \cdots, (i, a_k)$ where a_j is the assignments that satisfy C_i. Note that $S = \cup_i SC_i$ whose size is to be estimated. The universal set $U = \{(i, a_i) : a_i \in SC_i\}$. Then $|U| = \sum_i |SC_i|$.

The sampling can be carried out as follows: sample (i, a_i) uniformly at random from U, and add the count if the sample belongs to S, just as the previous algorithm.

To sample uniformly at random, we compute $|SC_i|$, pick i with probability $\frac{|SC_i|}{|U|}$, and then pick a random assignment to satisfy C_i. Also, it is easy to see that $\frac{|S|}{|U|} \geq \frac{1}{m}$ since each assignment cannot satisfy more than m clauses. Furthre analysis could show that $\mathcal{O}(\frac{\ln(2/\delta)}{\epsilon^2 \mu})$ samples are sufficient. Now the entire process can be finished within polynomial time.