COMS W6998-5: Algorithms through Geometric Lens (Fall17) Sept 6, 2017

Lecture 1 – Introduction,Dimension Reduction,Approximation Algo

Instructor: *Alex Andoni, Ilya Razenshteyn* Scribes: *Aishwarya Srinivasan*

1 Introduction

The class is based on advanced algorithms and their analysis using the concepts of graph theory. The aim of the course is to understand ways by which the real world problems which are solved by certain algorithms are optimized by inter-relating them with the concepts of graphs and interpretations, similar to dimension reduction of data points for reducing space complexity of the problem etc.

2 Course Topics

- Sketching/Streaming Algorithms
- Dimension reduction and applications
- Nearest Neighbour Search (NNS)
- Graph algorithms based on semi definite programming
- Spectral Graph Algorithms
- Metric embeddings with applications
- Distance oracles
- Discrepancy minimization

3 Pre-requisites for the course

For understanding the course well, grasping the concepts and putting then to application, mathematical maturity is needed. Moreover, the student needs to be familiar with the below mentioned courses:
- basics of probability theory,linearity of expectation, variance, Markov/ Chebyshev bound;
- basic linear algebra (eigen values, eigenvectors);
- asymptotic analysis of algorithms, runtime analysis;
- most basic algorithms, such as hashing or binary search;
- graphs

4 Evaluation and grading

The courses will be evaluated by the following:
- Scribing (1 lecture): 10%;
- 3 homeworks: 45%;

• Project: 45%, including 5% for project proposal, 10% for oral presentation, and 30% for the final write-up.

5 Probability Recap

These are a few probability tools we will use in the analysis of algorithms.
Let x and y be random variables.

Definition 1. *(Expectation) For a discrete random variable x, the expectation of x, E[x] is*

$$E[x] = \sum_{v} v.Pr[x = v]$$

For a continuous random variable x, the expectation of x, E[x], is

$$E[x] = \int v.\phi(v)dv$$

where ϕ is the probability distribution function of x.

Lemma 2. *(Linearity of Expectation). Let X and Y be two random variables.*

$$E[x + y] = E[x] + E[y]$$

Lemma 3. *(Markov Inequality) for*

$$x \geq 0 \quad \lambda \geq 0$$

$$Pr[x > \lambda] \leq (E[x])/\lambda$$

Definition 4. *(Variance). The variance of a random variable x, denoted Var[x], is*

$$Var[x] = E[(x - [x])^2] = E[x^2] - (E[x])^2$$

Lemma 5. *(Chebyshev Inequality) for $\lambda \geq 0$*

$$Pr[|x - E[x]| > \lambda] \leq (Var[x])/\lambda^2$$

6 Streaming algorithms

Let us consider a router, through which some data is transmitter. The type, dimension or size of the incoming data may or may not be similar to the outgoing data. In this scenario, we wish to compute some complex analytics and statistics with the data received by the router.

For the purpose, the data received by the router needs to be processed. If all the data received by the router is saved, that will exceed the available storage. In order to have an optimized way of storing

and processing the data, we look upon the concept of dimension reduction, which is explained in later sections of the scribes.

7 Dimension Reduction

Let $x \in R^n$ and $y \in R^n$, and $f : R^n \to R^k$, where k<<n be a function which maps the data points of n-dimension to k-dimension. The function which maps is known as a hash function.

$$x \to (x) \quad and \quad y \to (y)$$

The function is mapped in such a way that the difference between any two data after the dimension reduction still remains comparable.

$$|(|x - y|)| \approx ||f(x) - f(y)||$$

$$HammingDistance : ||x - y|| = \sum_{i=1}^{n} |x_i - y_i|$$

$$EuclideanDistance : ||x - y|| = \sqrt{\sum (x_i - y_i)^2}$$

Problem: Doing faster linear regression?
Solution: Given a data matrix A and vector b, find

$$argmin||Ax - b|| \qquad\qquad x \in R^n$$

8 Nearest Neighbour Search/Similarity Search

Nearest neighbour search (NNS), as a form of proximity search, is the optimization problem of finding the point in a given set that is closest (or most similar) to a given point. Closeness is typically expressed in terms of a dissimilarity function: the less similar the objects, the larger the function values. Formally, the nearest-neighbour (NN) search problem is defined as follows: given a set S of points in a space M and a query point $q \in M$, find the closest point in S to q.

Let us consider a set of points $P \in R^d$, number of points being n.Pre-processing of the P is required such that given $q \in R^d$. The output for the nearest neighbour search would be arg?min $||q - p||$, where $p \in P$.

This method is applied in Deep Learning neural network to classify images.

9 Graph Algorithms (Based on semi definite programs)

Graphs are broken into vectors and operations are performed to find the solutions to the problem of graphs. Some of the concepts that can be applied are linear algebra, decompose matrix and Eigen vector.

- MAXCUT problems
- Approximate Co louring Problem

10 Distance oracles

A distance oracle (DO) is a data structure for calculating distances between vertices's in a graph.
Given weighted graphs G, build a data structure on G, with n nodes and m edges, such that for $\forall(x, y)$, it outputs $d_G(x, y)$.
Naive Solution:
Store all distances $d_G(x, y)$ for all nodes (x,y).
Space = n^2 for n vertices's.

The aim is to coin an algorithm that uses minimal computational time and space. An example of that is Dijkstra Algorithm, which takes $O(m+nlog(n))$ in time, and requires no extra space than the graph itself.

11 Approximation Algorithms

Approximation algorithms are efficient algorithms that find approximate solutions to NP- hard optimization problems with provable guarantees on the distance of the returned solution to the optimal one.

$$a \rightarrow real \quad answer$$

$$\alpha \quad approximation \qquad a \leq a^{\leq}\alpha.\hat{a}$$

$$\alpha - \beta \quad approximation \qquad a/\beta \leq a^{\leq}\alpha.\hat{a}$$

A simple example of an approximation algorithm is one for the Minimum Vertex Cover problem, where the goal is to choose the smallest set of vertices's such that every edge in the input graph contains at least one chosen vertex. One way to find a vertex cover is to repeat the following process: find an uncovered edge, add both its endpoints to the cover, and remove all edges incident to either vertex from the graph. As any vertex cover of the input graph must use a distinct vertex to cover each edge that was considered in the process (since it forms a matching), the vertex cover produced, therefore, is at most twice as large as the optimal one. In other words, this is a constant factor approximation algorithm with an approximation factor of 2.