Error Correcting Codes: Combinatorics, Algorithms and Applications (Spring 2010)

Lecture 40: GS List Decoder and Data Stream Algorithms

April 19, 2010

Lecturer: Atri Rudra *Scribe: Xinglei Zhu, Atri Rudra*

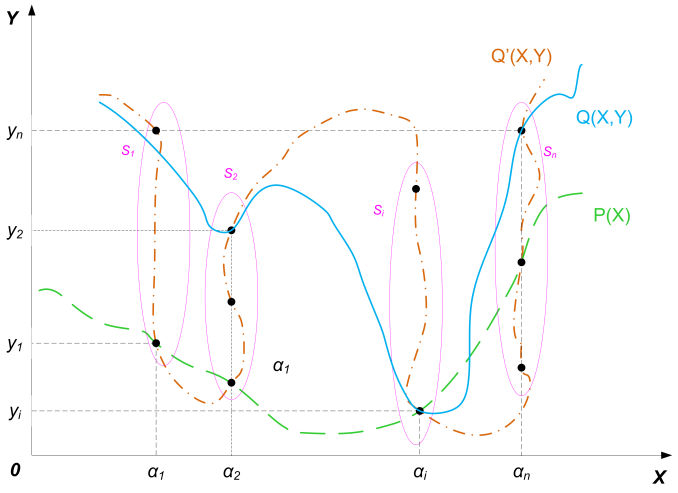
1 Guruswami-Sudan List Decoding

Recall that a better list decoding algorithm for RS codes was based on the following observation with definition of *multiplicity*. This is the new idea in the Guruswami-Sudan algorithm over Sudan's algorithm for list decoding RS codes. Today we will see how this idea can be extended to solve a generalization of list decoding.

1.1 Multiplicity

Recall the definition of multiplicity: $Q(X,Y)$ has *multiplicity roots* r at $(0,0)$ if it has no monomial of degree less than r. $Q(X,Y)$ has multiplicity r at (α,β) if $Q_{\alpha,\beta}(X,Y) = Q(X+\alpha, Y+\beta)$ has *multiplicity* r roots at its origin.

We now observe that in the previous version of the GS-list decoding algorithm from last lecture, the $\alpha_i s'$ need not be *distinct*. For the same value of α_i, there could be multiple $y_i' s$. An example is shown in the following Figure. $Q(X,Y)$ is a bivariable poly with $(1,k)$ degree $\leq D$ with *multiplicity* 1 for each (α_i, y_i), $1 \leq i \leq n$ as before. However, the GS algorithm even works if the input has multiple y_i^j values for the same α_i. Let (α_i, y_i^j) be distinct. The $Q(X,Y)$ is a bivariable polynomial that passes through all the (α_i, y_i^j) points. Set S_i contains the points (α_i, y_i^j) with same α_i. Let ℓ_i denote the number of points in S_i, i.e. $\ell_i = |S_i|$. Note that till now we have considered the case $\ell_i = 1$.



1.2 The modified Guruswami-Sudan list decoding algorithm

We now consider the case when the input is (α_i, S_i) where $S_i \subseteq \mathbb{F}_q, 1 \leq i \leq n$. We will have multiple constraints.

Step 1: Compute non-zero $Q(X, Y)$ with $(1, k)$ degree $\leq D$ with multiplicity r for each $(\alpha_i, y_i),\ 1 \leq i \leq n$. As long as the number of coefficients are more than the number of constraints (recall from the same α_i now), the desired solution exists. Therefore if we have

$$\frac{D^2}{2k} > \sum_{i=1}^{n} |S_i| = \sum_{i=1}^{n} \ell_i \tag{1}$$

then $\exists$ a non-zero $Q(X, Y)$ of $(1, k)$ degree $\leq D$ s.t. $Q(\alpha_i, \beta_i) = 0$ for $1 \leq i \leq n$ and $\beta_i \in S_i$.

Step 2: Output $P(X)$ of degree less or equal to k, if ***i)*** $(Y - P(X))|Q(X, Y)$ and ***ii)*** $P(\alpha_i) = y_i$ for no less than t values of i. $P(\alpha_i) \in S_i$ for at least t values of i.

Using the argument from last lecture, the above algorithm works as long as $rt > D$.

Definition [List Recovery]: Given a code $C \in \Sigma^n$ and input $(\alpha_i, S_i), S_i \subseteq \Sigma, 1 \leq i \leq n$, output all $(c_1, ..., c_n) \in \mathbb{C}$, such that $c_i \in S_i$ for no less than t values of i.

Definition [Error-less List Recovery]: List recovery with $t = n$, i.e. $\forall i, c_i \in S_i$

1.3 List recovery bounds for GS algorithm

Let $\ell = \frac{1}{n}\sum_{i=1}^{n} \ell_i$ denote the average input list size. We now look the instantiates of the parameters so that the algorithm above works.

In *Step 1*, for successful interpolation we need,

$$\frac{D^2}{2k} > n\ell\binom{r+1}{2} \text{ which is satisfied for } D = \left\lceil \sqrt{n\ell k r(r+1)} \right\rceil \tag{2}$$

In *Step 2*, for successful decoding, we must have $rt > D$, therefore, we have

$$rt > D \Leftrightarrow t > \frac{D}{r} \Leftrightarrow t > \left\lceil \sqrt{n\ell k(1 + \frac{1}{r})} \right\rceil \tag{3}$$

If we select $r = 2n\ell k$, $n\ell k(1 + \frac{1}{r}) = n\ell k + \frac{1}{2}$. Because t has to be integer, if $t > \sqrt{n\ell k}$, we have $t > \sqrt{n\ell k + \frac{1}{2}}$ and by the above equation, $rt > D$ as desired.

Thus, for error-less list recovery, we have $n > sqrt(n\ell k)$ Therefore the GS algorithm works as long as

$$n > \sqrt{n\ell k} \Leftrightarrow \sqrt{n} > \sqrt{\ell k} \Leftrightarrow \ell < \left\lfloor \frac{n}{k} \right\rfloor \tag{4}$$

In the above equation, the bound $\ell < \left\lfloor \frac{n}{k} \right\rfloor$ is known to be tight. It has been shown that super-poly list size is required when $\ell > \left\lfloor \frac{n}{k} \right\rfloor$ [1].

1.4 Soft decoding

In the previous algorithm, we assume that the multiplicities for all α_i are all the same as r. This assumption could be relaxed such that for different i, the multiplicity $\omega_{i,\beta} \geq 0$ could be different for $(\alpha_i, \beta), 1 \leq i \leq n, \beta \in \mathbb{F}_q$. List recovery with different multiplicities is also called *soft decoding* [2]. The list decoding problem with same multiplicity r is indeed a special situation of soft decoding with following $\omega_{i,\beta}$:

$$\omega_{i,\beta} = \begin{cases} r, & \text{if } \beta \in S_i \\ 0, & \text{otherwise} \end{cases}$$

It can be checked tat the GS algorithm can output all codewords $c_1, \cdots, c_n$ s.t.

$$\sum_{i=1}^{n} \omega_{i,c_i} \geq \sqrt{\ell k \sum_{i=1}^{n} \sum_{\beta \in \mathbb{F}_q} \binom{\omega_{i,\beta}+1}{2}} \tag{5}$$

in polynomial time.

2 Data Stream Algorithms

Let's consider the problem of tracking updates on stock trades. Given a set of trades $(i_1, u_1), \cdots, (i_m, u_m)$, where i_j is the stock id for the j^{th} trade, u_j is the amount of the stocks in the j^{th} trade. The problem is to keep track of the top d stocks. Such a problem is also called hot items/ heavy hitters problem.

Let n be the total number of stocks in the market. This problem could be solved in $O(m) + O(n \log n) \approx O(m)$ time and $O(n)$ space by setting a $O(n)$ size buffer to record the total number of trading for each stock and then sort the buffer later. However, m could be million level for one minute's trading, e.g. in the first minute of April 19, 2010, there are 8077600 stocks were traded. Taking the huge amount of trades into consideration, such an algorithm is not practical.

A practical *data stream algorithm* is expected to have the following properties:

- does not store the entire input (due to the huge size of input) - only uses poly-log space
- 1-pass over data - since multiple pass requires storing the whole data
- poly-log updating time - linear time is not acceptable due to the size of problem
- poly-log reporting time - same as above

Definition [Hot Items Problem]: Let f_ℓ denote the total count for the stock id ℓ. Initially $f_\ell = 0$, given (ℓ, u_ℓ), $f_\ell = f_\ell + u_\ell$. Given $(i_\ell, u_\ell), 1 \leq \ell \leq m, i_\ell \in [n]$, output all $j \in [n]$, such that $f_j > (\sum_{i=1}^{m} u_\ell)/d$. Note that at most d items match this condition.

More detailed algorithm will be presented in next lecture. Check a survey paper [3] if interested.

References

[1] V. Guruswami and A. Rudra. Limits to list decoding reed-solomon codes. *IEEE Transaction on Information Theory*, 52(8):3642–3649, 2006.

[2] R. Koetter and A. Vardy. Algebraic soft-decision decoding of reed-solomon codes. *IEEE Transaction on Information Theory*, 49(11):2809–2825, 2003.

[3] S. Muthukrishnan. Data streams: Algorithms and applications. *ACM-SIAM Symposium on Discrete Algorithms*, 2003.