| | |
|---|---|
| **CS 681: Computational Number Theory and Algebra** | **Lecture 2: Reed Solomon code** |
| **Lecturer: Manindra Agrawal** | **Notes by: Anindya De** |
| | **August 8, 2006.** |

# 1 Reed Solomon codes-Encoding

Let $b_0, b_1, \ldots, b_n$ be a binary sequence which is to be coded for handling a maximum of t errors. Fix a $k < n$ and split $b_0, b_1, \ldots, b_n$ into $n/k$ blocks of k bits each. Let these be $c_0, c_1, \ldots c_k$. View each $c_i$ as an element in $\mathbf{F}_{2^k}$.
Define $P(x) = \sum_{i=1}^{n/k-1} c_i x^i$.
Let $d_j = P(e_j)$ for $e_0, e_1, e_2, \ldots, e_{m-1} \in \mathbf{F}_{2^k}$.
We will output $d_0, d_1, d_2, \ldots, d_{m-1}$ as the encoded message. The input size is $n$ bits as compared to the output size which is $mk$ bits. Also we assume that the number of errors is atmost $t$ i.e. atmost $t$ out of the $m$ $d_i$ get corrupted.
Note that though theoretically it can correct only upto t errors, the number of errors it can correct in practice is much larger. This is because we assume that the $t$ bits that get corrupted are in $t$ different $d_i$'s but usually errors occur in blocks. Hence it can correct upto $tk$ errors.

# 2 Decoding

To decode the message, we must have $m \geq n/k$ (without any errors). In case, the message does not have any errors and we get the $d_i$'s, we can decode it as follows:
In order to find $c_i$'s, we can solve the following system of linear equations.

$$Ec = d \tag{1}$$

$$E = \begin{pmatrix} 1 & e_0 & e_0^2 & \ldots & e_0^{n/k-1} \\ 1 & e_1 & e_1^2 & \ldots & e_1^{n/k-1} \\ \vdots & \vdots & \vdots & \ddots & \\ 1 & e_{n/k-1} & e_{n/k-1}^2 & \ldots & e_{n/k-1}^{n/k-1} \end{pmatrix}$$

$$c = \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_{n/k-1} \end{pmatrix}$$

$$d = \begin{pmatrix} d_0 \\ d_1 \\ \vdots \\ d_{n/k-1} \end{pmatrix}$$

**Fact 2.1** *The determinant of matrix $E$ is $\prod_{i>j}(e_i - e_j)$. Hence with distinct $e_i$'s the matrix is always invertible.*

Hence, we don't have any error in the message it can be easily decoded. Now, suppose that there are errors ($< t$) in the message. Let the position of the errors be $i_1, i_2, i_3, \ldots, i_t$. (In case, there are less than $t$ errors, analysis would still be correct). Also let the corrupted message be $d'_0, d'_1, d'_2, \ldots, d'_{m-1}$
Let $Q(x)$ be a polynomial such that $Q(x) = \prod_{j=1}^{t}(x - e_{i_j})$.
$Q(x)$ is called the error locator polynomial. Important property of the error locator polynomial is that:
$d'_j Q(e_j) = d_j Q(e_j)\ \forall j$ (2.1)
$\implies d'_j Q(e_j) = P(e_j)Q(e_j)\ \forall j$
$\implies d'_j Q(e_j) = R(e_j)\ \forall j$
Here $R(x) = P(x)Q(x)$. deg($R$)=deg($P$)+deg($Q$)=$n/k - 1 + t$. Also deg($Q$)=$t$. If we consider the coefficients of $R(x)$ and $Q(x)$ as variables, then we have $m$ linear equations that can be solved to get the values of these variables. Once we know $R(x)$ and $Q(x)$, we can obtain $P(x)$ by dividing them. i.e.
Let $R(x)$=$\sum_{j=0}^{n/k+t-1} \alpha_j x^j$
Let $Q(x)$=$\sum_{j=0}^{t} \beta_j x^j$
$\forall e_i \quad d'_i \sum_{j=0}^{t} \beta_j e_i^j = \sum_{j=0}^{n/k+t-1} \alpha_j e_i^j$
If the number of equations $m$ is greater than the number of variables $n/k + 2t + 1$, then the equations may be solved. (There may be more than one solution for $Q(x)$ or $R(x)$ but $P(x)$ will be the same for all cases). One can also show that there are atmost $n/k + 2t + 1$ linearly independent equations among the $m$ equations.

# 3 Analysis of the scheme

Requirements for the scheme to work:

1. $m \leq 2^k$
2. $m \geq n/k + 2t + 1$

Hence, $mk \geq n + 2tk + k$. We want to minimise $mk$ and hence $2tk + k$. The least value $k$ can have is $k = [log_2 m]$ where [] is the ceiling function. i.e. min $mk$=$n + 2t[log_2 m] + [log_2 m]$. Hence, we are roughly adding $2[log_2 m]$ redundant bits for every error. If $n \sim 5GB$ and $t \sim 50MB$ then we need to add about $4GB$ of redundancy. Since $m \leq n$ (usually), then we have to add $O(tlogn)$ redundant bits for $t$ errors.