A parallel version of a special case of the Sudan decoding algorithm

Daniel Augot
INRIA, Domaine de Voluceau
F78153 Le Chesnay CEDEX
e-mail: `Daniel.Augot@inria.fr`

Abstract — **We consider the Sudan decoding algorithm of Reed-Solomon codes, in the particular case when the Y-degree of the bivariate $Q(X,Y)$ is equal to one. It is very similar to the Welch Berlekamp algorithm. The equation for finding $Q(X,Y)$ can be solved using an algorithm from R.R. Nielsen. Then we remark that all the univariate polynomials in these computations are of degree less than n, the length of the code, and that they can be represented by their evaluation on the support of the code. This leads to a simpler arithmetic on polynomials, which can also be parallelized. But, by the end of the process, there are unkown values at the positions of the errors, and they can be reconstructed using linear algebra.**

I. Introduction

Let $x_1,\ldots,x_n$ be n distinct elements of $\mathbb{F}_q$, where q is the finite field with q elements. We denote by ev the evaluation map:

$$\mathrm{ev}: \begin{array}{ccc} \mathbb{F}_q[X] & \to & \mathbb{F}_q^n \\ f(X) & \mapsto & \mathrm{ev}(f(X)) = (f(x_1),\ldots,f(x_n)) \end{array}$$

Let C be the Reed-Solomon code of dimension k with support $x_1,\ldots,x_n$, it is equal to

$$C = \{\mathrm{ev}(f(X));\ f(X)\in\mathbb{F}_q[X];\ \deg f(X) < k\},$$

and the minimum distance of C is $n = k+1$. The Sudan [3] list decoding algorithm can correct up to $\tau = n-\sqrt{2kn}$ errrors.

Let $y = (y_1,\ldots,y_n)$ be the received word to be decoded. The Sudan algorithm proceeds in two steps

1. find a polynomial $Q(X,Y)$ with $\mathrm{wdeg}\, Q(X,Y) < \tau$, and $Q(x_i,y_i) = 0$, $i = 1\ldots,n$.
2. find the roots $f(X)$ of $Q(X,f(X)) = 0$.

Now we consider the particular case where $\deg_Y Q(X,Y) = 1$. We have to find $Q(X,Y) = f(X)+Yg(X)$, with $\deg f(X) < n-\tau$ and $\deg g(X) < n-\tau-(k-1)$. Finding the root is easy: the solution is $f(X)/g(X)$. This case is very similar to the Welch-Berlekamp decoding algorithm [1, 2] and the algorithm can correct up to $t = (n-k+1)/2$ errrors.

II. R.R.N. algorithm

R.R. Nielsen has devised an algorithm for finding the polynomial $Q(X,Y)$ [4, 5]. In the case where $\deg_Y Q(X,Y) = 1$, it is very similar to algorithm described by Berlekamp in [1]. Let us describe the algorithm as follows. It can be seen that the complexity is $O(n^2)$.

for $i = 0$ to 1 do $g_i(X,Y) = Y^i$.
for $i = 1$ to n do
  $f(X,Y) \leftarrow g_{j_0}(X,Y)$, where $g_{j_0}(X,Y)$ is minimal and $g_{j_0}(x_i,y_i) \neq 0$.
  for $j = 0$ to 1 do
    if $j = j_0$ then $g_{j_0}(X,Y) \leftarrow (x-x_j)f(X,Y)$.
    else
      $g_j(X,Y) \leftarrow g_{j_0}(x_i,y_i)g_j(X,Y) - g_j(x_i,y_i)g_{j_0}(X,Y)$.
return $g_j(X,Y)$ minimal.

III. Adaptation in the transform domain

Observing that the polynomials $f(X)$ and $g(X)$ in $Q(X,Y) = f(X)+Yg(X)$ are of degree less than n, the above algorithm can be translated in the transform domain, using arrays of n values instead of polynomials. Multiplication of a polynomial by a constant is just scalar multipliaction of an array by a constant etc. The bivariate polynomials are represented by a polynomial in Y with coefficients polynomials in X, represented in the transform domain. From the point of view of implementation, the operations on the corresponding arrays are performed componentwise independantly, that is, in parallel. Thus, if n circuits for finite fields arithmetic are used, each one dedicated to one of the x_i's, the running time of this algorithm in the transform domain is $O(n)$.

IV. Missing values

The last step of the decoding procedure is to perform the division $f(X)/g(X)$, where $Q(X,Y) = f(X)+Yg(X)$. In the transform domain, this means to set the codeword equal to $f(x_i)/g(x_i)$, $i = 1\ldots n$. This step can not be performed for the x_i such that $g(x_i) = 0$. The polynomial $g(X)$ is known to be the locator polynomial of the errors of y. Thus when $Q(X,Y)$ is computed using arrays for $f(X)$ and $g(X)$, the positions where $g(x_i) = 0$ are read on the array. The direct output of the algorithm is then the value of the codeword at correct position, and undefined at positions where errors have occured. However, since at least k positions will be correct, the unknown value can computed form the known values, by a sort of interpolation mecanism.

References

[1] Elwyn R. Berlekamp. "Bounded Distance +1 Soft-Decision Reed-Solomon Decoding", *IEEE Transactions on Information Theory*, vol 42, n. 3, 1996.

[2] Elwyn R. Berlekamp and Loyd R. Welch, "Error correction of algebraic block codes" US Patent Number 6,633,470, 1986.

[3] Madhu Sudan "Decoding of Reed-Solomon codes beyond the error-correction bound", *Journal of Complexity*, vol. 13, 1997.

[4] Rasmus R. Nielsen "Decoding AG codes beyond half the minimum distance" Technical University of Denmark, available at http://www.student.dtu.dk/~p938546/public.html

[5] Tom Hoeholdt and Rasmus R. Nielsen, "Decoding Reed-Solomon codes beyond half the minimum distance" *International Conference on Coding Theory, Cryptography and Related Areas*, 1998.