ABSTRACT

The reverse engineering problem for genetic networks is the problem of determining the network that describes functional relations between genes, given a set of experimental data. A finite field model for reverse engineering is addressed in this work. In this model, reverse engineering large genetic networks involves intensive arithmetic computations over finite fields, where the most time consuming operation is multiplication. FPGA capabilities for accelerating finite field arithmetic are exploited for speeding up reverse engineering in a hardware/software environment. The main task in the overall design for reverse engineering is an interpolation algorithm, which demands successive multiplications. An important goal in this work is to carry out the whole interpolation algorithm on FPGAs. A space-saving architecture based on Newton's method is proposed for an efficient interpolation. In order to accelerate interpolation we are proposing a fast FPGA implementation of a finite field multiplier based on the Mastrovito matrix. In addition, NTT convolution is considered for accelerating multiplication over fields of odd prime characteristic.

THEORY AND METHODOLOGY

Interpolation

- Using interpolation for reverse engineering large genetic networks requires intensive arithmetic computations over finite fields. Namely, additions, multiplications, inversions, and exponentiations.

- Multiplication is the most costly operation in terms of computation time and circuit complexity. Inversion and exponentiation are performed by repeated multiplications.

- Taking advantage of our fast multiplier, we have implemented an efficient architecture for inversion based on Itoh-Tsujii's algorithm [4].

- We are proposing an architecture based on Newton's interpolation. This method uses more multiplications \((\log n)^2\)) than Lipson's algorithm \((\log n)\). However, the former method demands a prohibitive amount of storage resources.

- An important goal in this work is to carry out the whole interpolation algorithm on the FPGA. By shifting more workload to the FPGA, a better optimization of the ratio of computation time to communication time can be achieved.

Finite Field Multiplication

- Let \(\alpha(0), \alpha(1), \alpha(2), \ldots, \alpha(m)\) elements in \(GF(2^m)\) and \(F0\) the irreducible polynomial generating \(GF(2^m)\). Then the finite field multiplication \(\alpha(0) \cdot \alpha(1) \mod F0\) is accomplished by calculating

\[
\alpha(i) = \alpha(0) \cdot \alpha(1) \mod F0
\]

where \(\alpha(1)\) denotes polynomial multiplication

- In the Mastrovito method [6], finite field multiplication is expressed as

\[
C = ZB
\]

where \(B\) is a \(m \times m\) matrix, involving coefficients of \(\alpha(1)\). \(Z\) is called the Mastrovito matrix. \(B\) and \(C\) are vectors of dimension \(m\), consisting of the coefficients of \(\alpha(0)\) and \(\alpha(1)\) respectively, i.e.,

\[
B = [b0, b1, b2, \ldots, b_{m-1}]
\]

\[
C = [c0, c1, c2, \ldots, c_{m-1}]
\]

Example: Let \(\alpha(0), \alpha(1), \alpha(2), \ldots, \alpha(7)\) elements in \(GF(2^7)\) for \(F0 = x^7 + x^6 + 1\) an irreducible trinomial generating \(GF(2^7)\). The multiplication \(\alpha(0) \cdot \alpha(1)\) can be expressed as

\[
\begin{pmatrix}
\alpha(0) \\
\alpha(1)
\end{pmatrix} =
\begin{pmatrix}
\alpha(2) \\
\alpha(3) \\
\alpha(4) \\
\alpha(5) \\
\alpha(6)
\end{pmatrix}
\]

In \[3\] we describe a quite simple architecture that takes advantage of symmetries in the Mastrovito matrix.

In this research, Reverse Engineering is considered in the context of the univariate model for finite fields \(GF(2^m)\) [1]. In this model, a solution to the reverse engineering problem can be described as follows:

Given a sequence of points

\[
f(k) = s_k, f(k) = s_{2k}, \ldots, f(k) = s_{Nk}
\]

determine

\[
f(k) = f(x) + g(x)
\]

where \(f(x)\) is a polynomial that passes through the points \((k, s_k), (2k, s_{2k}), \ldots, (Nk, s_{Nk})\) and \(g(x)\) is a polynomial that vanishes on these points.

The polynomial \(f(x)\) can be found by interpolation and \(g(x)\) can be determined by further biological experiments.

EXPERIMENTAL RESULTS

- Target platform: Cray XD1 supercomputer including six FPGAs units tightly integrated to 12.2 GHz Opteron AMD processors through a high bandwidth interconnection system. FPGA units are Xilinx Virtex II-Pro xcv2p50

- Performance measurements: When compared against other methods, our multiplication approach shows better time performance [8].

- Inversion in \(GF(2^m)\) using Itoh-Tsujii’s algorithm with our multiplier and a multiplier based on the ordinary polynomial multiplication shows better time performance [3].

- Preliminary results in relation to Newton’s interpolation are promising. We have implemented an architecture which uses reasonably the space in the FPGA.

CONCLUSIONS AND FUTURE WORK

- An efficient FPGA-based implementation of a finite field multiplier has been introduced.

- Newton’s interpolation can be implemented in a space-saving architecture. So it is possible to carry out the whole interpolation algorithm in the FPGA.

- We expect to interpolate in a reasonable number of points over finite fields of the type \(GF(2^m)\) \((m \leq 500)\) in a single FPGA.

- For fields of odd prime characteristic, the number of operations used in multiplication can be asymptotically reduced by using convolution.

REFERENCES


