Parallel Reduction Algorithm of Multiple-Precision over Finite Field GF(2^n)

Yongnan Li[1], Limin Xiao[1], Aihua Liang[1], Zhenzhong Zhang[1], Li Ruan[1]

[1] School of Computer Science and Engineering, Beihang University, Beijing, 100191, China
{liyongnan1984, liangah, zzzhang}@cse.buaa.edu.cn {xiaolm, ruanli}@buaa.edu.cn

Abstract. This paper presents a parallel reduction algorithm concerning multiple-precision integer over finite field GF(2^n). The data dependency of the sequential reduction algorithm is analyzed to design the parallel algorithm. We take one time clock as the computation time unit to compute the time complexities of parallel algorithm and sequential algorithm. The speedup shows high efficiency of the proposed parallel algorithm.

Keywords: parallel algorithm; finite field GF(2^n); reduction

1 Introduction

The finite field GF(2^n) is one of the common used mathematical sets for constructing some cryptosystems including conic curves cryptosystem [1,2] and elliptic curves cryptosystem [3,4]. In recent years, the parallel algorithms of some fundamental operations over finite field GF(2^n) have received considerable attention due to massive computation caused by the increased security demands. However, there is less deep study on fast parallel algorithms concerning multiple-precision integers over finite field GF(2^n). Our previous works have designed several parallel algorithms about some basic operations of multiple-precision integers over two other mathematical sets [5-8]. This paper proposes an efficient parallel reduction algorithm of multiple-precision over finite field GF(2^n) to accelerate the speed of this basic operation over finite field GF(2^n).

2 Reduction Algorithm over GF(2^n)

The following algorithm is the traditional algorithm for computing reduction over GF(2^n).

Reduction over GF(2^n)
Input: module $f(Z) = Z^n + r(Z)$,
polynomial $C(Z) = C_{2n-2}Z^{2n-2} + \cdots + C_1 Z + C_0$.
Output: $C(Z) mod\ f(Z)$.
1. for *i* from $n-2$ to 0, repeat:

1.1 if $C_{i+n} = 1$, then
$j \leftarrow \lfloor i/W \rfloor, \quad k \leftarrow i - Wj, \quad C\{j\} \leftarrow u_k(Z) \oplus C\{j\}.$
2. return $(C[t-1], \cdots, C[1], C[0])$.

3 Parallel Reduction Algorithm over GF(2^n)

This section discusses the proposed parallel reduction algorithm over finite field GF(2^n). We take one time clock as the computation time unit to compute the time complexities of parallel algorithm and sequential algorithm.

The parameter *W* represents the word length of the computer and the parameter *n* denotes the degree of the modular polynomial $f(Z)$. The parameter $u_k(Z)$ means $Z^k r(Z)$ where $0 \leq k \leq W-1$. We define $m = \lceil (2n-1)/W \rceil$, so the polynomial $C(Z)$could be expressed as $(C[m], \cdots, C[1], C[0])$. The parameter $C\{j\}$ means $(C[m], \cdots, C[j+1], C[j])$.

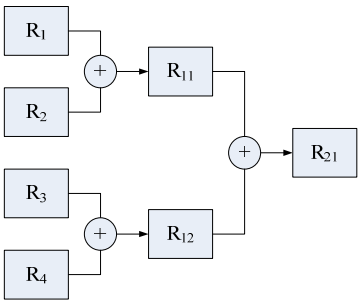

Fig. 1. An example of incorporating the intermediate results

In the first step, one comparison must be executed to decide whether or not to execute the two arithmetic operations and one logical operation. The average number for computing the other three operations in all rounds of the first step is $(n-1)/2$ for the reason that the probability of C_{i+n} equal to 1 is 0.5. In substep 1.1, operation $j \leftarrow \lfloor i/W \rfloor$ denotes computing the word length of $\{C(i+n), \ldots, C(n)\}$ and it could be calculated in one time clock. Operation $k \leftarrow i - Wj$ is used to get the number of the bits in the highest word of $\{C_i Z^i + \cdots + C_1 Z^1 + C_0\}$, so the operation of multiplication and operation of deduction are not needed to be computed. Only one time clock is needed to obtain the result of this operation. Operation $C\{j\} \leftarrow u_k(Z) \oplus C\{j\}$ means executing XOR for every bit of $u_k(Z)$ and $C\{j\}$ from the lowest bit to the highest bit and it also needs one time clock. To sum up, the total runtime of the sequential procedure for computing reduction over GF(2^n) is $5(n-1)/2$.

For the parallel procedure, every round of the first step can be calculated simultaneously except the operation of $C\{j\} \leftarrow u_k(Z) \oplus C\{j\}$. To obtain the final value of $C(Z) mod\ f(Z)$, the intermediate results of $u_k(Z)$ in substep 1.1 should be incorporated after the other operations finished. Obviously, the average number of the temporary result of $u_k(Z)$ is $(n-1)/2$. As depicted in Fig.1, we use the merging principle

to incorporate the intermediate results of $u_k(Z)$ and it will cost $\lceil log_2((n-1)/2) \rceil$ time clocks. Then the total parallel runtime of reduction over GF(2^n) is $\lceil log_2((n-1)/2) \rceil + 2$.

Therefore, the speedup is

$$S = \frac{5(n-1)/2}{\lceil log_2((n-1)/2) \rceil + 2}. \quad (1)$$

4 Conclusions

In this paper, we presented a fast parallel reduction algorithm of multiple-precision over finite field GF(2^n). The parallel algorithm is designed by analyzing the data dependency of the sequential algorithm. Time complexities of the parallel algorithms and the sequential algorithms are discussed to show the high efficiency of the proposed algorithm. We only discussed the method for paralleling reduction operation over finite field GF(2^n) in this paper. The future works may focus on parallelization of other basic operations over finite field GF(2^n).

Acknowledgments. This study is sponsored by the National "Core electronic devices high-end general purpose chips and fundamental software" project under Grant No. 2010ZX01036-001-001, the Hi-tech Research and Development Program of China (863 Program) under Grant No. 2011AA01A205 and the National Natural Science Foundation of China under Grant No. 60973008.

References

1. Cao, Z.: A public key cryptosystem based on a conic over finite fields Fp (in Chinese). Advances in Cryptology: Chinacrypt98, Science Press, pp.45–49 (1998)
2. Cao, Z.: Conic analog of RSA cryptosystem and some improved RSA cryptosystems. Natural Science Journal of Heilongjiang University. 16 (4), pp.5–18 (1999)
3. Koblitz N.: Elliptic curve cryptosystems. Mathematics of Computation, 48:pp.203-209 (1987)
4. Miller V.: Uses of elliptic curves in cryptography. Advances in Cryptology-Crypto'85. Lecture Notes in Comput. Sci, vol. 218, pp. 417–426, Springer-Verlag, Berlin(1986)
5. Li, Y., Xiao, L., Hu, Y., Liang, A., Tian, L.: Parallel algorithms for cryptosystem on conic curves over finite field Fp. In: 9th International Conference on Grid and Cloud Computing, pp. 163–167. IEEE Press, Nanjing, China (2010)
6. Li, Y., Xiao, L., Liang, A., Wang, Z.: Parallel point-addition and point-double for cryptosystem on conic curves over ring Zn. In: 11th International Conference on Parallel and Distributed Computing, Applications and Technologies, pp. 317–322. IEEE Press, Wuhan, China (2010)
7. Li, Y., Xiao, L.: Parallel point-multiplication for conic curves cryptosystem. In: 3rd International Symposium on Parallel Architectures, Algorithms and Programming, pp.116–120. IEEE Press, Dalian, China (2010)
8. Li, Y., Xiao, L., Chen, S., Tian, H., Ruan, L., Yu, B.: Parallel Extended Basic Operations for Conic Curves Cryptography over Ring Zn. In: 9th IEEE International Symposium on Parallel and Distributed Processing with Applications Workshops, pp. 203–209. IEEE Press, Busan, Korea (2011)