Square over Finite Field GF(2^n) using Self-Assembly of DNA Tiles

Yongnan Li[1,2], Limin Xiao[1,2], Li Ruan[1,2], Ke Xie[1,2] and Guangchao Yao[1,2]

[1] State Key Laboratory of Software Development Environment, Beihang University, Beijing, 100191, China
[2] School of Computer Science and Engineering, Beihang University, Beijing, 100191, China

liyongnan.buaa@gmail.com {xiaolm, ruanli}@buaa.edu.cn
{xieke89, yutianzuijin}@cse.buaa.edu.cn

Abstract

The tile assembly model is a highly distributed parallel model of molecular computation. Plenty of experiments have proved that the simple binary arithmetic could be fulfilled by the process of self-assembly of DNA (deoxyribonucleic acid) tiles. Finite field GF(2^n) is one of the most commonly used mathematic sets. A DNA computing system was designed based on the tile assembly model and applied to perform the operation of square over finite field GF(2^n). One concrete example is proposed to show the details of our tile assembly system.

Keywords: *DNA computing, Finite field GF(2^n), Square, Tile assembly model*

1. Introduction

The field of DNA based computing is a new technique of simulating biomolecular structure of DNA. Two major advantages of DNA computing lie in huge memory capacity and high parallelism. The tile assembly model [1, 2], one of the most important DNA computing model, is a formal model of crystal growth. The self-assembly process is defined as autonomous organization of combining components into structurally well-defined aggregates. The tile assembly system is a concept of the computation through self-assembly process of the DNA tiles based on the theoretical underpinnings of tile assembly model. It was fully explained that the crystal growth of the tile assembly system could accomplish such mathematic computations as addition and multiplication [3, 4].

The implementation of the tile assembly system is a process of folding a single long scaffold strand into an arbitrary shape by using small helper strands [5]. The tile assembly systems usually grow crystal by using double-crossover complexes as tiles [6]. Our tile assembly system is also designed based on the theoretical basis of these double-crossover complexes.

Finite field GF(2^n), in which carry bits do not need to be propagated, is one of the most commonly used mathematic sets for elliptic curves cryptosystem [7, 8] and conic curves cryptosystem [9-12]. This paper proposes one tile assembly system which could compute the operation of square over finite field GF(2^n). There is very little research that has been proposed to compute the arithmetic over finite field GF(2^n) in the tile assembly model except the researches [13, 14] based on a six-tuple tile assembly model, which is different from the four-tuple tile assembly model adopted in this paper. The four-tuple tile assembly model is a more sophisticated model used in recent years.

The rest of this paper is organized as follows. Section 2 introduces tile assembly model. Section 3 presents the tile assembly system of square over finite field GF(2^n). The computation complexity is discussed in Section 4. Section 5 provides a conclusion and a brief discussion of potential future research ideas.

2. Tile Assembly Model

The tile assembly model must be introduced firstly. $\sum$ is a four-tuple $\{\sigma_N, \sigma_S, \sigma_W, \sigma_E\} \in \sum^4$, including the binding domains on the north, south, west and east. The set of directions $D = \{N, S, W, E\}$ is a set of four functions from positions to positions i.e. $\mathbb{Z}^2$ to $\mathbb{Z}^2$.

The positions (x, y) and (x', y') are neighbors if $\exists d \in D$ such that $d(x, y) = (x', y')$. For a tile t, for $d \in D$, $bd_d(t)$ is referred as the binding domain of tile t on d's side. A special tile $empty =< null, null, null, null >$ represents the absence of all other tiles. The position relationships are listed as follows:

$$\begin{cases} E(x, y) = (x + 1, y) \\ W(x, y) = (x - 1, y) \\ S(x, y) = (x, y - 1) \\ N(x, y) = (x, y + 1) \end{cases} \tag{1}$$

A strength function $g : \sum \times \sum \to \mathbb{R}$, where g is commutative and $\forall \sigma \in \sum$, $g(null, \sigma) = 0$, denotes the strength of the binding domains, the value of which may be 0, 1 or 2 (called *null*, *weak*, *strong* bonds, respectively). It is common to assume that $g(\sigma, \sigma') = 0 \iff \sigma \neq \sigma'$. The binding domains determine the interaction between tiles when two tiles attach to each other. Finally, a tile system $\mathbb{S}$ is a triple $< T, g, \tau >$, where T is a finite set of tiles containing *empty* tile, g is a strength function, and $\tau \geq 0$ is a parameter about the temperature. This paper uses $g = 1$ to denote $\forall \sigma \in \sum$, $g(\sigma, \sigma) = 1$ and $\forall \sigma \neq \sigma'$, $g(\sigma, \sigma') = 0$.

If A is a configuration, then within $\mathbb{S}$, a tile *t* can attach to *A* at position (x, y) and produce a new configuration A'. The conditions are listed as follows:

$$\begin{cases} A(x, y) = empty \\ \Sigma_{d \in D}\ g(bd_d(t), bd_{d^{-1}}(A(d(x, y)))) \geq \tau \\ \forall (u, v) \in \mathbb{Z}^2, (u, v) \neq (x, y) \Rightarrow A'(u, v) = A(u, v) \\ A'(x, y) = t \end{cases} \tag{2}$$

Given a tile system $\mathbb{S} =< T, g, \tau >$, a set of seed tiles Γ, and a seed configuration $S : \mathbb{Z}^2 \leftarrow \Gamma$, one may attach tiles of *T* to *S* if the above conditions are satisfied. A tile can attach to a configuration only in empty positions and only if the appropriate binding domains match the tiles in neighboring positions.

Configuration produced by $\mathbb{S}$ on *S* is the process of attaching tiles from *T* to *S*. If this process terminates, the final configuration with no more attachments could be produced. If all possible final configurations are identical for every sequence of tile attachment, then $\mathbb{S}$ is said to produce a unique final configuration on *S*.

3. Square System

3.1. Square over Finite Field GF(2^n)

According to the characteristic of finite field GF(2^n), carry bits do not need to be propagated in the process of mathematic computation. For $a = a_{n-1}x^{n-1} + \cdots + a_1 x + a_0$ over finite field GF(2^n), the value of a^2 is $a_{n-1}x^{2n-2} + \cdots + a_2 x^4 + a_1 x^2 + a_0$.

3.2. Tile Assembly System of Square

Theorem 1 Let $\Sigma = \{00, 01, 0\#, 10, 11, 1\#, \#\#\}$, $g = 1$, $\tau = 2$, and *T* be a set of tiles over Σ. Then $\mathbb{S} =< T, g, \tau >$ computes the function $C = a^2$ over finite field GF(2^n).

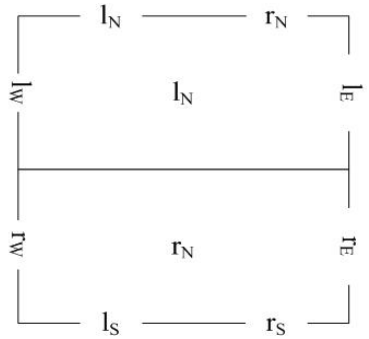


Figure 1. The Concept Tile

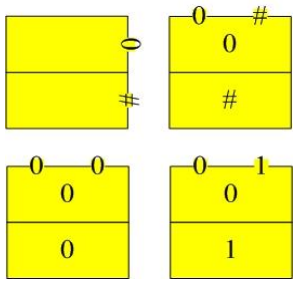


Figure 2. Boundary Tiles

Figure 1 shows the concept tile of the tile assembly system. The tile has two input sides (west and south) and two output sides (east and north). The first bit of every side is *l* and the second bit of every side is *r*. The boundary tiles and the computation tiles are showed in Figure 2 and Figure 3, respectively. Note that the colors are only used for a better understanding, the tiles themselves have no sense of color. The boundary tiles are used for constructing the seed configuration of tile assembly system. The magenta tiles assign the parameter $a_i (0 \leq i \leq n-1)$ into the final answer and identify the white tiles. The white tiles insert the '0' bit into the final solution. The green tiles perform the operation of a right-shift for parameter a_i $(0 \leq i \leq n-1)$ that has not been processed. The gray tiles are only used for passing the parameters from west side to east side or from south side to north side. A example of 1111^2 over finite field GF(2^n) is showed in Figure 4 and Figure 5. Two parameters, *a* and *C*, are coded as 1111 and 0000000 on the bottom row.

All tiles on leftmost column are coded as 0# . The top row reads the solution: 1010101 in Figure 5.

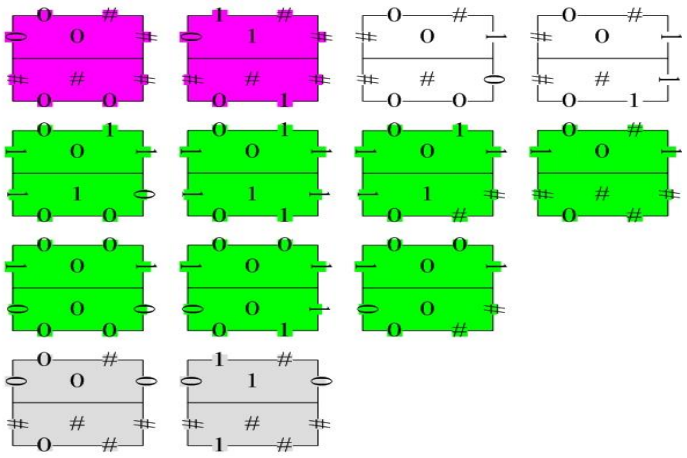


Figure 3. Computation Tiles

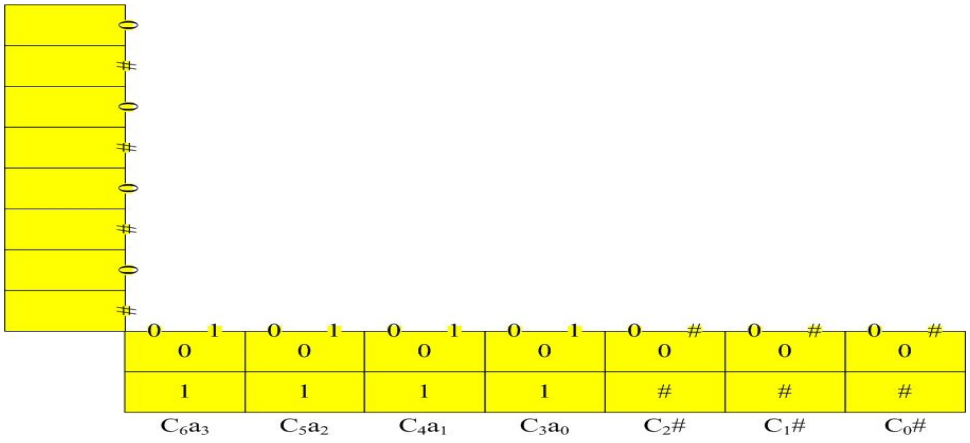


Figure 4. The Seed Configuration of a Sample Input of a=1111

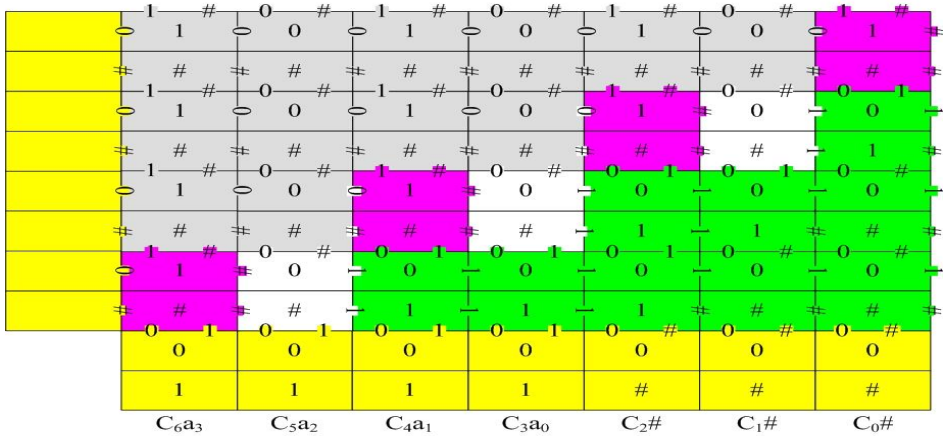


Figure 5. The Final Configuration of the Example 1111^2

Proof of Theorem 1 Consider the tile system $\mathbb{S}$. Let *a* be the input parameter of square over finite field GF(2^n). Let *C* be the result. The sizes, in bits, of *a* and *C*, are *n* and *2n-1*, respectively. For all $i \in \mathbb{N}$, let $a_i, C_i \in \{0,1\}$ be such that $a = a_{n-1} \ldots a_0$, and $C = C_{2n-2} \ldots C_0$. The initial value of parameter *C* is *0*.

Let $\Gamma = \{\ \alpha_{01} =< 01, null, null, null >,\ \alpha_{00} =< 00,\ null, null,\ null >,\ \alpha_{0\#} =< 0\#,$ *null, null, null* $>,\ \beta =<$ *null, null, null*, $0\# >\}$.

Then the seed configuration $S : \mathbb{Z}^2 \to \Gamma$ is such that

- $\forall i \in \{0, \ldots, n-1\}, S(i,-1) = \alpha_{0a_{n-1-i}}$
- $\forall i \in \{n, \ldots, 2n-2\}, S(i,-1) = \alpha_{0\#}$
- $\forall j \in \{0, \ldots, n-1\}, S(-1,j) = \beta$
- $for\ all\ other (x,y) \in \mathbb{Z}^2, S(x,y) = empty$

Σ has 13 types of computation tiles with the west side and the south side as the input sides, and the east side and the north side as the output sides. There would be only one single position where a tile may attach to *S* since its west neighbor tile and south neighbor tile are fixed.

Obviously, the self-assembly process begins from the position (*0,0*). For $\forall t \in T$, the two-tuple $< bd_S(t), bd_W(t) >$ is unique. It is certain that $\mathbb{S}$ produces a unique final configuration on *S*. The abutting binding domains of two tiles have to match each other when a tile attaches to *S*.

Let the final configuration be *F*. For all $0 \le i \le 2n-2$, $0 \le j \le n-1$, *S* and *F* agree on $S(i,-1)$ and $S(-1,j)$. Therefore, for $0 \le i \le n-1$, $bd_N(F(i,-1)) = 0a_{n-1-i}$; for $n \le i \le 2n-2$, $bd_N(F(i,-1)) = 0\#$; for $0 \le j \le n-1$, $bd_E(F(-1,j)) = 0\#$.

For the tiles with two-bit binding domains bd, let $l(bd)$ be the first bit and $r(bd)$ be the second bit of the binding domain. Let $t = F(i,j)$. Since t binds with two neighbor tiles, for $0 \le i \le 2n-2$, $0 \le j \le n-1$, $bd_S(t) = bd_N(F(i,j-1))$, $bd_W(t) = bd_E(F(i-1,j))$. For all $t \in T$, let $v(t) = l(bd_N(t))$. Thus, the initial inputs of all binding domains are

- $0 \le i \le 2n-2$, $l(bd_S(F(i,0))) = 0$
- $0 \le i \le n-1$, $r(bd_S(F(i,0))) = a_{n-1-i}$
- $n \le i \le 2n-2$, $r(bd_S(F(i,0))) = \#$
- $0 \le j \le n-1$, $l(bd_W(F(0,j))) = 0, r(bd_W(F(0,j))) = \#$

For $0 \le i \le 2n-2$, $0 \le j \le n-1$, all the followings are true in the process of tile self-assembly:

- $$l(bd_N(t)) = \begin{cases} r(bd_S(t)), & \text{if } l(bd_W(t)) = 0 \text{ and } r(bd_S(t)) \neq \# \\ 0, & \text{if } l(bd_W(t)) = \# \\ l(bd_S(t)), & \text{if } l(bd_W(t)) = 1 \\ l(bd_S(t)), & \text{if } l(bd_W(t)) = 0 \text{ and } r(bd_S(t)) = \# \end{cases}$$
- $$r(bd_N(t)) = \begin{cases} \#, & \text{if } l(bd_W(t)) = 0 \text{ and } r(bd_S(t)) \neq \# \\ \#, & \text{if } l(bd_W(t)) = \# \\ r(bd_W(t)), & \text{if } l(bd_W(t)) = 1 \\ r(bd_S(t)), & \text{if } l(bd_W(t)) = 0 \text{ and } r(bd_S(t)) = \# \end{cases}$$

- $l(bd_E(t)) = \begin{cases} \#, & \text{if } l(bd_W(t)) = 0 \text{ and } r(bd_S(t)) \neq \# \\ 1, & \text{if } l(bd_W(t)) = \# \\ 1, & \text{if } l(bd_W(t)) = 1 \\ l(bd_W(t)), & \text{if } l(bd_W(t)) = 0 \text{ and } r(bd_S(t)) = \# \end{cases}$
- $r(bd_E(t)) = \begin{cases} \#, & \text{if } l(bd_W(t)) = 0 \text{ and } r(bd_S(t)) \neq \# \\ r(bd_S(t)), & \text{if } l(bd_W(t)) = \# \\ r(bd_S(t)), & \text{if } l(bd_W(t)) = 1 \\ r(bd_W(t)), & \text{if } l(bd_W(t)) = 0 \text{ and } r(bd_S(t)) = \# \end{cases}$

4. Complexity Discussion

As $\tau = 2$, only one tile with two neighbors may attach at any time in this system. Therefore, no tile may attach to the configuration unless its west neighbor and south neighbor have already existed. When $F(2n-2, n-1)$ tile attaches to the position $(2n-2, n-1)$, this parallel molecular computation of square over finite field GF(2^n) will terminate. Obviously, the assembly time of this system is $T(n) = 2n - 1 + n - 1 = 3n - 2 = \Theta(n)$ and the space complexity is $S(n) = (2n-1)n = \Theta(n^2)$. This system of square requires 13 types of computation tiles and 4 types of boundary tiles.

5. Conclusion

A tile assembly system is proposed to compute the operation of square over finite field GF(2^n). Many researches have been proposed to deal with parallel computation of basic operations over this specific finite field. These studies mainly focus on reducing computing unit [15], accelerating computing speed [16] and lowering power consumption [17]. Differing from these researches, our work contributes on figuring out the result in linear assembly time, and it is supposed to obtain the solution space within as less steps as possible. This system could fulfill the process of self-assembly and figure out the solution in linear assembly time. The assembly time of this system is $T(n) = \Theta(n)$ and the space complexity is $S(n) = \Theta(n^2)$.

This tile assembly system was extended from the methods of implementing arithmetic computations used by Brun for binary addition and multiplication [3]. The described tile assembly system is designed theoretically based on the condition that all DNA operations are perfect. Although DNA computation has the problem of high error-rates, there are some existing methods of error control and error correction [18]. These researches suggest a bright future for DNA computing.

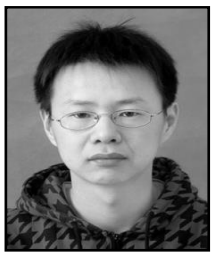
Acknowledgments

This study is sponsored by the fund of the State Key Laboratory of Software Development Environment under Grant No. SKLSDE-2012ZX-06, the Hi-tech Research and Development Program of China (863 Program) under Grant No. 2011AA01A205, Beijing Natural Science Foundation under Grant No. 4122042 and the National Natural Science Foundation of China under Grant No. 61232009.

References

[1] S. Rebecca and W. Erik, "Programmable control of nucleation for algorithmic self-assembly", Proceedings of 10th International Workshop on DNA Computing, Milan, Italy, (**2005**) June 7-10.
[2] W. Erik, "Algorithmic self-assembly of DNA", Proceedings of 2006 International Conference on Microtechnologies in Medicine and Biology, Okinawa, Japan, **(2006)** May 9-12.
[3] Y. Brun, "Arithmetic computation in the tile assembly model: Addition and multiplication", Theoretical Computer Science, vol. 378, **(2007)**, pp. 17-31.
[4] Y. Brun, "Nondeterministic polynomial time factoring in the tile assembly model", Theoretical Computer Science, vol. 395, **(2008)**, pp. 3-23.
[5] P. W. K. Rothemund, "Folding DNA to create nanoscale shapes and patterns", Nature, vol. 440, no. 7082, **(2006)**, pp. 297-302.
[6] P. Sa-Ardyen, A. V. Vologodskii and N. C. Seeman, "The flexibility of DNA double crossover molecules. Biophysical Journal, vol. 84, **(2003)**, pp. 3829-3837.
[7] F. B. Muhaya, Q. A. Al-Haija and L. Tawalbeh, "Applying hessian curves in parallel to improve elliptic curve scalar multiplication hardware", International Journal of Security and its Applications, vol. 4, no. 2, **(2010),** pp. 27-38.
[8] K. Edoh, "Elliptic curve cryptography on PocketPCs", International Journal of Security and its Applications, vol. 3, no. 3, **(2009),** pp. 23-34.
[9] Y. Li, L. Xiao, Y. Hu, A. Liang and L. Tian, "Parallel algorithms for cryptosystem on conic curves over finite field Fp", Proceedings of 9th International Conference on Grid and Cloud Computing, Nanjing, Jiangsu, China, **(2010)** November 1-5.
[10] Y. Li, L. Xiao, A. Liang and Z. Wang, "Parallel point-addition and point-double for cryptosystem on conic curves over ring Zn", Proceedings of 11th International Conference on Parallel and Distributed Computing, Applications and Technologies, Wuhan, China, **(2010)** December 8-11.
[11] Y. Li, L. Xiao, G. Qin, X. Li and S. Lei, "Comparison of three parallel point-multiplication algorithms on conic curve", Proceedings of 11th International Conference on Algorithms and Architectures for Parallel Processing, Melbourne, VIC, Australia, **(2011)** October 24-26.
[12] Y. Li and L. Xiao, "Parallel point-multiplication for conic curves cryptosystem", Proceedings of 3rd International Symposium on Parallel Architectures, Algorithms and Programming, Dalian, China, **(2010)** December 18-20.
[13] C. Zhen, "Computation of Multiplicative Inversion and Division in GF(2(n)) by Self-Assembly of DNA Tiles", Journal of Computational and Theoretical Nanoscience, vol. 9, **(2012)**, pp. 336-346.
[14] R. Barua and S. Das, "Finite field arithmetic using self-assembly of DNA tilings", Proceeding of the 2003 Congress on Evolutionary Computation, Canberra, Australia, **(2003)** December 8-12.
[15] A. E. Cohen and K. K. Parhi, "Fast reconfigurable elliptic curve cryptography acceleration for GF(2 m) on 32 bit processors", Journal of Signal Processing Systems, vol. 60, no. 1, **(2010)**, pp. 31-45.
[16] Y. Li and L. Xiao, "Parallelization of two arithmetic operations over finite field $GF(2^n)$", International Journal of Security and its Applications, vol. 6, no. 2, **(2012)**, pp. 223-228.
[17] S. C. Seo, D. Han and S. Hong, "TinyECCK16: An efficient field multiplication algorithm on 16-bit environment and its application to Tmote Sky sensor motes", Ieice Transactions on Information and Systems. E92-D, **(2009)**, pp. 918-928.
[18] Y. Baryshnikov, E. Coffman, N. Seeman and T. Yimwadsana, "Self-correcting self-assembly: Growth models and the hammersley process", Proceeding of 11th International Workshop on DNA Computing, London, ON, Canada, **(2006)** June 6-9.

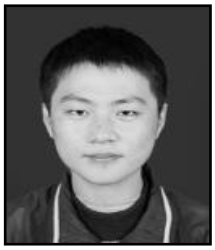
Authors



Yongnan Li is a Ph.D. student at School of Computer Science and Engineering, Beihang University. His main research areas are computer architecture, cloud computing, parallel computing and information security.

Limin Xiao is a professor of the Institute of Computer Architecture, Beihang University. He is also a senior membership of China Computer Federation. His main research areas are computer architecture, computer system software, high performance computing, virtualization and cloud computing.

Ruan Li is a lecturer of the Institute of Computer Architecture, Beihang University. She is also a senior membership of China Computer Federation. Her main research areas are virtualization and cloud computing, computer system software, high performance computer.



Ke Xie is a graduate student in Beihang University. His main research areas are computer architecture, parallel file system, computer system software, virtualization and cloud computing.

Guangchao Yao is a graduate student in Beihang University. His main research areas are computer architecture, computer system software, virtualization and cloud computing.