Int. J. Open Problems Comp. Math., Vol. 1, No. 3, December 2008

Integer and Vector Multiplication by Using DNA

Essam Al-Daoud[1], Belal Zaqaibeh[2], and Feras Al-Hanandeh[3]
[1]Computer Science Department, Zarka Private University, Jordan.
e-mail: essamdz@zpu.edu.jo
[2]Department of Computer Science, Jadara University, Jordan
[3]Department of Computer Information System, Hashemite University, Jordan.

Abstract

This paper presents a new procedure to perform the integer and vector multiplication by using the self assembly of 3D DNA nanostructure. The suggested procedure simulates the vector multiplication by using a look up table that represents many pairs of the vectors. The multiplication of each corresponding numbers can be performed by adding the carry and the result in each internal 3D tile of the assembled superstructures. The final result can be detected in the first row of the sum layer, where the sum layer uses the sticky ends to accumulate the result from each vertical layer. The suggested procedure is less energy consumer and can be used at very low cost.

Keywords: 3D Tile**,** Self-assembly, Nanostructure, Vector Multiplication, Matrix Multiplication.

1 Introduction

DNA computing is a form of computing which uses DNA, biochemistry and molecular biology to store information and perform complex calculations, instead of the traditional silicon-based computer. DNA computing is fundamentally similar to parallel computing in that it takes advantage of the many different molecules of DNA to try many different possibilities at once. DNA computing overlaps with, but is distinct from, DNA nanotechnology. The latter uses the specificity of Watson-Crick basepairing and other DNA properties to make novel structures out of DNA. These structures can be used for DNA computing[1].

In 1994, University of Southern California computer scientist Leonard Adelman suggested that DNA could be used to solve complex mathematical problems. Adelman found a way to harness the power of DNA to solve the traveling salesman. In his experiment he solved an instance of the Directed Hamiltonian Path Problem, using DNA strands to encode the problem and biological operations to simulate the computation [2]. Since then, algorithms for different combinatorial problems have been proposed, different models of computation with DNA have been considered and many experiments have been conducted, see [3] and [4]. DNA's key advantage is that it will make computers smaller than any computer that has come before them, energy efficiency, while at the same time holding more data. More than 10 trillion DNA molecules can fit into an area no larger than 1 cubic centimeter. With this small amount of DNA, a computer would be able to hold 10 terabytes of data, and perform 10 trillion calculations at a time [5].

In this paper, a novel procedure for performing the vector multiplication based on self-assembly of 3D DNA nanostructures is proposed. DNA computers using the suggested procedure can solve substantially large size problem because of their massive parallelism.

The remainder of this paper is organized as follows. Section 2 presents the most useful 2D and 3D DNA nanostructures that can be made from two or more DNA strands. Section 3 illustrates a new integer multiplication procedure, in this procedure the result of each bit in each row is added to the carry. Section 4 describes the suggested vector multiplication procedure.

2 2D and 3D Nanostructure

Winfree [6] first came up with the idea of computing using the automatic assemble of DNA nanostructure, called tiles, into complex superstructure. Tiles self-assembly of DNA can perform Turing-universal computation—which implies that any algorithm can in principle be embedded in, and guide, a potentially a periodic crystallization process. In this paradigm, a set of molecular Wang tiles is viewed as the program for a particular computation or molecular fabrication task. This model offers not only new capabilities for computation and construction, but also presents a new range of physical phenomena and experimental challenges as well. DNA nanostructure have sticky single- stranded ends “pads” behave as tiles in a two or three dimensional assembly and can be used to construct complex superstructure. Due to the extremely small size of DNA strands, as many as 10^{18} DNA tiling assemblies may be made simultaneously in a small test tube, and each superstructure may also occur at many locations via local parallelism. Thus DNA tiles can be used to perform a specific algorithm very fast. Algorithm's values or symbols can be encoded by using the sticky ends of tile. The tile that attaches to this tile would have the complementary sticky end encoding the same value or symbol. Tiles can be designed to encode, pass and process the data. Many different two-dimensional tiles can be made from two or more DNA strands such as the following tiles:

DX: consists of two double-helices interlocked by exchange of oligonucleotide strands at two separate crossover points. DX complexes have four termini, one at each end of the two strands. DX tiles come in five varieties (DAO, DAE, DPE, DPOW and DPON) that differ from one another in the geometry of the strand exchange and the topology of the strand paths through the tile.

TAO: Triple helix with Anti-parallel crossover and an Odd number of helical half turns between crossovers. TAO tile forms by the annealing of four DNA single strands and it has four sticky ends at the four corners.

TAE: Triple helix with Anti-parallel crossover and an Even number of helical half turns between crossovers. TAE tile has six sticky ends and formed by six DNA strands three on the left and three on the right. Figure 1 shows six TAO tiles joined together diagonally, then the reporter passing through the assembly from bottom left to top right.

Holliday junction: This structure is named after Robin Holliday. Four-arm junction can be made using four individual DNA strands which are complementary to each other in the correct pattern [7].

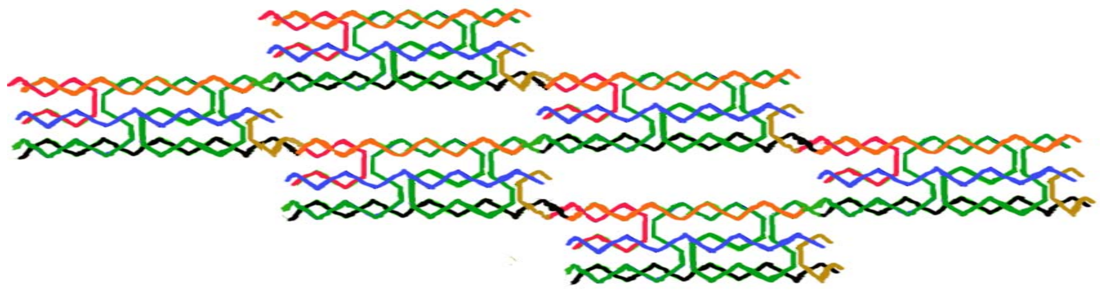
Fig. 1: Six TAE tiles joined diagonally

Winfree used the first and last layers of the assembly for input and output, respectively; this allows the input and output to be DNA single strands rather than assemblies of tiles, the output single strand (the reporter) can be used to confirm the success of formulation of the desired structure and to extract the results of a computation assembly. The input can be a long DNA strands capable of serving as nucleation points for assembly [8].

3D Nanostructure has six sticky ends and can be formed by many DNA strands. Kao and Ramachandran introduced new algorithms to build a Hollow cube, which is intuitively one of the simplest 3D structures to construct [9]. Hollow cube can be modified to attach the required sticky ends as in figure 2.

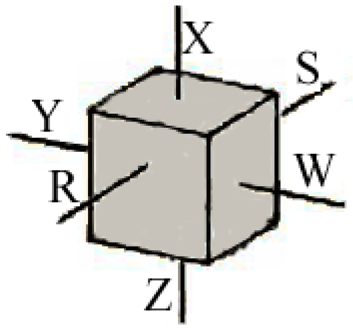

Fig. 2: Hollow cube with six sticky ends

3 Integer Multiplication

A huge number of DNA Tiles (around 10^{18} DNA tiles) can be used to encode the integer numbers and to simulate multiplying of many integer pairs simultaneously; therefore a large look up table of integer multiplication can be constructed. Integer multiplication can be calculated by multiplying and adding the carry and the result for each bit in each row. The same process can be simulated by using the self-assembly of the DNA nanostructures. Figure 3 shows the base tiles that will be used in the integer multiplication procedure, each base tile with a diagonal reporter can be designed by using for example TAE. To force the DNA nanostructures to assemble in the required way, we have to encode the four sticky ends. Figure 4 shows a tile with the encoded sticky ends where the ends X and W in the base tiles are replaced by a random binary number from 0 to 7, while the ends Y and Z are replaced by three bits that are described in figure 4, where:

$$\text{Carry}_{k+1} = \text{The left bit of } (q_j * p_i + \text{Carry}_k + \text{Result}_s)$$
$$\text{Result}_{s+1} = \text{The right bit of } (q_j * p_i + \text{Carry}_k + \text{Result}_s)$$

Procedure 1 is a new procedure can be used to simulate the integer multiplication in Figure 4, where the number p or q can be extracted from the constructed superstructures by denaturing the selected superstructures and analyzing the reporter using PCR and gel electrophoresis

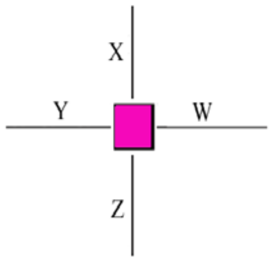

Fig. 3: Base tile with four sticky ends

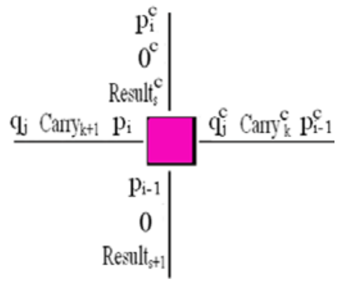

Fig. 4: Tile with the encoded sticky ends

Procedure 1: Integer Multiplication
Input: two integer numbers p and q.
Output: the result of p*q
Steps:

1. Prepare the base tiles as figure 1, and Let T_0 contains around 10^{18} base tiles.
2. Encode the base tiles by appending a new sticky ends that is described in procedure 2 (in this step we have to encode the tiles to force it to simulate the integer multiplication).
3. Merge the encoded tiles (in this step the self assembly will occur at many points simultaneously and therefore massive parallel integer multiplication will be performed. Each multiplication is represented by two dimensional superstructures).
4. Pick up the superstructure that contains p and q. This step can be achieved by applying the magnetic bead separation technique to extract the required superstructure.

Procedure 2: Encoding of the tiles
Input: Base Tiles
Output: Encoded Tiles
Steps:

1. Let A=000, B=001,C=010, D=011, E=100, F=101, G=110, H=111, I=00e, J=01e, K=10e, L=11e.
2. Let $Data_1$=A, $Data_2$=B, …, $Data_{12}$=L, $Data_{13}$=A^c, $Data_{14}$=B^c,…,$Data_{24}$=L^c.(Note

that $A^c \neq 111$ but A^c is the complement of the nucleotides that are used to represent 000).

3. Encode the top and the right tiles in Figure 5 which will be used to represent p and q. To achieve this step, take a small amount from tube T_0 and move it to Tube T_1, append A to the end Z of the tiles in T_1, these tiles will be used to represent the bit 0 in the number p, and then move T_1 content to T_2. Take another amount from T_0 to T_1 and append B to the end Z, these tiles will be used to represent the bit 1 in the number p, and then move T_1 content to T_2. Repeat the same procedure for the number q, but append A and E to the end Y to represent 0 and 1 respectively.
4. Encode the intermediate tiles as follows:

```
For i=1 to 12
 For j=1 to 12
   Take small amount from tube $T_0$ and move it to Tube $T_1$. Do the following
   operations on the tiles in $T_1$:
      Append the complement of $Data_i$ to the end X
      Append the complement of $Data_j$ to the end W
      Append U to the end Y
      Append V to the end Z
      Where
      $U^1$ (the first bit)=$(E_i)^1$
      $U^2$ (the Second bit)= The carry (left bit) of
                  $[(Data_i)^1*(Data_j)^3+(Data_i)^3+(Data_j)^2]$
      Note that If $(Data_i)^1$=e then $(Data_i)^1*(Data_j)^3=0$
      $U^3$ (the third bit)=$(E_j)^3$
      $V^1$ (the first bit)=$(E_j)^1$
      $V^2$ (the Second bit)= 0
      $V^3$ (the third bit)= The result (right bit) of
                  $[(Data_i)^1*(Data_j)^3+(Data_i)^3+(Data_j)^2]$
     Note that: If $(Data_i)^1$=e then $(Data_i)^1*(Data_j)^3=0$
     If $U^1$ =e or $(Data_j)^3$=e then cut or block the left and the top ends of the
     current tiles
  Move all $T_1$ content to $T_2$
```

5. Return T_2.

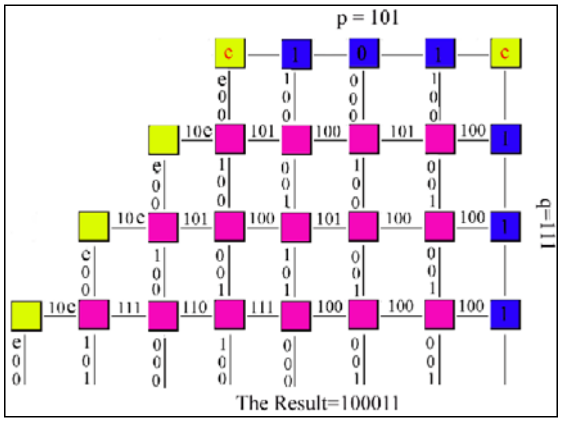

Fig. 5: Superstructure represents the result of p*q

4 Vector Multiplication

Figure 6 is a superstructure represents the two vectors (5, 2, 9, 10, 6, 4, 3) and (7, 5, 7, 11, 9, 5, 8)^t. A look up table can be prepared to represent many pairs of the vectors. The multiplication for each corresponding numbers can be performed by using procedure 1. The result of the two vectors multiplication can be found by passing and adding the numbers through the sticky ends R and S. Figure () shows an example of a vector multiplication where the sum layer uses the sticky ends R and S to accumulate the result from each vertical layer. The final result can be detected in the first row of the sum layer in Figure7.

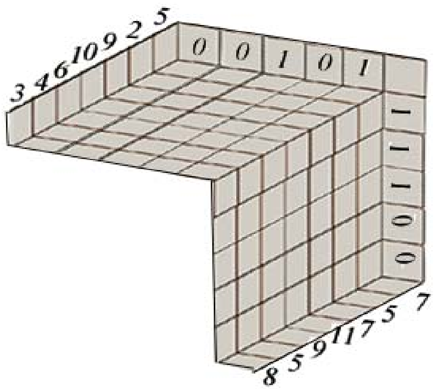

Fig. 6: Superstructure represents two vectors

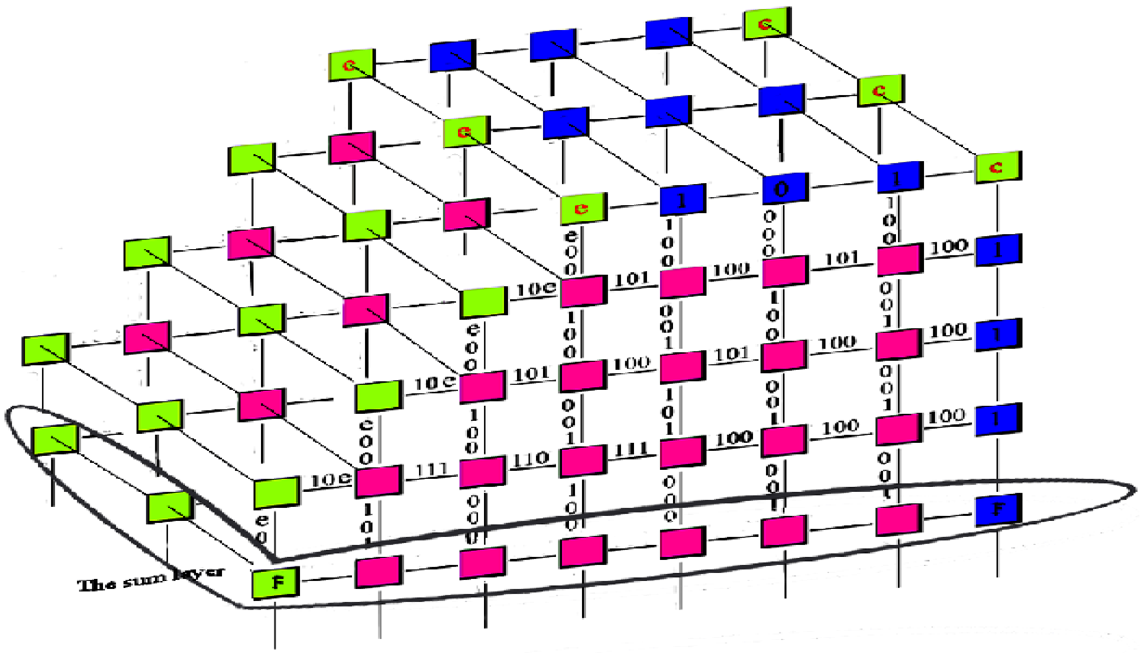

Fig. 7: Superstructure represents two vectors multiplication with the sum layer.

5 Conclusion and Open Problems

DNA computing is a fast developing interdisciplinary area. R&D in this area concerns theory, experiments and applications of DNA computing. The unique data representation of the DNA tiles and the ability to perform many parallel operations, allow us to look at a computational problems from a different point of view. This paper presented a new procedure for performing vector multiplication by using the self assembly of 3D DNA nanostructure. The final result can be detected in the first row of the sum layer, where the sum layer uses the sticky ends to accumulate the result from each vertical layer. DNA computers using the suggested procedure can solve substantially large size problem because of their massive parallelism. However there are many questions must be answered before DNA computer is adopted:

- How can we use the parallel molecular computation using DNA tiling self-assembly in which a large number of distinct inputs are simultaneously processed?
- What are the shapes or the tasks that can be achieved by self assembly?
- How DNA tiles can be used to recognize Chomsky Language?
- What is the minimum number of steps to make a shape or to perform a task?
- What is the minimum number of tiles to make a shape or to perform a task?

References

[1] J. H. Reif, “Parallel molecular computation” *SPAA*, (1995), pp. 213-223.
[2] L. Adleman, ”Molecular computation of solutions to combinatorial problems,” *Science 266*, (1994), pp. 1021-1024.
[3] D. Soloveichik, and E. Winfree, “Complexity of self-assembled shapes,” *SIAM J. Comput*, vol. 36, (2007), pp. 1544-1569.
[4] C. Mao,T. LaBean, J. H. Reif, and N.C. Seeman, “Logical computation using algorithmic self-assembly of DNA triple-crossover molecules,” *Nature 407*, vol. 6803, (2000), pp. 493–496.
[5] Y. M. Baryshnikov, E. G. Coffman, and J. P. Momcilovic, “DNA-based computation times” *DNA*, (2004), pp. 14-23.
[6] T. LaBean, E. Winfree, and J. Reif, "Experimental progress in computation by self-assembly of DNA tilings," *DIMACS Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, Providence*, vol. 54, (2000), pp. 123-140.
[7] K. V. Gothelf and T. H. LaBean, “DNA-programmed assembly of nanostructures,” *Organic & Biomolec. Chem*., vol. 3, (2005), pp. 4023-4037.
[8] Y. M. Baryshnikov, E. G. Coffman, and J. P. Momcilovic, “On Times to Compute Shapes in 2D Tile Self-assembly” *DNA*, (2006), pp. 215-222.
[9] M. Kao and V. Ramachandran “DNA Self-Assembly For Constructing 3D Boxes” *Lecture Notes in Computer Science, Springer Berlin, Heidelberg* , (2001), pp. 429-441.