

A Nano-DNA-Origami Probe Machine

JIANZHONG.CUI¹, XUEMEI.SHI^{1,*}

^{1,*}School of Information and Engineering, Huainan Union University, 232038 AnHui, P.R. China

*Corresponding author: XueMei Shi (e-mail: 784909013@qq.com).

This work was supported by Natural Science Foundation of Anhui Province [Grant No. 2024AH040222], Huainan Union University 2024 University-Level Quality Engineering Project [Grant No.XTSZY2404, XXQKC2403, XJZD2404].

ABSTRACT This paper presents a practical implementation of a Probe Machine (PM) using DNA nano-origami for efficiently solving the Boolean satisfiability (SAT) problem. We encode variable assignments of a 3-SAT formula into DNA-based data structures and design specific probes to interact with complementary pairs. Through a single-step probe operation on a DNA origami platform, the satisfying assignment is generated rapidly and detected via Atomic Force Microscopy (AFM). This approach allows the direct physical readout of solutions without exhaustive search. We demonstrate the model using a hard 4-variable 3-SAT instance and provide a general probe-based criterion for determining satisfiability. The encoding process scales linearly with the number of variables, while solution generation occurs in constant time, $O(1)$. The key advantage of this method is its highly parallel and scalable architecture, which leverages molecular parallelism to overcome limitations of conventional and bio-molecular SAT solvers. This work offers a tangible and scalable bio-molecular computing platform with potential applications in computational verification, optimization, and beyond.

INDEX TERMS SAT problem, DNA origami, molecular computation.

Date of Submission: 08-09-2025

Date of acceptance: 19-09-2025

I. INTRODUCTION

The Boolean satisfiability problem (SAT) occupies a central role in computer science, with critical applications spanning hardware verification, artificial intelligence, automated reasoning, and optimization[1,2]. As a canonical NP-complete problem, SAT solvers are indispensable in practice; however, their computational demands increase dramatically as problem size increases, challenging the limits of conventional silicon-based computing paradigms.

Traditional approaches to SAT can be broadly categorized into two classes: complete algorithms implemented on Turing machines (e.g., DPLL and its enhancements) and bio-molecular methods employing DNA computing techniques[3,4]. While software-based SAT solvers have made remarkable progress through advanced heuristics and learning strategies, their underlying sequential architecture inherently limits parallelism. On the other hand, early DNA computing models achieved massive parallelism through molecular operations but suffered from exponential resource growth and experimental complexity, restricting scalability[5-7].

The recently proposed Probe Machine (PM) offers a new computational model that transcends these limitations by leveraging intrinsic physical and molecular parallelism. In the PM framework (Figure.1), computation is achieved via interactions between data and probes, mimicking synaptic connectivity in neural systems. This structure supports a non-von Neumann computational paradigm capable of generating solutions in constant time, independent of problem size[8-10].

In this study, we implement a PM-based solution to the 3-SAT problem using a nano-DNA-origami architecture. By encoding variables and clauses into DNA nanostructures and designing specific probes to represent logical constraints, we demonstrate that the satisfying assignment can be produced in a single operational step and visually resolved via atomic force microscopy (AFM). This approach effectively transforms a symbolic satisfiability question into a biomolecular assembly process with physical readout.

The remainder of this paper is organized as follows. Section 2 provides background on the SAT problem and reviews existing solution approaches. Section 3 details the architecture and biomolecular design of the proposed Nano-DNA-Origami PM. Section 4 presents complexity analysis and theoretical results. Finally, Section 5 concludes with a discussion of implications and future directions.

II. MATERIALS AND METHODS

A. SATISFIABILITY PROBLEM

The Boolean satisfiability problem (SAT) is a fundamental decision problem in computational theory and logic. Formally, let $V = \{x_1, x_2, \dots, x_n\}$ be a set of Boolean variables. A literal L is either a variable x_i or its negation $\neg x_i$. A clause C_i is a disjunction (\vee) of literals. A propositional formula F in conjunctive normal form (CNF) is a conjunction (\wedge) of clauses. An assignment is a function $f: V \rightarrow \{T, F\}$. The formula F is satisfiable if there exists an assignment such that F evaluates to T ; otherwise, it is unsatisfiable. For instance, the following formula F is satisfiable since there exists an assignment that makes the formula *True*. And the assignment is $(x_1=T, x_2=F, x_3=T, x_4=F)$.

$$F = (\neg x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \neg x_3) \wedge (x_1 \vee \neg x_2 \vee x_3) \wedge (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_3 \vee \neg x_4) \wedge (\neg x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee \neg x_3)$$

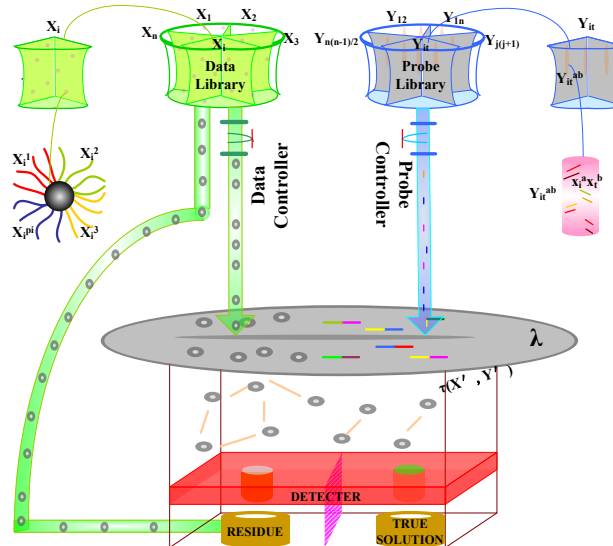


FIGURE 1. Schematic diagram of PM

Existing algorithms for SAT can be broadly classified into two categories: those implemented on Turing-machine (TM) architectures and those employing biomolecular paradigms.

TM-based algorithms predominantly derive from the Davis-Putnam-Logemann-Loveland procedure and its enhancements. These methods systematically explore the assignment space through branching, deduction, and backtracking. Modern solvers integrate advanced techniques including clause learning, non-chronological backtracking, and restarts, significantly improving performance on structured instances. Nevertheless, their worst-case time complexity remains exponential, reflecting the intrinsic difficulty of NP-complete problems.

Biomolecular approaches leverage massive parallelism inherent in chemical reactions and molecular interactions. Adleman's seminal DNA computing experiment demonstrated the feasibility of solving combinatorial problems using biological operations. Subsequent research extended this to SAT, employing techniques such as sticker models, ligase chain reactions, and membrane computing. While these methods achieve theoretical parallelism, they often require pre-synthesized pools of DNA strands representing all possible assignments, leading to exponential growth in material and time resources. Moreover, issues such as error rates, yield efficiency, and signal detection impose practical constraints on scalability.

Recent efforts have sought to combine molecular parallelism with addressable nanostructures to mitigate these limitations. In particular, DNA origami provides a programmable substrate for organizing molecular components with nanoscale precision, enabling localized computation and simplified readout.

Against this background, we propose a hybrid architecture: the Nano-DNA-Origami Probe Machine, which integrates the theoretical framework of the Probe Machine with the structural control of DNA origami. This approach encodes logical variables and constraints into DNA nanostructures and uses hybridization probes to perform satisfiability testing in a highly parallel and spatially organized manner. The aim is to overcome key limitations of both conventional and earlier biomolecular methods by reducing biochemical complexity and enabling direct physical observation of solutions.

B. PM-BASED COMPUTING MODEL

For the given 3-SAT instance introduced in subsection A, each possible variable assignment was encoded as a 4-digit binary sequence, where the number of digits corresponds to the number of variables. In this encoding, a bit value of ‘1’ indicates that the corresponding Boolean variable is assigned True (*T*), while ‘0’ represents False (*F*). The position of each bit, from left to right, corresponds to a specific variable x_i (where $i=1,2,3,4$). Each binary sequence was then transformed into a hyphen-delimited string by inserting hyphens between adjacent bits. For instance, the satisfying assignment ($x_1=T, x_2=F, x_3=T, x_4=F$) was first represented as the binary number 1010, and subsequently converted into the string ‘1-0-1-0’, which contains three hyphens. This procedure generated a total of 16 distinct strings, each comprising 7 characters. Each character in the string is referred to as either a variable character (at position i , where $j=1,2,3,4$) or a hyphen (at position j , where $j=1,2,3$). A key challenge addressed here is how to design the data and probe library of the proposed model in order to create the string satisfying the given formula?

To answer this question, the adjacency relationship between variable character x_i and x_j ($i, j=1,2,3,4, i \neq j$) were investigated pairwise. Apparently, variable character x_1 is solely adjacent to x_2 on the right of x_1 via the 1th hyphen. That is to say, x_1 has only one neighbor x_2 on the right. Variable character x_2 is not only adjacent to x_1 on the left of x_2 but also adjacent to x_3 on the right of x_2 . That is to say, x_2 has both a left and a right neighbor. Similarly, x_3 has both a left and a right neighbor, as well. x_4 has only one left neighbor. The adjacency relationships of variable characters were organized into the following table I.

TABLE I
THE ADJACENCY RELATIONSHIP OF VARIABLE CHARACTERS

	X ₁	X ₂	X ₃	X ₄
X ₁		L		
X ₂	R		L	
X ₃		R		L
X ₄			R	

Then the table was re-organized into an adjacent matrix as follows:

$$\begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ & L & & \\ R & & L & \\ & R & & R \\ & & R & \end{pmatrix}$$

Based on the matrix, the design of data was presented as follows:

$$\begin{pmatrix} x_1 & \neg x_1 & x_2 & \neg x_2 & x_3 & \neg x_3 & x_4 & \neg x_4 \\ x_1^R & \neg x_1^R & x_2^L & \neg x_2^L & x_3^L & \neg x_3^L & x_4^L & \neg x_4^L \\ & & x_2^R & \neg x_2^R & x_3^R & \neg x_3^R & & \end{pmatrix}$$

The adjacent matrix was transformed into a new matrix with 8 columns, in contrast to pre-transformed 4 columns, which corresponded to 8 types of data, respectively. Data were enumerated separately on the top of each columns, denoted by x_i or $\neg x_i$ ($i=1,2,3,4$). Beneath each data in each column, data fibers (fiber) belonging to the data were (was) enumerated vertically. Denote data fibers by x_i^j or $\neg x_i^j$ ($i=1,2,3,4, j=L,R$). 2 types of data body was chosen to attach these 12 types of data fiber, one for data x_i , the other for data $\neg x_i$. Data bodies were not listed in the transformed matrix. Therefore, once data fibers and data bodies were determined, the structures of all data were subsequently determined. That completed the design of data that targeted at the string in the proposed model. There were a total of 8 types of data, which consisted of 12 types of data fiber attached to 2 types of data body. The schematic diagram of all data was presented in Figure 2. Data bodies were represented by 2 types of sphere with different size.

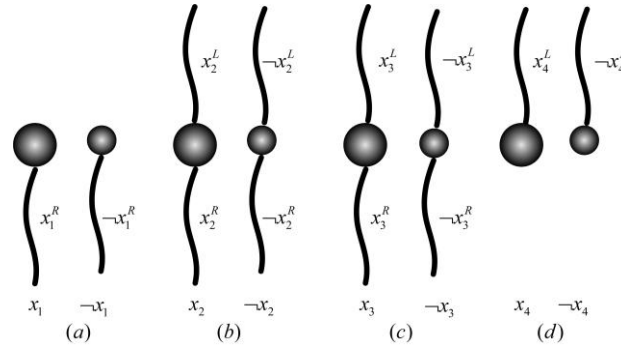


FIGURE 2. Schematic diagram of 8 types of data. (a), (b), (c), and (d) correspond to variable character x1, x2, x3, and x4, respectively.

For each type of data, a data sub-library was constructed, denote by X_i ($i = 1, 2, \dots, 8$). The constructed data sub-libraries were listed as follows:

$$X_1 = \{x_1\}, X_2 = \{-x_1\}, X_3 = \{x_2\}, X_4 = \{-x_2\}, \\ X_5 = \{x_3\}, X_6 = \{-x_3\}, X_7 = \{x_4\}, X_8 = \{-x_4\}$$

Data library X was ultimately constructed and listed as follows:

$$X = X_1 \cup X_2 \cup \dots \cup X_8.$$

Next the design of probes was presented. The main task of the design is to figure out compositions of all probes in probe library. As described in Introduction, probe consists of two complements of data fibers belonging to different data. For example, if there exists a probe between data fiber x_1^R and x_2^L , the complement of x_1^R , denoted by $\overline{x_1^R}$, together with the complement of x_2^L , denoted by $\overline{x_2^L}$, form that probe, denoted by $\overline{x_1^R x_2^L}$. Under certain circumstances, data itself is of the main concern, instead of data fibers. In such case, data x_1 and x_2 are called potential probe, which means there exists at least one type of probe between data fibers of data x_1 and x_2 .

In order to figure out compositions of probes, so that the satisfying string can be created, all possible values assigned to variables in the formula were arranged in a binary tree and presented in the following Figure 3.

For the first clause in the formula, $c_1 = (\neg x_1 \vee x_2 \vee x_3)$, the unsatisfying assignments is $(x_1 = 1, x_2 = 0, x_3 = 0)$. Since the assignment evaluates clause c_1 F , hence evaluates the formula F . Therefore, leafs 1000 and 1001 at the bottom in the tree, must be unsatisfying assignments. That is to say, according to clause c_1 , leafs 1000 and 1001 should be pruned from the tree. Two c_1 s were marked under leafs 1000 and 1001, respectively.

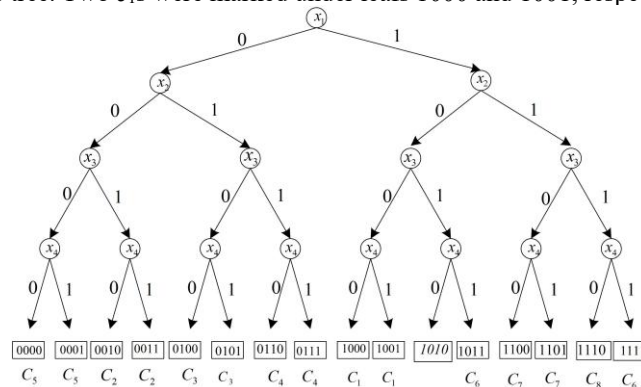


FIGURE 3. The pruning of the binary tree according to the given formula.

For the second clause, $c_2 = (x_1 \vee x_2 \vee \neg x_3)$, the unsatisfying assignments is $(x_1 = 0, x_2 = 0, x_3 = 1)$. According to clause c_2 , leafs 0010 and 0011 should be pruned from the tree. Two c_2 s were marked under leafs 0010 and 0011, respectively.

The same procedure was applied to the remaining clauses of the given formula. The algorithm pruned the leaf nodes representing assignments 0000, 0001, 0100, 0101, 0110, 0111, 1011, 1100, 1101, 1110, and 1111 in a clause-wise manner. It is worth emphasizing that the pruning operations can be executed concurrently. After processing all clauses, the unique satisfying assignment, 1010, was identified. This corresponds to the hyphen-delimited string "1-0-1-0", which implies the variable assignment: $x_1 - \neg x_2 - x_3 - \neg x_4$.

Therefore, based on the constructed data library X , compositions of all probes were determined in the probe library. There exist 3 types of probe : $\overline{x_1^R - x_2^L}$ between data fiber x_1^R of data x_1 and data fiber $-x_2^L$ of data $-x_2$, probe $\overline{-x_2^R x_3^L}$ between data fiber $-x_2^R$ of data $-x_2$ and data fiber x_3^L of data x_3 , and probe $\overline{x_3^R - x_4^L}$ between data fiber x_3^R of data x_3 and data fiber $-x_4^L$ of data $-x_4$. That is to say, data x_1 and $-x_2$, $-x_2$ and x_3 , x_3 and $-x_4$ are potential probe. These 3 types of probe were presented in the following Figure 4.

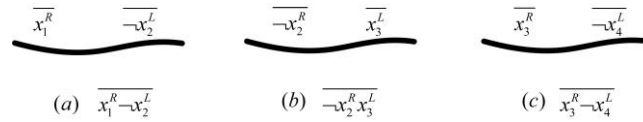


FIGURE 4. Schematic diagram of 3 types of probe.

Thus, targeted at the string and based on the constructed data library X , the design of probes completed. For each type of probe, a probe sub-library was constructed, denote by $Y_{i(i+1)}^{RL}$, ($i=1,2,3$). There were a total of 3 types of probe sub-library. The constructed probe sub-libraries were listed as follows:

$$Y_{12}^{RL} = \{\overline{x_1^R - x_2^L}\}, Y_{23}^{RL} = \{\overline{-x_2^R x_3^L}\}, Y_{34}^{RL} = \{\overline{x_3^R - x_4^L}\}.$$

Probe library Y was ultimately constructed and listed as follows:

$$Y = Y_{12}^{RL} \cup Y_{23}^{RL} \cup Y_{34}^{RL}.$$

In general, the design of probes is based on the following principles.

Principle1: For the data in data sub-library X_i , there does not exist probe between data fiber pair belonging to the data.

Principle2: Probe only possibly exists between data fibers belonging to different data.

Data controller took an appropriate amount of data from data library; meanwhile probe controller took an appropriate amount of probes from probe library, and placed them into computing platform to execute probe operation. In the computing platform, probes paired with complementary data fibers in massive parallel, leading to the connection of data via probe. When probe operation terminated, the product corresponding to the string satisfying the given formula, $x_1 - x_2 - x_3 - x_4$, was thus created, as presented in Figure 5. Here, the preceding question was answered.

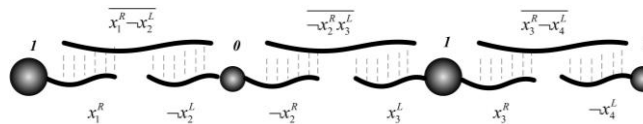


FIGURE 5. Schematic diagram of the product of probe operation.

Products corresponding to true solution that contained 4 types of data were decoded by detector. The type of data body of each data reported the specific value assigned to variable in the given formula. In Figure 5, detector reported the satisfying assignment is 1010.

True solutions were placed into true solution storage. Non-solutions were decomposed into residue collector.

That completed the computation process of proposed computing model for solving satisfiability problem based on PM. There exists a unique assignment 1010, i.e., $(x_1 = T, x_2 = F, x_3 = T, x_4 = F)$ that satisfies the given formula. Therefore, the given formula is satisfiable.

In general, for any 3-SAT instance comprising n variables and m clauses, a theorem is established that determines its satisfiability based on the existence of potential probes.

Theorem. The given formula is satisfiable iff pairs of data representing distinct variables are potential probe, or else, the given formula is unsatisfiable.

Proof. If the formula is satisfiable, then there exists at least one assignment that satisfies it. This implies that at least one leaf node in the binary tree remains after pruning. Based on the probe design, each leaf corresponds to $n - 1$ types of probes. Hence, data pairs representing distinct variables must form potential probes. Conversely, if such data pairs are potential probes, then at least $n - 1$ probe types are present, enabling the generation of at least one valid string through probe operation. This indicates the existence of a satisfying assignment, confirming that the formula is satisfiable.

If the formula is unsatisfiable, all leaf nodes are pruned. Consequently, no probe exists between at least one critical pair of data, meaning that not all variable-representing data pairs form potential probes. The converse also holds. This completes the proof.

Based on this theorem, it is concluded that the general satisfiability problem can be decided and solved within the proposed model.

C. NANO-DNA-ORIGAMI PM

The key to the realization of PM based model is the solution detection technology. How can solutions be readily detected without losing the massive parallelism inherent in the model? What materials can be employed to realize data and probe? And how to decide whether the product resulted from probe operation is the true solution to the given problem or not? Specifically, in the proposed PM based computing model, since assigned values to variables were encoded into 2 types of data body, the question is how to detect the type of data body?

To answer these questions, targeted again at the satisfying string, $x_1 - -x_2 - x_3 - -x_4$, a rectangular 2D DNA origami platform with anchored hairpin-shaped oligonucleotides was devised. By pairings of probes with both data and anchored hairpins, on one hand, data were fixed at pre-designed locations, termed *Station* ($S_i, i = 1, 2, 3, 4$), on the origami platform. On the other hand, hairpin structures of oligonucleotides were unfolded by probes, and opened hairpins triggered the chain reaction, resulted in the generation of DNA paths representing hyphens in the target string. Followed by, Atomic Force Microscope (AFM) imaging of the platform was performed. Therefore, each type of data body was determined.

Next, designs for creating the satisfying string on the DNA origami platform were presented in detail.

1) DNA ORIGAMI PLATFORM

DNA origami is a new method of DNA self-assembly with nanoscale addressability and programmability. The principle of this method is to fold a long DNA strand, termed scaffold, to construct a pre-designed nanostructure by shorter DNA strands, termed staples, on the basis of complementary base pairing. Recent progresses in this field include DNA walker [33], DNA navigator [34], intelligent computing devices [35], etc. Here, a ring-shaped bacteriophage (M13mp18) was folded into a rectangular 2D DNA origami platform by staples, which was presented as a rectangle in gray in the following Figure 6.

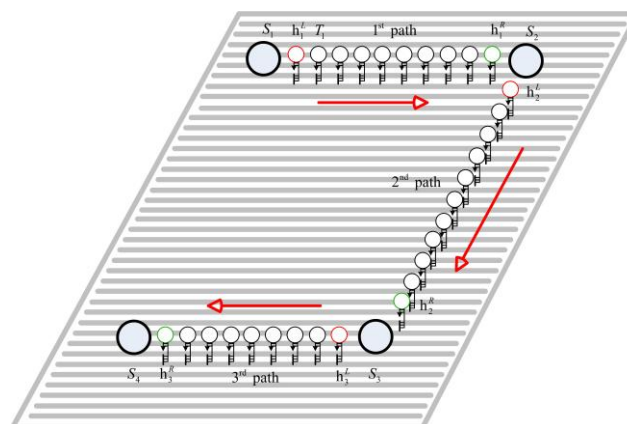










FIGURE 6. The layout of the origami platform.

On the origami platform, hairpin-shaped oligonucleotides were anchored and paved at a precisely designed distance in 2 horizontal and 1 vertical lines, denoted by 1st, 2nd, and 3rd path, respectively. All paths started with a hairpin h_i^L (in red), passed through hairpins T_1 (in black), and ended by a hairpin h_i^R (in green). Arrow (in red) indicated the direction of a DNA path, and also the direction of the chain reaction. Note here, although the starting and terminal hairpin in all paths shared the same color, red and green, their sequences were different. Additionally, there was another type of hairpin (in yellow) T_2 . The hairpin (presented in table II) did not require to be anchored on the platform. Instead, it diffused freely in the solution that contained the platform. Therefore, hairpin T_2 was not presented in Figure 6. Sequences of each type of hairpin were given in table II. Sub-sequence b and b^* of all hairpins are self-complementary, which linear oligonucleotides were shaped into hairpin structure. There were a total of 8 types of hairpins. Finally, 4 Black circles represented *Station* S_i , which were pre-designed locations for data to be fixed on the platform.

TABLE II
SEQUENCES AND STRUCTURES OF ALL HAIRPINS

	1 st Path	2 nd path	3 rd path		
h_1^L :	 $5'-a_1^S-b-c-d-3'$	h_2^L :	 $5'-a_2^S-b-c-d-3'$	h_3^L :	 $5'-a_3^S-b-c-d-3'$
T_1 :		$5'-d-b-c-b^*-3'$			
h_1^R :	 $5'-a_1^T-b-c-d-3'$	h_2^R :	 $5'-a_2^T-b-c-d-3'$	h_3^R :	 $5'-a_3^T-b-c-d-3'$
T_2 :		$5'-b^*-d^*-b-c^*-3'$			

2) DATA LIBRARY

2 types of gold nanoparticle (AuNP) with 2.5nm and 5nm in diameter were chosen as data bodies, while data fibers were realized by 12 types of single-stranded oligonucleotides. Attach oligonucleotides to AuNP to fabricate 12 types of data. The data library was listed as follows:

$$X = \{x_1\} \cup \{-x_1\} \cup \{x_2\} \cup \{-x_2\} \cup \{x_3\} \cup \{-x_3\} \cup \{x_4\} \cup \{-x_4\} .$$

3) PORBE LIBRARY

Each potential probe between pairs of data was transformed into potential probes between data and hairpins anchored on the origami platform. For example, since there existed a probe $\overline{-x_2^R x_3^L}$ between data fiber $-x_2^R$ and data fiber x_3^L , then the pair of data $(-x_2, x_3)$ was potential probe. The probe $\overline{-x_2^R x_3^L}$ was transformed into 2 types of probe: $\overline{-x_2^R h_2^L}$ between data fiber $-x_2^R$ and hairpin h_2^L , $\overline{x_3^L h_2^R}$ between data fiber x_3^L and hairpin h_2^R . In this way, based on the probe library Y constructed and hairpins devised, original 3 types of probe in the constructed probe library were transformed into 6 types of probe. The transformed probe library was listed as follows:

$$Y = \{\overline{x_1^R h_1^L}\} \cup \{\overline{-x_2^L h_1^R}\} \cup \{\overline{-x_2^R h_2^L}\} \cup \{\overline{x_3^L h_2^R}\} \cup \{\overline{x_3^R h_3^L}\} \cup \{\overline{-x_4^L h_4^R}\} .$$

According to sequences of data fibers and hairpins, oligonucleotides complementary to corresponding subsequences were synthesized to fabricate probes.

Appropriate amount of data and probes were placed into buffer containing DNA origami platform and T_2 hairpin to execute probe operation. On the origami platform, dark blue probes (presented in Figure 7) paired with both data fibers and anchored hairpins in massive parallel. For example, one half of the $\overline{-x_2^R h_2^L}$ probe, $\overline{h_2^L}$ (in dark blue), recognized and paired with the starting hairpin h_2^L (in red) in 2nd DNA path on one hand. The h_2^L hairpin structure was unfolded. Then, the opened h_2^L hairpin captured and paired with T_2 hairpin (in yellow) in the buffer. Next, the opened T_2 hairpin recognized and paired with the T_1 hairpin (in black) that was adjacent to the starting hairpin. The chain reaction occurred in this way, T_2 hairpin and T_1 hairpin were alternatively unfolded. The DNA path was generated on the platform. The path terminated at h_2^R hairpin (in green), when T_1 hairpins anchored in this DNA path exhausted. On the other hand, the other half of the $\overline{-x_2^R h_2^L}$ probe, $\overline{-x_2^R}$ paired with $-x_2^R$ data fiber. The right data fiber of $-x_2$ was fixed on the platform. It followed that the left data fiber was fixed on the platform, as well. Therefore, data $-x_2$ were fixed to station S_2 on the platform. When probe operation terminated, all data were fixed at pre-designed locations on the platform, as presented in Figure 7. Atomic Force Microscope (AFM) imaging was performed to detect each type of data body.

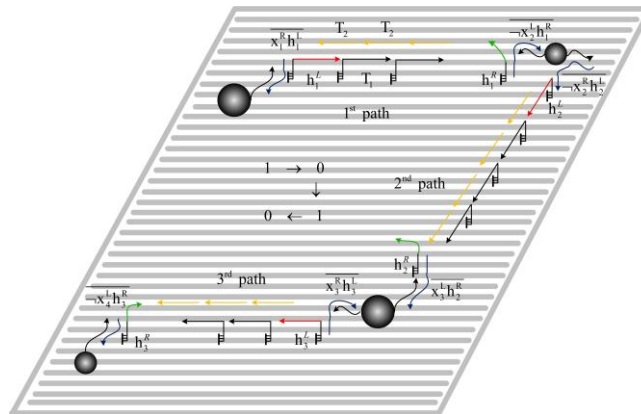


FIGURE 7. Schematic diagram of probe operation on the origami platform.

III. Discussion

Conventional Turing machine (TM) based algorithms are inherently sequential, whereas the proposed Nano-DNA-Origami PM exhibits a high degree of parallelism that scales with problem size. Leveraging the intrinsic parallel search capability of the probe machine (PM), a natural question arises: can the tractable size of NP-complete problems be further extended?

The answer depends fundamentally on the physical implementation of the PM. Historically, von Neumann's use of semiconductors to realize the Turing machine model led to the development of the modern stored-program computer. Previous work [8] proposed a realization of PM using nanoparticles and single-stranded DNA. In contrast, this study employs gold nanoparticles, single-stranded oligonucleotides, and a DNA origami platform combined with atomic force microscopy (AFM) imaging—a approach termed Nano-DNA-Origami—to implement a PM-based computational architecture.

This design is motivated by the fact that the products of probe operations form three-dimensional structures in solution, referred to as probe operation graphs [8]. For such a structure to represent a valid solution, it must satisfy two conditions: it must contain exactly n data bodies, and the 3D probe operation graph must be isomorphic to a predefined 2D planar graph. However, verifying graph isomorphism under these conditions requires the detection of 3D molecular structures, which remains challenging with current biotechnology.

To overcome this limitation, our approach converts the probe operation graph from a 3D solution-based structure into a 2D layout on a DNA origami substrate. This transformation significantly simplifies solution detection via AFM. Moreover, this conversion process nearly doubles the variety of probe types involved, thereby preserving—and even enhancing—the massive parallelism inherent in the PM model. Thus, the Nano-DNA-Origami PM offers a feasible and efficient implementation of the PM computing framework without compromising its parallel processing capabilities.

IV. Conclusions

This paper presents a Nano-DNA-Origami Probe Machine (PM) designed to solve the Boolean satisfiability problem. Specifically, variable assignments of a 3-SAT instance are encoded into structured data, and probes are constructed to mediate interactions between complementary data pairs. Through a single probe operation executed on a DNA origami platform, the unique satisfying assignment for a hard 3-SAT formula is generated. The solution is then identified via atomic force microscopy (AFM) imaging.

Moreover, the approach is generalized to arbitrary 3-SAT instances with n variables and m clauses. A method utilizing the concept of potential probes is introduced to determine satisfiability. Complexity analysis indicates that the encoding process requires $O(n)$ operations, while the solution is generated in constant time, $O(1)$.

The proposed Nano-DNA-Origami PM fundamentally differs from both bio-molecular and Turing machine (TM) -based approaches. Its key advantages are twofold: first, solutions to NP-complete problems are generated in one step rather than discovered through exhaustive search; second, the model exhibits scalable parallelism that grows with problem size. Theoretically, a single probe operation can explore 2^n possible assignments. This represents a significant computational advance over TM-based methods, with the potential to substantially extend the solvable size of NP-complete problems.

At present, several obstacles must be overcome to be overcome in the realization of PM, for example, solution detection technology, error-prone nature of bio-chemical reaction, etc. Although DNA origami combined with AFM imaging was proposed to facilitate solution detection, the number of variables that the proposed Nano-DNA-Origami PM can handle was limited by the size of DNA origami platform ($100 \times 70 \text{nm}^2$).

A feasible way to this limitation is to adopt fractal self-assembly to increase the size of the DNA origami platform. Despite these, we believe that the advent of PM is an important contribution to computational theory. The proposed Nano-DNA-Origami PM only made a small step towards the becoming into reality of PM.

REFERENCES

- [1]. M. Davis, G. Logemann, and D.Loveland, "A machine program for theorem-proving", *Communications of the ACM*, vol. 5, no. 7, pp. 394-397, 1962.
- [2]. Y. Vizel, G. Weissenbacher, and S. Malik, "Boolean satisfiability solvers and their applications in model checking", *Proceedings of the IEEE*, vol.103, no. 11, pp. 2021-2035, 2015.
- [3]. C. Sinz and M. Iser, "Problem-sensitive restart heuristics for the DPLL procedure", *Theory and Applications of Satisfiability Testing-SAT* pp. 356-362, 2009.
- [4]. A. Dal Palu, A. Dovier, A. Formisano, and E. Pontelli, "Cud@ sat:Sat solving on GPUs", *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 27, no. 3, pp. 293-316, 2015.
- [5]. C. N. Yang and C. B. Yang, "A DNA solution of SAT problem by a modified sticker model", *BioSystems*, vol. 81, no. 1, pp. 1-9, 2005.
- [6]. X. L. Wang, Z. M. Bao, J. J. Hu, S. Wang, and A. B. Zhan, "Solving the SAT problem using a DNA computing algorithm based on ligase chain reaction", *BioSystems*, vol. 91, no. 1, pp. 117-125, 2008.
- [7]. B.S. Song, M. J. Pérez-Jiménez, and L.Q. Pan, "An efficient time-free solution to SAT problem by P systems with proteins on membranes", *Journal of Computer and System Sciences*, vol. 82, no. 6, pp. 1090-1099, 2016.
- [8]. J. Xu, "Probe machine", *IEEE transactions on neural networks and learning systems*, vol. 27, no. 7, pp. 1405-1416, 2016.
- [9]. Wang D , Liu J , Sun H ,et al. "A Fully Parallel Approach of Model Checking Via Probe Machine", *International Journal of Software Engineering and Knowledge Engineering*, 2022. DOI:10.1142/S0218194021400210.
- [10]. Jianzhong C, Zhixiang Y, Zhen T, Jing Y, "Probe Machine Based Computing Model for Maximum Clique Problem", *Chinese Journal of Electronics*, 2022, 31(2), 304-312. doi: 10.1049/cje.2020.00.293.