Available online at www.sciencedirect.com

ScienceDirect

BioSystems 90 (2007) 242–252

BioSystems

www.elsevier.com/locate/biosystems

Solving satisfiability problems using a novel microarray-based DNA computer

Che-Hsin Lin[a,1], Hsiao-Ping Cheng[a,1], Chang-Biau Yang[b,2], Chia-Ning Yang[c,*]

[a] *Department of Mechanical and Electro-Mechanical Engineering, National Sun Yat-sen University, Kaohsiung 804, Taiwan*
[b] *Department of Computer Science and Engineering, National Sun Yat-sen University, Kaohsiung 804, Taiwan*
[c] *Department of Medical Imaging and Radiological Sciences, I-Shou University, Kaohsiung 840, Taiwan*

Received 12 June 2006; received in revised form 21 August 2006; accepted 23 August 2006

Abstract

An algorithm based on a modified sticker model accompanied with an advanced MEMS-based microarray technology is demonstrated to solve SAT problem, which has long served as a benchmark in DNA computing. Unlike conventional DNA computing algorithms needing an initial data pool to cover correct and incorrect answers and further executing a series of separation procedures to destroy the unwanted ones, we built solutions in parts to satisfy one clause in one step, and eventually solve the entire Boolean formula through steps. No time-consuming sample preparation procedures and delicate sample applying equipment were required for the computing process. Moreover, experimental results show the bound DNA sequences can sustain the chemical solutions during computing processes such that the proposed method shall be useful in dealing with large-scale problems.
© 2006 Elsevier Ireland Ltd. All rights reserved.

Keywords: SAT problem; DNA computing; DNA array; Sticker model

1. Introduction

DNA computing uses molecular biology laboratory procedures to solve mathematical problems. The concept of DNA computing was first developed by Adleman in 1994 (Adleman, 1994), who solved the *traveling salesperson problem* (*TSP*), in which it is necessary to determine a path for a seven-vertex directed graph with designated start and end nodes. Inspired by Adleman's success, Lipton proposed a method for solving the satisfiability (SAT) problems (Lipton, 1995), whose goal is to determine appropriate assignments of a set of Boolean variables with values of either "true" or "false" such that the output of the whole Boolean formula is true (Cormen et al., 2001). The SAT instance solved by Lipton is $F=(x\vee y)\wedge(\bar{x}\vee\bar{y})$, where x and y are the Boolean variables, $(x\vee y)$ and $(\bar{x}\vee\bar{y})$ are two clauses, $\vee$ the logical operation OR and $\wedge$ the logical operation AND, and the notation $\bar{x}$ is the negation of x, which dedicates that $\bar{x}=0$ if $x=1$, and $\bar{x}=1$ if $x=0$.

Of the various computational problems which have been successfully solved using Adleman's approach (Braich et al., 2002; Chen and Ramachandran, 2001; Diaz et al., 2001; Liu et al., 2000, 2002; Ouyang et al., 1997; Pirrung et al., 2000; Rozenberg and Spaink,

* Corresponding author at: No. 1, Section 1, Hsueh-Cheng Road, Ta-Hsu Hsiang, Kaohsiung 840, Taiwan. Tel.: +886 7 6577711x6814; fax: +886 7 6577056.
E-mail addresses: chehsin@mail.nsysu.edu.tw (C.-H. Lin), cbyang@cse.nsysu.edu.tw (C.-B. Yang), cnyang@isu.edu.tw (C.-N. Yang).
[1] Present address: No. 70, Lien Hai Road, Kaohsiung 804, Taiwan. Tel.: +886 7 5252000x4240; fax: +886 7 525 4299.
[2] Present address: No. 70, Lien Hai Road, Kaohsiung 804, Taiwan. Tel.: +886 7 5252000x4333.

0303-2647/$ – see front matter © 2006 Elsevier Ireland Ltd. All rights reserved.
doi:10.1016/j.biosystems.2006.08.009

2003; Sakamoto et al., 2000; Schmidt et al., 2004; Su and Smith, 2004; Wang et al., 2000; Yoshida and Suyama, 1999) the SAT problem has emerged as a useful benchmark for exploring the feasibility of DNA-based algorithms and experiments. For example, Sakamoto et al. used autonomous molecular computing with the delicate hairpin formation to solve an instance with four variables (Sakamoto et al., 2000); Liu et al. demonstrated the feasibility of DNA computing with surface-bound sequences rather than a test tube approach by solving an instance with six variables (Liu et al., 2000); Braich et al. employed a sticker model (Paun and Rozenberg, 1998) to solve an instance with 20 variables (Braich et al., 2002).

In general, DNA computing algorithms commence by constructing a data pool of all possible candidate solutions, expressed in the form of either single- or double-stranded DNA (ssDNA or dsDNA). The incorrect solutions are progressively eliminated from the pool by performing biological separation processes. The surviving sequences are then considered to represent the correct solutions of the problem. Adopting this approach, the n-variable SAT problem can be solved as follows: (1) create a data pool containing the 2^n sequences corresponding to all of the possible variable assignments (note that a total of 2^n possible solutions exist because each variable can be assigned by only value of either "true" or "false"); (2) eliminate the incorrect variable assignments by applying logic restrictions; (3) collect and read out the surviving sequences. In the traditional computation model, this strategy corresponds to a complete binary decision tree of $n+1$ levels in which every one node has two branches, attached with two children, each representing the true value or false value of one variable. Each path starting from the root to one leaf, consisting of n edges, corresponds to one possible assignment of the n variables, as indicated in Fig. 1a for an instance with three variables.

However, to find all true assignments or to eliminate all false assignments, we may not need to examine all possible 2^n assignments. In our approach, we examine the input Boolean formula by considering its falsity one clause by one clause. For instance, consider $F=(x\vee\bar{y})\wedge(\bar{x}\vee z)\wedge(\bar{y}\vee\bar{z})$. The decision tree now is transformed into an AND/OR tree, as shown in Fig. 1b. The AND/OR tree consists of $m+1$ levels if m clauses are considered. The branches on level i correspond to the literals of clause i, where the level of the root is counted as 1. In other words, the number of branches on level i is d_i, which is the number of literals included in clause i. In addition, the branches attached to a node perform the OR operation, and the path from the root to one leaf should perform the AND operation. Fig. 1b shows the complete AND/OR tree for this instance. However, the tree can be trimmed by transforming some conflict branches and then terminating some falsified branches, as shown in Fig. 1c. When a new coming clause i is checked, the following two procedures are applied to expand the tree branches: (1) transformation: if a newly assigned value of variable x in clause i is complementary to the previ-

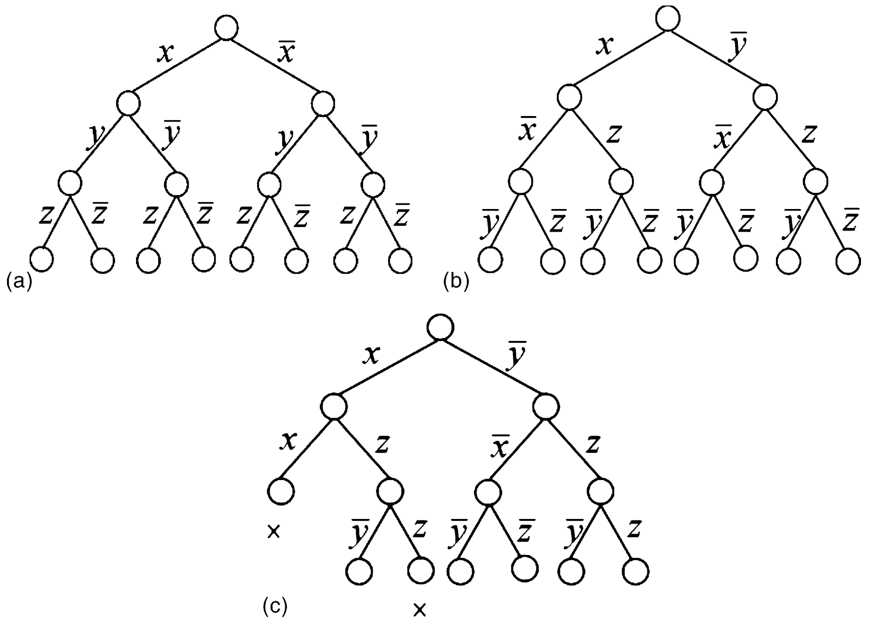


Fig. 1. Decision trees for (a) a general strategy in other studies, (b) our strategy, and (c) our computing process.

ously assigned one, the value of x stays unchanged since x has been assigned and it cannot be assigned with two values; (2) termination: extract the values assigned for the variables on the path from the root to the current leaf, check whether these assigned values satisfy at least one literal of clause i. If none of literals of clause i can be satisfied, then this branch is terminated since it becomes a falsified branch. Accordingly, in Fig. 1c, the first left node in level 3 is terminated due to the falsity of the second clause $(\bar{x} \vee z)$ and the second left node in level 4 is terminated due to falsity of the third clause $(\bar{y} \vee \bar{z})$. The leaf nodes in level 4 represent correct solutions to the formula and from left to right they are (x = true, y = false, z = true), (x = false, y = false, z = don't care), (x = false, y = false, z = false), (x = don't care, y = false, z = true), and (x = don't care, y = false, z = true), respectively. In all, the satisfiable assignments are concluded as $(x, \bar{y}, z)$, $(\bar{x}, \bar{y}, \bar{z})$, and $(\bar{x}, \bar{y}, z)$.

In this present work, we wish to provide an implementation of the above approach to solve SAT problems in which the requirement to construct an initial data pool is removed. Two instances of the SAT problems are considered. The first instance is that of the relatively straightforward 2-SAT (k-SAT is identified by k literals in each clause) $F = (x \vee \bar{y}) \wedge (\bar{x} \vee z) \wedge (\bar{y} \vee \bar{z})$ to demonstrate the associated experimental procedures. The second instance, $F = (x \vee y \vee z) \wedge (\bar{x} \vee \bar{w} \vee z) \wedge (\bar{y} \vee \bar{z}) \wedge (w \vee \bar{x} \vee y) \wedge (\bar{y} \vee z)$, has four variables and five clauses. This problem is presented to verify the feasibility of using the proposed method to solve a more complicate instance. An advanced MEMS-based microarray technology is chosen for the platform since not too many DNA computing methods were explored on surface in spite of its high through put property. The algorithm is based on our previously proposed theory termed modified sticker model (Yang and Yang, 2005) in spirit of building solutions in steps to satisfy one clause at a time and eventually the solutions satisfying the entire Boolean formula. That is, computation begins with a blank data pool, which is capable of carrying added information but carries no information at first place. In solving a 2-SAT instance, the blank pool is first divided into two parts and each part will get assigned one literal in the first clause. Combining these parts completes the OR operation in the first clause. The combined updated pool is again divided into two parts and each part will be assigned with one literal in the second clause. Further combining and dividing will be repeated to carry literal assignments in the remaining clauses. Yet a probing procedure is required after every combined pool forms to exclude members falsifying the considered clause. In the end, the surviving members represent the solutions assignments. In the present study, the blank pool is played by an array with immobilized DNA strands, whereas, the literal information is played by fluorescent dye labeled DNA strands complementary to the immobilized ones.

2. Materials and methods

2.1. Computation concept

To compute the 2-SAT problem $F = (x \vee \bar{y}) \wedge (\bar{x} \vee z) \wedge (\bar{y} \vee \bar{z})$, a 6×4 matrix array (six rows, four columns) is fabricated on a glass plate. In this array, each set of vertically adjacent x-, y- and z-sites corresponds to a Boolean variable assignment, and hence the array can accommodate a maximum of eight different assignments. The relationship between a given Boolean formula and the necessary array size is characterized later in this study. In the computations, the three variables, i.e. x, y and z, are represented by three 16-mer ssDNA sequences, i.e. X, Y and Z, respectively, and their complements X^c, Y^c, and Z^c. Prior to computation, the X, Y and Z sequences are covalently bound to the glass plate in accordance with their sites. Meanwhile, the complementary sequences, X^c, Y^c and Z^c, are labeled with green (FITC) and red (Cy3) florescent dye to enable the assignment of "true" and "false" signals, respectively, to the surface bound sequences. Because of the specific complementary property in the $X - X^c$, $Y - Y^c$ and $Z - Z^c$ pairs, the immobilized sequences X, Y and Z on the glass plate recognize their corresponding complementary sequences and are therefore assigned as either true or false when the green or red fluorescent dye-labeled sequences are applied to the sites in the array. In other words, during the computations, an x-site containing ssDNA X is assigned as true (or false) when green (or red) X^c sequences are annealed onto X. Moreover, once a particular site has been assigned a value, its value remains unchanged in subsequent steps when new sequence assignments are applied. The concept described here for the x-sites applies equally to the y-and z-sites in the array.

2.2. Variable assignment and immobilization

Table 1 summarizes the surface-bound oligonucleotides and their complements used in the present computations. All of the oligonucleotides and fluorescent dye labels were obtained from ScinoPharm (Tainan, Taiwan). The sequences were designed to maintain the GC content of 50% such that the number of hydrogen bonds in each matched duplex is the same and therefore a uniform thermodynamic stability is ensured. The five consecutive T nucleotides appended at the 5′ end of sequences X, Y and Z function as 'spacers', which isolate the hybridizing sequence from the support and therefore enhance the hybridization efficiency.

The protocol used in this study to modify the surface of the glass substrate is based on the approach originally described by Graham and Britland (Britland et al., 1992; Graham et al., 1992). Initially, the glass substrate was activated using absolute ethyl alcohol and acetone followed by immersion in

Table 1
Sequence assignments for *X*, *Y* and *Z* and their complements labeled with FITC (green dye) or Cy3 (red dye) to represent true or false values, respectively

ssDNA sequence	Sequence 5′ → 3′
X	TTTTTAATTCCGGAAGGCCTT
X^c with FITC label	FITC-AAGGCCTTCCGGAATT
X^c with Cy3 label	Cy3-AAGGCCTTCCGGAATT
Y	TTTTTTTCCTTCCAAGGAAGG
Y^c with FITC label	FITC-CCTTCCTTGGAAGGAA
Y^c with Cy3 label	Cy3-CCTTCCTTGGAAGGAA
Z	TTTTTAATTAATTCCGGCCGG
Z^c with FITC label	FITC-CCGGCCGGAATTAATT
Z^c with Cy3 label	Cy3-CCGGCCGGAATTAATT

a 5% (v/v) 3-aminopropyltrimethoxy-silane solution (APTS, Aldrich, USA) for 8 h. APTS was then covalently attached to the glass surface. The modified substrate was again rinsed for 5 min with acetone, absolute ethyl alcohol and DI water, respectively. The substrate was then rinsed with phosphate buffered saline (PBS, pH 7.4, Sigma, USA) and immersed in a 2% glutaraldehyde solution (GA, Sigma, USA) for 2 h. The exposed –CHO functional group on the attached GA was used for ssDNA immobilization. Subsequently, the substrate was rinsed with phosphate buffer (PB, Sigma, USA) and DI water, respectively. Finally, the substrate was dried in an oven at 40 °C for 1 h. The ssDNA molecules, *X*, *Y* and *Z*, were then attached to the modified substrate surface by covalent bonding.

2.3. Experimental details for microarray fabrication

A simple and low-cost process for the fabrication of a microarray-based DNA computer utilizing home-made photomask and back-side exposure method was developed in this study (Yang et al., 2006). Fig. 2 presents a schematic illustration of the experimental procedure. To solve the 2-SAT problem described above, a 4 × 6 array of microwells with a diameter of 300 μm and a pitch of 300 μm was patterned on a glass slide using standard photolithography processes. Prior to patterning, the substrate was coated with a thin Cr film with a thickness of 100 nm to enhance the quality of the emitted fluorescence images. The surfaces of the microwells were firstly modified with 3-aminopropyltrimethoxy-silane (APTS) and glutaraldehyde (GA) to facilitate ssDNA immobilization (Fig. 2A). The *X*, *Y* and *Z* ssDNA templates were diluted to 100 μM with DI water and then attached to the appropriate wells via the covalent bonding of the functional groups of $-NH_2$ on ssDNA and –CHO on GA (Fig. 2B). A second layer of positive photoresist, patterned to open specific wells for DNA hybridization was then coated on the prepared substrate (Fig. 2C). The patterned substrate was then immersed in the sample solutions

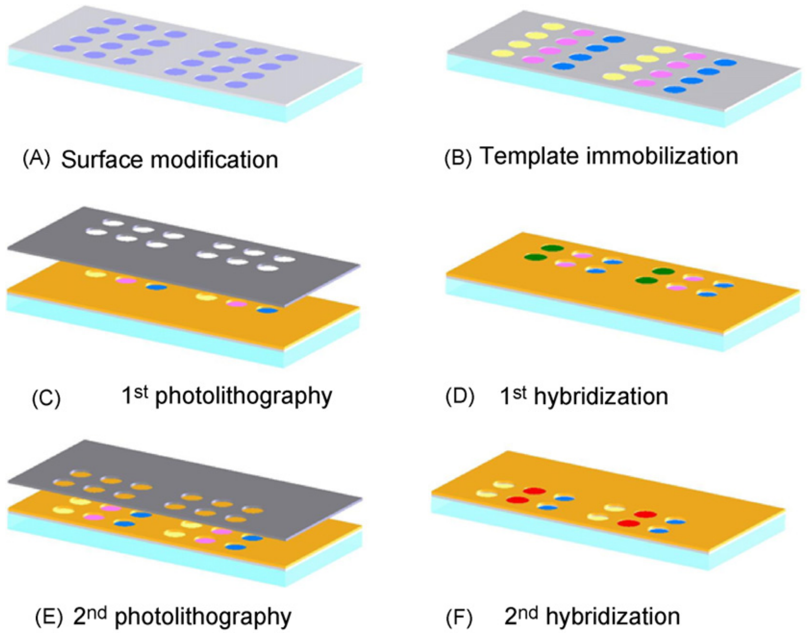


Fig. 2. Simplified illustration of the experimental procedures involved in the proposed microarray-assisted DNA computing procedure: (A) A 4 × 6 array of microwells is patterned on a Cr-coated glass slide using standard photolithography processes. The exposed microwells are then modified with APTS and GA; (B) single-stranded DNA templates of *X*, *Y* and *Z* are attached to their corresponding wells, (C) computation commences by coating the substrate with a layer of positive photoresist patterned to open the well areas corresponding to the first computing procedure (Step 1(a) in Fig. 6A), (D) the glass slide is immersed in the sample solutions for DNA hybridization to carry out the computations for *x* = true, (E) and (F) similar processes to those shown (C) and (D) are performed to compute the second computing procedure (Step 1(b) in Fig. 6A). A similar processing to that shown in (C)–(F) is repeated until the whole problem has been solved.

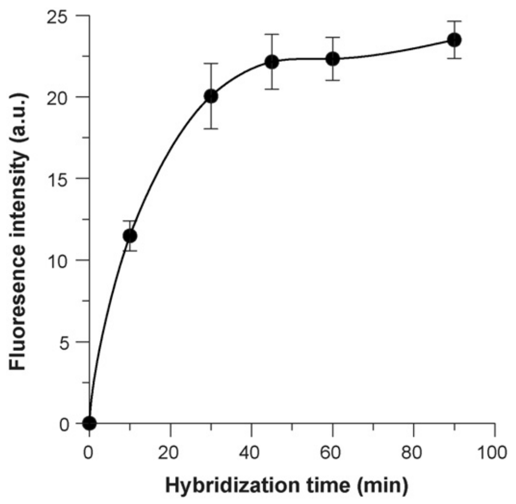


Fig. 3. Variation of the emitted fluorescence intensity with the hybridization time for the ssDNA hybridization reaction in the microwells. The results suggest that the hybridization process becomes saturated approximately 45 min after the initial reaction. Note that the values of each measured point represent the average of 5 individual microwell readings.

with the corresponding complementary ssDNA to produce the first hybridization (Fig. 2D). The substrate was then blown with N_2 gas to remove the residual solution. The first DNA computing procedure was then finished by stripping the photoresist layer using acetone and rinsing the substrate with ethyl alcohol and DI water. The second computing process was performed by carrying out similar photolithography and hybridization processes, as shown in Fig. 2E and F, respectively. A total of six photolithography-hybridization procedures were therefore performed to solve the 2-SAT problem.

In place of a delicate microarray scanner, this study used a low-cost standard LIF (laser induced fluorescence) detection system to read the experimental results directly. The system comprised a fluorescence microscope (E-400, Nikon, Japan) and a mercury lamp module for the excitation and observation of the fluorescent dye emissions from the microwells. The images of the computing results were acquired by a 2× objective lens fitted to the fluorescence microscope and recorded using a digital camera (4500, Nikon, Japan).

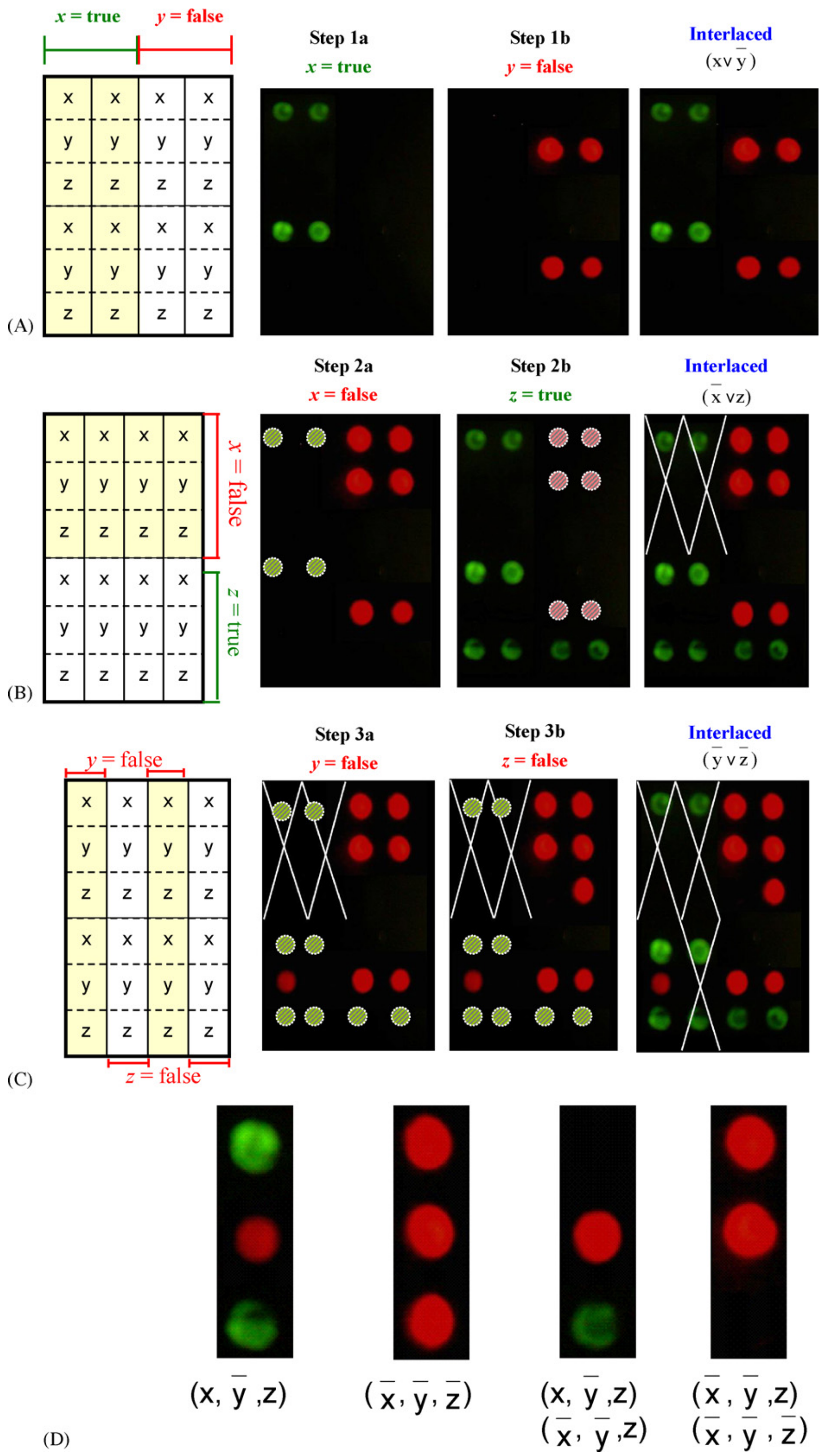
Prior to executing the DNA computing procedures described above, a preliminary investigation was conducted to establish the minimum time required for ssDNA hybridization. The hybridization degree was determined from the emitted fluorescence intensity of the hybridization at times of 10, 30, 45, 60 and 90 min, respectively, following the initial reaction process. The emitted fluorescence from five individual microwells was measured and the corresponding optical intensities of the five individual microwells were calculated and used to represent the relative DNA concentration in the microwells. Fig. 3 illustrates the variation of the emitted fluorescence intensity with the hybridization time for the ssDNA hybridization reaction in the microwells. It is observed that the hybridization process saturates approximately 45 min after the initial reaction. Therefore, 45 min was specified as the standard hybridization reaction time in each step of the current DNA computing procedure such that the overall computing time was minimized. Furthermore, the assigned microwells were not affected by subsequent sequence assignments in the later steps.

2.4. Computation procedures

Using the array with the attached X, Y and Z sequences and the complementary X^c, Y^c and Z^c sequences labeled with FITC or Cy3 fluorescent dye, the current 2-SAT problem $F = (x \vee \bar{y}) \wedge (\bar{x} \vee z) \wedge (\bar{y} \vee \bar{z})$ was solved in a total of three steps, where each step dealt with one particular clause in the formula. Fig. 4A–C show the experimental results obtained after each step.

- Step 1: first clause in the 2-SAT formula, i.e. $(x \vee \bar{y})$. Since the solution assignments must satisfy x or $\bar{y}$, the array was divided into two parts, i.e. a left part and a right part, such that each part contained two columns. As shown in Fig. 2A, the true X^c and false Y^c sequences were applied such that the left part of the array was encoded with x while the right part was encoded with $\bar{y}$. To accomplish this, the right part of the array was initially masked when true X^c sequences were added to the left part (Step 1a; Fig. 4A). Similarly, the left part was then masked when false Y^c sequences were added to the right part (Step 1b; Fig. 4A). Since the images acquired by the fluorescence microscope cannot show both green and red objects simultaneously, the green spots for the x sites in Step 1a are not visible in the image corresponding to Step 1b. Therefore, double-exposure was utilized to take both green and red fluorescence in a single image to generate the satisfied assignments for $(x \vee \bar{y})$ ("interlaced"; Fig. 4A).
- Step 2: second clause in the 2-SAT formula, i.e. $(\bar{x} \vee z)$. Again, the solution assignments must satisfy $\bar{x}$ or z, and

Fig. 4. (A, B, C) Steps involved in 2-SAT computation. For each step, the schematic array and sequence descriptions indicate the division of the glass array into two equal portions and identify the sequences assigned to define the variables in the considered clause. Since a fluorescence microscope cannot take both green and red images simultaneously, only green or red spots are shown in the original images corresponding to different applications of fluorescent dye-labeled sequences (i.e. the images corresponding to Steps (a) and (b) in each case). The third image ("Interlaced") is a double exposed image for illustration purpose. In the images corresponding to Steps 2a, 2b, 3a, and 3b, respectively, the shaded green and red spots represent the assignments given in previous steps. (D) Output readouts of the 2-SAT computation. Four patterns of output are obtained: (I) $(x, \bar{y}, z)$; (II) $(\bar{x}, \bar{y}, \bar{z})$; (III) $(x, \bar{y}, z)$ and $(\bar{x}, \bar{y}, z)$ (since y = false and z = true already satisfy the studied formula, the value of x can be either true or false); (IV) $(\bar{x}, \bar{y}, z)$ and $(\bar{x}, \bar{y}, \bar{z})$. Overall, the solutions of the 2-SAT problem are $(x, \bar{y}, z)$, $(\bar{x}, \bar{y}, \bar{z})$, and $(\bar{x}, \bar{y}, z)$.


x = true
y = false
Step 1a
x = true
Step 1b
y = false
Interlaced
$(x \vee \bar{y})$
(A)
Step 2a
x = false
Step 2b
z = true
Interlaced
$(\bar{x} \vee z)$
x = false
z = true
(B)
y = false
Step 3a
y = false
Step 3b
z = false
Interlaced
$(\bar{y} \vee \bar{z})$
z = false
(C)
$(x, \bar{y}, z)$
$(\bar{x}, \bar{y}, \bar{z})$
$(x, \bar{y}, z)$ $(\bar{x}, \bar{y}, z)$
$(\bar{x}, \bar{y}, z)$ $(\bar{x}, \bar{y}, \bar{z})$
(D)

hence the updated array was separated into two equivalent parts, namely the upper part and the lower part. Therefore, as shown in the schematic of the array presented in Fig. 4B, each part contained three rows of the 6×4 matrix. Furthermore, each array site retained the true or false assignments achieved in Step 1. In this step, the upper part of the array was assigned to carry $\bar{x}$ and the lower part to carry z by applying the false X^c and true Z^c sequences, respectively. Accordingly, the two upper right x-sites take the $\bar{x}$ value, while the two upper left x-sites retain the value of x assigned in the previous step (Step 2a; Fig. 4B). In the lower part of the array, all four z-sites take a value of z (Step 2b; Fig. 4B). Note that the Step 2a image in Fig. 4B shows four shaded green spots. These spots are not actually visible in the red image, but are shown here to illustrate why that the two upper left x-sites remain true. The shaded red spots in the Step 2b image in Fig. 4B serve the same purpose.

Before moving on to Step 3, the array was checked, and computation was terminated on those sets not satisfying the clause $(\bar{x} \vee z)$. Specifically, the complete array was probed and sets containing neither false X^c nor true Z^c sequences were blocked for any further x, y or z assignments. Note that it is necessary to probe for the existence of $\bar{x}$ and z here since at the completion of Step 1, each set already satisfies the first clause and accordingly it is necessary to verify whether or not the updated assignment sets also satisfy the second clause. The interlaced image shown in Fig. 4B indicates that the two upper left sets were terminated because these two sets satisfy the first clause, but fail to satisfy the second clause.

- Step 3: third clause in the 2-SAT formula, i.e. $(\bar{y} \vee \bar{z})$. Again, the surviving sites from Step 2 were divided into two equal parts. As shown in Fig. 4C, the first and third columns were assigned $\bar{y}$ sequences, while the second and fourth columns were assigned $\bar{z}$ sequences. The resulting assignments are shown in the Steps 3a and 3b images, respectively. As in Step 2, the array was then probed to check for satisfaction of the clause $(\bar{y} \vee \bar{z})$ as shown in Fig. 4C, the sets without sequences representing $\bar{y}$ or $\bar{z}$ were terminated. The interlaced image in Fig. 4C shows that four outcomes were obtained.

2.5. *Answers readout*

Four patterns of computing outcome (shown in the interlaced image of Fig. 4C) are given in Fig. 4D. Pattern (I) was read as (green, red, green), and thus the answer was $(x, \bar{y}, z)$. The same rationale was applied to pattern (II) for $(\bar{x}, \bar{y}, \bar{z})$. As to pattern (III), the signal reading for variable x was undefined, while the y and z were defined as false and true. Since the assigned $\bar{y}$ and z values already satisfied the Boolean formula, either x equal true or false would not affect the satisfaction. In this case, the answers in pattern (III) were $(x, \bar{y}, z)$ and $(\bar{x}, \bar{y}, z)$. Likewise, pattern (IV) was read as $(\bar{x}, \bar{y}, z)$ and $(\bar{x}, \bar{y}, \bar{z})$. In all, $(x, \bar{y}, z)$, $(\bar{x}, \bar{y}, \bar{z})$, and $(\bar{x}, \bar{y}, z)$ were concluded as solutions to $F = (x \vee \bar{y}) \wedge (\bar{x} \vee z) \wedge (\bar{y} \vee \bar{z})$.

3. Discussion

Although the current study uses a DNA array as a platform, the essential logic of the algorithm is based on the modified sticker model, which was originally designed for solution phase experiments (Yang and Yang, 2005). The original sticker model uses a memory strand to carry the information conveyed by stickers. In essence, the sticker model comprises a long single memory strand and a number of stickers. Each memory strand consists of l non-overlapping substrands, each of which has a length of k bases. Each sticker also has a length of k bases and is complementary to exactly one of the l sections in the memory strand. During computation, each section is defined as “true” or “false” depending on whether or not its corresponding sticker is annealed or the other way around. A memory strand with several sections annealed by the matching stickers is referred as memory complex in which the computation information is carried in binary format (Paun and Rozenberg, 1998). For example, for a 10-sectioned memory strand with the first, third and seventh sites annealed by stickers were coded as (1010001000). The modified sticker model employed in the current study differs from the original model only in that each section of a memory strand is assigned a value of true or false by annealing a true sticker or a false sticker, respectively, while an unannealed section corresponds to a variable with an undefined value (Yang and Yang, 2005).

In the current array, each vertically adjacent set of x-, y- and z-sites (characterized by their covalently bound X, Y and Z sequences) represents a memory strand. The FITC- or Cy3-labeled X^c, Y^c and Z^c sequences act as true or false stickers for the three variables. The computing course commences with memory strands, which carry no binary codes, and gradually constructs memory complexes in such a way as to satisfy the entire formula. Since the formula for the current 2-SAT problem has three clauses, with two literals in each clause, a total of three steps are required to obtain the solutions. In each step, the total pool of memory strands or memory complexes is divided into two parts, with each part satisfying one literal in the considered clause via the appropriate annealing of either FITC- or Cy3-labeled complementary sequences. The outputs from the two parts of the array in each step are then combined and divided into another two equal parts to satisfy the next clause in the formula. This combining and dividing procedure is similar to the procedures performed in solution phase experiments (Yang and Yang, 2005) but the microarray scheme is beneficial for complicate problem. Before moving to the next clause (i.e. the next step), a probing process is performed to eliminate the unsatisfied upgraded assignments, i.e. those sets in which none of the values of the currently considered clause are shown. Unsatisfied assignments can arise for one of two reasons. For example, an unsatisfied assignment is produced if all of the variables in the considered clause are already assigned values opposite to that of the current literals. Since all of the clauses in the current 2-SAT problem are connected by the logical AND operation, falsifying one clause is sufficient to falsify the entire formula. Unsatisfied assignments also occur when some of the considered variables are assigned as their opposite values and the remaining variables in the considered clause are unassigned. If such assignments remain in the pool, further application of sequences with opposite values will satisfy the further clauses, but will falsify the current clause. In the 2-SAT problem considered above, the entire data pool is bisected three times, i.e. once for each of the three clauses, and hence a total of eight memory strands are required in the array.

3.1. Computation of general SAT problems

Having solved the 2-SAT problem, this study now considers the general SAT case. Any SAT problem with n variables in m clauses, with d_i ($1 \leqq i \leqq m$) literals in the ith clause, requires an array with $d_1 \times d_2 \cdots d_{m-1} \times d_m$ "memory strands" and can be solved in m steps. To generate a big microarray is not a problem for MEMS technique since all the microwells can be patterned in a single process. Of course, m computing steps are required to solve this SAT problem with longer experimental time. As to the instance $F = (x \vee y \vee z) \wedge (\bar{x} \vee \bar{w} \vee z) \wedge (\bar{y} \vee \bar{z}) \wedge (w \vee \bar{x} \vee y) \wedge (\bar{y} \vee z)$, it requires an array with 108 (i.e. $3 \times 3 \times 2 \times 3 \times 2$) memory strands arranged in an 18×24 matrix, i.e. 108×4, to support a maximum of 108 assignments of the four variables. Note that the first three clauses in the formula are treated by dividing the array horizontally, while the last two clauses are treated by dividing the array vertically. Five computing steps and a total of 13 photolithography procedures and an examination of the computing results after each step are taken. It is known that a photo-bleaching effect is caused when repeated fluorescent images of the same object are taken. Consequently, the fluorescent intensity in the microwells will decay, and the image quality will deteriorate, as the number of acquired images increases. In solving the SAT problem with five clauses, this study overcomes this problem by fabricating six identical microarrays on a single glass slide. The corresponding step-by-step computing procedure is shown in Fig. 6. In the first computing step, all six microarrays were proceeded to satisfy the first clause of this instance, i.e. $(x \vee y \vee z)$ (Step 1; Fig. 6). The results of the first computing step were then determined by using a fluorescence microscope to acquire images of only the first microarray in Fig. 5. After removing the unsatisfied assignments in the two upper-left blocks (Step 2; Fig. 6), three masks for computing the second clause solutions of $\bar{x}$, $\bar{w}$ z, respectively, were generated. The remaining five microarrays were patterned and computed for the second clause via three photolithography procedures and three DNA hybridization procedures. A fluorescent image of the second microarray was then taken (Step 2; Fig. 5) and the unsatisfied assignments for the second computing step (Step 3; Fig. 6) were again removed. This process was repeated until the fifth clause had been computed. The fifth and sixth microarrays were anticipated to show the same fluorescent images for the computed results of the SAT problem, and hence the solutions were read out from the sixth array and then double-checked against those of the fifth array. Using this step-by-step approach provides a convenient means of overcoming the photo-bleaching problem since increasing the number of microarrays on a single slide makes very little difference during the computing steps in the proposed MEMS-based microarray technique.

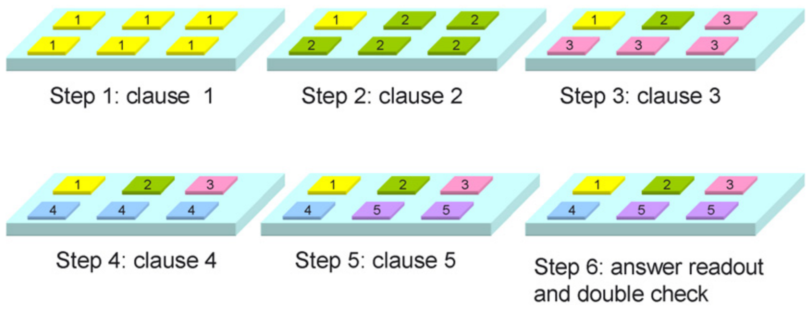


Fig. 5. Well-to-well experimental steps involved in computing the second SAT problem. The problem caused by the photo-bleaching effect is overcome since the fluorescence emissions of each microarray are shot only once.

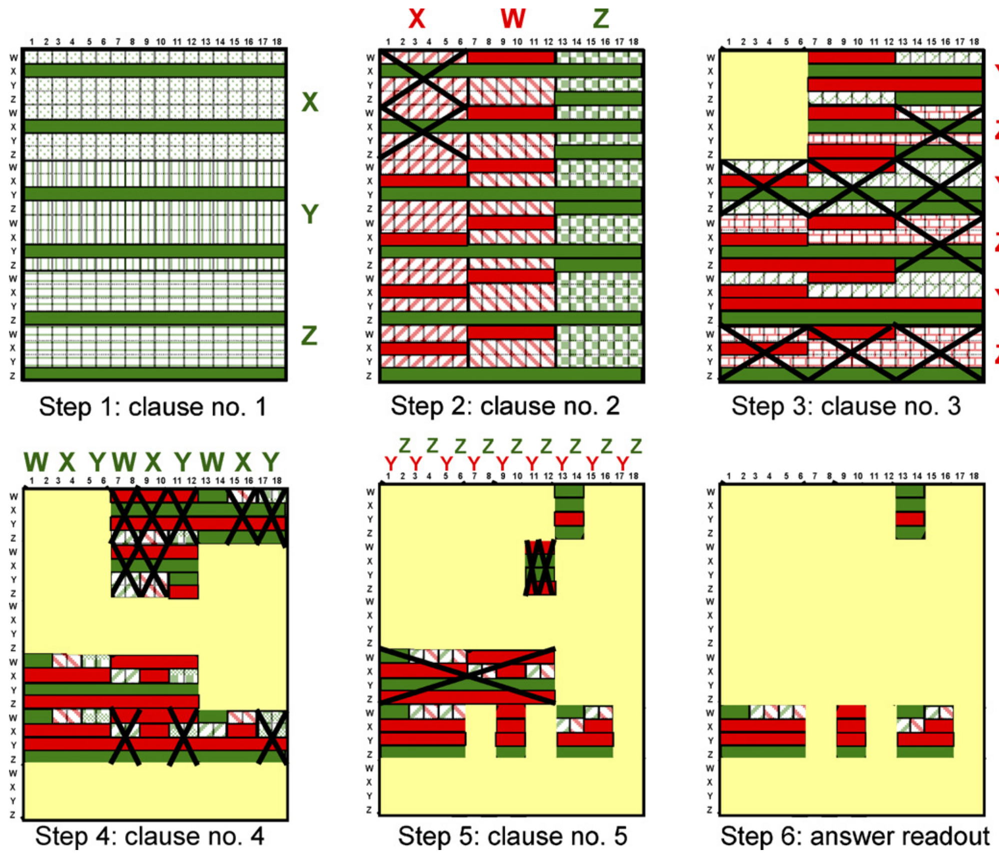
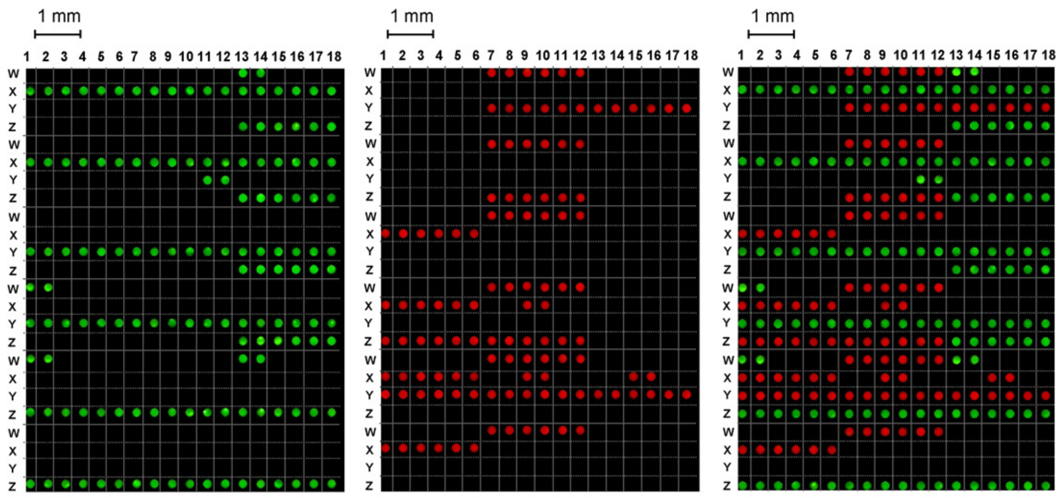


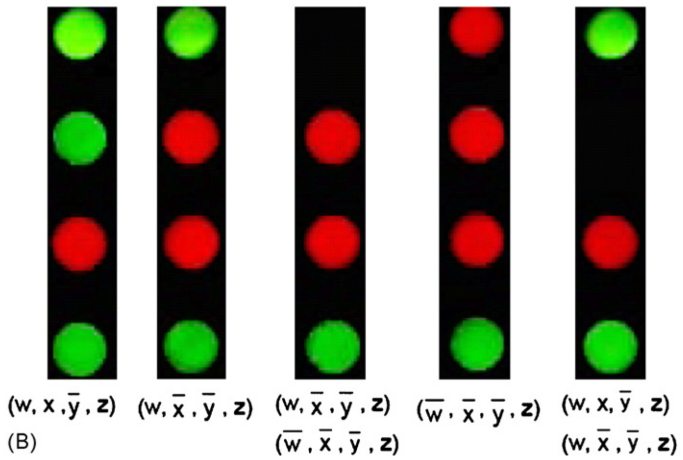
Fig. 6. Step-by-step computing procedure and mask generation protocol for solving $F = (x \vee y \vee z) \wedge (\bar{x} \vee \bar{w} \vee z) \wedge (\bar{y} \vee \bar{z}) \wedge (w \vee \bar{x} \vee y) \wedge (\bar{y} \vee z)$.

The fluorescent images acquired after computing the fifth clause are presented in Fig. 7A, while the satisfied assignments are shown in Fig. 7B. The solution $(w, x, \bar{y}, z)$ is located in the 13th and 14th columns in the first row. Meanwhile, the solution $(w, \bar{x}, \bar{y}, z)$ is located in the first and second columns in the fifth row. The solutions $(w, \bar{x}, \bar{y}, z)$ or $(\bar{w}, \bar{x}, \bar{y}, z)$ are found in the third, forth, fifth and sixth columns in the fifth row. The solution $(\bar{w}, \bar{x}, \bar{y}, z)$ is located in the 9th and 10th columns in the fifth row. The solutions $(w, x, \bar{y}, z)$ or $(w, \bar{x}, \bar{y}, z)$ are found in the 13th and 14th columns in the fifth row. Finally, the solutions $(w, \bar{x}, \bar{y}, z)$ or $(\bar{w}, \bar{x}, \bar{y}, z)$ are found in the 15th and 16th columns in the fifth row. After removing the repeated solutions, the final solutions for the SAT problem are found to be $(\bar{w}, \bar{x}, \bar{y}, z)$, $(w, \bar{x}, \bar{y}, z)$ and $(w, x, \bar{y}, z)$.

An acknowledged drawback of the proposed method is that the solution of a 3-SAT problem with m clauses and n variables requires a total of $n3^m$ microwells, i.e. the array size increases exponentially with m, but it is linear to n. However, this inefficiency represents a trade-off between the use of much simpler strands in the current computing procedures and the requirement to synthesize 2^n possible solution assignments in conventional algorithms. In practice, the 3^m possible assignments which must be considered in the array system proposed in this study are comparable to the pool size required when performing DNA computations using a hairpin formation, as proposed by Sakamoto et al. who generated 3^m literal strings initially and then excluded the conflicting strings via hairpin formations (Sakamoto et al., 2000). For example, an array with 75 (i.e. $5 \times 5 \times 3$) memory strands with a 25×24 matrix, i.e. 75×8, is required to solve an instance like $F = (t \vee y \vee z \vee u \vee v) \wedge (\bar{x} \vee \bar{w} \vee z \vee u \vee \bar{v}) \wedge (\bar{y} \vee \bar{z} \vee \bar{u})$. Therefore, as many as 75 assignments of eight variables will be utilized. In general, all initial possible $2^8 = 256$ solutions need to be generated to solve this instance. However, it is not necessary to generate all candidate solutions with the proposed approach if we first consider the first two clauses as horizontal while considering the last clause vertically. Therefore, the proposed method is much more efficient when the number of variables is large but the number of clauses is small.



(A)


$(w, x, \bar{y}, z)$
$(w, \bar{x}, \bar{y}, z)$
$(w, \bar{x}, \bar{y}, z)$
$(\bar{w}, \bar{x}, \bar{y}, z)$
$(\bar{w}, \bar{x}, \bar{y}, z)$
$(w, x, \bar{y}, z)$
$(w, \bar{x}, \bar{y}, z)$

(B)

Fig. 7. Output readouts of SAT computation extracted from interlaced result in Fig. 6C. Four patterns of output are obtained: (A) $(x, \bar{y}, z)$; (B) $(\bar{x}, \bar{y}, \bar{z})$; (C) $(x, \bar{y}, z)$ and $(\bar{x}, \bar{y}, z)$ (since y = false and z = true already satisfy the studied formula, the value of x can be either true or false); (D) $(\bar{x}, \bar{y}, z)$ and $(\bar{x}, \bar{y}, \bar{z})$. Overall, the solutions of the 2-SAT problem are $(x, \bar{y}, z)$, $(\bar{x}, \bar{y}, \bar{z})$, and $(\bar{x}, \bar{y}, z)$. (A) Images of the microarray following the computing processes. (B) Readout for $F = (x \vee y \vee z) \wedge (\bar{x} \vee \bar{w} \vee z) \wedge (\bar{y} \vee \bar{z}) \wedge (w \vee \bar{x} \vee y) \wedge (\bar{y} \vee z)$.

4. Summary

This study has presented a novel computing procedure for solving SAT problems utilizing a MEMS-based microarray technique. Although a fully automatic system to carry out the computing process has not been developed, the results of this study nevertheless confirm the feasibility of solving SAT problems on a solid surface using a modified sticker model. The proposed method has the advantage that as the size of the problem is scaled-up, it is necessary only to linearly increase the variety of sequences representing the variables and to augment the array size, instead of elongating sequences to accommodate the increased number of variables, since the conventional algorithms tend to encode all variables, each assigned as either true or false, in one sequence. A linear increase in the variety of sequences is far more satisfactory than the exponential increase of the data pool size required when a conventional algorithm is employed to solve scaled-up problems. In addition, it is less laborious in the sample preparation for those delicate sequence assignments to avoid unwanted overlaps among ssDNA sequences. A further advantage of the proposed method is that the requirements for time-consuming sample preparation procedures and sophisticated sample application equipment are avoided. Moreover, the experimental results have shown that the surface-bound DNA sequences sustain the chemical solutions during repeated computing processes, and hence the proposed method is applicable to the solution of large-scale problems.

The current study group intends to improve the efficiency of the proposed method when applied to more complicated SAT instances by exploiting enhanced image processing routines. Additionally, investigations into modifying the chemical design of the strands and developing alternative means of carrying the signals are currently underway. Although the microarray DNA computing method proposed in this study remains few limitations acknowledged above, the current authors believe that the proposed approach nevertheless represents a feasible basis for developing advanced algorithms and experimental designs to solve other problems in DNA computation.

Acknowledgement

The authors gratefully acknowledge the financial support provided to this study by the National Science Council of Taiwan under grant no. NSC 93-2113-M-214-001.

References

Adleman, L., 1994. Molecular computation of solutions to combinatorial problems. Science 266 (5187), 1021–1024.

Braich, R.S., Chelyapov, N., Johnson, C., Rothemund, P.W.K., Adleman, L., 2002. Solution of a 20-variable 3-SAT problem on a DNA computer. Science 296 (5567), 499–502.

Britland, S., Perez-Arnaud, E., Clark, P., McGinn, B., Connolly, P., Moores, G., 1992. Micropatterning proteins and synthetic peptides on solid supports: a novel application for microelectronics fabrication technology. Biotechnol. Progr. 8 (2), 155–160.

Chen, K., Ramachandran, V., 2001. A space-efficient randomized DNA algorithm for k-SAT. Condon.

Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C., 2001. Introduction to Algorithms. The MIT Press, Boston.

Diaz, S., Esteban, J.L., Ogihara, M., 2001. A DNA-Based Random Walk Method for Solving k-SAT. Springer–Verlag, Berlin.

Graham, C.R., Leslie, D., Squirrell, D.J., 1992. Gene probe assays on a fibre-optic evanescent wave biosensor. Biosens. Bioelectron. 7 (7), 487–493.

Lipton, R.J., 1995. DNA solution of hard computational problems. Science 268, 542–545.

Liu, Q.H., Wang, L.M., Frutos, A.G., Condon, A.E., Corn, R.M., Smith, L.M., 2000. DNA computing on surfaces. Nature 403 (6766), 175–179.

Liu, W.B., Zhang, F.Y., Xu, J., 2002. A DNA algorithm for the graph coloring problem. J. Chem. Inform. Comput. Sci. 42 (5), 1176–1178.

Ouyang, Q., Kaplan, P.D., Liu, S.M., Libchaber, A., 1997. DNA solution of the maximal clique problem. Science 278 (5337), 446–449.

Paun, G., Rozenberg, G., 1998. Sticker systems. Theor. Comput. Sci. 204 (1/2), 183–203.

Pirrung, M.C., Connors, R.V., Odenbaugh, A.L., Montague-Smith, M.P., Walcott, N.G., Tollett, J.J., 2000. The arrayed primer extension method for DNA microchip analysis. molecular computation of satisfaction problems. J. Am. Chem. Soc. 122 (9), 1873–1882.

Rozenberg, G., Spaink, H., 2003. DNA computing by blocking. Theor. Comput. Sci. 292 (3), 653–665.

Sakamoto, K., Gouzu, H., Komiya, K., Kiga, D., Yokoyama, S., Yokomori, T., Hagiya, M., 2000. Molecular computation by DNA hairpin formation. Science 288 (5469), 1223–1226.

Schmidt, K.A., Henkel, C.V., Rozenberg, G., Spaink, H.P., 2004. DNA computing using single-molecule hybridization detection. Nucleic Acids Res. 32 (17), 4962–4968.

Su, X.P., Smith, L.M., 2004. Demonstration of a universal surface DNA computer. Nucleic Acids Res. 32 (10), 3115–3123.

Wang, L.M., Liu, Q.H., Corn, R.M., Condon, A.E., Smith, L.M., 2000. Multiple word DNA computing on surfaces. J. Am. Chem. Soc. 122 (31), 7435–7440.

Yang, C.N., Chao, C.H., Cheng, S.P., Lin, C.H., 2006. A MEMS based DNA computer for solving SAT problems. In: Proceedings of the 1st IEEE International Conference of Nano/Micro Engineered and Molecular Systems, Zhuhai.

Yang, C.N., Yang, C.B., 2005. A DNA solution of SAT problem by a modified sticker model. Biosystems 81 (1), 1–9.

Yoshida, H., Suyama, A., 1999. Solution to 3-SAT by Breadth First Search. American Mathematical Society, Providence.