Towards a System for Simulating DNA Computing with Whiplash PCR

Akio Nishikawa and Masami Hagiya

Department of Information Science, Graduate School of the University of Tokyo, Tokyo, JAPAN
{nisikawa, hagiya}@is.s.u-tokyo.ac.jp

Abstract- Whiplash PCR, a reaction in which hairpin DNA structures are extended by polymerase, is a useful method for DNA computing. For example, state transitions can be implemented by this method. However, the feasibility of using whiplash PCR in a specific DNA algorithm should be carefully considered. We have implemented a general-purpose simulator to aid those who design protocols and algorithms for DNA computing, that includes whiplash PCR. In this paper, the simulator, including its data structures and a simulation algorithm, is described, and then the results of simulating state transitions with whiplash PCR are discussed.

1 Introduction

DNA computing has recently become a new research field. Most of the basic operations for DNA computing come from established biological techniques, including ligation, PCR, restriction digestion, and so on. In addition to these operations, whiplash PCR was proposed as a useful new operation [hagi97]. In whiplash PCR, DNA polymerase extends the 3′-end of a hairpin structure. If the subsequence at the 3′-end is considered to encode the current state of the molecule, this reaction can be considered as a state transition. Various applications of whiplash PCR have been reported [hagi97, saka98, win98a]. However, the experimental conditions for the operation must be considered carefully, to prevent unexpected competition between intra-molecular and inter-molecular reactions. Therefore, it is desirable to check the feasibility of whiplash PCR with simulations before performing actual experiments.

There are already some simulators for DNA computing. For example, Winfree analyzed the assembly of DNA tiles with his simulator [win98b]. Zhang and Shin developed a toolkit for simulating DNA reactions [zhang98] and simulated various DNA algorithms. We have also developed a general-purpose simulator for DNA computing [nishi99]. To our knowledge, our simulator is the only one that can simulate reactions involving hairpin structures. We recently extended the simulator, so that it can now simulate whiplash PCR. Using the simulator, we examined the feasibility of implementing state transitions by whiplash PCR, and observed the competition between intra-molecular and inter-molecular reactions.

This paper is organized as follows. In the next section, we briefly explain whiplash PCR. In Section 3, the implementation of the simulator is described. In Section 4, some simulation results by using it are reported in detail. Section 5 includes some discussion, and the final section contains our concluding remarks.

2 Whiplash PCR

Whiplash PCR is a reaction in which the 3′-end of a single-strand DNA molecule, taking a hairpin form, is extended using a subsequence of itself as the template for polymerization (Figure 1). If we regard the subsequence at the 3′-end as encoding the current state of the molecule, this reaction can be viewed as a state transition. Therefore, a DNA molecule can be a finite state machine whose transition table is encoded within the molecule itself.

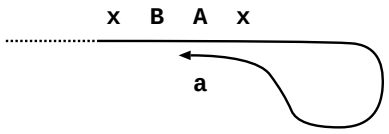

Figure 1: A hairpin structure for whiplash PCR

Stated more precisely, the transition table consists of subsequences in the form

$$stopper - state' - state,$$

where *state* denotes the state before a transition, and *state′* denotes the state after the transition.

A sequence complementary to that of the current state is attached to the 3′-end of the molecule. Assume that it is the complement of *state*. Since the molecule forms a hairpin structure, the 3′-end of the molecule anneals to the subsequence *state*, in the transition table of the molecule. The complement of *state′* is then added to the 3′-end by polymerization. The stopper sequence, denoted by *stopper*, stops polymerization right after the complement of *state′* is added. This function is called *polymerization stop*.

There is more than one method to implement polymerization stop. The method we employ is to remove one of the four bases from the polymerization buffer and use a string of the complement of the removed base as the stopper sequence.

Notice that the state machine described here does not read an input symbol at each transition, though one can supply input as a part of its initial state. In order to implement a machine that reads an input symbol at each transition, one can use a transition table consisting of subsequences of the following form.

$$stopper - state' - input - state$$

At each transition, an input symbol is first attached to the $3'$-end of the DNA molecule. However, as we explain below, we are more interested in successive transitions performed in a one-pot reaction (without interruption for input), because they enable $O(1)$-computation of NP-complete problems, etc.

If the hairpin structure formed to enable a transition is denatured, then a new hairpin structure can be formed to enable a further transition. Experimentally, we have verified that thermal cycling or the appropriate iso-thermal conditions can achieve successive transitions. The entire reaction just explained is called a *whiplash PCR*. By allowing successive transitions in a one-pot experiment, we can gain a high computational power with a constant number of bio-steps (biological steps). We assume here that

- Any number of transitions can occur by a single bio-step.
- The length and the amount of DNA molecules do not affect the number of bio-steps.

Under these assumptions, some NP-complete problems can be solved in $O(1)$-computation, where $O(1)$ means that the number of required bio-steps is independent of the problem size.

So far, we have proposed many applications for whiplash PCR. Parallel computation of Boolean formulas, $O(1)$-computation for solving various NP-complete problems [win98a], including the n-SAT problem, Hamiltonian path problem, vertex cover problem, direct sum problem and so on.

3 The Simulator

The authors have developed a general-purpose simulator for DNA computing, called VNA (Virtual Nucleic Acid) [nishi99]. It consists of four parts. There are two main modules: the hybrid analyzer for abstract sequences and the numeric calculator based on differential equations. They are closely related with each other. The other two parts are the graphical user-interface module and supplementary modules. Users can add functions to the simulator by preparing supplementary modules, if necessary. Figure 2 shows an overview of the simulator and summarizes the interactions between the modules.

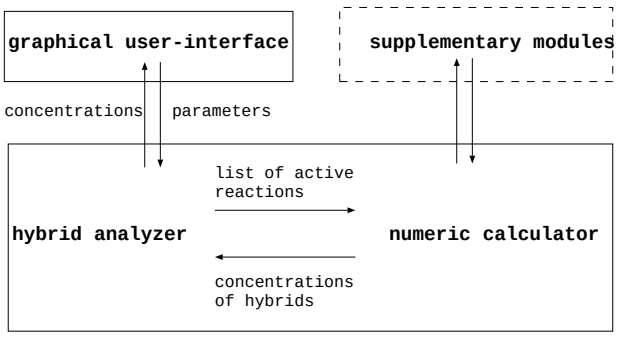

Figure 2: Summary of the interactions within the simulator

3.1 Data Structures

An abstract sequence is a finite sequence of abstract bases. A letter of the alphabet, either lowercase or uppercase, represents each abstract base. A lowercase alphabetic letter is considered as complementary to the corresponding uppercase letter and vice versa. Each abstract sequence has two ends; one is called the $5'$-end, and the other the $3'$-end. The sequence is usually written from the $5'$-end to the $3'$-end. Two abstract sequences are complementary to each other if they consist of complementary bases when one sequence is read from the $5'$-end to the $3'$-end, and the other is read from the $3'$-end to the $5'$-end. For example, the abstract sequences, `ab` and `BA`, are complementary to each other.

A hybrid consists of abstract sequences and bonds between them. Since a hybrid may contain more than one occurrence of the same abstract sequence, we use the word "strand" to denote an occurrence of an abstract sequence in a hybrid.

A bond associates two subsequences of strands that are complementary to each other. Each bond consists of the strands whose subsequences are associated, the position of each subsequence in the strand, and the length of the bond. For example, the following hybrid consists of two strands, `abc` and `FEDCB`, and a bond between them.

```
abc
 ||
 BCDEF
```

Notice that `FEDCB` is written from the $3'$-end to the $5'$-end in the figure. The bond consists of the two strands, the position of `bc` in `abc`, the position of `CB` in `FEDCB`, and the length of the bond, i.e., 2.

A bond can associate a strand with itself. For example, the following hybrid consists of one strand, `abcFEDCB`, and one bond, which associates the subsequences, `bc` and `CB`, in the same strand, as shown.

```
abcF
 || E
 BCD
```

It is always a vexing problem to judge the equality between linked data structures like hybrids in our simulator. In order to judge the equality of hybrids, we normalize hybrids into their normal forms. To obtain the normal form of a hybrid, we first choose the root strand of the hybrid according some criterion. Beginning with the root strand, we traverse the hybrid along the bonds, visiting strands of the hybrid in a depth-first manner. This determines a linear order among the strands.

However, if a hybrid contains the same abstract sequence more than once, there is no unique root strand, and therefore there is no unique normal form in general. This means that the equality of hybrids cannot be judged completely. This does not make simulations impossible, only inefficient due to the duplication of hybrids and reactions.

3.2 Reactions

Hybridization is a reaction in which two hybrids are hybridized to form a complex hybrid by a newly created bond between the two strands, one from each of the two hybrids. At least one of the newly associated subsequences must be at a free end, i.e., there should be no bond from the subsequence to one end of the strand. For example, the hybrids

```
abcF        abcF
 || E        || e
 BCD         BCD
```

cannot form a complex hybrid by a new bond between `e` and `E`, because neither subsequence is at a free end.

In hybridization, each new bond is created all at once independent of length of the new bond, using the maximal complementary subsequences. For example, when the one-strand hybrid, `abc`, and the one-strand hybrid, `FEDCB`, are hybridized, the following partial hybrids are not formed.

```
abc          abc
  |           |
 BCDEF       BCDEF
```

In this paper, the hybridization reaction constant is independent of the length of the new bond. (We are investigating the possibility of making it dependent on the length.)

Self-hybridization is a reaction in which a new bond is created inside a hybrid. A new bond either associates a strand with itself or associates two strands in the hybrid. A similar requirement applies to self-hybridization. In this paper, the self-hybridization reaction constant is independent of the length of the new bond as well as the topology of the existing bonds in the hybrid. The self-hybridization reaction constant is usually larger than the reaction constant of the hybridization reaction constant.

Denaturation is the inverse of hybridization or self-hybridization. In denaturation, a bond in a hybrid is broken. As a consequence, the hybrid may be split into two. The denaturation reaction constant is defined as kf^n, where n is the length of the bond to be broken, f is the denaturation factor and is less than 1, and k is the denaturation constant.

After a bond is removed from a hybrid in denaturation, any bond that is adjacent to a free end of a strand is automatically extended. For example, consider the following hybrid (left). If the bond between `abc` and `FEDCB` is removed, the bond between `FEDCB` and `cde` is extended to obtain the hybrid at the right.

```
abc
 ||
BCDEF          BCDEF
  ||            |||
 cde            cde
```

Extension is a reaction in which the 3′-end of a strand is extended according to its complementary sequence. For example, the hybrid

```
abc
 ||
BCDEF
```

is extended to produce either of the following hybrids.

```
abc            abcdef
|||             |||||
ABCDEF         BCDEF
```

Both of these hybrids are further extended, and will produce the same hybrid

```
abcdef
||||||
aBCDEF
```

For extension, it is possible to specify the stopper base. Extension stops before the stopper base. If `E` is specified as the stopper base in the above example, extension stops before `E` to give

```
abcdef
 |||
BCDEF
```

The reaction constant of extension is of the form k/n, where n is the number of abstract bases to be extended.

Ligation concatenates two strands if they have a bond to the same strand and the two bonds are adjacent. The strands, `abc` and `def`, in the following hybrid are concatenated.

```
abc   def
  \\ ///
   BCDEF
```

Ligation may produce a circular strand. However, in the current implementation of the simulator, circular strands are not allowed, because molecules including circular strands cannot be identified by the module for the linear ordering of strands. Ligation reactions that lead to a circular strand are ignored after a warning message.

Digestion is a reaction in which a strand is cut at a specified *restriction site* if the subsequence at the site is double stranded. For example, if `cd` is specified as a

restriction cite, the following hybrid at the left is cut into the two hybrids at the right.

```
abcdef        abc def
 |||||         || |||
 BCDEF         BC DEF
```

3.3 Simulation Algorithm

The simulator supports two simulation algorithms: *threshold* and *stochastic* [nishi99]. To simulate whiplash PCR, we employed the simulation algorithm with the threshold. Of course, whiplash PCR can also be simulated by the stochastic approach. However, the simulator cannot calculate the concentration of the hybrids by the approach. Therefore the simulator cannot select the major reaction paths by it. The simulator keeps a set of hybrids and a set of active reactions. The former is initialized with the initial hybrids which are usually one-strand hybrids. The latter is initialized to the empty set. These sets are enlarged as a simulation proceeds. In the current implementation, they are never decreased. Once a reaction is activated, it is never inactivated.

The algorithm iterates the loop consisting of the following steps:

- Update the concentration of each hybrid using the active reactions. The increase or decrease in a concentration is computed from the current concentrations of the reactants of the reaction, the reaction constant, and the duration of each time step.
- Enumerate new reactions whose reactants all have a concentration that exceeds the threshold concentration. Add the reaction products to the set of hybrids if they are not already in the set.
- Activate the enumerated reactions.

Note that after a reaction is made active, it remains active in the rest of a simulation and contributes to the concentration computations, even when its products have concentrations less than the threshold concentration.

4 Simulation results of whiplash PCR

As described in Section 2, whiplash PCR is a useful method for DNA computing. For example, state transitions can be implemented by this method [hagi97, saka98, win98a]. Since state transitions simulated with whiplash PCR require the polymerization stop at the stopper sequence, we newly added this capability to the simulator. In our simulator, the stopper sequence is implemented as one abstract base. It is set to `x` in the following simulations. Since the simulator was already able to simulate self-hybridization and form hairpin structures, we were able to try to simulate DNA computing algorithms with whiplash PCR immediately after we added the capacity of polymerization stop.

In this section, we describe the results of several simulations with whiplash PCR. They are simulations of three state transitions, five state transitions, and competition between intra-molecular and inter-molecular reactions.

4.1 Three state transitions with whiplash PCR

First, we tried to simulate whiplash PCR for three state transitions. The only initial sequence was `JxBAxCBxDCxa`, which could form a hairpin structure by self-hybridization between `A` and `a`, as shown in Figure 1.

The conditions for the simulation are summarized in Table 2 and are the same as those used for simulating five and seven state transitions, described later.

Figure 3 shows the result of simulating three state transitions with the graphical user interface. Each curve in the graph corresponds to one hybrid and only the ten most abundant hybrids are plotted, omitting minor products. At some time steps the tenth product changes.

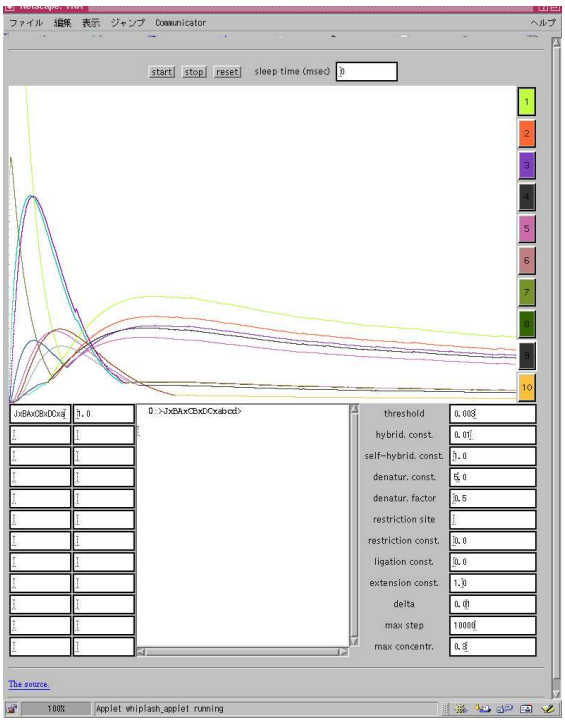

Figure 3: Simulating three state transitions

Figure 4 shows the change of strands in three state transitions. The strands are extended by whiplash PCR and the 3'-end of each strand indicates a state. For example, the state of the strand `JxBAxCBxDCxa` is `a` and that of `JxBAxCBxDCxab` is `b`. The sequence `BA` in the molecule works as a transition rule from the state `a` to the state `b`. By this transition mechanism, Winfree implemented GOTO programs [win98a].

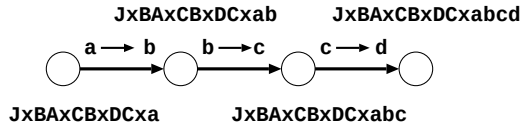

Figure 4: The three state transitions

	Three state transitions	
	Sequences	Yield
Initial	JxBAxCBxDCxa	1.0
Final	JxBAxCBxDCxabcd	0.71
Final	JxBAxCBxDCxabc	0.13
Final	JxBAxCBxDCxab	0.11
Final	JxBAxCBxDCxa	0.04

Table 1: Simulating three state transitions

While Figure 3 is a graph of the concentrations of the hybrids, the graph shown in Figure 5 plots the concentrations of the abstract sequences of the hybrids in the simulation.

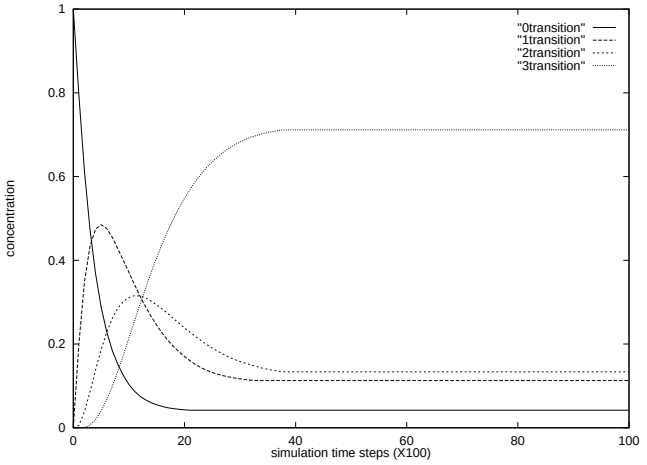

Figure 5: The plot of the concentration of each sequence in three state transitions

The result shows that the first and second transitions are efficient compared with the third one. The first tall peak in Figure 3 is the product of the first transition. The next two peaks, which nearly overlap, are of the product of the second transition. One peak is for the hairpin form, and the other is for the denatured form.

4.2 Five and seven state transitions with whiplash PCR

The initial sequences for five and seven state transitions were JxBAxCBxDCxEDxFExa and JxBAxCBxDCxEDxFExGFxHGxa, respectively.

Parameters	Values
threshold	0.003
hybridization const.	0.1
self-hybridization const.	1.0
denature const.	5.0
denature factor	0.5
restriction site	
restriction const.	0.0
ligation const.	0.0
extension const.	1.0
period of each step	0.01
max steps	10,000
max concentration	0.3

Table 2: The conditions used for simulating three, five and seven state transitions

	Five state transitions	
	Sequences	Yield
Initial	JxBAxCBxDCxEDxFExa	1.0
Final	JxBAxCBxDCxEDxFExabcdef	0.40
Final	JxBAxCBxDCxEDxFExabcde	0.18
Final	JxBAxCBxDCxEDxFExabcd	0.14
Final	JxBAxCBxDCxEDxFExabc	0.13
Final	JxBAxCBxDCxEDxFExab	0.11
Final	JxBAxCBxDCxEDxFExa	0.04

Table 3: Simulating five state transitions

The tendency of the result of simulating five state transitions shown in Table 3 and Figure 6 is almost the same as that of three state transitions. The yield of the final product is, however, smaller than that of three state transitions.

The simulation of seven state transitions shows a different tendency from those of three and five state transitions. It generated almost no final product, even though we simulated 40,000 steps and more. The result of the 10,000-step simulation is shown in Table 4 and Figure 7.

4.3 Competition between intra-molecular and inter-molecular reactions

We also tried to simulate the competition between intra-molecular and inter-molecular reactions, that is, self-hybridization and normal hybridization.

Tables 5 and 6 show the results of two simulations of competitive reactions. All the basic conditions for these two simulations are the same, except for the hybridization and self-hybridization constants. The shared conditions are summarized in Table 7.

Comparing the results in Tables 5 and 6, we can see the effect of the hybridization and self-hybridization constants. If these two constants are equal, as in Table 6,

	Seven state transitions	
	Sequences	Yield
Initial	JxBAxCBxDCxEDxFExGFxHGxa	1.0
Final	JxBAx.....xHGxabcdef	0.23
Final	JxBAx.....xHGxabcdefg	0.17
Final	JxBAx.....xHGxabcde	0.17
Final	JxBAx.....xHGxabcd	0.14
Final	JxBAx.....xHGxabc	0.13
Final	JxBAx.....xHGxab	0.11
Final	JxBAx.....xHGxa	0.04
Final	JxBAx.....xHGxabcdefgh	0.007

Table 4: Simulating seven state transitions

	Hybrid. const. = 0.01 Self-hybrid. const. = 1.0	
	Sequences	Yield
Initial	JxPAxCPxQCxa	1.0
Initial	JxBAxCBxDCxa	1.0
Final	JxPAxCPxQCxapcq	0.60
Final	JxBAxCBxDCxabcd	0.60
Final	JxBAxCBxDCxab	0.16
Final	JxPAxCPxQCxap	0.16
Final	JxBAxCBxDCxabc	0.15
Final	JxPAxCPxQCxapc	0.15
Final	JxBAxCBxDCxa	0.08
Final	JxPAxCPxQCxa	0.08
Final	JxPAxCPxQCxab	0.004
Final	JxBAxCBxDCxap	0.004

Table 5: Simulating competitive state transitions

	Hybrid. const. = 0.1, Self-hybrid. const. = 0.1	
	Sequences	Yield
Initial	JxPAxCPxQCxa	1.0
Initial	JxBAxCBxDCxa	1.0
Final	JxPAxCPxQCxa	0.37
Final	JxBAxCBxDCxa	0.37
Final	JxPAxCPxQCxap	0.34
Final	JxBAxCBxDCxab	0.34
Final	JxPAxCPxQCxab	0.14
Final	JxBAxCBxDCxap	0.14
Final	JxBAxCBxDCxabc	0.13
Final	JxPAxCPxQCxapc	0.13
Final	JxBAxCBxDCxabcd	0.02
Final	JxPAxCPxQCxapcq	0.01

Table 6: Simulating competitive state transitions

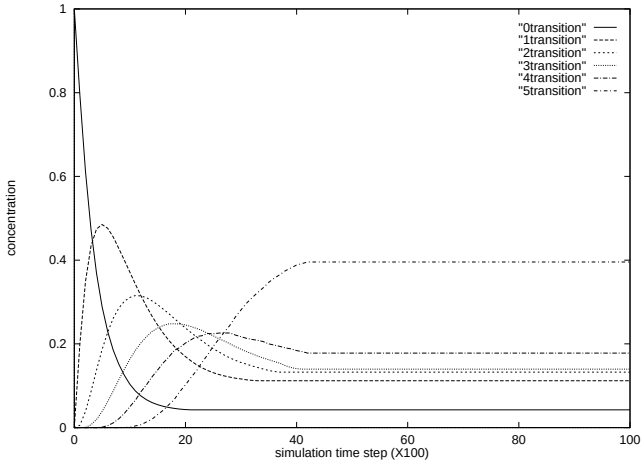

Figure 6: The plot of the concentration of each sequence in five state transitions

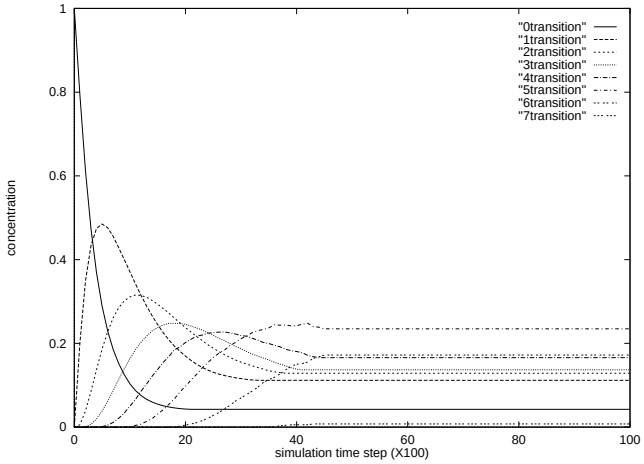

Figure 7: The plot of the concentration of each sequence in seven state transitions

the hybrids produced by inter-molecular hybridization increase. In contrast, if the self-hybridization constant is larger, the hybrids produced by inter-molecular hybridization become minor and negligible.

5 Discussion

We initially planned to simulate real DNA sequences rather than abstract sequences, and thought that we could find possible mis-hybridizations by string matching between DNA sequences. However, this approach did not seem practical for large-scale simulations. Even with abstract sequences, simulations of reasonable size, such as the simulation of seven state transitions, take days on the fastest workstations.

Although it is more efficient, the simulator based on abstract sequences, has several disadvantages. First, ab-

parameters	values
threshold	0.01
denature const.	5.0
denature factor	0.5
restriction site	
restriction const.	0.0
ligation const.	0.0
extension const.	1.0
period of each step	0.01
max steps	10,000
max concentration	0.3

Table 7: The shared conditions for the simulations

stract sequences in the current implementation do not contain information on the length and GC content of the corresponding real DNA sequence. Therefore, the physical properties of abstract sequences, including various reaction constants, should be given *a priori*. Second, simulations based on abstract sequences ignore mishybridizations and some other experimental errors. In addition, since self-hybridizations do not take the distance between strands into account, the simulator produces hybrids that are physically impossible. To minimize these disadvantages, we must add some new attributes to abstract sequences in the future.

Whiplash PCR utilizes intra-molecular reactions, especially self-hybridization. The initial sequences for whiplash PCR include at least one pair of complementary subsequences that can form a hairpin structure. Since the pair of sequences can also form a normal hybrid consisting of two strands, the competition between intra-molecular and inter-molecular reactions cannot be avoided. In general, intra-molecular reactions are faster than inter-molecular reactions. (It is even possible to force intra-molecular reactions by attaching the molecules to beads.) Of course, we can tune the parameter values of the simulator so that the ratio of the two rate constants gives reasonable results. We need to refine the reaction model and verifying it with *in vitro* experiments.

6 Concluding Remarks

This paper presented a simulator for DNA computing with whiplash PCR. The simulator is still under development. The paper described the current stage of the development and discussed the simulations we have. Based on this discussion, the simulator can be refined to be a better tool for designers of DNA computation algorithms.

Acknowledgements

This research was supported by the Japan Society for the Promotion of Science under the *Research for the Future* Program (JSPS-RFTF 96I00101).

Bibliography

[hagi97] Masami Hagiya, Masanori Arita, Daisuke Kiga, Kensaku Sakamoto and Shigeyuki Yokoyama, "Towards Parallel Evaluation and Learning of Boolean μ-Formulas with Molecules," preliminary Proc. of Third DIMACS Workshop on DNA Based Computers, 1997.

[nishi99] Akio Nishikawa, Masami Hagiya and Masayuki Yamamura, "Virtual DNA Simulator and Protocol Design by GA," GECCO'99 *to appear*, 1999.

[ogi98] Mitsunori Ogihara and Animesh Ray, "DNA-Based Self-Propagating Algorithm for Solving Bounded-Fan-In Boolean Circuits," Genetic Programming 98, pp.725–730, 1998.

[saka98] Kensaku Sakamoto, Daisuke Kiga, Ken Komiya, Hidetaka Gouzu, Shigeyuki Yokoyama, Shuji Ikeda, Hiroshi Sugiyama and Masami Hagiya, "State transitions by molecules," preliminary Proc. of Fourth International Meeting on DNA Based Computers, pp.87–99, 1998.

[win98a] Erik Winfree, "Whiplash PCR for O(1) computing," preliminary Proc. of Fourth International Meeting on DNA Based Computers, pp.175–188, 1998.

[win98b] Erik Winfree, "Simulations of Computing by Self-Assembly," preliminary Proc. of Fourth International Meeting on DNA Based Computers, pp.213–239, 1998.

[win96] Erik Winfree, Xiaoping Yang and Nadrian C. Seeman, "Universal Computation via Self-assembly of DNA:Some Theory and Experiments," preliminary Proc. of Second Annual Meeting on DNA Based Computers, pp.172–190, 1996.

[zhang98] Byoung-Tak Zhang and Soo-Yong Shin, "Molecular Algorithms for Efficient and Reliable DNA Computing," Genetic Programming 98, pp.735–742, 1998.