

DNA Sequence Design Using Templates^{*1}

Masanori ARITA

*PRESTO, JST and Computational Biology Research Center, AIST
2-41-6 Aomi, Koto-ku, Tokyo 135-0064 JAPAN*

m-arita@aist.go.jp

Satoshi KOBAYASHI

*Department of Computer Science, University of Electro-Communications
1-5-1 Chofugaoka, Chofu-shi, Tokyo 182-8585 JAPAN*

satoshi@cs.uec.ac.jp

Received 15 September 2001

Revised manuscript received 15 February 2002

Abstract Sequence design is a crucial problem in information-based biotechnology such as DNA-based computation. We introduce a simple strategy named *template method* that systematically generates a set of sequences of length l such that any of its member will have approximately $l/3$ mismatches with other sequences, their complements, and the overlaps of their concatenations.

Keywords: DNA Computing, DNA Array, Error Correcting Code.

§1 Introduction

DNA computation attempts to harness the computational power of molecules for information processing, and many computational models have been proposed for solving mathematical problems in laboratory experiments.¹⁾ To achieve success, a good coding strategy for sequences is crucial, because the code determines the way to process information on sequences in the experiments. Such experiments with artificially coded sequences are called *in vitro* computation. The necessity of sequence design arises not only in computation *in vitro* but in other biotechnologies, such as the design of DNA chips for mutational analysis and for sequencing. In all approaches, sequences are designed so that each element uniquely hybridizes to its complementary sequence, but not to any other sequence.

^{*1} This paper has been selected to receive the New Generation Computing Award for Distinguished Papers.

Due to the differences in experimental requirements, however, it seems impossible to establish an all-purpose library of sequences that effectively caters to the requirements of all laboratory experiments. Therefore, researchers must design a new set of sequences for each experiment depending on various design constraints.

One particularly common constraint for preventing experimental errors is to minimize the similarity between sequences under some distance measure, such as the Hamming distance. Another constraint is to standardize the GC content of sequences, which is a good indicator of the melting temperature of short oligonucleotides. Usually these constraints are also imposed on the reverse complements of the sequences. Additional constraints are necessary when a part of the sequences is pre-determined, or when some sequences are enzymatically processed during *in vitro* computation. One typical instance is the use of restriction or ligation enzymes to split or concatenate sequences.

There is a trade-off between the tightness of these constraints and the number of sequences we can design. Because of the variety of the constraints, however, the design problem is not straightforward and at present there is no standard design protocol. Previous approaches tested combinatorial design, random generation, and genetic algorithms.^{3,5,6)}

In this paper, we introduce a design method that uses two types of binary words, i.e. *templates* and error-correcting codewords. The basic idea is to design sequences in two steps by fixing [AT] and [GC] positions for all sequences. In the first step, positions for [AT] and [GC] are determined. These positions are specified by a template $x_1x_2 \cdots x_l$ ($x_i \in \{0, 1\}$), where 1 indicates [AT], and 0 indicates [GC] (or vice versa). In the second step, either A or T is chosen for template positions $x_i = 1$, and either G or C for positions $x_i = 0$ ($1 \leq i \leq l$). These choices are specified by an error-correcting codeword.

The idea originated in the work of Frutos *et al.*,⁶⁾ but its practical utility was overlooked. Here we develop their idea to obtain a more powerful result. The new method does not guarantee optimality in terms of the number of sequences we can design, but it is flexible enough to find application in broad areas of biotechnology.

§2 Template Method

Given a sequence $x = s_1s_2s_3 \cdots s_{l-1}s_l$, by $\langle x \rangle$, we denote the subsequence $s_2 \cdots s_{l-1}$ (a subsequence without a terminal letter at either end). Throughout the paper, only sequences of length $l \geq 2$ are considered. If not otherwise mentioned, by l , we denote the length of a sequence. Since no particular experimental model is presupposed in this paper, we assume any pair of sequences may be concatenated in the computation *in vitro*. The problem we will consider is as follows:

Problem 2.1

Design the set S of length l DNA sequences such that any sequence $x \in S$ or its reverse complement mismatches in at least d positions from any other sequence $y \in S$ ($x \neq y$) or the overlap region of two sequences, i.e. $\langle yz \rangle$ ($y, z \in S$).

Moreover, all sequences must share the same GC content.

Hereafter, the word ‘sequence’ is used for DNA sequences to be designed (i.e. 4-alphabet words), whereas ‘string’ is for templates or binary codewords. The length of a string is denoted $|x|$. For a string x , x^R and x^{-i} denote the reverse string of x , and the string obtained by cyclic shifting the block of the first i bits to the end of the string, respectively. The Hamming distance $H(x, y)$ between two strings $x = x_1x_2 \dots x_l$ and $y = y_1y_2 \dots y_l$ is the number of indices i such that $x_i \neq y_i$. To count the mismatches between strings of different lengths, the notation $H_M(x, y)$ ($|y| \geq |x|$) indicates the minimum Hamming distance between x and $|y| - |x| + 1$ substrings of length $|x|$ in y . Note that $H_M(x, \langle xx \rangle) = \min_{0 < i < |x|} H(x, x^{-i})$.^{*2}

Definition 2.1

To count the least number of possible mismatches, the *mass* of x (denoted $\|x\|$) is defined as:

$$\|x\| \stackrel{def}{=} \min(H(x, x^R), H_M(x, \langle xx \rangle), H_M(x, \langle x^R x^R \rangle), H_M(x, \langle xx^R \rangle), H_M(x, \langle x^R x \rangle)).$$

Example 2.1

For a template $x = 110001$, $\|x\| = 0$, because $H_M(x, \langle x^R x^R \rangle) = 0$.

The terms $H_M(x, \langle xx^R \rangle)$ and $H_M(x, \langle x^R x \rangle)$ in Definition 2.1 may be dropped when sequences are never concatenated with their reverse complements in the computation *in vitro*. This variant will be discussed in the next section.

In the template method, the original problem is decomposed into two subproblems.

1. Find a single template x satisfying $\|x\| = d$.
2. Find an error-correcting binary code E of minimum distance d .

From the template x and any codeword in E , words over 4 alphabet letters are derived as a product $S = \{x \cdot y \mid y \in E\}$ (Fig. 1), which we write $S = x \cdot E$, for simplicity. It is immediately obvious that sequences in S mismatch at least d positions from other sequences including their reverse complements and concatenations. The number of sequences we can design depends on the factor code E , and this subproblem has been studied extensively in code theory.¹⁰⁾ Therefore, our only task is the first subproblem: finding a good template.

template	1	1	0	1	0	0
codeword	1	0	1	0	1	0
	↓	↓	↓	↓	↓	↓
DNA	A	T	G	T	G	C

Fig. 1 Design Example Using Template 110100

^{*2} This measure is not a distance, for it does not satisfy the triangle inequality. Although the notation is different, the idea of shifting sequences is the same as in Garzon *et al.*⁷⁾

2.1 Finding Templates

Since typical sequences used in biotechnology are shorter than $l = 30$ bases, good templates can be found by an exhaustive search, i.e. by calculating the mass for all $O(2^l)$ strings. In this section, we show several bounds for the search space, given a sequence length and a desired mass value. Throughout this section, let us deal with a simpler problem where Definition 2.1 is replaced with:

Definition 2.2

$$\|x\| \stackrel{def}{=} \min(H(x, x^R), H_M(x, \langle xx \rangle), H_M(x, \langle x^R x^R \rangle)).$$

Lemma 2.1

For any template x , $\|x\|$ is always even.

Proof

It is immediately recognized that $H(x, x^R)$ is even. $H(x, x^{-1})$ is even because for each consecutive region of bit 1 in x , 2 mismatch candidates arise. By induction on the number of shifts, we can prove that $H_M(x, \langle xx \rangle)$ is even. In the same way, $H(x, \langle x^R x^R \rangle)$ is shown to be even. ■

Lemma 2.2

Mass is invariant under cyclic shifts and reversals.

Proof

Let $\|x^{-i}\| = d$. First, we prove $d = \|x\|$. If $d = H_M(x^{-i}, \langle x^{-i} x^{-i} \rangle)$, then for some j ($i \neq j$), $d = H(x^{-i}, x^{-j}) = H(x, x^{-(j-i)}) \geq H_M(x, \langle xx \rangle) \geq \|x\|$. If $d = H_M(x^{-i}, (x^{-i})^R \langle (x^{-i})^R \rangle)$, then for some j , $d = H(x^{-i}, (x^{-j})^R) = H(x^{-i}, (x^R)^{-j}) = H(x, (x^R)^{-j-i}) \geq H_M(x, x^R \langle x^R \rangle) \geq \|x\|$. Thus, $d \geq \|x\|$. Conversely, we can show $\|x\| \geq d$ in a similar way. Next, we will prove $\|(x^R)^{-i}\| = \|x\|$. In case of $i = 0$, we have:

$$\begin{aligned} \|x\| &= \min(H(x, x^R), H_M(x, \langle xx \rangle), H_M(x, \langle x^R x^R \rangle)) \\ &= \min(H(x, x^R), H_M(x^R, \langle x^R x^R \rangle), H_M(x^R, \langle xx \rangle)) \\ &= \|x^R\|. \end{aligned}$$

By repeating the first argument again with x^R , the proof is complete. ■

Lemma 2.2 shows that $\{x^{-i} \mid 0 \leq i < l\} \cup \{(x^R)^{-i} \mid 0 \leq i < l\}$ forms a certain code. For finding a template, one only needs to find one codeword in each set.

Lemma 2.3

Every template x of length l belongs to a cyclic code E of minimum distance $d = \|x\|$ and size $|E| \geq 2l$.

Proof

We construct a cyclic code E consisting of l cyclic shifts of both x and x^R . Suppose a pair of words $u, v \in E$. When both u and v are cyclic shifts of x , $H(u, v) = H(x^{-i}, x^{-j}) = H(x, x^{-(j-i)}) \geq d$. The proof when both u and v are cyclic shifts of x^R is similar. Suppose either one is a cyclic shift of x^R , say $u =$

x^{-i} and $v = (x^R)^{-j}$. Then, $H(u, v) = H(x, (x^R)^{-j}) \geq H_M(x, x^R x^R) \geq d$. The lemma follows. ■

When the block of bits formed by reversing the order of code bits is always another codeword in the same code, the code is called *reversible*.¹¹⁾ The code E constructed in Lemma 2.3 is thus reversible.

The next few lemmas give constraints based on the number of bit 1's in the template x , denoted $weight(x)$.

Lemma 2.4

For any template x , there always exists a template x' such that $\|x'\| = \|x\|$ and $weight(x') \leq \lfloor l/2 \rfloor$.

Proof

Since $weight(x)$ is equal for all $x \in E$, flipping all bits for all x gives another code of the same property. ■

By Lemma 2.4, it suffices to search for a template x such that $weight(x) \leq \lfloor l/2 \rfloor$. Therefore, in the rest of this section we assume $weight(x) \leq \lfloor l/2 \rfloor$.

Lemma 2.5

For any template x of length l , $\|x\| \leq \lfloor l/2 \rfloor$.

Proof

Let $w = weight(x)$. The maximum number of mismatches between x and the l cyclic shifts of x^R is $2w(l - w) \leq l^2/2$. Since $\|x\|$ is at most $H_M(x, x^R x^R)$, $\|x\| \leq \lfloor (l^2/2)(1/l) \rfloor = \lfloor l/2 \rfloor$. ■

The equation $weight(x) = l/2$ is a necessary condition for $\|x\| = l/2$. In this case, since $\|x\|$ is even, both the template length l and $weight(x)$ must be even.

Lemma 2.6

For any template x of length l and $\|x\| = d$, $weight(x) \geq \frac{l - \sqrt{l^2 - 2dl}}{2}$.

Proof

Let $w = weight(x)$. Note that $weight(x \wedge x^{-i})$ gives the number of matched bits between x and its cyclic shift x^{-i} . Therefore, the Hamming distance between x and x^{-i} is $2(w - weight(x \wedge x^{-i}))$. Since the total number of matches of 1 between x and all the shifts of x^R is w^2 , the mass for the given w satisfies $d \leq 2(w - \lceil w^2/l \rceil)$. By solving this, the lemma follows. ■

A simpler bound $weight(x) \geq \|x\|/2 + 2$ is easier to obtain as follows. For any template x such that $weight(x) \leq 2$, we have $\|x\| = H_M(x, x^R x^R) = 0$. The maximum number of mismatches is therefore $2(weight(x) - 2) \geq \|x\|$.

The next lemma counts the number of candidate templates based on the Hamming distance $H(x, x^R)$. From Lemma 2.1, we can write $H(x, x^R) = 2e$ and $weight(x) = e + p$, where $p \geq 0$ (Fig. 2).

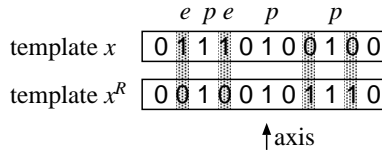


Fig. 2 $e = 2$ Asymmetric Bits and $p = 3$ Symmetric Bits in a Template x of Length 11. Note that $H(x, x^R) = 2e$ and $weight(x) = e + p$.

Lemma 2.7

Let $W_l(e, p)$ be the number of template candidates x of length l such that $H(x, x^R) = 2e$ and $weight(x) = e + p$.

$$W_l(e, p) = \begin{cases} 0 & p : \text{odd and } l : \text{even} \\ \binom{\lfloor l/2 \rfloor}{\lceil (p-1)/2 \rceil} \binom{\lfloor l/2 \rfloor - \lceil (p-1)/2 \rceil}{e} 2^{e-1} & \text{otherwise} \end{cases}$$

Proof

Total p bit 1's are located at symmetric positions. When l is even, the axis of symmetry is between digits, and therefore p must be even. When l is odd, $\lfloor l/2 \rfloor$ th digit becomes the axis.

There are $\binom{\lfloor l/2 \rfloor}{\lceil (p-1)/2 \rceil}$ choices for p bits. Once they are positioned, e bits are chosen among $\lfloor l/2 \rfloor - \lceil (p-1)/2 \rceil$ indices, for each of which 2^e arrangements of 0 and 1 exist. However, both x and x^R are counted in the calculation. Therefore, half of this count is the right number. ■

In finding templates of mass d or more, we only need to consider candidates satisfying $H(x, x^R) \geq d$. Templates such that $H(x, x^R) = 2e$ for a given e ($\geq d/2$) are generated as in Lemma 2.7, with a stepwise increment of e . This process can be systematically applied in the decreasing order of $weight(x)$.

Theorem 2.1

The number of words to be tested with Definition 2.2 for finding templates of

$\|x\| \geq d$ and $weight(x) = w$ is upper-bounded by:
$$\sum_{e=d/2}^{\lfloor l/4 \rfloor} \sum_{p=w-e}^{\lfloor l/2 \rfloor - e} W_l(e, p).$$

Proof

From Lemma 2.5, we only need to check templates which satisfy $d \leq H(x, x^R) = 2e \leq \lfloor l/2 \rfloor$. From Lemma 2.4, $\lfloor l/2 \rfloor \geq e + p$. The theorem follows. ■

Example 2.2

Let us consider searching templates of length 23. By Lemmas 2.1 and 2.5, the possible largest mass is 10. A template of this mass can only be found in the range $weight(x) = 8 \sim 11$ by Lemmas 2.4 and 2.6. Note that the $weight$ determines the GC content of designed sequences. From Lemma 2.5, we need not test templates such that $H(x, x^R) > 10$. For each value of $weight(x)$, therefore, we test $W_{23}(5, weight(x) - 5)$ words (Lemma 2.7). The total number of tested words

here, e.g. $W_{23}(5, 11 - 5) = 147840$, is far less than $O(2^l)$ and is within a feasible amount for an exhaustive search.

The search result of the largest masses for $l = 6 \sim 32$ is shown in Table 1. Interestingly, the mass is not monotonically non-decreasing, as seen for templates of $l = 23$ and $l = 24$.

Table 1 Largest Mass $\|x\|$ (Definition 2.2) for Templates of Length $l = 6 \sim 32$

$\ x\ $	l	$\ x\ $	l
2	6 ~ 10	8	19 ~ 22, 24
4	11 ~ 15	10	23, 25 ~ 27
6	16 ~ 18	12	28 ~ 32

In order to compute the original mass in Definition 2.1, each template found with a mass in Definition 2.2 is cyclic shifted to check $H_M(x, \langle xx^R \rangle)$ and $H_M(x, \langle x^R x \rangle)$. These masses are shown in Table 2. The odd distance results from the additionally checked terms.

Table 2 Largest Mass $\|x\|$ (Definition 2.1) for Templates of Length $l = 6 \sim 32$

$\ x\ $	l	$\ x\ $	l	$\ x\ $	l
2	6 ~ 10	7	19	10	26, 27
4	11 ~ 15	8	20 ~ 22, 24	11	28, 29
6	16 ~ 18	9	23, 25	12	30 ~ 32

In these tables, we see certain word lengths are more appropriate for biotechnology. Suppose that we impose more than $l/3$ mismatches for each unexpected hybridization. The possible lengths are then $l = 11, 16, 17, 19, 20, 21, 22, 23, 25, 26, 27, 28, 29, 30, 31, 32$. In the appendix, we list all the templates achieving the largest masses for up to $l = 30$.

§3 Melting Temperature

From a template x such that $\|x\| = d$, words over 4 alphabet letters are derived as a product of x and an error-correcting code E of minimum distance $\geq d$. Although the GC content is uniform for all designed sequences $S = x \cdot E$, their melting temperatures may actually differ.

To examine the uniformity of hybridization more accurately, let us consider the nearest-neighbor (NN) thermodynamics. When sequences are synthetic and melt in a single cooperative transition, the melting temperature T_M is calculated simply as $\Delta H^\circ / \Delta S^\circ$.¹⁵⁾ Since ΔH° (enthalpy) and ΔS° (entropy) depend mainly on the arrangement of [GC] and [AT] (Fig. 3), we can conclude that sequences designed by the template method share close melting temperatures in the NN model as well.

Three groups in Fig. 3 correspond to the three patterns of 1 and 0 in a template: a run of 1's, alternating 0's and 1's, and a run of 0's. From the

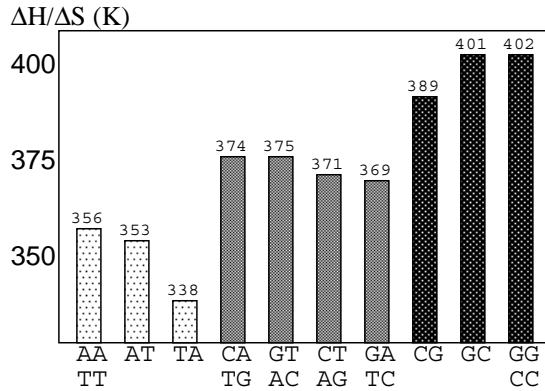


Fig. 3 $\Delta H^\circ/\Delta S^\circ$ Parameters for DNA/DNA Dimers (From Allawi *et al.*²⁾)

parameters of Allawi *et al.*,²⁾ the most T_M -estranging base composition in each pattern is calculated as in Table 3.

pattern	difference	example (high > low)
all 1's	around 11	AAAA...AA and TATA...TA
alternating	around 6	ACAC...AC and TCTC...TC
all 0's	around 2	GCGC...GC and CCCC...CC

Although previously reported NN parameters are somewhat inconsistent among laboratories, the qualitative trend of base stability is not: $GC/CG = CG/GC > GG/CC > CA/GT = GT/CA = GA/CT = CT/GA > AA/TT > AT/TA > TA/AT$.¹⁵⁾ From this trend together with the data in Table 3, we can extrapolate that templates with shorter runs of 1's can deliver sequences of less T_M difference.

Example 3.1

Let us consider the template 01111 01011 00110 01010 000 of length 23, where bit 1 means [AT], and 0 means [GC]. Using the above qualitative trend, let us choose high T_M sequence CAAAA CACAA GCAAG CACAC GCG and low T_M sequence GTATA GTCTA GCTAG CTCTC CCC. According to the NN-based computer program *Melting*,⁸⁾ the T_M difference between them is less than 7 °C when the salt concentration = 50 mM and the single-strand concentration = 250 pM. The actual difference will be even less, because we do not consider the optimal factor code in this example.

§4 Application

4.1 DNA Computation

The number of sequences we can design depends on the factor code E . Its maximum size is extensively investigated in code theory. For example, the Golay

code of length 23 provides $2^{12} = 4096$ words at minimum distance 7. This is a perfect code, i.e. the optimality of the code size is proven. For another example, BCH code of length 15 provides at least $2^7 = 128$ words at minimum distance 5. These sizes suffice for DNA computation currently performed in laboratories.

Restriction enzymes are standard tools in DNA computation. Usually, restriction sites must appear only at specified positions in the designed sequences. In the template method, their occurrence can be easily controlled, for all sequences share the same template. To control the bit patterns in the product code, we only need to set its information bits at the corresponding position. A problem arises only when no templates contain an appropriate substring for restriction enzymes at specified positions, or when the number of required sequences is more than we can design. In these cases, we need to lower the mass value until we obtain a matching template.

Example 4.1

Let us consider the template 01111 01011 **00110 01010** 000 of length 23. Suppose we use enzyme *BamHI*, whose restriction site is GGATCC. The restriction site will appear at 11–16th region in the template (in bold face) only, even when sequences and their complements are concatenated.

4.2 Universal DNA Arrays

For array-based hybridization assay, several groups proposed the advantage of universal arrays. In these arrays, artificially designed oligonucleotides are planted on a glass slide, where each probe strongly hybridizes only to its complementary sequence, but not to any other sequence. Since array probes are never concatenated in hybridization assays, the mass in Definition 2.2 suffices for sequence design. Therefore, any cyclic shift of any string in the appendix can be used as a template. Also note that the number of available templates exceeds those listed in the appendix, because Table 4 lists only templates achieving the largest mass in Definition 2.1.

Previous approaches for array design use De Bruijn sequences.^{4,12)} This is a cyclic sequence where each n -mer, for a given n , appears only once. The construction of De Bruijn sequences was a topic of mathematical interest and has been investigated in detail. For selecting probes from a De Bruijn sequence, Morris *et al.* proposed a greedy algorithm that picks probes piecewise until no more probes can be added. Since its performance was unknown, Ben-Dor *et al.* gave an optimal construction method for the problem.

Unlike templates, the De Bruijn method does not constrain the order and composition of DNA bases. It can design, therefore, more sequences than the template method. The advantage of the template method, aside from the consideration of overlap regions of probes, is that the number of mismatches is guaranteed. In the De Bruijn method, even if no selected probes share any n -mer, they may share multiple subsequences shorter than n . This means, for example, that probes of length $l = 23$ and $n = 9$ may share $l - \lfloor l/(n-1) \rfloor = 21$ bases. Another advantage is the flexibility to include predetermined subsequences. Since

[AT] and [GC] positions are fixed, it is possible to include a restriction site for some probes, while precluding it from others. The control is easy by manipulating the information bits of the factor code. Thus, the template method can accommodate to a more complicated hybridization assay such as the intelligent chip by Sakakibara *et al.*¹⁴⁾

4.3 Code Theory

Encoding and decoding information using the code in Lemma 2.3 seems difficult because of its nonlinearity. We only note that its code size can sometimes be doubled as follows. The mass for $l = 14$ is 4, and 28 words are constructed from a single template (Lemma 2.3). One such template is $x = 11101\ 00100\ 0000$, where $\text{weight}(x) = 5$. Let x' be a template where all bits in x are 01-flipped. From x' , additional 28 words are constructed, each of which is at least $9 - 5 = 4$ bits distant from the first set. Thus, by adding $00 \cdots 0$ and $11 \cdots 1$, 58 words at minimum distance 4 are obtained. Since the currently known nonlinear 2 error-correcting code (minimum distance 5) of length 14 is at most 128 words,¹³⁾ our simple construction is interesting as a reversible cyclic code of fixed weight distribution. Similarly, 98 words at minimum distance 8 are obtained by using 11000 01101 10101 00000 0000 for length 24.

In summary, the template method has the following advantages:

- Given a template x such that $\|x\| = d$ and an error-correcting code E of minimum distance d , we can design $|E|$ sequences.
- All sequences share close melting temperatures.
- The occurrence of specific subsequences can be easily controlled.

§5 Conclusion

This paper introduced the template method for designing sequences in biotechnology. Using the templates in Table 4, we can design sequences of length $l = 11 \sim 30$ such that any sequence will have more than $l/3$ mismatches with other sequences and with their overlap regions, including their complements. Moreover, their melting temperatures are close to uniform.

The number of mismatches $l/3$ was empirically determined by the outcome of our search for relatively short (~ 32) strings. While theoretical analysis remains to be investigated, our practical applications encounter no problems, because sequences of length 30 are sufficiently long for the use in computation *in vitro* and in hybridization assays.

The drawback of our method is the limited number of words that can be designed. The maximum number of words from a single template is limited to at most thousands. Therefore, it is natural to attempt using multiple templates. To use multiple templates simultaneously, however, each template must mismatch at more than d positions from overlap regions of the other templates. In code theory, this property is called *comma free* (or synchronizing), and our problem corresponds to finding an error-correcting reversible comma-free code such that $x \neq x^R$. For the case without reversibility, Levy partly solved the problem by considering a coset of the original cyclic code.⁹⁾ However, the extra polynomial

to obtain a good coset was discovered by “a nonanalytic procedure which may best be described as a series of educated guesses,” and its theoretical validation remains to be investigated.

From the thermodynamic viewpoint, our analysis lacks the consideration of the secondary structure, especially hairpin loops. It is not difficult to filter out hairpin-forming templates using dynamic programming. We skipped this selection, however, because the criterion for hairpin loops depends on experimental conditions. More important is the distribution of mismatches between designed sequences.

Since duplex formation starts with a transient nucleation complex and proceeds to the zippering process, it is better that the mismatches between sequences are scattered as much as possible. The simplest heuristic is to shuffle information- and check-bits before applying the code to a template. For algebraic codes such as cyclic codes, however, this restriction may be analyzed by considering the maximum length of consecutive matches (or mismatches).

In summary, the following problems are still open for characterizing templates.

- Theoretical analysis on the mass of templates
- Systematic construction of an error-correcting, reversible, comma-free code
- Analysis of the distribution of correctable errors

Finally, to justify our design strategy, verification of sequences in laboratory experiments, the most practical criterion in biotechnology, must be undertaken.

Acknowledgements

We thank the anonymous referee for the very careful readings of the manuscript and helpful suggestions. We also thank Dr. Osamu Gotoh for helpful discussions.

References

- 1) Adleman, L., “Molecular Computation of Solutions to Combinatorial Problems,” *Science*, 266, 5187, pp. 1021–1024, 1994.
- 2) Allawi, H. T. and SantaLucia Jr., J., “Nearest-neighbor Thermodynamics of Internal AC Mismatches in DNA: Sequence Dependence and pH Effects,” *Biochemistry*, 37, 26, pp. 9435–9444, 1998.
- 3) Ben-Dor, A., Karp, R., Schwikowski, B. and Yakhini, Z., “Universal DNA Tag Systems: A Combinatorial Design Scheme,” in *Proc. of the 4th Annual International Conference on Computational Molecular Biology (RECOMB2000)*, ACM Press, pp. 65–75, 2000.
- 4) Brenner, S., “Methods for Sorting Polynucleotides using Oligonucleotide Tags,” US Patent 5 604 097, 1997.
- 5) Deaton, R., Garzon, M. Rose, J. A., Franceschetti, D. R., Murphy, R. C. and Stevens Jr., S. E., “Reliability and Efficiency of a DNA Based Computation,” *Physical Review Letters*, 80, pp. 417–420, 1998.

- 6) Frutos, A. G., Liu, Q., Thiel, A. J., Sanner, A. M. W., Condon, A. E., Smith, L. M. and Corn, R. M., "Demonstration of a Word Design Strategy for DNA Computing on Surfaces," *Nucleic Acids Research*, 25, 23, pp. 4748–4757, 1997.
- 7) Garzon, M., Neathery, P., Deaton, R., Murphy, R. C., Franceschetti, D. R. and Stevens Jr., S. E., "A New Metric for DNA Computing," in *Proc. of the 2nd Annual Genetic Programming Conference*, Morgan Kaufmann, pp. 472–478, 1997.
- 8) Le Novere, N., "Melting, Computing the Melting Temperature of Nucleic Acid Duplex," *Bioinformatics*, 17, 12, pp. 1226–1227, 2001.
- 9) Levy, J. E., "Self-synchronizing Codes Derived From Binary Cyclic Codes," *IEEE Transactions on Information Theory*, IT-12, 3, pp. 286–290, 1966.
- 10) MacWilliams, E. J. and Sloane, N. J. A., "The Theory of Error-Correcting Codes," North-Holland, 1977.
- 11) Massey, J. L., "Reversible Codes," *Information and Control*, 7, pp. 369–380, 1964.
- 12) Morris, M. S., Shoemaker, D. D., Davis, R. W. and Mittmann, M. P., "Methods and Compositions for Selecting Tag Nucleic Acids and Probe Arrays," European Patent Application 97302313, 1997.
- 13) Nordstrom A. W. and Robinson, J. P., "An Optimum Nonlinear Codes," *Information and Control*, 12, pp. 613–616, 1967.
- 14) Sakakibara, Y. and Suyama, A., "Intelligent DNA Chips: Logical Operation of Gene Expression Profiles on DNA Computers," in *Proc. of the 11th Workshop on Genome Informatics*, Universal Academy Press, pp. 33–42, 2000.
- 15) SantaLucia Jr., J., "A Unified View of Polymer, Dumbbell, and Oligonucleotide DNA Nearest-neighbor Thermodynamics," in *Proc. of Natl. Acad. Sci. USA*, 95, pp. 1460–1465, 1998.

Appendix

Table 4 List of Templates of the Largest Mass in Definition 2.1 (word size $l = 11 \sim 27$). Reverse strings and 01-flipped strings are omitted. Note that their cyclic shifts (but not all shifts) may achieve the same mass.

size ($ x $)			
11 (4)	01110100100		01001111010101000
12 (4)	000111011010		01010000010011011
	001011100110		01100011110100000
	001111010100		01110001001101010
	010011011100		01110101100101000
	010111100010		10000011101010010
	011010100110		10011000010111100
	101001100000		10110001110010000
	101100001000		10110010111000100
	111001011000		10111001100010100
13 (4)	0000101100010		11000111011010000
	0000111011010		11010100110100000
	0001011001110		11101010001100100
22	0001011100110		11110010001100001
words	0001011100110	18 (6)	(209 words)
	0001110110010	19 (7)	1010111100100110000
	0010010011100	20 (8)	1000010110011001011
	0010100101110		11010011101110001000
	0010100111010		11011101000100111000
	0010110010110	21 (8)	000101101001111001100
	0011110101000		001001011011100010110
	0100010111000		010101000001110011011
	0110010100000		01010111000110110000
	0110011110000		011010001010011101100
	0110101001100		011110100000100110110
	1000110110100		100110110101110000010
	1000111010000		101000001100010011110
	1001011100000		101011110011011000000
	1010010011000		111100110000011010100
	1010110010000	22 (8)	(409 words)
	1010110110000	23 (9)	01111010110011001010000
	1010111001000	24 (8)	(10760 words)
	1101100101000	25 (9)	0000100011011010011101010
14 (4)	(79 words)		0000101011000110110100110
15 (4)	(180 words)		0000110010101100011110010
16 (6)	0001100011110100	20	00001000101101001011100110
	0010011100011010	words	0001000101100101011001000
	0011010011101000		0010000110110001111010100
	0101000010011011		00100111000011011010100
	0101101110001000		0010100110001101011110000
	1000001110110100		0011110101100110010100000
	1001111000001001		0101000001100110001111010
	1100101101010000		0101001101001110110001000
17 (6)	00001000100110111		0101110011010010100110000
	00001011100101100		0110011100010100001011010
	00010010101100110		0110100011000110100000111
26	00010101011011000		0111100110010000110101000
words	00011000111110100		1000001010001100111010110
	00011101101001000		1011001110010101011000000
	00100101011111000		1101010011100110100010000
	00100111000101100		1110010100110011010100000
	01000011110110010		1110011001000001010110100
	01000110011110000	26 (10)	(330 words)
	01001011000101110	27 (10)	(2272 words)
	01001011101100010		



Masanori Arita, Ph.D.: He is a researcher at Computational Biology Research Center (CBRC) in National Institute of Advanced Industrial Science and Technology (AIST). He received his M.S. (1996) and Ph.D. (1999) from Department of Information Science, The University of Tokyo. He also works as a fellow of Precursory Research for Embryonic Science and Technology (PRESTO) in Japan Science and Technology Corporation (JST), and a project associate professor at Institute for Advanced Biosciences (IAB) in Keio University. Main research interests are bioinformatics and DNA computing.



Satoshi Kobayashi: He received the B.E., M.E. and D.E degrees from the University of Tokyo in 1988, 1990 and 1993, respectively. He has been an associate professor of Department of Computer Science, University of Electro-Communications since 2000. His research interests include inductive inference of formal languages, bioinformatics, and the theory of DNA computing. He is a member of ACM, Japanese Society for Artificial Intelligence (JSAI), Information Processing Society of Japan (IPSJ), and Institute of Electronics, Information and Communication Engineers, Japan (IEICE).