This is to certify that the

thesis entitled

PERMUTED SUBSTRING MATCHING, WITH
APPLICATIONS TO COMPUTATIONAL BIOLOGY

presented by

Houman Alborzi

has been accepted towards fulfillment
of the requirements for

Master of Science degree in Computer Science

Eric Torng

Major professor

Date 8-20-1997

Permuted Substring Matching, with Applications to Computational Biology

By

Houman Alborzi

A Thesis

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Master of Science

Department of Computer Science

1997

Abstract

Permuted Substring Matching, with Applications to Computational Biology

By

Houman Alborzi

We propose a new inexpensive technique to find the DNA sequence of genes. The technique is based upon finding the location of a fragment of a genome using the restriction map of the genome and the restriction pattern of the fragment. As there may be more than one location on the genome matching the restriction pattern of the fragment, we investigate the effectiveness of our technique based on how precisely it can determine the location of the fragment. We show that the size of the restriction pattern of the fragment is the primary parameter that affects the effectiveness of our technique. We have also designed and implemented fast algorithms for locating the fragment.

Dedicated to the advancement of frontiers.

ACKNOWLEDGMENTS

I would like to thank Dr. Eric Torng, my academic advisor who was always trying to put order in my chaotic work. Without his patience and support, completing this thesis would not have been possible.

Dr. Sakti Pramanik and Dr. James Cole were always eager to discuss my work. I am grateful for their help and time allotted to this project.

My technical thanks to the staff of the computer science department computing services who provided the support and maintenance needed for this project.

Finally, I would like to thank the faculty, staff, and students of the computer science department with whom I worked. They were always encouraging.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

In this thesis we investigate the problem of locating a fragment of a previously sequenced genome on the genome. More specifically, we try to develop a new procedure to find the genetic sequence of genes. In this procedure, a genome that has been previously sequenced is broken into fragments, perhaps using restriction enzymes. Then, after biological experiments have determined that some of the fragments are biologically interesting, we try to find the DNA sequence of those fragments. Usually finding the DNA sequence is expensive. However, since the particular DNA sequence is part of our previously sequenced genome, the problem of finding the DNA sequence of the gene reduces to finding the location of the gene on the genome.

1.1 Genome, Genes, and DNA

Genetics goes back to the pioneering work of Gregor Mendel in 1865 [LW95]. Mendel hypothesized that traits are affected by discrete factors, which an offspring inherits from its parents. Today, we call those factors *genes*. In 1952, Alfred Hershey and Martha Chase showed that genes are encoded in DNA molecules which exist in any living organism's cells. Later, in 1953, Watson and Crick presented a model of

the structure of a DNA molecule. The DNA molecules are linear polymers of four basic chemical structures called nucleotides named adenine, thymine, cytosine, and guanine, and abbreviated as A, T, C, and G.

Each gene is usually associated with one or more DNA subsequences. These subsequences will be translated into protein sequences, which are considered to be the main chemicals affecting the traits of an organism. The whole set of genes of an organism constitutes its *genome*. In this manuscript we use the word genome to refer to a single sequence comprising the sequences of all DNA molecules of an organism.

The DNA nucleotides are joined by a sugar-phosphate backbone to form a DNA molecule. While some bacteria have single stranded DNA molecules, the DNA of humans and most other organisms consists of two strands making a double helix in which the bases (nucleotides) pair up (Figure 1.1).

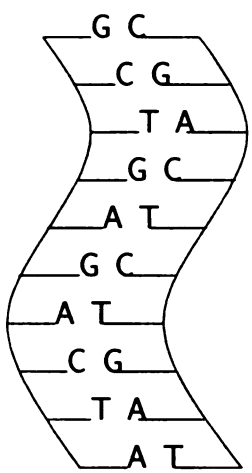


Figure 1.1: A schematic of an unfolded DNA double helix.

This pairing of bases is not arbitrary. A always pairs with T and C always pairs with G. Hence, the sequence of either strand can be determined from the sequence

of the other one. The length of a DNA molecule is measured in units of base pairs (bp), which is the number of bases in a single strand of the DNA molecule. Usually, we represent a strand of DNA by a single string over the alphabet of $\{A, T, C, G\}$.

DNA molecules are usually *linear*; that is they have two endpoints. However, in some organisms the DNA is *circular*, so there is no starting or ending point of the DNA.

1.2 Restriction Enzymes

There are enzymes that can cleave a DNA molecule into several fragments. These enzymes are called *restriction enzymes*, and the process of cleaving the DNA molecule into fragments is called *digestion*. Each restriction enzymes has a *recognition sequence*, which is a sequence of nucleotides where the enzyme cuts the DNA molecule. For example, the *EcoRI* enzyme cuts a DNA molecule at all occurrences of the sequence GAATTC. The *cut site* of a restriction enzyme determines exactly where in the recognition sequence the DNA molecule will be cut. For the *EcoRI* enzyme the cut site is 1, which means that the DNA molecule will be cut after the first base of the recognition sequence. For example, if the DNA sequence is GCTGAGAATTCGCAAGGAATTCGCAG, then the *EcoRI* enzyme will cleave it into three fragments, with the sequences of: GCTGAG, AATTCGCAAG, AATTCGCAG (Figure 1.2).

The *restriction sites* of a DNA molecule for a given restriction enzyme is the set of locations of all the enzyme's cut sites on the molecule. The ordered list of fragment lengths is the *restriction map* of the DNA molecule for the given enzyme. For instance,

in our previous example, the restriction map, will be $(6, 10, 9)$. A *restriction pattern* is the multi-set of fragment lengths of a restriction map. That is, the order of fragments in the restriction map is forgotten. For example the restriction pattern of our example is the set $\{6, 9, 10\}$.

We can obtain the restriction map of a DNA molecule using the gel electrophoresis experiment. This procedure is much simpler and cheaper than existing methods for obtaining the corresponding restriction map. In the gel electrophoresis experiment, the restriction fragments are placed in an electric field over a gel coated medium. The fragments then are separated based on their length. There is a logarithmic relationship between the size of DNA fragment and the distance it migrates on a gel. By measuring the distance each fragment has migrated, the length of the fragment can be found.

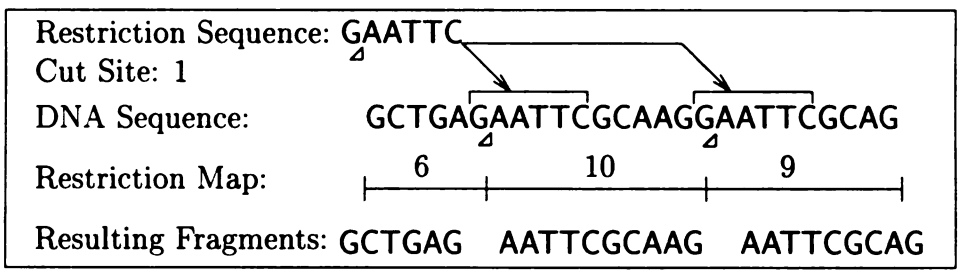


Figure 1.2: An example of a digestion process.

There is a measurement error associated with this experiment; that is, a fragment of length l will be read with some error ϵ which results in a measured length of $l(1+\epsilon)$. Usually, ϵ is a random variable with a normal distribution. However, we can safely assume that it is uniformly distributed over a range of $(-\delta, \delta)$, where δ is a parameter of the precision of the devices used in the experiment. Wherever needed, we assume

that $\delta = 0.05$. For example, considering this measurement error, a restriction pattern of $\{100, 105, 300, 400\}$ could be read as $\{98, 100, 297, 415\}$.

The process of digesting a DNA molecule by an enzyme can be partial or complete. In a *complete digestion*, a DNA molecule is broken at all of its recognition sequences. In a *partial digestion*, not all of the recognition sequences are broken. For example the result of a partial digestion in Figure 1.2 could be the GCTGAG and AATTCGCAAGAATTCGCAG fragments. Depending on the duration of the digestion process and the amount of restriction enzyme used, we can have various degrees of digestion.

The next section formally defines the terms described before. It can be skipped in a first reading.

1.3 Formal Definitions

Most definitions in this section are based on those in [KCTP96]. Nevertheless, we use more definitions to facilitate the communication.

Definition 1.1. *The restriction map $RM_e(g)$, for the DNA sequence g and the restriction enzyme e, is the ordered tuple $(s_1, s_2, ..., s_n)$ such that the first restriction site of e on g is at s_1, the second one at $s_1 + s_2$, and the ith one at $\sum_{i=1}^{i} s_i$. If the DNA sequence g is a circular molecule, then we arbitrarily choose s_1 as the length of one of the restriction fragments and assume that the cut site of its beginning is at index 0 of g.*

Definition 1.2. *For the multi-set RP and the ordered tuple RM, we define $RP \sim$*

RM if there exists a one to one mapping between their elements.

Definition 1.3. *The restriction pattern $RP_e(g)$ for the DNA sequence g and the restriction enzyme e is the multi-set $\{s_1, s_2, \ldots, s_n\}$ such that $RP_e(g) \sim RM_e(g)$.*

Definition 1.4. *We define $x \leq^\delta y$ if and only if $x \leq y(1+\delta)$. In this case we say x lower-matches y with error δ. Also, $x \geq^\delta y$ if and only if $x \geq y(1-\delta)$. In this case we say x upper-matches y with error δ. Finally, $x =^\delta y$ if and only if $x \leq^\delta y$ and $x \geq^\delta y$; that is, $y(1-\delta) \leq x \leq y(1+\delta)$. In this case we say x matches y with error δ.*

Definition 1.5. *We say the restriction map RM' matches restriction map RM with error δ, denoted $RM' =^\delta RM$, if and only if $\forall i\ RM'_i =^\delta RM_i$.*

Definition 1.6. *We say the restriction pattern RP matches restriction map RM with error δ if and only if, there exists some permutation of RP, denoted by RM', such that $RM' =^\delta RM$. We denote this by $RP \sim^\delta RM$.*

1.4 Problem Statement

There is a huge amount of ongoing research to obtain the complete genome of various organisms where the results will be stored in gene banks. Some micro-biologists believe that the functionality of some genes is based on their DNA sequences only; hence, by removing or adding a DNA sequence from or to an organism, we can modify the functionality of that organism. Thus, there is a great deal of research focused on finding the DNA sequence of genes responsible for specific traits.

Here, we provide a new procedure for finding the sequence of a gene without trying to sequence it directly. The basic idea is that if we know that a DNA fragment contains genes, and it is already part of an already sequenced genome, we can use its restriction pattern data, which is cheaply available, to find its location in the original genome, and hence learn its underlying DNA sequence.

The problem is based on the following procedure:

1. Choose a completely sequenced genome that has some interesting characteristics.

2. Break down the genome using a partial digestion experiment into fragments, denoted as *isolates.*

3. Perform biological tests on the isolates in order to identify some interesting isolates.

4. Completely digest an interesting isolate, and use gel electrophoresis to obtain its restriction pattern.

5. Use the restriction pattern of the isolate to find its location on the genome in order to obtain information about its underlying DNA sequence.

The last item in the list above is the main goal of this thesis; that is, as we wish to develop fast and practical algorithms to search for locations on the restriction map of the genome which match the restriction pattern of interesting isolates. However, one important question is "how useful is the proposed procedure?" In other words, given a restriction pattern of an isolate, how many locations on the genome could have the same restriction pattern. We denote this attribute of a restriction pattern as its *resolution* with respect to the genome.

1.5 Background on Searching

The problem of pattern matching has been extensively studied. The basic problem is to search a text for all the locations that match a given search pattern [Aoe94]. This match could be an exact or an approximate one. For an exact match algorithm, the result will be all the locations in the text which match the pattern exactly. In most approximate matching contexts, the text and pattern can have up to k mismatches.

The search pattern is usually a sequence of characters. However variants where the pattern is a regular expression, a sequence of characters, or even a context free grammar have been investigated [Aoe94, LW95]. In these pattern matching variants, the characters of the search pattern and the text are usually from a finite alphabet. Also, the characters have an equivalence relation, under which any two characters that are matched together will equivalently match or not match any third character. In [KMP77], Knuth, Morris and Pratt propose an algorithm that solves the problem in only one scan of the text by building a finite state automata from the search pattern. Their algorithm takes $O(n+m)$ time, where n is size of the text and m is length of the pattern. In [BM77], Boyer and Moore, provide a faster algorithm that basically skips some parts of the text while scanning it. It is another $O(n+m)$ algorithm. This work assumes the input text will be stored in random access memory, so the cost of skipping text is zero. By this assumption, the Boyer-Moore algorithm and its variants are considered to be the fastest algorithms available with an average running time of $O(\frac{\log m}{m}n)$ which is optimal [GB91]. Karp and Rabin [KR87] proposed a randomized algorithm to improve the brute force search. In their algorithm, a hash function is

used to compare the pattern and substrings of the text. Whenever these two match, the algorithm performs a character by character matching.

Our application differs from the other pattern matching problems as follows. Firstly, we are basically doing an unordered pattern matching, where the pattern is a multi-set of characters rather than a sequence of characters. Secondly, the characters of the pattern and text not only are from an infinite alphabet but also do not have an equivalence relation. Fortunately they do have a partial-order relation which we exploit in developing efficient algorithms.

1.6 Outline

In the following chapters, we first try to provide a probability distribution model for the restriction fragment lengths. After that, we provide discussion and empirical results on the effectiveness of the restriction pattern data. Finally we provide some restriction pattern search algorithms and some empirical results.

Chapter 2

A probability distribution model for restriction fragment lengths

2.1 Restriction Fragment Lengths have an Approximate Exponential Distribution

It is useful for the analysis and design of our algorithms to have a model of restriction site distribution along the genome. We consider a model in which the restriction sites are uniformly distributed across a genome; that is, we assume that a restriction site uniformly occurs at any location on the genome with probability p. Given this assumption, we can use basic probability theory to show that fragment lengths have a geometric distribution. That is, a fragment has length l with probability $p(l) = p(1-p)^l$. If we consider l to be continuous rather than discrete variable, we obtain,

$$p(l) = \lambda e^{-\lambda l} \tag{2.1}$$

where

$$\lambda = -\ln(1-p) \tag{2.2}$$

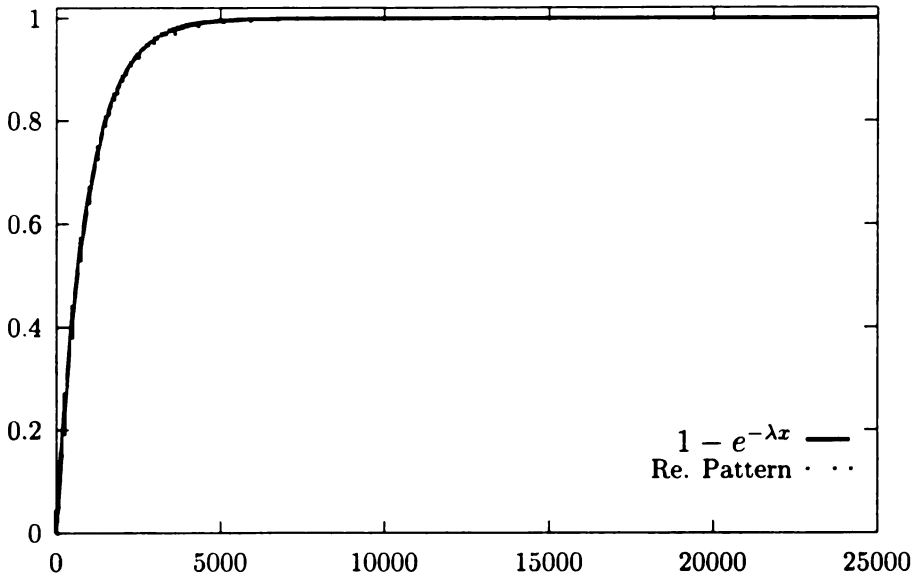
The same model for restriction sites has been proposed before in [Wat95]. We attempted to confirm the model by running the Chi-Square test [Dev95] for the Haemophilus influenzae Rd genome [FAW+95] with different recognition sites. The results of the Chi-Square test (Table 2.1) show that in most cases, the hypothesis that restriction sites are uniformly distributed is rejected with a high confidence level. However, we find that the exponential distribution for fragment lengths is quite acceptable even for enzymes which do not meet the uniform distribution assumption for cut sites. For example, we plot the cumulative distribution function (CDF) (Figure 2.1, solid line) of restriction fragment lengths for H.influenzae genome when cut by a recognition sequence of ATTAAT – which was rejected with confidence level of 1.000 in the Chi-Square test – and compare it with the exponential distribution function (solid line) where its λ parameter is derived using Equation 2.2.

To obtain p for a given genome and a restriction enzyme, we use the following equation:

$$p = \frac{\text{Number of restriction sites}}{\text{Length of genome in bp}} \tag{2.3}$$

As it can be seen from Figure 2.1, the assumption that the restriction pattern has an exponential distribution is indeed a very good approximation. Throughout this text, we use the parameter p as the probability that a restriction site occurs at any point in the genome and λ as the parameter for the exponential distribution function of fragment lengths. Note that p and λ are related as in Equation 2.2. For small values of p we have

$$\lambda \approx p \tag{2.4}$$



Genome	Haemophilus influenzae Rd
Recognition sequence	ATTAAT
Dotted line	The fragment length data
Solid line	Estimated exponential distribution

Figure 2.1: CDF of a restriction pattern and its approximation.

While the measured fragment lengths have an error of $\pm\delta$, they essentially preserve their exponential distribution as δ is very small. We do assume the same probability distribution function for both the exact restriction pattern data and those obtained from gel electrophoresis experiment.

As the restriction pattern data has a known distribution, we consider ways to use this knowledge in order to speed up our algorithms. Mainly we need to search and sort restriction pattern data. If the data has a uniform distribution, we can use interpolation sort with time complexity of $O(n)$ and interpolation search with time complexity of $O(\log\log n)$ [GB91]. In order to use these fast sort and search algorithms, we should build a key of restriction pattern data that has uniform distri-

bution. For any distribution, the function needed to make it a uniform distribution is its CDF function. In our case the CDF function is $1 - e^{-\lambda l}$, where l is the fragment length.

2.2 Normalized fragment lengths

Definition 2.1. *The normalized value of a restriction fragment length l, denoted by x is defined as*

$$x = 1 - e^{-\lambda l} \tag{2.5}$$

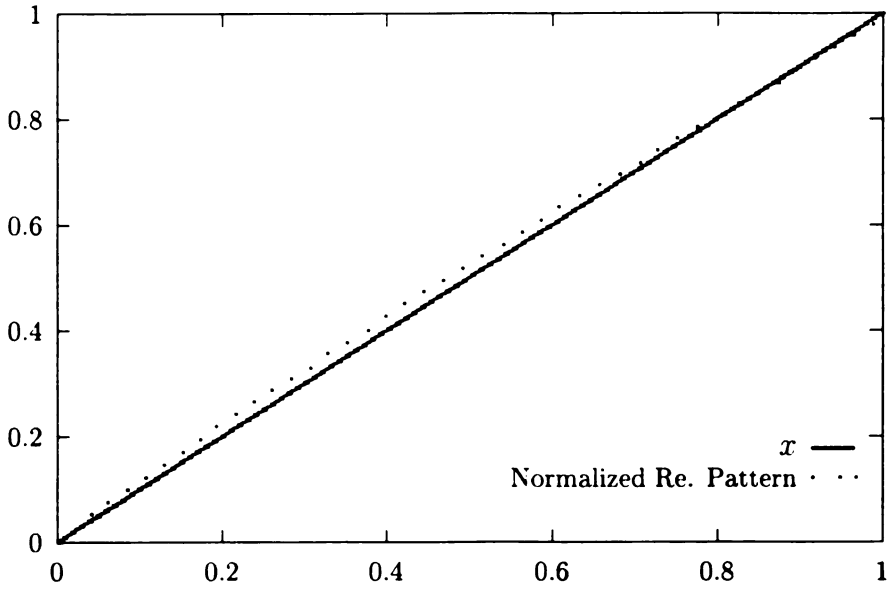
Also,

$$l = \frac{-\ln(1-x)}{\lambda} \tag{2.6}$$

Again, to justify our method, a graph of the normalized fragment lengths is shown in Figure 2.2. The genome and the restriction enzyme are the same as in Figure 2.1. While this plot shows more difference between the real data and the approximation, notice that in interpolation search and sort, the most important characteristic that affects the running time is the linearity of the CDF of the data. Here, for most of the graph, we have a good linear approximation. Furthermore, note that the enzyme we have chosen to plot has the most statistical difference with the approximated distribution function.

We usually use l, l', l_1, etc. to refer to a fragment length, and x, x', x_1, etc. to refer to a normalized fragment length.

It will be easier for us to do our analysis using the normalized fragment length data. Therefore, we state a relationship for matching with error δ two normalized



Genome	Haemophilus influenzae Rd
Recognition sequence	ATTAAT
Dotted line	The normalized fragment length data
Solid line	Uniform distribution

Figure 2.2: CDF of a normalized restriction pattern and its approximation.

fragment lengths. It is interesting to note that this relationship is independent of the λ parameter of the CDF.

Theorem 2.1. *For two fragment lengths l_1 and l_2, we have $l_1 \leq^{\delta} l_2$ if and only if $x_1 \leq 1-(1-x_2)^{1+\delta}$, and $l_1 \geq^{\delta} l_2$ if and only if $x_1 \geq 1-(1-x_2)^{1-\delta}$. Also, $l_1 =^{\delta} l_2$ if and only if $1-(1-x_2)^{1-\delta} \leq x_1 \leq 1-(1-x_2)^{1+\delta}$*

Proof. Proof follows directly from Definition 1.4 and Equation 2.6. □

Theorem 2.2. *The probability that two restriction fragments match with error δ is approximately $\frac{\delta}{2}$.*

Proof. Two distinct fragments, l_1 and l_2 will match if their normalized values x_1 and x_2 satisfy the relation $1-(1-x_2)^{1-\delta} \leq x_1 \leq 1-(1-x_2)^{1+\delta}$. As x_1 and x_2 are

uniformly distributed, the probability that two fragments match can be approximated as follows:

$$\begin{aligned}
\int_0^1 \int_{1-(1-x_2)^{1-\delta}}^{1-(1-x_2)^{1+\delta}} dx_1\, dx_2 &= \int_0^1 (1-x_2)^{1-\delta} - (1-x_2)^{1+\delta}\, dx \\
&= \frac{1}{2-\delta} - \frac{1}{2+\delta} \\
&= \frac{2\delta}{4-\delta^2} \\
&\approx \frac{\delta}{2}
\end{aligned}$$

□

Pattern	Conf. Level	Pattern	Conf. Level
AAATTT	0.9998	GAATTC	0.5023
AACGTT	0.9043	GACGTC	0.1665
AAGCTT	0.6948	GAGCTC	0.2867
AATATT	1.0000	GATATC	0.9973
ACATGT	0.2435	GCATGC	0.2677
ACCGGT	0.7415	GCCGGC	0.9488
ACGCGT	0.5107	GCGCGC	0.9654
ACTAGT	0.0805	GCTAGC	0.3993
AGATCT	0.9548	GGATCC	0.9363
AGCGCT	0.7770	GGCGCC	0.2100
AGGCCT	0.7601	GGGCCC	0.4751
AGTACT	0.2167	GGTACC	0.1388
ATATAT	0.9990	GTATAC	0.7571
ATCGAT	0.7109	GTCGAC	N/A
ATGCAT	0.8591	GTGCAC	0.5255
ATTAAT	1.0000	GTTAAC	0.7709
CAATTG	0.9975	TAATTA	0.9998
CACGTG	0.5277	TACGTA	0.9942
CAGCTG	0.9710	TAGCTA	0.8844
CATATG	0.9905	TATATA	0.9989
CCATGG	0.6005	TCATGA	0.9744
CCCGGG	0.2212	TCCGGA	0.8858
CCGCGG	0.9713	TCGCGA	0.3433
CCTAGG	0.9922	TCTAGA	0.6418
CGATCG	0.6466	TGATCA	0.7699
CGCGCG	0.9064	TGCGCA	0.4748
CGGCCG	0.8858	TGGCCA	0.9344
CGTACG	0.8777	TGTACA	0.5256
CTATAG	0.4512	TTATAA	0.9998
CTCGAG	0.8510	TTCGAA	0.9806
CTGCAG	0.9498	TTGCAA	0.7565
CTTAAG	0.6770	TTTAAA	0.9965

Hypothesis: Restriction sites are uniformly distributed over the H.influenzia genome.

Table 2.1: Chi-Square test confidence levels for distribution of restriction sites

Chapter 3

Effectiveness of the Procedure

As stated earlier, the restriction pattern obtained from a gel electrophoresis experiment is not precise. So, in practice, after searching a genome for the restriction pattern, we can end up with more than one matching location on the genome. As the number of matching locations or matches increases, the procedure becomes less effective. We try to investigate the general effectiveness of the procedure and answer the following questions in this chapter:

- How does δ affect the effectiveness of the procedure?
- Which restriction enzyme(s) should we choose for digesting the isolate?

We first need to define a metric for the effectiveness of the procedure. In the next section, we propose three different metrics to be used later in our discussion.

3.1 Metrics for Effectiveness

We first assume that the number of matches is a good indicator for the general effectiveness of the procedure. Based on this assumption, we define our first metric

as follow.

Definition 3.1. *We define the resolution of a restriction pattern RP over a genome g with restriction enzyme e, denoted* $r_e(RP, g, \delta)$*, as the number of subsequences* RM_s *of* $RM_e(g)$ *such that* $RP \sim^{\delta} RM_s$.

However, in practice, two different matched locations may overlap (Figure 3.1). We believe that the biologist can still use this procedure even if the search results in multiple matches with significant overlap.

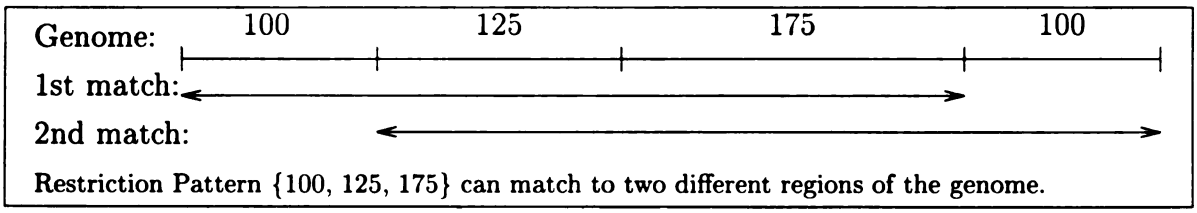


Figure 3.1: Possible overlapping matches to a restriction pattern.

We conjectured that isolates with resolution greater than one are actually overlapping with their false matches in most cases. We developed a second metric to study our conjecture where each possible isolate of the genome for a certain enzyme was put into one of three different categories:

1. The isolates which result only in a single match.
2. The isolates which result in multiple matches, but all false matches overlap with the original isolate.
3. The isolates which result in at least one false match which does not overlap the original isolate.

We refer to this metric as the *Category* metric.

The isolates in category 1 are the perfect ones for our case. Based on experiments we observed that the isolates in category 3 can be avoided easily as they are very rare. The main difficulty arises with the numerous isolates in category 2. This metric does not distinguish between isolates that generate two highly overlapped matches and isolates that generate two matches that barely overlap. In order to study this effect better, we developed a third metric(*Score*). Let l denote the isolate length and l' denote the portion of the isolate shared by all the matches. An isolate receives a score of $\frac{l'}{l}$. Note that all isolates in category 1 have a score of 1, and those in category 3 have a score of 0.

3.2 Resolution Metric

We experimentally measured the resolution of isolates for the *Mycoplasma genitalium* genome [FGW+95] using the TGATCA recognition site which cuts the genome into 500 fragments. We plotted them in Figure 3.2 for $\delta = 0.04$ and Figure 3.3 for $\delta = 0.10$. In both cases the resolution of isolates increases with the length of the isolate. However, for the smaller error value, the slope is much lower. Also, in both cases, we have high values of resolution for very short isolates.

Note that the resolution metric is very dependent on the error value, and we

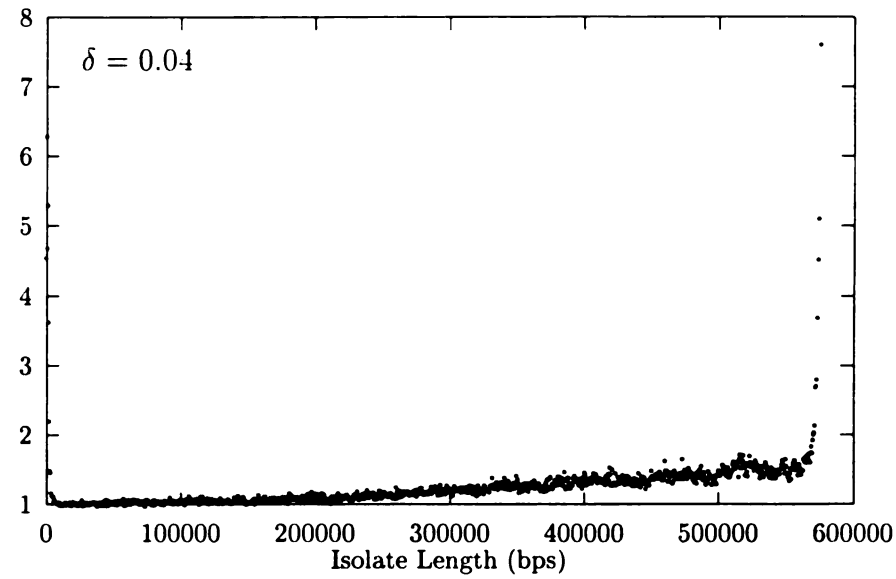


Figure 3.2: Average resolution of isolates, $\delta = 0.04$.

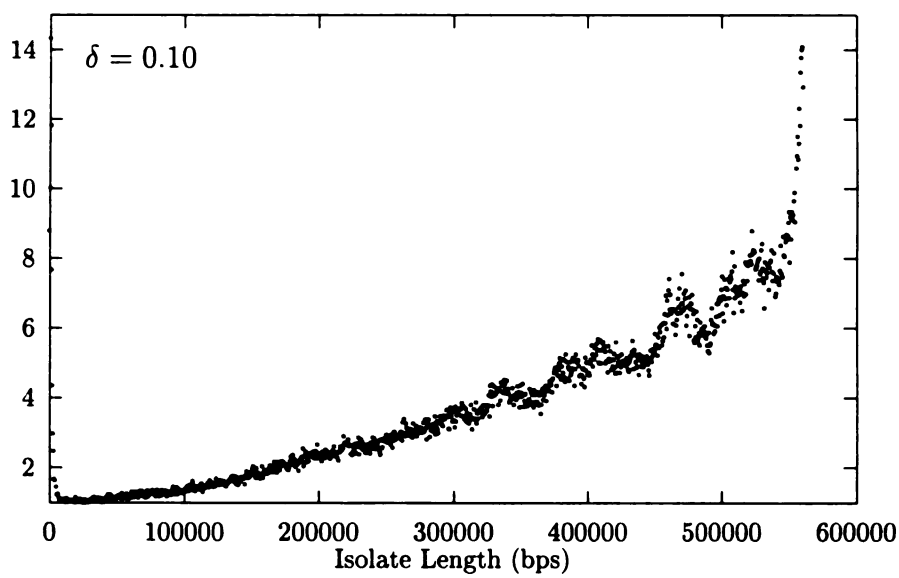


Figure 3.3: Average resolution of isolates, $\delta = 0.10$.

cannot infer much using it, when we don't know much about the error value.

3.3 Category Metric

Definition 3.2. *Let $P(N, k, i)$ denote the probability that an isolate with k restriction fragments obtained from a genome with N restriction fragments is in category i.*

First, we try to find the probability that an isolate is in category 3, i.e. $P(N, k, 3)$. As stated before in Section 2.2, the normalized fragment lengths have uniform distribution, and it is easier for analysis purposes to use the normalized lengths of fragments.

An isolate I is not in category 3 if all other non-overlapping isolates of the genome do not match with it. Computing the exact probability that two independent isolates match is rather difficult; therefore we use the following approximation.

We first derive a lower bound for $P(N, k, 3)$. The probability that two restriction fragments match is $\frac{\delta}{2}$, and the probability that two isolates match is at least $\left(\frac{\delta}{2}\right)^k$. There are $\frac{N}{k}$ disjoint isolates on the genome, and the probability that there is no other matching isolate will be less than:

$$\left(1 - \left(\frac{\delta}{2}\right)^k\right)^{\frac{N}{k}-1}$$

That is,

$$P(N, k, 3) \geq 1 - \left(1 - \left(\frac{\delta}{2}\right)^k\right)^{\frac{N}{k}-1}$$

We next derive an upper bound on $P(N, k, 3)$ by assuming that all permutations of the restriction pattern are likely to match and by considering all other $N - 2k + 1$

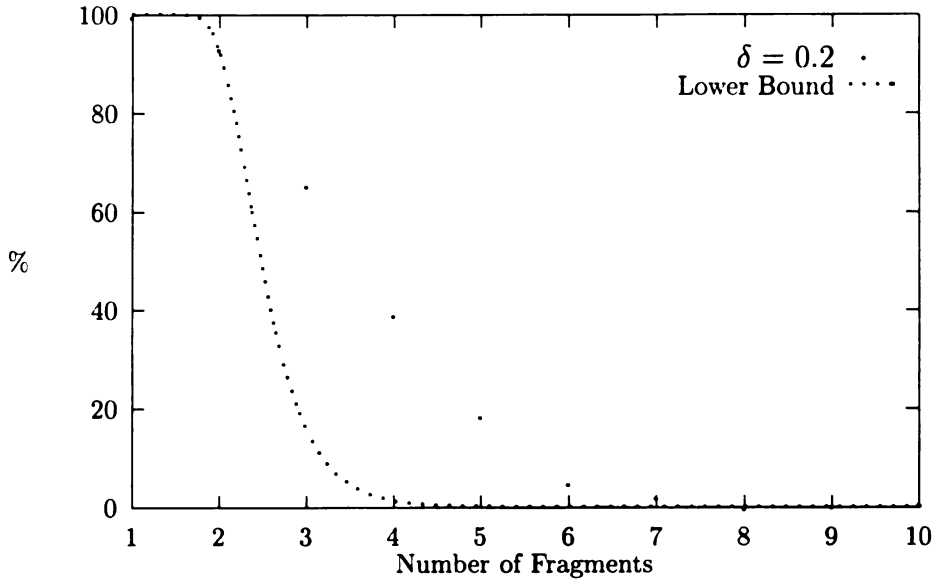


Figure 3.4: Percentage of isolates in category 3.

non-overlapping isolates of the genome.

$$P(N,k,3) \leq 1 - \left(1 - \left(\frac{\delta}{2}\right)^k\right)^{k!(N-2k+1)}$$

In Figure 3.4, the lower bound and the real percentage of isolates in category 3 is shown. The genome is *Mycoplasma genitalium* genome [FGW+95], and the recognition site is TGATCA which cuts the genome into 500 fragments. As it can be seen, if the isolate has more than 8 fragments, we don't have any isolate in category 3 even for a big error value of 20%. So, by carefully choosing the restriction enzyme, we can can assure that the isolates are not in category 3.

We now focus on finding the probability that an isolate is in category 2. We derive a lower bound for $P(N,k,2)$. First we observe that at least $\frac{\delta}{2}$ of the isolates are in category 2. The reason is that the first fragment of an isolate can match the very

first fragment of the genome after the isolate with that probability. To improve this lower bound for the probability, we compute the probability that two isolates match in all but one fragment. In other words, isolates that share all the fragments except one. Let's name these two fragments as x_1 and x_2. They could match together with a probability of $\frac{\delta}{2}$, hence resulting in the matching of the isolates. Or, they can match through another common fragment, say x_3, such that x_1 matches x_3 and x_3 matches x_2. The probability that this happens is:

$$\int_0^1 \int_{1-(1-x_2)^{1-\delta}}^{1-(1-x_2)^{1+\delta}} \int_{1-(1-x_3)^{1-\delta}}^{1-(1-x_3)^{1+\delta}} dx_1\, dx_3\, dx_2 \approx \frac{8}{27}\delta^2$$

There are $k-1$ such shared fragments between the two isolates', hence, we can find the following lower bound on the probability that an isolate is in category 2:

$$P(N,k,2) \geq 1 - \left(1 - \frac{\delta}{2}\right)\left(1 - \frac{8}{27}\delta^2\right)^{k-1}$$

We can improve this lower bound by considering a chain of shared fragments to match x_1 and x_2. However our current model is acceptable for small values of k.

In Figure 3.5 and Figure 3.6, the lower bound and the observed probability of an isolate being in category 2 are depicted. Note that as the number of fragments increases, it is more likely that an isolate is in category 2. On the other hand, the probability of an isolate being category 3 decreases rapidly with the number of fragments (Figure 3.4).

To maximize the percentage of category 1 isolates, we should choose a restriction

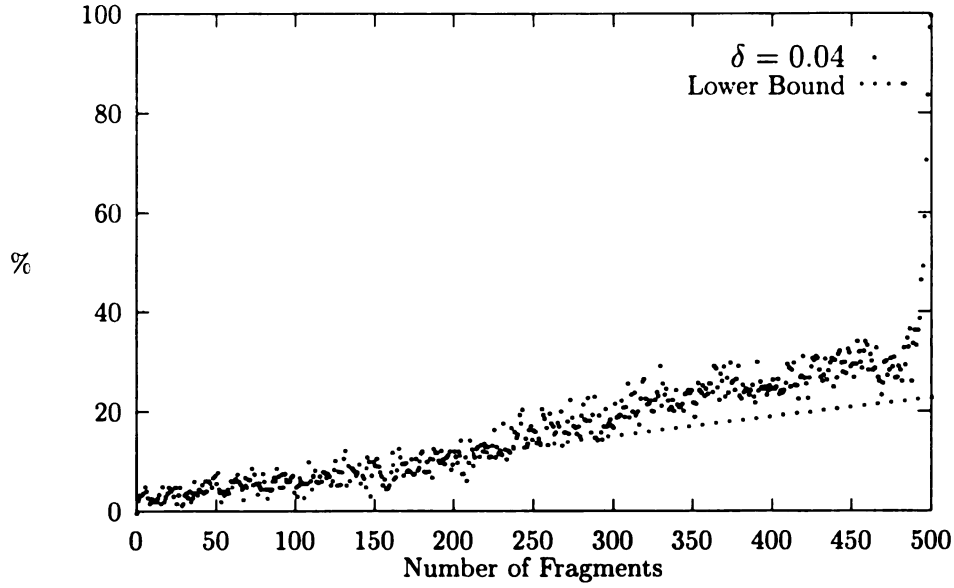


Figure 3.5: Percentage of isolates in category 2, $\delta = 0.04$.

enzyme which cuts the isolate into have very few fragments (approximately 10). Since the length of the isolate is usually known before doing the digestion, we can choose an appropriate restriction enzyme.

3.4 Score Metric

As stated before, the score metric measures the ratio of overlapped length of matches of an isolate to the length of the isolate. For a relatively high error of $\delta = 0.10$, computer experiments showed that, in average, most isolates achieve a score of 0.95. This indicates that, in most cases, we can find the location of the isolate with a 95% accuracy. If this approximate knowledge of location of the isolate on the genome is sufficient for the biologist, then this procedure can be used successfully. The maximum score was when the isolate has a small length, which is due to the

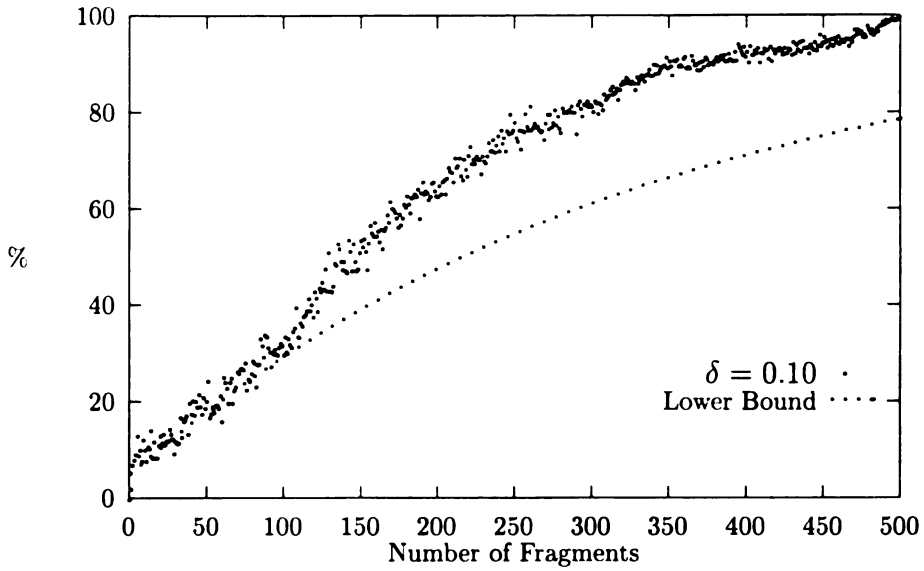


Figure 3.6: Percentage of isolates in category 2, $\delta = 0.10$.

smaller number of fragments it has. This was discussed in Section 3.3.

Figure 3.7 and Figure 3.8 show the score metric for isolates when the genome is *Mycoplasma genitalium* and the recognition site is TGATCA. Notice that with the lower error of δ we obtain a much better score. Based on the experiments we can conjecture that the score of a moderate length isolate is about $1 - \frac{\delta}{2}$.

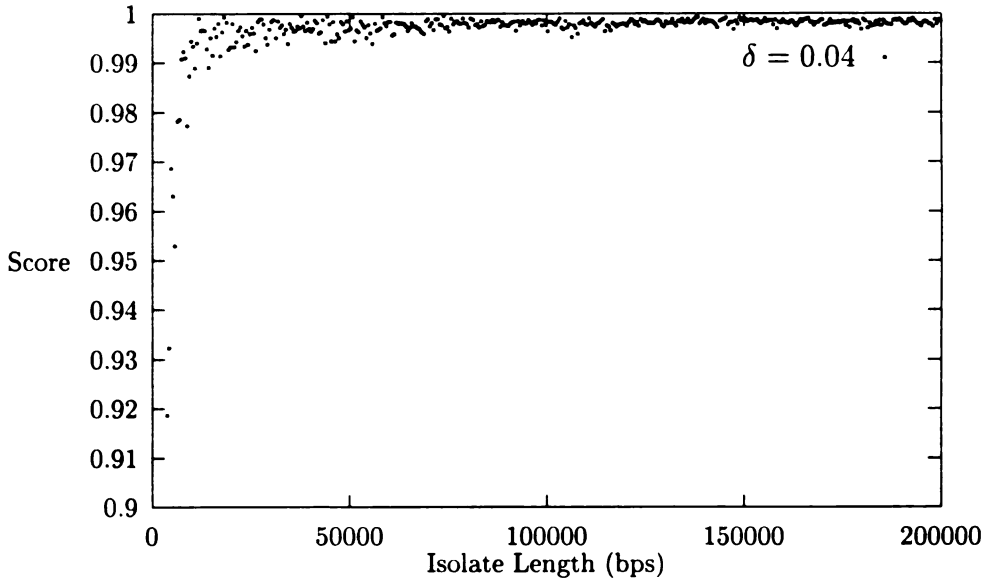


Figure 3.7: Score metric, $\delta = 0.04$.

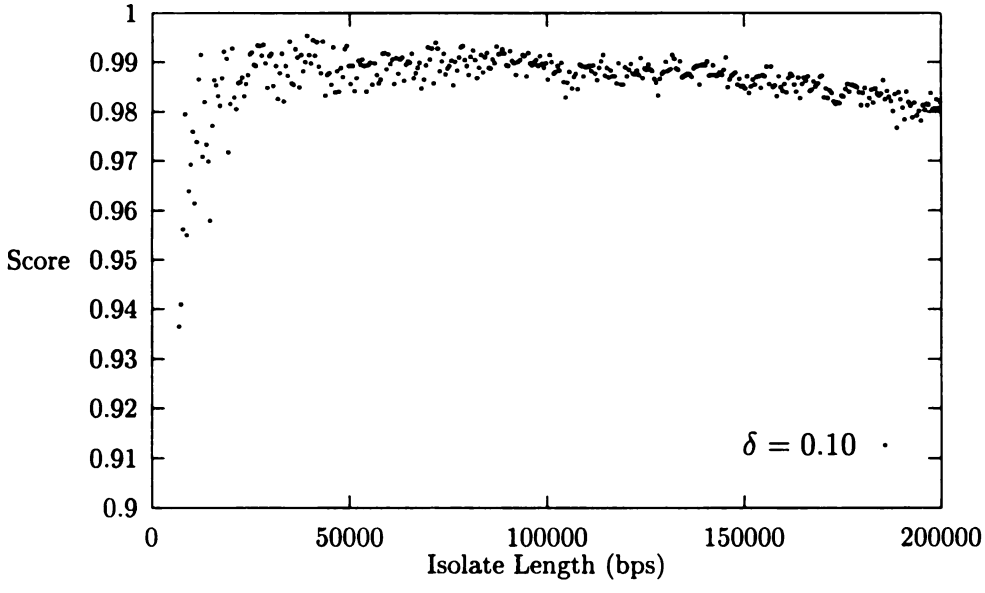


Figure 3.8: Score metric, $\delta = 0.10$.

Chapter 4

A Matching Algorithm

In this chapter we consider the design of an algorithm that searches the restriction map data of a genome for a location which matches the restriction pattern of a query isolate. The restriction map of the genome contains N fragments, and the restriction pattern of the isolate consists of k fragment lengths. The algorithm should find all locations of the genome which match the isolate data.

In the next section we provide a theorem which is the basis of our algorithms. In the following section we provide the algorithm and improvements of that algorithm.

4.1 The Basis Theorem

Definition 4.1. *For any restriction map RM, we define its sorted restriction map $\overleftarrow{RM}$ as an ordered tuple consisting of the elements of RM such that,*

$$\forall i, j, i < j : \overleftarrow{RM}_i \geq \overleftarrow{RM}_j$$

Definition 4.2. *For any restriction pattern RP, we define its sorted restriction map*

$\overleftarrow{RP}$, *as an ordered tuple consisting of the elements of* RM *such that,*

$$\forall i, j, i < j : \overleftarrow{RP}_i \geq \overleftarrow{RP}_j$$

Lemma 4.1. *For numbers* x_1, x_2, y_1, *and* y_2 *such that* $x_1 \leq x_2$ *and* $y_1 \geq y_2$, *if* $x_1 =^\delta y_1$ *and* $x_2 =^\delta y_2$, *then* $x_2 =^\delta y_1$ *and* $x_1 =^\delta y_2$.

Proof.

$$\begin{aligned}
y_1(1-\delta) \leq x_1 \wedge y_1 \geq y_2 &\Rightarrow y_2(1-\delta) \leq x_1 \\
x_2 \leq y_2(1+\delta) \wedge x_1 < x_2 &\Rightarrow x_1 \leq y_2(1+\delta) \\
y_1(1-\delta) \leq x_1 \wedge x_1 < x_2 &\Rightarrow y_1(1-\delta) \leq x_2 \\
x_2 \leq y_2(1+\delta) \wedge y_1 \geq y_2 &\Rightarrow x_2 \leq y_1(1+\delta)
\end{aligned}$$

□

Theorem 4.1. $RP \sim^\delta RM$ *if and only if* $\overleftarrow{RP} =^\delta \overleftarrow{RM}$.

Proof. The converse of the theorem is trivial and simply follows from Definition 1.6.

Now, we have to show that if $RP \sim^\delta RM$, then $\overleftarrow{RP} =^\delta \overleftarrow{RM}$. Assume that there exist RP and RM such that $RP \sim^\delta RM$ but not $\overleftarrow{RP} =^\delta \overleftarrow{RM}$. We define an *inversion* of an ordered tuple S, as the number of occurrences of i, such that $S_i < S_{i+1}$. Notice that when the number of inversions is 0, then $S = \overleftarrow{S}$. Consider the minimum

inversion permutation on RP, the ordered tuple RP', such that $RP' =^{\delta} \overleftarrow{RM}$. The number of inversions should be positive by assumption. Let i denote the first index i of RP' such that $RP'_i < RP'_{i+1}$. As $RP' =^{\delta} \overleftarrow{RM}$, we have

$$RP'_i \quad =^{\delta} \quad \overleftarrow{RM}_i$$

$$(and)$$

$$RP'_{i+1} \quad =^{\delta} \quad \overleftarrow{RM}_{i+1}$$

By Lemma 4.1, we obtain

$$RP'_{i+1} \quad =^{\delta} \quad \overleftarrow{RM}_i$$

$$RP'_i \quad =^{\delta} \quad \overleftarrow{RM}_{i+1}$$

This means that there exists another permutation on RP matching $\overleftarrow{RM}$ with error δ with one inversion less than RP'. This contradicts our first assumption, and thus the theorem follows. □

4.2 Brute Force Algorithm

The algorithm is based on Theorem 4.1 which states that we can determine if the restriction pattern of an isolate matches the restriction map of a subsequence of the genome with k fragments in $O(k)$ when both the restriction pattern and restriction map are previously sorted.

A brute force approach is to sort each size k subsequence of the genome and compare it against the isolate. In this case we have at most, $k \log k + k$ comparisons for each subsequence of the genome resulting in at most $N(k + k \log k)$ comparisons (Algorithm 1). However, we can optimize this naive approach to perform much better. We describe these improvements in the remainder of this chapter.

Algorithm 1 Brute Force: Matching RP to RM with error δ

```
1: Sort RP, to obtain ←RP
2: for all RM' subsequences of RM, such that |RM'| = k do
3:    Sort RM', to obtain ←RM'
4:    Matched ← True
5:    for i = 1tok do
6:       if notRP_i =^δ RM'_i then
7:          Matched ← False
8:       end if
9:    end for
10:   if Matched then
11:      Report RM'
12:   end if
13: end for
```

4.3 Obtaining Sorted Subsequences in $O(Nk)$ comparisons

Algorithm 1 sorts each size k subsequence of RM once; hence it incurs a running time of $O(Nk \log k)$. We can improve this to $O(Nk)$ by using the fact that two overlapping subsequences which differ only in one element will be almost the same when sorted. We exploit this by making a moving window over the elements of RM in which we update only one element of RM' at each pass. If we store enough information to know which element of RM' should be removed for each update and

	$N = 2904$		$N = 1257$	
k	$\delta = 0.04$	$\delta = 0.10$	$\delta = 0.04$	$\delta = 0.10$
10	0.0061	0.0060	0.0026	0.0028
20	0.0081	0.0084	0.0037	0.0036
30	0.0103	0.0103	0.0045	0.0047
40	0.0122	0.0121	0.0056	0.0057

Table 4.1: Running time (in seconds) of the algorithm.

shift the elements of RM' as needed, the average number of shifts and comparisons needed for each update will be

$$\frac{\sum_{i=1} k \sum_{j=1} k|i-j|}{k^2} = \frac{k}{3}$$

That means we'll have a total of $(N-1)\frac{k}{3} + k \log k + Nk$ comparisons.

In Table 4.1 the running time of the algorithm is shown for various values of N, k, and δ. Notice that the running time is linearly increasing for both k and N, and it is independent of δ.

4.4 Limiting the search area to the most promising areas

The next improvement we make is to limit the search area. Rather than searching the entire genome, we focus on the areas which are most likely to succeed. We know that all members of RP should be present in the matched subsequences of RM. Hence, based on the information about RP, we can limit our search in RM. In particular, if any fragment of RP is extremely rare in RM, that could be used to focus our search on only a few areas. However in order to gain the most from this

approach, we should know which elements of RP are least likely to have matches in RM. We use a probabilistic model to help us identify these fragment lengths.

A normalized fragment x' matches normalized fragment x whenever (Theorem 2.1)

$$(1-x)^{1-\delta} \geq 1-x' \geq (1-x)^{1+\delta}$$

As x' is a uniformly distributed random variable, the fraction of RM which match to x will be:

$$(1-x)^{1-\delta} - (1-x)^{1+\delta}$$

Using second order Taylor approximation we derive

$$-2\delta(1-x)\ln(1-x) + O(\delta^3)$$

Ignoring the third order term of δ and using the fragment length l in place of x, we obtain

$$2\delta\lambda l e^{-\lambda l}$$

So, we try to filter out the RM using any member of RP which achieves the lowest value of the function above.

We can use all the RP data iteratively to filter out the RM. However, keeping track of filtered RM requires more time than is needed to search RM with brute force. Hence in practice, we use only the first few members of RP for filtering.

In Table 4.2 the running time of the algorithm is shown when it uses only one

	$N = 2904$		$N = 1257$	
k	$\delta = 0.04$	$\delta = 0.10$	$\delta = 0.04$	$\delta = 0.10$
10	0.0019	0.0029	0.0009	0.0014
20	0.0026	0.0037	0.0012	0.0017
30	0.0031	0.0045	0.0015	0.0020
40	0.0042	0.0058	0.0019	0.0023

Table 4.2: Running time (in seconds) of the algorithm with 1 limiting fragment.

	$N = 2904$		$N = 1257$	
k	$\delta = 0.04$	$\delta = 0.10$	$\delta = 0.04$	$\delta = 0.10$
10	0.0025	0.0030	0.0011	0.0013
20	0.0028	0.0037	0.0013	0.0016
30	0.0027	0.0036	0.0014	0.0019
40	0.0030	0.0048	0.0014	0.0020

Table 4.3: Running time (in seconds) of the algorithm with 3 limiting fragments.

fragment of RP to filter out the RM. Here, the running time is dependent on the value of δ. It can be seen that the running time has been improved. In Table 4.3 the running time of the algorithm is shown when it uses three fragments of RP to filter out the RM. For smaller values of k, the overhead of using 2 more fragments to filter the data does not compensate the gain in limiting the search area. However for higher values of k we have a better running time. It is still an open question to find the optimal number of limiting fragments for given values of N, k, and δ.

4.5 More Improvement

We observed that, in practice, the RM' changes locally; that is, new RM' differs in only one area from the previous RM'. We divide RM' into three consecutive sections. The first section $\overleftarrow{RM'}_1 \ldots \overleftarrow{RM'}_{l-1}$ matches $\overleftarrow{RP}_1 \ldots \overleftarrow{RP}_{l-1}$, the second sec-

tion is a single fragment ($\overleftarrow{RM'_l}$) that does not match $\overleftarrow{RP_l}$, and the third section is $RM'_{l+1} \ldots RM'_k$. We only compare the elements of RM' and RP if RM' has changed in the area of $RM'_1 \ldots RM'_{l'}$, where l' is the value of l from the previous iteration.

There was not much difference in running times after this improvement. It is due to the fact that the comparison of two sorted restriction patterns takes less time than generating them.

4.6 More on Comparing RP and RM

Up to this point, we have assumed that the restriction pattern of the isolate is a subsequence of the genome's restriction map. However, this assumption is not always true (Figure 4.1). In general the two tail fragments of RP can be a partial fragment of their corresponding ones on RM.

Definition 4.3. *We say the restriction pattern RP inner-matches restriction map RM with error δ, denoted $RP \approx^{\delta} RM$, if and only if*

$$\exists \{x, y\} \subset RP :$$

$$\left(RP - \{x, y\} =^{\delta} RM - \{RM_1, RM_k\}\right) \wedge$$

$$\left(\left(x \leq^{\delta} RM_1 \wedge y \leq^{\delta} RM_k\right) \vee \left(y \leq^{\delta} RM_1 \wedge x \leq^{\delta} RM_k\right)\right)$$

where $k = |RM|$.

In other words a restriction pattern RP inner-matches a restriction map RM when there is relaxed matching between elements of RP and RM. Figure 4.1 illustrates this definition.

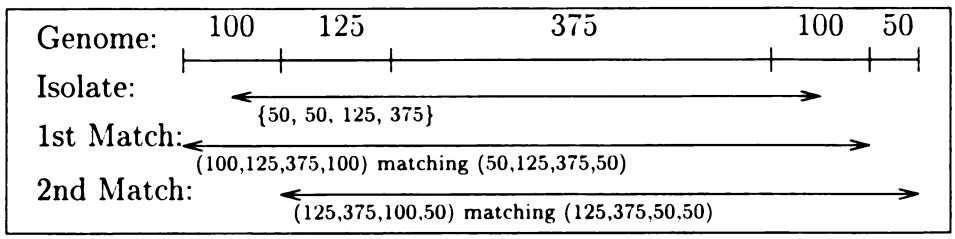


Figure 4.1: Inner-Matching.

We cannot use Theorem 4.1 to compare RP and subsequences of RM for this case. However, with a little modification, we can still determine whether or not $RP \approx^{\delta} RM$ in $O(k)$ time. Suppose the elements of RP and RM are already sorted in decreasing order. Two of the elements of RM can have lower values in RP. We then skip those elements of RM when trying to compare RM and RP (Algorithm 2).

Theorem 4.2. *$RP \approx^{\delta} RM$ if and only if Algorithm 2 terminates with a non-negative skips value.*

Proof. Using Definition 4.3 it is easy to see that the algorithm terminates with a negative *skips* value unless $RP \approx^{\delta} RM$.

Now, we have to show that if $RP \approx^{\delta} RM$, then the algorithm terminates with a non-negative *skips* value. If $RP \approx^{\delta} RM$, then there is a matching between elements of RP and RM. Consider such a matching, and assume that the $\overleftarrow{RP_i} \leq^{\delta} \overleftarrow{RM_1}$ and $\overleftarrow{RP_j} \leq^{\delta} \overleftarrow{RM_k}$. Build a new set RP' such that it has all elements of RP except RP_i and RP_j, with $RM_1(1+\delta) \in RP', RM_k(1+\delta) \in RP'$. By Theorem 4.1, we have $\overleftarrow{RP'} \sim^{\delta} \overleftarrow{RM}$. Now replace $RM_1(1+\delta)$ of $\overleftarrow{RP'}$ with RP_i, and $RM_k(1+\delta)$ of $\overleftarrow{RP'}$ with RP_j. The order of elements of $\overleftarrow{RP'}$ and $\overleftarrow{RP}$ are exactly the same except that RP_i and RP_j have higher indexes in $\overleftarrow{RP}$. Consider Figure 4.2 for an illustration where

Algorithm 2 Inner Match: Is $RP \approx^{\delta} RM$?

Sort RP, to obtain $\overleftarrow{RP}$
Sort RM, to obtain $\overleftarrow{RM}$
$skips \leftarrow 0$
$i \leftarrow 1$
while $i \leq k$ **and** $skips \geq 0$ **do**
  if $\overleftarrow{RM}_i$ is RM_1 or RM_k **then**
    if $RP_{i-skips} \leq^{\delta} RM'_i$ **then**
      $skips \leftarrow skips + 1$
      $SkipBack \leftarrow$ **False**
    else
      $SkipBack \leftarrow$ **True**
    end if
  else
    if $RP_{i-skips} =^{\delta} RM'_i$ **then**
      $SkipBack \leftarrow$ **False**
    else
      $SkipBack \leftarrow$ **True**
    end if
  end if
  if $SkipBack$ **then**
    $skips \leftarrow skips - 1$
  else
    $i \leftarrow i + 1$
  end if
end while
return $skip \geq 0$

the solid lines show a matching.

The algorithm matches all elements of $\overleftarrow{RP}$ and $\overleftarrow{RM}$ until it reaches RP_i to be matched against RM_k. There are two cases. Either (i) they match or (ii) they don't. In case (i) the algorithm corrects the *skips* value immediately, and continues. In case (ii), the algorithm does not correct the *skips* value immediately. In either case it does not terminate with a negative *skips* value.

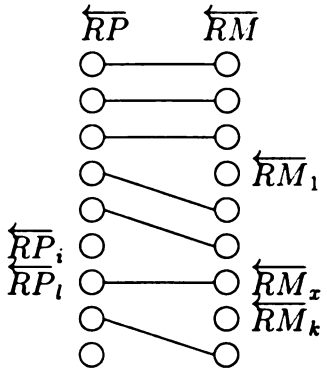


Figure 4.2: Proof of Theorem 4.2.

□

Bibliography

Bibliography

[Aoe94] Jun-ichi Aoe. *Computer Algorithms: String Pattern Matching Strategies.* IEEE Computer Society Press, 1994.

[BM77] Robert S. Boyer and J. Strother Moore. A fast string matching algorithm. *Communications of the ACM*, 20(10):62–72, October 1977.

[Dev95] Jay L. Devore. *Probability and statistics for engineering and the sciences.* Duxbury Press, 4th edition, 1995.

[FAW+95] R. D. Fleischmann, M. D. Adams, O. White, R.A. Clayton, E. F. Kirkness, A. R. Kerlavage, C. J. Bult, J. F. Tomb, B. A. Dougherty, J. M. Merrick, et al. Whole-genome random sequencing and assembly of haemophilus infuenzae Rd. *Science*, 269(5223):496–512, July 1995.

[FGW+95] C. M. Fraser, J. D. Goycayne, O. White, M. D. Adams, R. A. Clayton, R. D. Fleischmann, C. J. Bult, A. R. Kerlavage, G. Sutton, J. M. Kelley, et al. The minimal gene complement of mycoplasma genitalium. *Science*, 270(5235):397–403, October 1995.

[GB91] Gaston H. Gonnet and Ricardo Baeza-Yates. *Handbook of Algorithms and Data Structures.* Assison-Wesley, 2nd edition, 1991.

[KCTP96] Jin Kim, James R. Cole, Eric Trong, and Sakti Pramanik. Inferring relatedness of a macromolecule to a sequence database without sequencing. In *Proceedings of the Fourth International Conference on Intelligent Systems for Molecular Biology*, pages 125–133, June 1996.

[KMP77] Donald E. Knuth, James H. Morris, and Vaughan R. Pratt. Fast pattern matching in strings. *SIAM Journal on Computing*, 6(2):323–350, June 1977.

[KR87] Richard M. Karp and Michael O. Rabin. Efficient randomized pattern-macthing algorithms. *IBM Journal of Research and Development*, 31(2):249–260, March 1987.

[LW95] Eric S. Lander and Michael S. Waterman. *Calculating the Secrets of Life.* National Academy Press, 1995.

[Wat95] Michael S. Waterman. *Introduction to Computational Biology.* Chapman and Hall, 1995.