

# Biological Mathematics: A General DNA Splicing Model



Garret Suen

CPSC601.73

Wednesday, January 30, 2002



## Forward...

The following is a brief summary of a general DNA splicing model system as outlined by *Lila Kari* in her article:

*DNA Computing: arrival of biological mathematics.*



## What we've learned so far...

So far, we've seen some basic DNA models for computation as outlined by Adelman, and others.

We have looked at a general proposed rule-set used to generate operations that can be used for computation.

These proposed models can seemingly solve many problems, including those of the NP variety.



## Extending the Model

The previous models all have one aspect in common: They treat DNA as a static non-mutable data structure.

In biology, however, DNA may exist in ever-changing, non-static forms.

Dynamically changing DNA serves as a form of protection in bacteria against invading bacteriophage DNA.



## Splicing in Nature

Amongst higher-level organisms, most are invaded by bacteria that can cause disease.

Bacteria, are cellular organisms that can cause serious harm.

At the same time, there are specific organisms that can attack a bacteria and cause it serious harm.

These are known as phages, and in specific bacteriophages.



# Bacteriophages and how they work

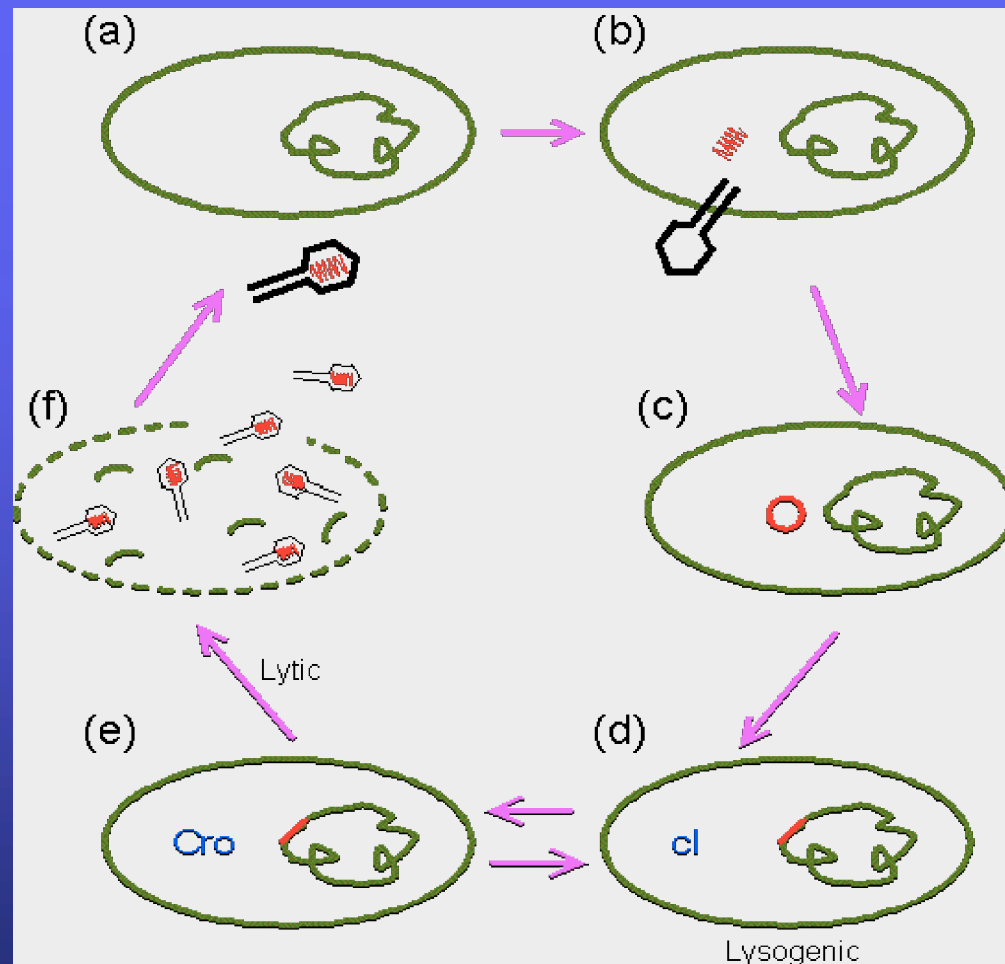
Bacteriophages are parasitic organisms that attack bacteria (they are harmless to humans).

They attack bacteria by injecting their DNA into the bacteria.

The phage DNA then incorporates itself into the bacterial DNA and forces it to construct new phages.

These new phages are then released to infect other bacteria.

# Phage Life Cycle...



Courtesy of:

<http://www.web-books.com/MoBio/Free/Ch1F1.htm>



# Fighting a Phage

Just as we as humans have an immune system to fight off bacteria, bacteria have also developed special mechanisms to fight off phages.

Most bacterial systems fight phages by targeting phage DNA.

If a bacteria can render phage DNA inoperable, then it really can't do anything to the bacteria.



# Restriction Enzymes

Bacteria have developed specialized enzymes to fight against infecting phage DNA.

These enzymes are known as Restriction Enzymes.

Restriction Enzymes can recognize specific subsequences of DNA and cut them into pieces.

This effectively renders the DNA inoperable as it can not be transcribed and translated.



## So What?

Restriction Enzymes have become vastly important in Biotechnology over the years.

The ability to cleave sequences of DNA have led to the ability of producing important proteins such as insulin for diabetes patients (recombinant DNA).



## How do they work?

Restriction Enzymes work by recognizing a specific sequence embedded into a DNA strand.

These sequences are usually 4-6 nucleotides long.

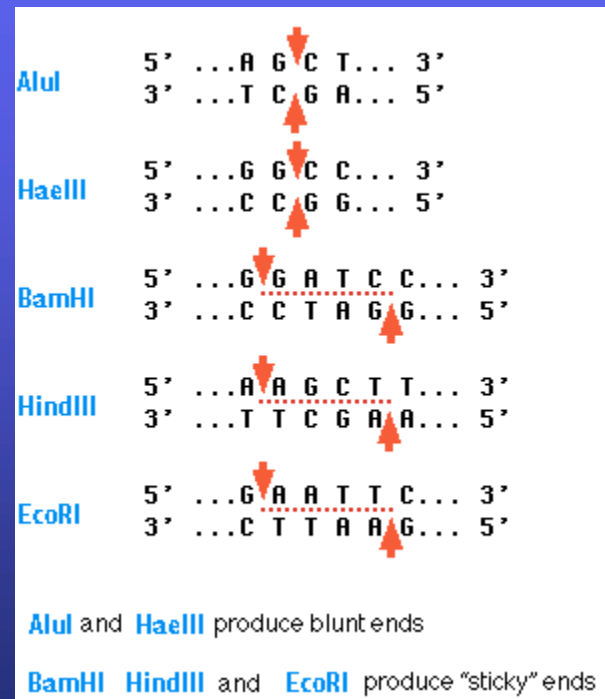
Each Restriction Enzyme has a specific cut site, a site where it actually does the cutting.

The cut site can be anywhere from position 2 to 5 on a 6-nucleotide Restriction Enzyme.



# Some Examples

Here are some common examples of Restriction Enzymes and their cut sites.



Courtesy of:

<http://www.ultranet.com/~jkimball/BiologyPages/R/RestrictionEnzymes.html>



## Sticky Ends

Once a Restriction Enzyme has cut a piece of DNA, it produces two pieces of DNA.

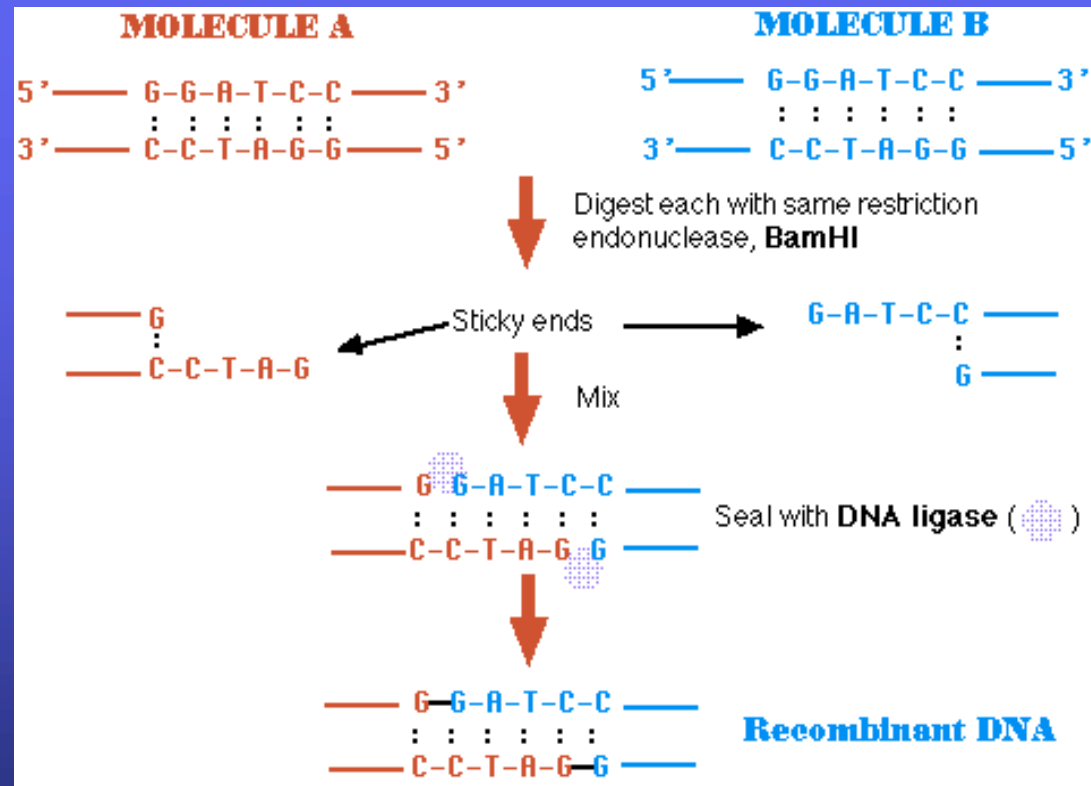
Each of the two pieces contain “sticky ends” where the Restriction Enzyme made its cut.

These sticky ends may be a small (1-4) nucleotide sequence that are not base paired.

These sticky ends are available to bind to other such sticky ends as long as they are complement (through the use of DNA ligase).

Thus one can cross-join two strands of DNA using the same Restriction Enzyme.

# Sticky Ends in Action...



Courtesy of:

<http://www.ultranet.com/~jkimball/BiologyPages/R/RecombinantDNA.html>



## A Splicing Model

The use of restriction enzymes in bacteria is sometimes referred to as a splicing model.

Splicing is most commonly found in processing of mRNA before it is passed out of the nucleus for translation.

Splicing is simply the removal of specific sequences within a DNA/RNA molecule.



# Applying it to DNA Computation

The ability to cut sequences of DNA and paste complement sequences of DNA together allows us new operation to include into our DNA computational model.

We call this a DNA splicing model.

To define this system, we introduce a formal language to describe the splicing model.



# A Formal Language...

Formal are useful for understanding complex systems by adding a level of abstraction.

For our purposes, we will abstract the DNA splicing model into a formal language and show how the cut and paste rules can work under such a language.



# The Alphabet

We define an alphabet as being:  $\Sigma$  (looking one level down, we note that our alphabet is simply A, C, G, T).

We can define words based on this alphabet by denoting that words are simply concatenations of our alphabet, called:  $\Sigma^*$ .

We also define that our language has a null element, or the empty string as:  $\epsilon$ .



# The Splicing Rule (I)

Given some alphabet  $\Sigma$  and two strings  $x$  and  $y$  over  $\Sigma$ , we can define a splicing rule  $r$  as follows:

- $R$  consists of two steps:
  1. Cut  $x$  and  $y$  at certain positions.
  2. Paste the resulting prefix of  $x$  with the suffix of  $y$  and the prefix of  $y$  with the suffix of  $x$ .
- From this we can say that a splicing rule  $r$  over  $\Sigma$  is a word of the form  $\Sigma_1 \# \Sigma_1 \$ \Sigma_2 \# \Sigma_2$ .



## The Splicing Rule (II)

We can say that  $z$  and  $w$  are obtained by splicing  $x$  and  $y$  according to the following splicing rule  $r = \square_1 \# \square_1 \$ \square_2 \# \square_2$  as:

$$(x, y) \square r(z, w)$$

If and only if:

$$x = x_1 \square_1 \square_1 x'_1$$

$$y = y_1 \square_2 \square_2 y'_1$$

$$z = x_1 \square_1 \square_2 y'_2$$

$$w = y_2 \square_1 \square_1 x'_1$$

For some  $x_1, x'_1, y_1, y'_1 \square \square^*$



## The Splicing Rule (III)

What we've outlined is a simple abstraction of the splicing rule that is found in DNA.

The formal language definition simply allows us to apply this splicing rule to any alphabet that we chose to use.

The Splicing Rule can be used to do a variety of operations such as addition (see pg.10-11 of the article).



# Defining a Splicing System

Now that we have defined how a splicing rule works, we can apply this rule to sets of alphabets.

If a simple alphabet is used as a language (or a set), we note that there are no restrictions to splicing and that:

1. The splicing sections  $x$  and  $y$  are not used up by the splicing action, and are available to be used again.
2. There is no restriction on the number of copies of  $z$  and  $w$  that we obtain, as they can be easily duplicated.

However, we note that with DNA, this is not the case.

Only some DNA strands are available for use some of the time, and in limited quantities.



# Splicing over Multisets

To solve this issue, we introduce the idea of multisets.

With multisets, we can keep track of the number of copies of strands from moment to moment.

With a multiset, we simply define sets for each given splice application.

Thus, from our splicing rules, we see that there is an initial set that consists of strands  $x$  and  $y$  ( $\{x, y\}$ ).

After the splicing rule has been applied, we get another set that consists of strands  $z$  and  $w$  ( $\{z, w\}$ ).

With use of multisets we can keep track of what strands are going where.



# A Formal Splice System

We define a multiset to be over  $\Sigma^*$ .

We define  $M(w)$  as being a function of the multiset that returns the number of copies of the word  $w$  in that multiset (in DNA there are potentially an infinite amount of any give word  $w$ ).

*Definition:* A splicing system is a quadruple  $\Sigma = (\Sigma, T, A, R)$ , where  $\Sigma$  is an alphabet,  $T \subseteq \Sigma$  is the terminal alphabet,  $A$  is the multiset over  $\Sigma^*$ , and  $R \subseteq \Sigma^* \# \Sigma^* \$ \Sigma^* \# \Sigma^*$  is the set of splicing rules.



# Splice System Characteristics (I)

Given the previous definition, the following holds:

1.  $M(x) \geq 1, M(y) \geq 1$ , if  $(x \neq y)$  (resp.  $M(x) \geq 2$  if  $x = y$ ).
  - *This simply states that there must be two viable strands for splicing,  $x$  and  $y$ , and that there must exist at least one copy of both  $x$  and  $y$  for splicing to occur. Respectively if strand  $x = y$  then it follows that there must be at least two copies of  $x$  (or  $y$ ) for splicing to occur.*



## Splice System Characteristics (II)

2.  $(x, y) \sqsupseteq r(z, w)$  according to the splicing rule  $r$  (defined above).
3.  $M'(x) = M(x) - 1, M'(y) = M(y) - 1$  if  $x \neq y$   
(resp.  $M'(x) = M(x) - 2$  if  $x = y$ )
  - *This simply states that the contents of the next set of  $\{x, y\}$  will have two strands deleted from it, one of type 'x' and another of type 'y' (as they have been spliced and joined to form new strands which are now not part of the set  $\{x, y\}$ ). Respectively, if  $x = y$  then this set loses 2 strands both of type x.*



## Splice System Characteristics (III)

4.  $M'(z) = M(z) + 1$ ,  $M'(w) = M(w) + 1$  if  $z \neq w$   
(resp.  $M'(z) = M(z) + 2$  if  $z = w$ ).
  - *This simply states that the contents of the set of  $\{z, w\}$  will have two strands added to it (the new strands created from splicing), one of type 'z' and another of type 'w'. Respectively, if  $x = y$  then this set gains 2 strands both of type z (or w).*



## Great, but how does this help?

Generally speaking, splicing rules are simply another operation that we can add to our programming tool box.

They add a new level of flexibility, and allow us to cut and paste parts of DNA, and ultimately, data from one memory strand to another.

It is postulated that a *universal programmable DNA computer* can be constructed based on splicing operations.

The Cut and Paste aspect of splicing is very much akin to union and intersection in set theory.



# Feasibility of DNA Computing

One of the questions of DNA computing beyond the theoretical aspect is whether or not the operations outlined are feasible.

Here we will outline some basics on DNA manipulation techniques and their feasibility.



# Constructing DNA

DNA synthesis is a relatively simple process that can be done efficiently in a lab.

You can tailor make a DNA strand by simply starting with the nucleotide of your choice.

This nucleotide is attached to a glass bead for a secure anchor.

A solution of the next nucleotide in the sequence is added to the test tube, and the nucleotides are allowed to anneal.

Then you wash the solution free of excess nucleotides and repeat the process with the next nucleotide of your choice.



# DNA Hybridization

DNA hybridization is the process of taking two single strands of DNA and attempting to anneal the two together (if they are complementary).

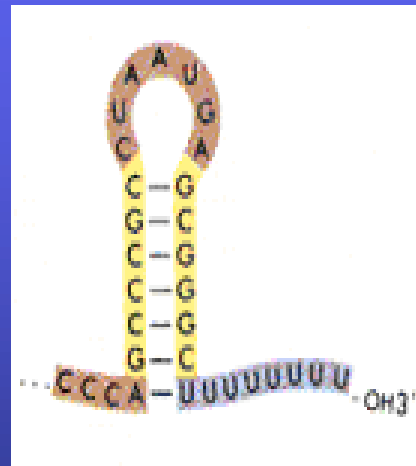
Generally, one can hybridize two strands of DNA by simply heating the mixture and changing the reaction conditions (salt concentration, etc).

One must be careful however, in self-annealing, the process whereby a single strand of DNA will anneal to itself, forming hairpin loop structures.



# Hairpin Loops

Hairpin loops are caused by complementary sections of nucleotides located on the same strand.



Courtesy of:

<http://www.mun.ca/biochem/courses/3107/Lectures/Topics/transcription.html>



# Cleaving DNA

We have seen that DNA can be cleaved rather easily through the use of Restriction Enzymes.

However, Restriction Enzymes are not 100% error-free.

Often times, partial digests of DNA occur.



# Ligating DNA

Ligating is the process of attaching strands of DNA together.

This is done through the use of DNA ligase, an enzyme used primarily in DNA replication.

One can attach complementary sticky ends created by restriction enzymes together (pasting DNA together).

While DNA Ligase is rather accurate, it is not perfect (leading to some mutations in DNA).



# Detection and Sequencing

DNA detection and Sequencing is the process of determining the exact sequence of a given DNA strand.

This is done usually through gel electrophoresis and such specialized techniques as the Sanger Method for DNA sequencing.

These methods are costly, time-consuming and not very reliable.



# DNA Replication

Done through the use of PCR.

Relatively cheap and fast.

Requires knowledge of the primer sequence of a known DNA sequence.

Not reliable past 10kb of amplification, although this is improving as better DNA Polymerases are being synthesized.



# Will we ever see a DNA Computer?

The theory of DNA computation is rather sound.

The bottleneck, however, lies within the ability of biotechnology to perform the operations that we have discussed.

As new techniques in biotechnology emerge, it may be feasible to consider DNA computation as a reasonable alternative to conventional computation.