A Systematic and Extensible Approach to DNA Primer Design for Whole Gene Synthesis

by

Amanda Victrix Allen Wozniak

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2005

Author .....
Department of Electrical Engineering and Computer Science
August 25, 2005

Certified by .....
Thomas F. Knight Jr.
Senior Research Scientist
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Department Committee on Graduate Theses

A Systematic and Extensible Approach to DNA Primer Design for Whole Gene Synthesis

by

Amanda Victrix Allen Wozniak

Submitted to the Department of Electrical Engineering and Computer Science
on August 25, 2005, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

The future of synthetic biology research hinges upon the development of accurate and inexpensive whole gene synthesis technologies. Recent advances in the purification of solid-phase manufactured oligonucleotides make it possible to manufacture whole genes by polymerase chain reaction methods. Yet, despite the improvement in laboratory methods, whole gene synthesis is not rapidly progressing because most gene design software takes an excessively naive approach to the complex problem of designing component oligonucleotides for whole gene synthesis. The synthetic biology community needs a flexible, robust and optimal primer design tool.

We present the software design for a tool which designs oligonucleotides that are compatible with a wide variety of oligo purification and whole gene assembly protocols. Our design strategy uses physical sequence feature identification, optimal artificial intelligence search techniques, and sequence optimisation via intelligent codon substitution to produce near-optimal oligonucleotide arrays. We address all aspects of the oligonucleotide design problem, from physical constraints to the computational overhead involved in searching for an optimal solution, and provide an extensive set of data structures and algorithms.

Thesis Supervisor: Thomas F. Knight Jr.
Title: Senior Research Scientist

Acknowledgments

For their knowledge, oversight and influence in my academic development I principally want to thank Dr. Thomas K. Knight Jr, my thesis advisor, and Professor Gerald Jay Sussman who advised me in both my undergraduate and graduate academic coursework, inspired me to change majors and referred me to TK. I also appreciate the priviledge of having many excellent instructors, including but not limited to Professor William Thilly, Professor Roger Mark, Professor Steve Burns, Professor Steven B. Leeb, Professor Peter Dourmashkin and Dr. Claiborne Skinner.

I would like to thank Jake Beal for his time, friendship and patient explanations of artificial intelligence search techniques (both as my TA and as a fellow graduate student). Eliot Gable has provided invaluable technical support for PHP and my project development environment. I must also thank Greg Martin for his willingness to argue about data abstractions and the loan of his personal textbook collection.

For their patience in reviewing the text of my thesis, my thanks to KC Kerby and Jen Mitchel. No braver soul has ever tackled the superfluous adverb and dangling preposition.

For innumerable favors and offers of assistance, my sincerest thanks to Ken Clary, Riad Wahby, Chris Porter, Rob Morrison, John Hawkinson and the members of the Student Information Processing Board.

Finally, mere thanks can not communicate my gratitude for the personal support and friendship of Mark Feldmeier, Limor Fried, KC Kerby, Anne Alvarado, Laura Nichols, Jen Mitchel, Carmen Phillips, Kristin Josephson, Shervin Fatehi, Jim Paris, Dan McAnulty, Jake Beal, Abi Harper, Chris Vogt, Josh Lifton, Ken Clary, Ali Mohammed and his ever-entertaining cadre of CSAIL graduate students and UROPs, my family and other friends, select residents of MIT's Senior House and the patriots of Fort Awesome.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This thesis addresses a predominant problem in synthetic biology: how one can manufacture long pieces of DNA in a laboratory and have the process be consistently accurate and inexpensive. Influenced by ongoing research in oligonucleotide purification [2] [7] [26], we set out to implement a tool which systematically designs DNA oligonucleotides that are compatible with a variety of oligonucleotide purification and whole gene assembly techniques. While we do not include the complete implementation of a functional software package, we do present a complete software design, including pseudocode for the critical algorithms and methods.

This document includes an analysis of the problems involved in whole gene synthesis (WGS) including restrictions imposed by standard biology laboratory methods and our limited understanding of DNA hybridization kinetics. We then present a systematic approach to designing arrays of oligonucleotides which both conform to the necessary restrictions of the user's purification techniques and are suitable for use in whole gene synthesis. The presentation of our software design includes substantial discussion of alternate methods for oligonucleotide design.

Our approach treats the primer design problem as a classic constraint satisfaction problem. The appropriate laboratory methods, as selected by a virtual end-user, are translated into a set of physical constraints which individual oligonucleotides and the final array of oligonucleotides must satisfy. Common computer science techniques, specifically string processing algorithms, are used both to identify relevant physical

features of the target DNA and to evaluate whether oligonucleotides satisfy the end-user's constraints. This approach is presented in a hierarchical manner from high-level discussions to low-level descriptions of the required data structures and pseudocode for the required algorithms.

A capable software engineer should be able to take our primer design approach to a final software tool given only this document, a working knowledge of PERL and SCHEME or another suitable language.

1.1 The Motivating Problem: Oligo Synthesis

Synthetic biology as a discipline includes protein engineering, engineering novel cellular sensors, designing protein-DNA logic circuits and creating synthetic organisms. It is a derivative of genetic engineering, but the requirements of synthetic biology go well beyond the capacity to clone natural DNA and splice those sequences into new host systems. In order to fully develop the field of synthetic biology, researchers need the ability to create entirely artificial genes and genomes.

Current DNA synthesis techniques are very limited, however. The standard protocols for manufacturing DNA are all variations on solid-phase phosphoramidite synthesis. [9] The solid phase synthesis technique grows oligonucleotides on a solid support matrix[1] one base at a time. Each base addition involves a series of four separate chemical reactions: deprotection of the growing strand; coupling of the next nucleotide to the growing strand; capping of the unreacted reagents; and oxidation of the successfully elongated strands. The deprotection, capping and oxidation steps are very efficient, so the yield of each base addition is limited by the efficiency of the coupling reaction. A typical coupling efficiency for solid-phase synthesis is only 98.5%, implying that any given base addition has a less than 98.5% yield.[2] This makes it physically impossible to directly synthesize oligonucleotides which are longer than 100

[1]Typically borosilicate glass beads.

[2]For the detailed chemistry of solid-phase synthesis techniques, please refer to [9] or a standard laboratory protocol.

bases with any significant yield.[3]

Even though synthesis yields are currently low for longer oligonucleotides, solution chemistry is constantly improving. We may see a 99.9% efficient base addition protocol developed in the next few years. But even if the yields are improved, it will still be problematic to directly synthesize *accurate* oligonucleotides (oligos) which are long enough to use as synthetic genes. The reason for this is that the solid-phase synthesis process has errors which don't decrease the yield of the reaction. Types of errors include miscapping and mis-deprotection reactions which result in oligos that are missing one base or have an extra base inserted. The probability of error associated with each base addition is P_e. If the number of bases in an oligo is N, as N increases, the probability of any one strand of synthesis product being correct, $P_c = (1 - P_e)^N$ approaches zero. [4]

As already stated, the most common error in solid-phase synthesis is the deletion of bases or early failure of the process due to mis-capping. [9] The mis-capping error mode typically results in oligos which are significantly shorter than the target length. These are straightforward to purify from the full-length sequences using gel electrophoresis. [4] The base deletion error mode, commonly called the N-minus-mer problem, is harder to detect. Gel electrophoresis can not reliably discriminate between a strand of length N and a strand of length $(N - 1)$ for oligos greater than 50 base pairs long. [25] Many researchers are currently developing lab techniques to purify full-length, correct synthetic oligonucleotides from oligonucleotides with a single base deletion by flowing the oligos through some elaborately engineered gel. [1] [2] [11] [16] [19], [26] [24] Many of these methods discriminate correct oligomers from N-minus-1-mers using by exploiting the thermal kinetic penalty of mis-annealing events.

If one flows manufactured oligos over a matrix of statistically correct complements,

[3]If there are N bases in a target oligonucleotide and the yield of a single base addition is P_a, the total yield for the target oligonucleotide is ${P_a}^N$. When $P_a = 0.985$, as is the case for standard solid-phase synthesis techniques, $N = 100$ has a 22% yield and $N = 150$ has only a 10% yield.

[4]For a typical oligonucleotide that is 100 bases long, there is a 98.5% yield for each base addition and a 0.5% chance of error per base addition. For this protocol there will be $({P_c}^N)({P_a}^N) = 13.6\%$ correct yield. A typical gene that is 2000 bases long will have a correct yield of $7.5\text{x}10^{14}$, or less than 1 in 13 trillion attempts.

those oligos which contain errors will have a slower kinetics for duplex formation with the matrix than than perfectly correct oligos. [14] Incorrect sequences will therefore be less likely to bind to the matrix of complements at a given temperature, and similarly, the correct oligos will be retarded. Current techniques for this kind of purification are highly varied and are sophisticated enough to purify out oligos with as few as one base error, but require that the oligonucleotides conform to specific length and target melting temperature criteria. One example of restriction on length is the phenomenon that, as oligo length increases, the thermal or kinetic penalty for duplex formation with a single base error becomes less significant. [14]

It is physically impossible to directly synthesize oligomers which are larger than 100 base pairs in length and similarly difficult to purify oligomers which are longer than 100 bases in length. Synthetic biology requires the manufacture of genes which are thousands of nucleotides long. The most hotly pursued approach to this gene synthesis problem is to splice short sequences together to form longer, gene-length, sequences. These methods are called *whole gene synthesis* (WGS) techniques. Current DNA synthesis and purification techniques guarantee that we can manufacture perfectly correct short oligonucleotides. All that remains is to find a method to determine the set of oligonucleotides that can be used to construct a target gene in an accurate and effective manner using WGS protocols.

1.2 Whole Gene Synthesis Techniques and Primer Design

One method of whole gene synthesis is the primer assembly method, a "primer" being an oligonucleotide which is typically used to prime polymerase chain reactions (PCR). At its simplest, the primer assembly method is as follows: one starts with the sequence for a long piece of DNA; designs a set of short primers which, when assembled, constitute the original strand; and assembles those primers using PCR protocols. The primer assembly method is illustrated in Figure 1-1. While the problem seems

straightforward, it is non-trivial to design a set of primers which can correctly assemble into a target sequence. One "bad" primer will cause an entire gene assembly to fail, and these failures often remain undetected until after the assembly process is complete. If an assembly fails to produce a full-length target sequence, one is left with the tedious task of sequencing the result and analysing where the failure occurred, re-designing and re-manufacturing the component oligoprimers and repeating the assembly protocol. Mistakes represent significant losses in resources and laboratory time.

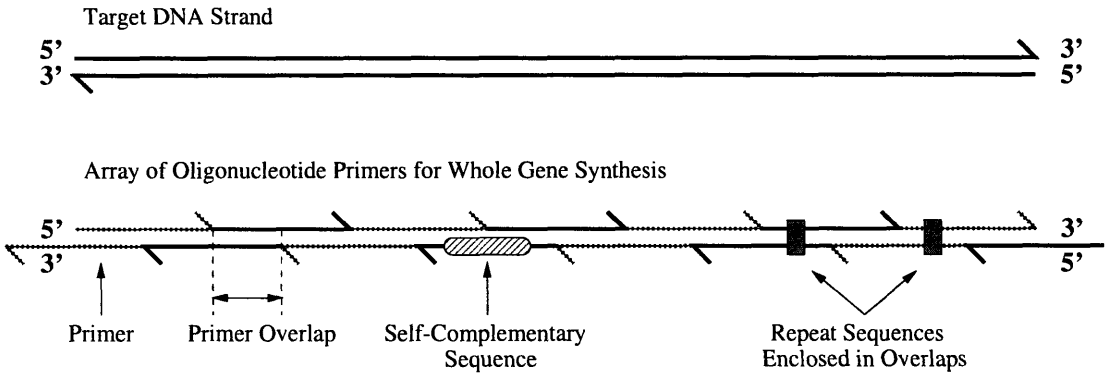


Figure 1-1: The generic method for building genes out of component oligonucleotides.

A primer can be "bad" for multiple reasons: it might bind to itself and become inactive (hairpinning); it may be too similar to another primer, causing an unindented substitution to occur; its temperature of duplex formation could fall outside of the range used by the assembly protocol, resulting in a failed assembly step. What makes a primer "good" or "bad" depends on the physical properties of the specific DNA sequence and on the specific protocols for purification and assembly.

1.3 Existing Research in Whole Gene Synthesis

It is our feeling that a comprehensive and systematic approach to designing primers for WGS methods is currently lacking in the biological engineering community. Those tools which are currently available are listed in Table A.1. All of these tools do not allow the user to specify more than a few restrictions on primer design, do not appear

to be computationally exhaustive, are not designed to find optimal primer sets for WGS techniques[5], and do not allow a user to evaluate, control or modify the internal algorithms which are responsible for primer design.

This thesis presents a systematic design for an extensible software tool explicitly for WGS primer design that is flexible enough to be compatible with a range of standard biology laboratory protocols. Furthermore, this thesis discusses and analyses many, if not all, of the problems involved in optimal primer design. It is our express hope that other software engineers will use this document as a framework from which to address the primer design problem; incorporate more extensive and effective analysis and search algorithms, or at least use my problem analysis as a starting point for their own software designs.

1.4 Document Overview

Understanding the physical behavior of DNA is key to tackling the problem of primer design, and this along with second order considerations of the biochemistry of synthetic biology is presented in Chapter 2. Relevant laboratory protocols along with a discussion of existing WGS techniques can be found in Chapter 3. From this background information, we determine exactly what criteria must be considered to intelligently design primers. We discuss the details of our primer design scheme in Chapter 4 and Chapter 5. In Chapter 4 we take the lessons from the physical properties of DNA from and the limitations of current whole gene synthesis protocols from Chapter 3, and compile the set of constraints on the primer design problem. These constraints are combined with a search strategy[6] to find an optimal set of primers for WGS. This search problem is extended in Chapter 5 to first address the possibility of optimising the DNA sequence via codon substitution and, second, what to do when all else fails. The implementation-level details of our primer design method are pre-

[5]Many of these tools attempt to be an all-in-one synthetic biology design tool, which typically indicates that they can do any one task poorly.

[6]Specifically, constraint propagation with forward checking, which is an optimal search strategy from Artificial Intelligence.

sented in Chapter 5 and the Appendix. Chapter 6 presents our reflections on future investigations which may advance whole gene synthesis techniques in the future.

Chapter 2

Biology Background

Unlike computational genomics or computational biology, primer design is not an abstract mathematical problem. Many of the difficulties involved in designing functional DNA primers center around problematic physical features and behavior of DNA in a laboratory environment. Some minimal background in cell biochemistry is necessary to understanding the approach to DNA primer design which is presented in this document.

2.1 DNA, General Cell Biochemistry

Note: this background material is taught in college biochemistry courses. All general biochemistry knowledge has been checked using [4].

In a cellular environment nucleic acids encode and transmit data. Deoxyribonucleic acid, or DNA, is the linear macromolecule that encodes the genetic information of all organisms, serving as the template for all cellular biochemistry. Enzymes in the nucleus and cytoplasm transcribe ribonucleic acid, or RNA, from DNA. The DNA serves as a template for transcription and is not modified or destroyed by the process of transcription. In the cytoplasm of the cell, RNA is non-destructively translated into chains of amino acids by organelles called *ribosomes*. These amino acid chains are subsequently processed, or *post-translationally modified* into proteins.

The bulk of genetic engineering and design concerns the physical identification

and manipulation of DNA, typically with the goal of producing novel cell biochemistry. As DNA is several steps removed from protein production, successful genetic engineering requires an in-depth understanding both of the physical properties and biochemistry of DNA, and of the enzymatic mechanisms and organelles which interact with DNA and the resultant RNA in both *in-vivo* and *in-vitro* environments. Insufficient consideration of the properties of DNA, RNA and cellular biochemistry inevitably leads to failure in the laboratory.

DNA can occur in several forms, typically either a single-stranded form (ssDNA) or a double-stranded form (dsDNA). As one might expect, dsDNA consists of two strands of ssDNA which form hydrogen bonds to one another and twist into a double-helical structure. Double-stranded DNA is very stable, though the enzymes and polymerases which process dsDNA do so by "unzipping" the dsDNA and operating on one or both of the composite ssDNA strands.

2.1.1 Composition

A single strand of DNA is a linear chain of nucleotides which consist of a sugar, a phosphate group and one of four bases: Adenine (A), Thymine (T), Cytosine (C) or Guanine (G). The sugars of the nucleotides form phosphodiester linkages to one another so that ssDNA (or any nucleic acid) resembles a series of bases projecting from a phospho-sugar backbone chain. The *DNA sequence* refers to the order of nucleotides as "read" from the 5' end to the 3' end of the phospho-sugar chain. The sequence is processed or "read" from 5' to 3' because the protein complexes which interact with ssDNA do so directionally, starting at the 5' end of the sugar backbone and proceeding in the 3' direction. The structure of DNA and the interpretation of "sequence" is depicted in Figure 2-1. While biochemically inaccurate, the terms 'nucleotide' and 'base' are often used interchangeably, and a nucleotide is frequently called by the name of the base which it contains. For the remainder of this document, 'base' will mean 'nucleotide' and 'adenine' will refer to a nucleotide containing adenine and so forth.

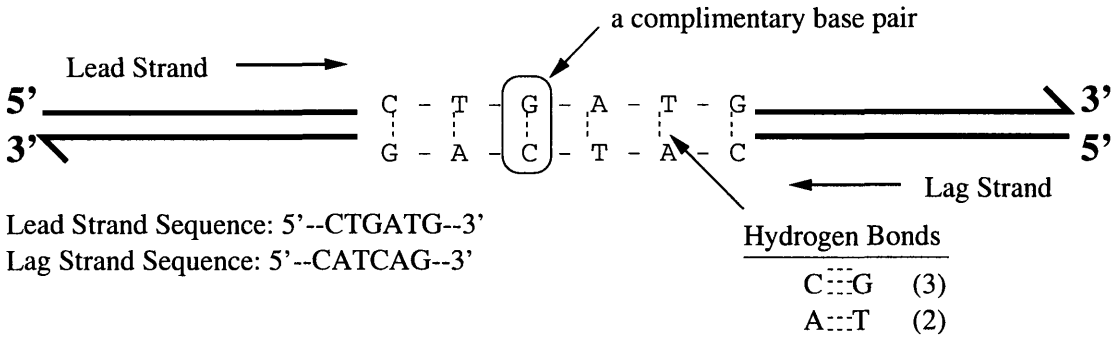


Figure 2-1: The Structure of DNA

2.1.2 Structure

Double-stranded DNA will spontaneously form whenever two *reverse complementary* strands of ssDNA are sufficiently near to one another at an appropriate ambient temperature.[1] "Reverse complementarity" is defined as follows: adenine is reciprocal to thymine, cytosine is reciprocal to guanine and if one strand is aligned 5' to 3' then the reverse complementary strand is aligned from 3' to 5'. Reverse complementarity is illustrated in Figure 2-1. When written, the top strand of dsDNA is called the "lead strand" and the bottom is the "lag strand". The enzymes which read dsDNA do not inherently distinguish between the lead and lag strands, they merely read one or the other strand from 5' to 3'; in this way, dsDNA can have a very high information density. In prokaryotes, it is quite common to see a single piece of dsDNA code for multiple proteins depending on the direction in which it is transcribed. [14]

The formation of dsDNA occurs when complementary base pairs between ssDNA strands undergo hydrogen bond formation: adenine forms two hydrogen bonds with thymine and guanine forms three hydrogen bonds with cytosine. The process by which two strands of ssDNA combine into a single strand of dsDNA is *annealing* or *hybridization* and the reverse process is *denaturation* or *melting*. The larger the number of hydrogen bonds in a given dsDNA strand, the higher the melting temperature. The detailed kinetics of DNA hybridization will be discussed later on.

[1] This behavior is critical to whole gene synthesis. If ssDNA will automatically form hybridized structures, all one must do to create desired dsDNA is design fragments of ssDNA which assemble in a known and well-behaved manner.

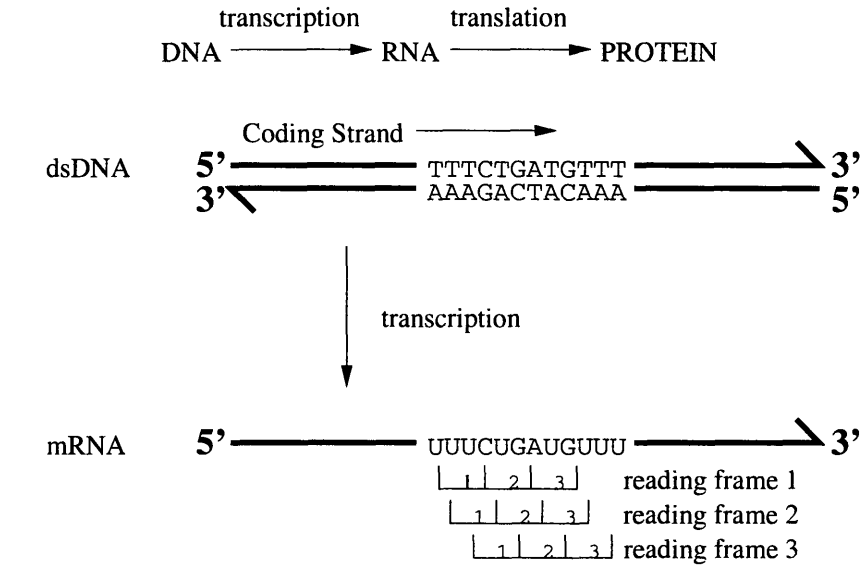
2.2 DNA to RNA to Protein

Researchers in synthetic biology frequently optimise genes when transporting a system between organisms or in order to adjust protein production levels. [25] These optimisation techniques exploit information redundancy in *translation*, the interpretation of RNA into protein, which I now will discuss in brief.

RNA is constructed from a different set of nucleotides than DNA. Where a DNA coding sequence will have Thymine, the corresponding RNA sequence will have Uracil. For example, the DNA coding sequence 5'-A**T**GAC**TT**CAGA-3', when transcribed into messenger RNA (mRNA), results in the sequence 5'-A**U**GAC**UU**CAGA-3'. During translation, a ribosome binds an mRNA sequence and reads three adjacent nucleotides at a time; this set of three adjacent nucleotides is referred to as a *triplet* or *codon*. For each of the 64 possible codons there is a corresponding transport RNA (tRNA) somewhere in the cell's cytoplasm that associates each that codon with a specific amino acid (AA). When translating mRNA into protein, the ribosome reads a codon, selects the tRNA corresponding to that codon, strips the amino acid off of that tRNA, adds that amino acid to the growing peptide chain, releases the tRNA and moves on to the next codon. When the ribosome reaches a termination point on the mRNA, or if the ribosome stalls or falls off of the mRNA, translation ends and the peptide chain is released to be processed into a protein. The process of translation is illustrated in Figure 2-2.

2.2.1 Codon Usage

There are 4^3 or 64 possible codons and only 20 amino acids in a typical organism, so several triplets can code for a single amino acid. This redundant mapping from codon to amino acid is generally referred to as the "genetic code". The genetic code is not universal, however. Between organisms, the same triplet may code for different amino acids. [27] The number of tRNAs present in the cytoplasm for a given codon also varies, meaning that one codon may be preferentially used over another based on the availability of the corresponding tRNA. This unequal use of synonymous



	codon 1	codon 2	codon 3	AA chain
reading frame 1:	UUC	UGA	UGU	...ser–stop
reading frame 2:	UCU	GAU	GUU	...ser–asp–val...
reading frame 3:	CUG	AUG	UUU	...leu–met–phe...

Figure 2-2: Translation and Transcription

codons in an organism is called "codon bias". [25] Codon bias can be determined by examining codon frequency patterns in a given genome; highly used codons generally have correspondingly high tRNA levels. [25] A number of codon frequency tables are freely available online [27] and we have have been included the table for *Escherichia coli* in Appendix A.

Current research indicates that codon bias strongly controls the kinetics of translation. Highly expressed genes tend to contain frequently used codons and the presence of rare codons can have a negative effect on gene expression or cause ribosomal stalling and the premature termination of translation. [25]

When a synthetic biologist optimises a gene to, for example, maximize protein production, he may do so by changing the DNA sequence so that the resulting mRNA contains a larger fraction of high-frequency codons for the same output amino acid sequence. This optimisation is made possible by the redundancy of the genetic code and is informed by the codon use frequency information for the target organism. If a given DNA sequence codes for a gene or contains an open reading frame (ORF)

for a protein, this same codon-substitution technique can be used to optimise that fragment of DNA for whole gene synthesis. One would use codon-substitution to do one of two things: to adjust the T_M of an oligonucleotide; to eliminate repeat sequences.

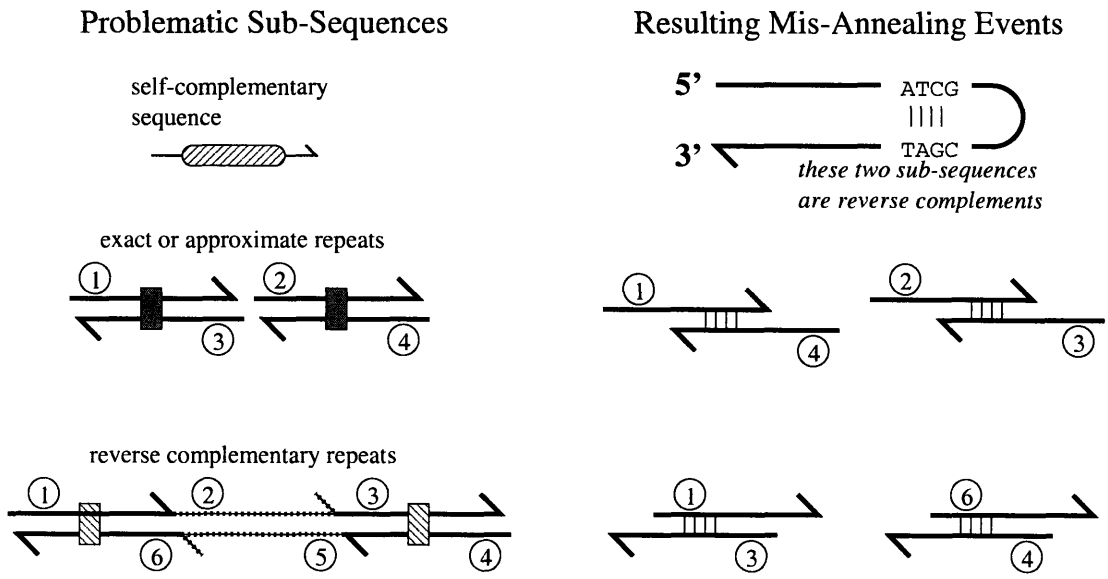


Figure 2-3: The different mis-annealing events which can occur during WGS.

2.3 Determining Melting Temperature

DNA melting temperature (T_M) is a statistical measurement used to determine protocols for DNA digests and ligations in addition to quantifying DNA stability. T_M is the temperature at which 50% of the DNA strands in a homogenous solution become thermally denatured. Denaturation occurs when increased thermal energy induces the dsDNA to uncoil and the hydrogen bonds between the two complementary strands break. There is an analagous measure, annealing temperature (T_A), which refers to the temperature at which 50% of strands in a solution have undergone duplex formation.

The melting temperature of a given DNA duplex can be determined using the solution salt concentration, pH, and a number of coefficients which represent the nearest-neighbor entropy of the different base pairs as determined by Santa Lucia et.

al. We use the salt-corrected nearest-neighbors method to predict T_M of different primers and primer overlaps, assuming that only perfect duplexes will form.[23] This quantitative model for DNA duplex stability is accurate and widely accepted as, although it is not the only such model. We include the equations for calculating T_M and the tables of nearest-neighbor parameters in Appendix A.

Unfortunately, while significant research has been done to evaluate the melting temperature of imperfect duplexes [2] [5] [17], the results to date have not produced a quantitative model for estimating the melting temperature of arbitrary hybrid duplexes. The lack of a quantitative model is unfortunate because the undesired formation of incorrect duplexes is a one of the problems that plagues whole gene synthesis. If a method existed to accurately predict the T_M of arbitrary mismatched duplexes, it would eliminate a significant amount of guesswork in primer design.[2]

2.4 Determining Codon Substitution Patterns

By combining the codon usage table with an accurate method for predicting T_M of arbitrary sequences, we can *rank* the codon usage table to reflect how particular codon substitutions will bias the melting temperature of a sequence. This means that a researcher, or a primer design program, can make informed choices for codon substitutions in order to adjust the T_M of an oligonucleotide.

One can construct the thermally ranked, or thermally biased, codon usage table by evaluating the melting temperature of each codon for the 16 combinations of flanking nucleotides.[3] One could calculate and rank the 1024 codon contexts by hand by using a script.[4] An example of a thermally biased codon usage table for *Escherichia coli* can be found in Appendix A.

[2]The relevance of this observation will be apparent in Chapter 3.3, where we address how to prevent undesired duplexes from forming during gene assembly.

[3]*(5' flanking nucleotide, 3' flanking nucleotide)*: (A,A) (A,T) (T,A) (T,T) (A,C) (C,A) (A,G) (G,A) (T,C) (C,T) (T,G) (G,T) (G,G) (G,C) (C,G) (C,C)

[4]One could script in BASH, PERL, PHP, C++, MATLAB or any number of programming languages.

2.5 Hybridization Kinetics

The process of annealing and denaturing DNA is passive and can be controlled by varying pH, salt concentration, DNA concentration, the similarity of the two strands and the ambient temperature [23], [6]. If the thermo-kinetic conditions permit, ssDNA will opportunistically bind to complementary sequences on other strands or will self-bind to complementary sequences on the same strand. Ways in which DNA can bind to itself are shown in Figure 2-3. Two pieces of DNA which are *sufficiently* complementary, as opposed to *perfectly* complementary, will also anneal to one another, albeit with different thermal kinetics. [20] DNA's promiscuous self-adherence is the primary problem facing the primer assembly method of whole gene synthesis. A primer, or oligonucleotide, is a short piece of ssDNA that is typically manufactured or synthesized in lab. In order for a set of primers to self-assemble in solution, as illustrated in Figure 1-1, all of the primers/oligos must denature/anneal under the same conditions and there must be exactly one way in which the primers can anneal to one another.

The problem of ensuring unique assembly is made more complex by this fact: that both perfect and mismatched primer pairs can hybridize to form a dsDNA duplex. The primary property which differentiates a mismatched duplex from a perfect duplex is the thermal stability of the dsDNA, which is commonly qualified by the temperature at which the duplex forms and denatures, referred to as the annealing/melting temperatures. What results is the critical constraint that all target primer pairs in a given whole gene assembly set strictly conform to a narrow range of acceptable thermodynamic parameters, as determined by the lab techniques used to manufacture and purify the primers and the techniques used for the primer assembly process. Assuring that this criterion is met in an optimal fashion requires extensive analysis of the target DNA strand, construction of a complete set of potential primers and an exhaustive primer selection process to build a final primer assembly array. Establishing a method by which this can be done was the main goal of this thesis.

2.5.1 Dynamics of Annealing and Denaturation

Despite the existence of many computational models for DNA hybridization dynamics[5], the understanding of how DNA anneals and denatures is primarily phenomenological. At its simplest, if one increases the ambient temperature above a duplex's melting temperature, T_M, the thermal energy is enough to overcome the hydrogen bonds which hold that duplex together. Similarly, if starts with the component strands of a potential duplex and decreases the ambient temperature below the potential duplex's annealing temperature, T_A, the two strands will spontaneously hybridize. Annealing and denaturation is a statistical phenomenon, but the exact physical dynamics of the process are not well-understood.

For example, current research suggests that the nucleotides at the end of a duplex are more important than nucleotides in the center. [?] There is not a consensus as to exactly how much the end nucleotides contribute to the overall thermal stability of a duplex, however. Additionally, the thermo-kinetics of annealing and denaturation for a duplex change if one is annealing together two offset sequences with dangling ends, much like our primers. The kinetics also change if one affixes one strand to a substrate and allows the complementary strand to be free in solution. [13]

2.5.2 The Thermal Properties of Mismatched Assemblies

As mentioned earlier, extensive research has been directed at determining the thermal stability of DNA duplexes with a single base mismatch and hairpin structures, but that is the extent to which the thermal properties of mismatched assemblies have been characterised. At present, no body of work exists which proposes a method of determining the melting temperature of a hybrid duplex with *more than one* base mismatch, or with a number of base deletions on one strand.

The consensus is that as few as one mismatch can significantly effect duplex melting temperature. Every mismatch represents a base location where hydrogen bonds can not form between the strands, making the duplex unstable and consequently low-

[5]The nearest-neighbors method of predicting duplex stability is among these.

ering its melting temperature. Particular mismatches may cause steric interactions between the two strands, further decreasing both its stability and melting temperature. [6]

Without a quantitative model for the dynamics of hybrid duplex formation, we can not hope to model the potential mis-annealing events which may occur during whole gene synthesis. Instead, the best we can do is identify any and all sub-sequences which will be prone to mis-annealing and somehow ensure that all potential mis-annealing events have a melting temperature which is well below the range of temperatures used in the primer assembly process.

2.6 Problematic Repeat Subsequences

The kinetics of DNA hybridization allow imperfect duplexes to form at a range of temperatures. These mismatched assemblies are most likely to occur when two different strands each contain a similar or exactly repeated subsequence. The following types of repeat structures will potentially interfere with WGS assembly:

Long runs of single bases	...AAAAAAAAAAAAA...GGGGGGGGGGG...
Long runs of tandem repeats	...ATATATATATA...ATGATGATGATGATG...
Exactly repeated subsequences	...ATGCTGA...ATGCTGA...
Approximately repeated subsequences	...ATGCTGA...AGGCTTGA...
Exact reverse complimentary subsequences	...ATGCTGA...TCAGCAT...
Approximate reverse complimentary subsequences	...ATGCTGA...TCAAGCCT...

Figure 2-4: The varieties of problematic repeat subsequences. Adapted from [20].

Each of these repeat types are problematic. Long strings of single bases will cause ribosomal slippage, and long strings of tandem repeats can result in mis-aligned annealing [25]. Exact and approximate repeats of any sort that are on the length order of a primer overlap can result in primer substitution. Reverse complimentary repeats can also result in primer substitution and reverse complimentary subsequences which are adjacent can form hairpin structures. These problems were illustrated previously in Figure 2-3 and must all be considered when designing primers for WGS assembly.

[6] A highly stable duplex will be able to resist significant amounts of thermal vibration. Therefore melting temperature tracks duplex stability.

Chapter 3

Oligo Purification And Whole Gene Synthesis Techniques

Many of the restrictions on DNA primer design for whole gene synthesis originate with the physical behavior of DNA; the remainder come from the lab protocols which a researcher uses to purify his manufactured oligos and the techniques by which he will attempt to assemble the oligoprimers into a complete gene. This chapter presents a brief summary of oligonucleotide purification and assembly protocols and highlights those aspects which effect WGS primer design.

3.1 Oligonucleotide Purification

The oligonucleotide synthesis procedure is far from error-free, but all of the component oligonucleotides for a whole gene synthesis attempt must be correct. Any errors in the oligos, particularly errors which are only present in some small fraction of oligos, will result in hard-to-detect errors in an unknown fraction of the final WGS product. Consider an example WGS array with 50 oligoprimers. Due to oligonucleotide manufacturing errors, 5% of the 37^{th} oligoprimer batch contains an A $\rightarrow$ T substitution. As a result, 5% of the genes produced by our WGS assembly protocol have an A $\rightarrow$T substitution in the middle region which corresponds to the 37^{th} primer. Although we can sequence the product of the WGS assembly protocol, we are not guaranteed to

find this one base substitution, although it is present in 5% of the genes.[1]

Synthesis errors – base deletions, insertions and substitutions – are catastrophic in synthetic biology. One missing base in the middle of a gene will result in a frameshift mutation[2]. A base substitution in the promoter region of a gene can interfere with the regulation of that gene's expression. A researcher's options for dealing with such errors are limited. One may either purify and error correct the synthetic genes that result from WGS assembly with faulty primers[3], or one can ensure that all of the component WGS oligonucleotides are perfectly correct before assembly. A third and less rigorous alternative exists, which is to ligate all of the synthetic genes into plasmids and transform those plasmids into cells. The synthetic genes in those cells which survive and exhibit the desired characteristics are sufficiently correct.

In a perfect world, we could manufacture perfect oligonucleotides which could be assembled into perfect synthetic genes. In reality, the oligos which one can buy from a large manufacturer invariably contain errors[4] due to the non-zero error rate of the solid-phase DNA synthesis process. The first step in whole gene synthesis is therefore to purify out all errors in all of the component oligonucleotides including single base deletions, single base insertions and single base substitutions. The limitations of existing purification techniques, in turn, dictate what kind of oligoprimers one can use for whole gene synthesis.

3.1.1 Techniques of Purification

Nearly all oligo purification techniques are variations on gel electrophoresis. *Electrophoresis* is the migration of charged molecules in an applied electric field; nucleic acid gel electrophoresis is the migration of negatively-charged single-stranded oligonucleotides through a gel or similar matrix in an applied electric field. Nucleotides have

[1] Gene sequencing is a statistical analysis, and even with the errors, the product of the WGS assembly is *statistically* correct, even if it is not *perfectly* correct.

[2] A nucleotide sequence is translated into an mRNA sequence and then "read" in one of three possible frames to build an amino acid chain or protein. A frameshift mutation is where a base deletion or insertion shifts the reading frame of the ribosome. This results in the expression of an incorrect and potentially non-functional protein.

[3] A very time-consuming task.

[4] Most commonly, mass-manufactured oligonucleotides will contain single-base deletions.

a fixed negative charge per unit length, and the force applied to a molecule in an electric field is directly proportional to that molecule's charge. Consequently, gel electrophoresis separates oligos purely on the basis of sequence length; longer oligos migrate more quickly through the gel for a given applied field.

The two most common electrophoretic media are agarose gels, which are porous and good for sorting very large molecules, and polyacrylamide gels, which are dense and used for high-resolution purification of short oligonucleotides. [4] One achieves different resolutions on gel electrophoresis by varying the gel medium and the strength of the applied field. There are electrophoresis protocols which claim to able to distinguish between an N-mer and an N-minus-1-mer, for an N on the order of 50-100 nucleotides. [1] [2]

While very high resolution can be achieved with certain polyacrylamide gels, such protocols are difficult to run consistently and can only catch base deletion or insertion errors. In order to have correct primers for whole gene synthesis, a user must be able to purify a correct oligo from an oligo with a single-base *substitution.*

Statistical Purification Techniques

Current research demonstrates that one can differentiate between oligos with single base deletion and substitution errors by means of statistical purification techniques. These techniques are a variant on gel electrophoresis, or capillary electrophoresis, where the electrophoretic medium is impregnated with the complement of the oligo one wishes to purify. [1] [2] [11] [16] [19], [26] [24] Although any one oligo in the matrix of complements is likely to contain an error, on the whole, the matrix of complements is statistically correct. If one flows a batch of synthetic oligos over this matrix of complements, the correct oligos will bind more readily to the matrix and incorrect oligos will flow by more quickly. Because this kind of purification technique relies on the DNA annealing kinetics to discriminate between correct and incorrect oligos, the resolution one can attain will be strongly dependent upon the ambient temperature. If one wishes to purify oligos *en masse*, such as on a plate or on a micro-fluidic array, then the operating temperature of the purification reactions would be uniform.

This in turn requires that all of the oligos for whole gene synthesis exhibit a uniform melting temperature characteristic that is compatible with the purification protocols.

3.1.2 Purification Requirements for Primer Design

Each purification technique places its own restrictions on oligonucleotide length and target melting temperature. Because we seek to purify a set of oligoprimers for whole gene synthesis, this requires the melting temperature of the primer solution set to have a low variance around some user-specified mean value. In order to ensure that all WGS primers are compatible with given purification techniques, the user must be able to specify the acceptable range of primer lengths, as well as the variance and mean value of the primer T_M.

3.2 Whole Gene Synthesis Techniques

All whole gene synthesis techniques are variations on a theme; the goal is to manufacture long sequences by splicing together short oligonucleotides. Each technique imposes a different set of restrictions on the component primers, however. We therefore present a brief summary of WGS techniques in order to motivate compatible primer design.[5]

3.2.1 Primer Extension

The fundamentals of the primer extension method are illustrated in Figure 3-1.

The process begins with a support-bound oligonucleotides in solution. The primers are added one at a time, allowed to anneal, then unbound primers are washed away. During this process of iterative primer addition, the ambient temperature is held constant.[6] Once the entire sequence is assembled, the primers are ligated together and the product is eluted from off of the substrate. [14]

[5]Many other WGS techniques exist, but these three are predominant.

[6]If the temperature were cycled, then the oligos annealed to the substrate would denature and wash away or might form intermediate products that would not be anchored to the substrate.

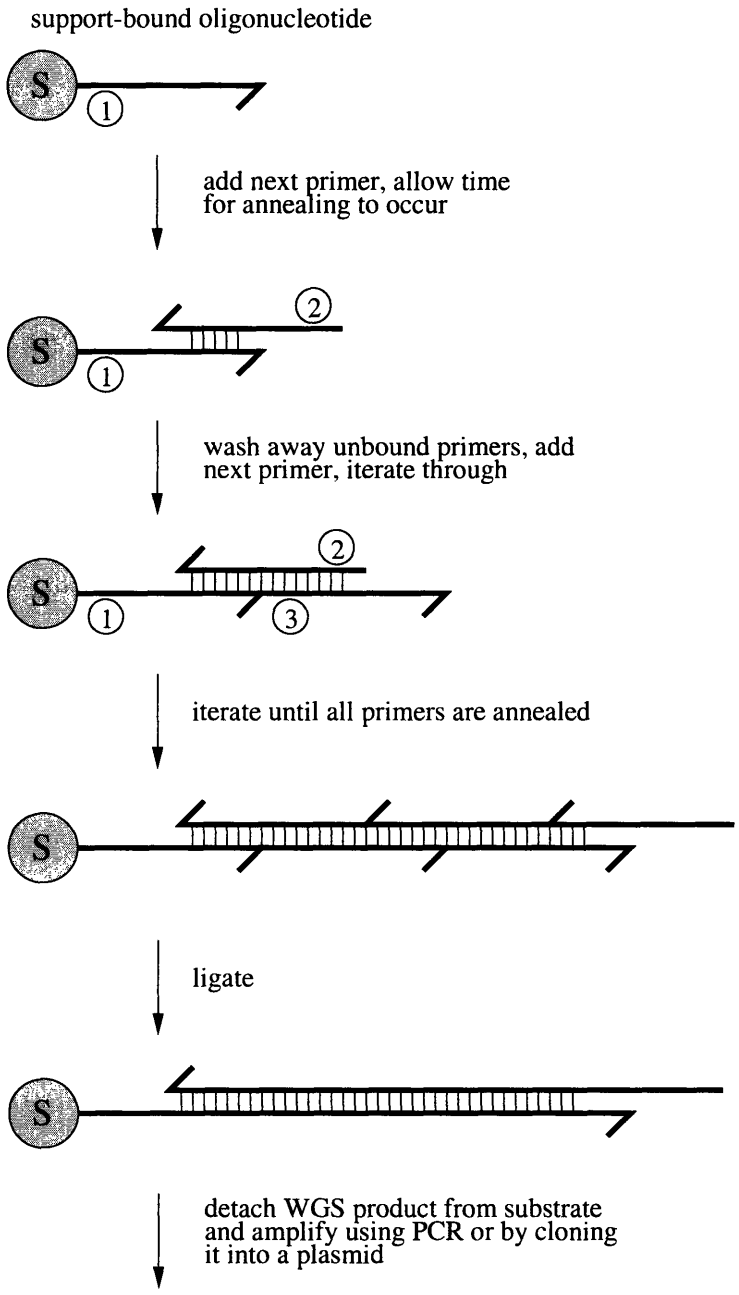


Figure 3-1: The primer extension method of WGS.

Primer extension assembly avoids the problem posed by repeated sub-sequences by only allowing one annealing event to occur at a time. However, it requires that all of the primer *overlaps* have a very uniform melting temperature.

3.2.2 Ligase Chain Reaction

In the ligase chain reaction, all the component primers are pooled together and thermally cycled to induce self-assembly. Once self-assembly occurs, the primers are ligated together and then the product is amplified. This process is illustrated in Figure 3-2. [3]

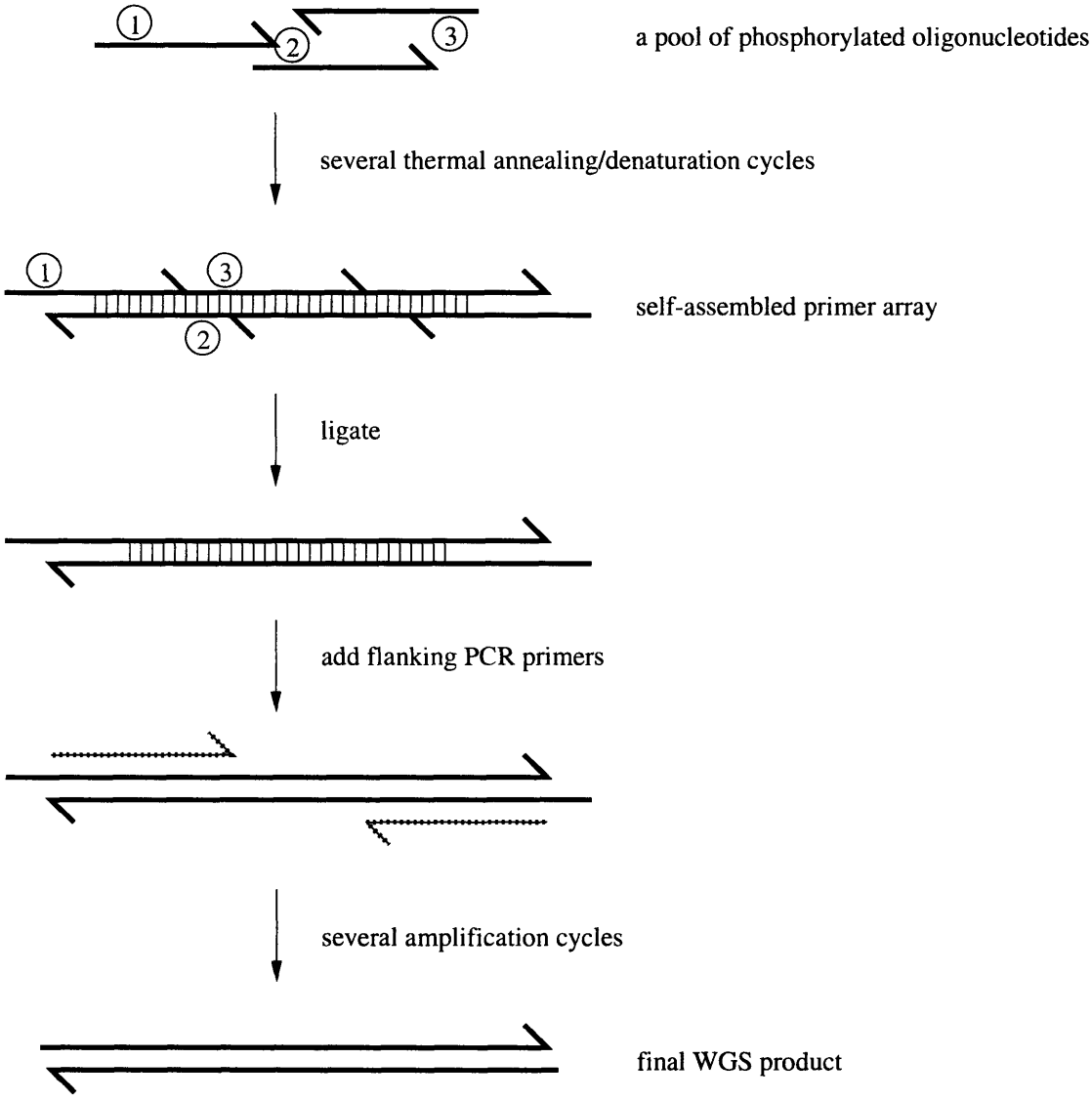


Figure 3-2: The Ligase Chain Reaction method of WGS.

Because all of the primers self-assemble simultaneously under the same thermal conditions, LCR assembly requires a very uniform primer overlap melting tempera-

ture. The process is also prone to mis-annealing events. All primer overlaps must therefore be unique enough to ensure that the melting temperature of a mis-annealing event falls well below the range used in the LCR protocol.

3.2.3 Polymerase Cycling Assembly

Polymerase cycling assembly also begins with a pool of component primers. This pool cycled through the steps of the polymerase chain reaction: denaturation; annealing and extension. After enough iterations, this results in a pool of intermediate oligonucleotides which can no longer extend. By adding extra flanking primers and continuing the PCR cycle, however, the intermediate oligonucleotides can be extended to form the full-length target gene.[7] PCA is illustrated in Figure 3-3. [25]

Like LCR, PCA is very sensitive to repeated sequences and mis-annealing events. It is additionally sensitive to regional variations in T_M across the DNA sequence. If the melting temperature varies too much over the length of the target sequence, it becomes problematic to form longer intermediate extension products. [25]

3.2.4 Summary of WGS Requirements

In aggregate, all whole gene synthesis protocols place the following restriction on WGS primers: all primer overlaps must be sufficiently unique disallow mis-annealing; the T_M of each primer overlap must conform to a narrow range. Note that the assembly protocols do not restrict the *primers* but the *primer overlaps.*

3.3 Requirements for a WGS Primer Design Tool

Now that we have a solid understanding of the physical constraints placed upon DNA primers by oligo purification and WGS assembly techniques, we can outline the high-level requirements for a WGS primer design tool. Any robust primer design tool will

[7]This assumes that the intermediate oligonucleotides were correct and that no mis-priming events occurred.

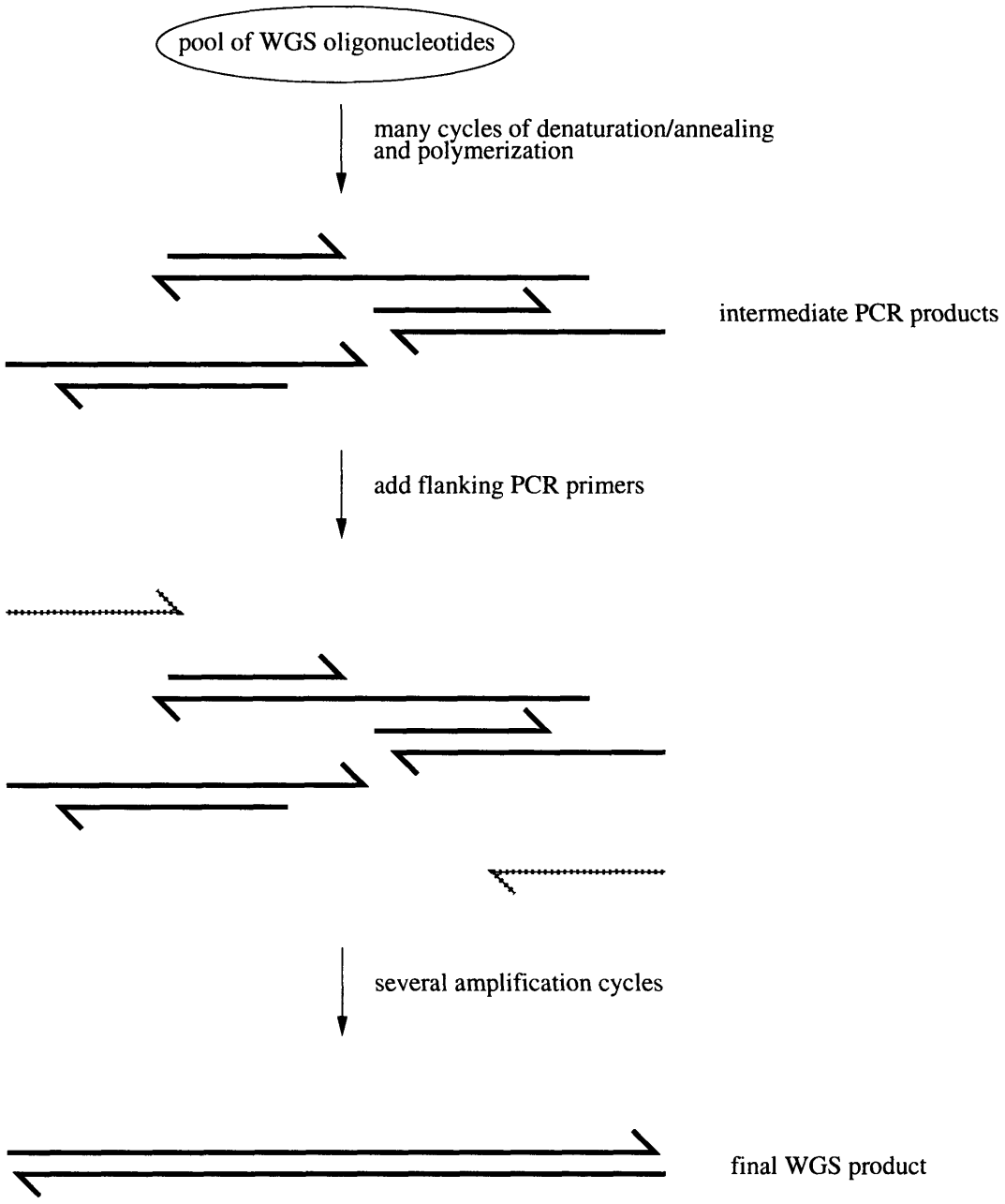


Figure 3-3: The generic Polymerase Cycling Assembly method of WGS. Adapted from [25].

allow the user a certain amount of control and flexibility while simultaneously limiting the user's ability to interfere with the design process.

What follows is a walk-through of the required user input parameters and internal design constraints for our approach to WGS primer design.

3.3.1 User Input

The user should be able to control the following program parameters:

1. **Target DNA Sequence**

   The user must be able to provide a target DNA sequence. He should be able to specify either a specific nucleotide sequence or a target amino acid sequence.

2. **Target Host Organism**

   The user must specify the target host organism for the synthetic gene if he inputs an amino acid sequence or if he wants to allow the program to optimise the sequence. *Escherichia coli* will be used by default.

3. **Primer Length Range [min, max, absolute]**

   The user must specify the minimum and maximum acceptable lengths for the WGS primers, according to the restrictions of his purification protocols. He may alternately constrain all primers, excepting the end primers[8], to some absolute uniform length.

   The acceptable primer *overlap* lengths are derived from the primer length range. The minimum primer overlap length is half of the minimum primer length, rounded up. The maximum primer overlap length is half of the maximum primer length, rounded down. This same formula applies when the primer maximum and minimum lengths are equal and absolute.

   A WGS primer design tool also needs to have its own restrictions on acceptable values for primer length. If the user attempts to build a gene with too-short primers, increasing the likelihood of unavoidable mis-assembly, the tool should flag the user and recommend a sub-assembly WGS technique.

4. **Primer T_M Range [mean, +range, -range]**

[8]There needs to be some flexibility in the length of the end primers, as a target sequence will not always be a clean multiple of a primer length. This is dealt with in practice by adding extra "dummy sequences" to potentially short end-primers. The dummy sequences are removed when the assembled gene is amplified using PCR techniques.

The user may specify a target primer melting temperature and the range of thermal variation allowed by his purification procedures. In order to accurately determine the actual T_M of the resulting primers, the user must also specify the solution [Na+] and pH of his protocols. From these parameters, the salt-corrected nearest-neighbors method [23] will be used to evaluate primer melting temperature.

If a target value of T_M is not specified by the user, the program will default to a mean T_M as determined by a cursory analysis of the input sequence and the user's protocol conditions. The default acceptable range will be +/-2°C. In this instance, both the calculated default mean T_M and the range must be reported to the user.

5. **Demarcation of Open Reading Frames**

   The user may specify ranges within the target DNA sequence which correspond to protein Open Reading Frames (ORFs). ORFs must be a multiple of three nucleotides in length, must begin with a valid methionine codon for the selected host organism and must end in a valid stop codon. If approved to do so, the primer design tool will modify the nucleotide sequence within ORFs[9] to increase the likelihood of finding a set of primers for WGS, or to attempt to make the set of primers optimal for WGS assembly.[10]

6. **Demarcation of Non-Optimisable Sub-Sequences**

   The user may additionally specify ranges within the DNA sequence which the program is not allowed to modify when optimising the sequence. This is a critical option to include in the event that the user's synthetic gene has already been designed with secondary mRNA structures in mind.

7. **Repeat Sub-Sequence Tolerance [min_size = N_R, threshold = L_D]**

   As mentioned in Chapter 2, a strand of DNA may contain any number of exact and relative repeat sequences. The most common kind of repeat one might

[9]Optimisation will modify the nucleotide sequence while preserving the amino acid sequence.
[10]The problem of optimisation is discussed at length in Chapter 4 and Chapter 5.

expect to see would be repeated codons, such as several instances of 'ATG' coding for methionine. Both exact and approximate repeated sub-sequences can result in priming mistakes in WGS assembly, as discussed earlier. Repeated sub-sequences represent a very complex phenomenon which biology is currently unable to accurately model. For the sake of simplicity, we abstract away the details of the different kinds of repeats and distill the problem down to two criteria: the minimum length which defines a repeat and how *similar* two sub-sequences must be in order to be considered *approximate* repeats.[11]

The need for a minimum size limit, N_R, on a repeat originates with how repeats are detected in a nucleotide or amino acid sequence. Most gene analysis and design tools use the BLAST family of algorithms [18] to find repeated subsequences. The BLAST algorithm uses a hash to create an exhaustive database of subsequences within a sequence or genome and searches that database for exact subsequence matches of some minimum length. Once an exact match is found, BLAST extends the boundaries these repeats until, by some threshold measure, they are no longer homologous sequences. BLAST is properly used to find homologous protein/amino acid sequences across entire genomes and between organisms. Some BLAST variants will fail to successfully find matches that are shorter than the order of 22 nucleotides[12], although new versions which support short nucleotide searches are available. BLAST has significant computational and memory overhead and is unnecessary[13] when analysing relatively short nucleotide sequences.

Rather than resorting to BLAST, we focus on using standard[14] string analysis tools to identify exact and approximate repeats in the target DNA sequence, specifically STRSTR(*search_string*, *sequence*) and LEVENSHTEIN(*string1*,

[11] Recall that both exact and approximate repeats can result in WGS mis-assemblies.

[12] This is because the older BLAST algorithms use a block hash, rather than a sliding hash, to create a the database of sub-sequences and save on space. If a search sequence spans two blocks, the algorithm will have difficulty identifying it as a match.

[13] *A.K.A.* Overkill.

[14] "Standard" refers to well-developed string manipulation libraries and functions for PERL, C++, JAVA, PHP, etc. These functions are commonly used in natural language processing.

string2).

The function STRSTR takes in a search string and a sequence and returns the location of the first instance of the search string in the sequence. This can be iteratively applied to build an exhaustive table of all exact repeated sub-strings for a given input.[15] As a string tool, STRSTR does not discriminate between nucleotides or amino acids; it can be used to find repeated sub-sequences at either levels. BLAST, on the other hand, is typically amino acid or nucleotide specific.

The function LEVENSHTEIN is commonly used in natural language processing. It calculates the Levenshtein Distance between two strings, L_D, which is the number of insertions, deletions and substitutions needed to make two strings equivalent [15]. We use it to define a threshold beyond which two subsequences are no longer sufficiently similar to be considered repeats of one another.

The user should be allowed to select both N_R and L_D based on his knowledge and experience with which types of repeats cause mis-priming events with his WGC protocols. We can also derive reasonable defaults for these parameters.

(a) ***minimum primer overlap length*** $\geq N_R \geq 4$: Searching for a minimum repeat size of three will result in the program flagging every instance of methionine[16] as a repeat.[17] However, one wants to set N_R to the smallest feasible value. We recommend looking for base repeats no shorter than 7-9 nucleotides long, or on the order of 2-3 amino acids.

(b) L_D **should scale with the *minimum primer overlap length.*** The hybridization kinetics of mismatched duplexes is not perfectly quantified, so there is no hard and fast rule for how similar primers must be such that mis-priming events occur in WGS. As the threshold L_D increases, more

[15]This function is implemented by the algorithm, FINDREPEATS, described in Chapter 5.

[16]There is typically only one codon for methionine, ATG.

[17]In our approach to primer design, this is problematic as we will use every identified repeat to limit our search space. If N_R is overly sensitive, it will affect our ability to find a solution primer array for whole gene synthesis.

dissimilar subsequences will be considered "repeats", so we want to chose the smallest possible threshold value of L_D.

Consider that our WGS assembly protocols already force the melting temperatures for all solution primer overlaps to conform to a narrow range. Then recall that DNA duplexes with mismatches are known to anneal with some thermal penalty. Even a single base mismatch has a significant penalty associated with it if the duplex is short enough. [14] Therefore, we choose the threshold L_D to be 10% - 20% of the minimum primer overlap length with the rationalization that a 10% dissimilarity results in enough of a thermal penalty that a mis-priming event won't occur during WGS assembly.[18]

3.3.2 Requirements of the Primer Design Strategy

Our approach to WGS primer design, described extensively in Chapter 4, comes with its own set of restrictions and input parameters. Whenever possible, one should provide the user with an input option to override the program's defaults.

1. **Hairpin Primers [min_size = N_H, offset = N_O, ignore_flag]**

   A hairpin is a secondary structure which forms when a piece of DNA contains both a sequence and its reverse compliment separated by 4-8 base pairs. If a primer contains a hairpin sequence, it will preferentially self-bind and disrupt the primer assembly process. [10] This is one example of DNA's problematic self-adherence, as discussed in Chapter 2, and should be avoided at all costs. In order to prevent a solution from containing any hairpinning primers, there must be a internal constraint that no two adjacent primer overlaps in the solution array may contain reverse-complimentary subsequences of a length, N_H, that is greater than 25% of the minimum primer length or that ends within N_O bases of either end of the primer.[19] Default values should be something like

[18]This assumption warrants confirmation via laboratory investigations.

[19]That is, if the sum total of bases in the hairpin exceeds 25% of the total primer overlap, it will

N_H = 25% of the minimum primer length and N_O = 3. One would want to allow a user to override these defaults based on his own working knowledge of hairpin formation kinetics.

However, if a user intentionally wants a particular primer to contain a hairpin sequence for whatever reason, he should be able to indicate this to the WGS primer design program so that the program's default constraints are suitably overridden. We allow this by including a boolean flag, *ignore_flag*. The result of setting this flag is to override the hairpin check constraint described above.

2. **Primer Overlap T_M Range [mean, +range, -range]**

   In all of the previously discussed WGS assembly techniques, the assembly protocol cycles over a narrow range of temperatures. If an overlap has a T_M outside of this functional range then mis-assembly is likely to occur. It is therefore necessary to ensure that all primer overlaps have a sufficiently uniform T_M characteristic while guaranteeing that the primers constructed from those overlaps will meet the user's criteria. We assume that a user will typically place strict restrictions on his primer T_M characteristics while having some degree of flexibility in his WGS assembly protocol. Therefore, rather than requiring the user to specify a separate overlap melting temperature criterion, we can *derive* the T_M constraints for the primer overlaps from the user's T_M restrictions for the primers.

3. **Primer Overlap Uniqueness Threshold [buffer_size]**

   In order to ensure that there is exactly one correct way in which a set of primers can assemble into a final sequence, all of the primer overlaps must be unique. A primer overlap is the fraction of the primer that is intended to anneal with a primer from the complementary strand, as illustrated in Figure 1-1. Each primer in the final solution array consists of two adjacent overlaps. It bears repeating

be considered likely to form a hairpin sequence. Additionally, the ends are much more important than the middle in mis-priming events [25] [5], therefore there must be N_O bases between the end of the hairpin structure and either end of the primer.

that this approach to WGS primer design is more correctly an approach to WGS "overlap design" that happens to later assemble the array of valid overlaps into an array of valid primers.

Ideally, all primer overlaps would be identically unique. However, because of the way genetic information is encoded in DNA using a small set of characters, one invariably finds repeated sequence elements. These repeated elements are problematic for whole gene synthesis because, as discussed in Chapter 2, DNA will form hybrid duplexes even if the two strands are only approximately matched. Therefore we need to establish a criterion by which all primer overlaps are "sufficiently unique" enough that mis-annealing events will not occur.

Luckily, in much the same way that researchers exploit thermal discrimination to purify out off-by-one and N-minus-1 sequences from perfect sequences, we can use thermal discrimination to prevent mis-annealing events.

Thermal discrimination assembly works as follows: one designs the set of overlaps so that the intrinsic melting temperatures of the entire set of primer overlaps conforms to a very narrow WGS operating range. Assume this set contains two similar overlaps which are prone to mis-annealing, as illustrated in Figure 2-3. If we can somehow guarantee that the mis-annealed duplex has a T_M well outside the WGS operating range then, then we can be confident that the mis-annealing event will not occur during WGS assembly. We will therefore *thermally discriminate* against the undesired annealing event by forcing the primer overlaps which contain the similar sub-sequences have "buffer zones" on either end.

The effect of the buffer zones of length *buffer_size* is to increase the thermal penalty against undesired duplex formation. Unfortunately, while significant research has been done to quantify the thermal effects of single and double base pair mismatches in DNA duplexes [24] [12], no quantitative model for the T_M of mismatched duplexes currently exists. We choose the default for *buffer_size* to be on the order of 4-8 nucleotides with no way to confirm that this is an

optimal default. Given that even single base mismatches can have a affect T_M for sufficiently short duplexes and the primer overlaps for WGS are short[20], however, we can be confident that this is a *sufficient* buffer size to ensure that even overlaps which contain exact repeats are sufficiently unique.

The buffering requirement has the additional effect that if the feature-identification protocols find any repeats which are longer than (*maximum_overlap_length* - 2**buffer_size*), the feature-driven WGS primer design tool will automatically fail.

3.3.3 Important Tradeoffs

The size of the table of valid overlaps for a given sequence will depend heavily on the user's choice of L_D, N_R and *buffer_size*. As one decreases the number of bases required in an exact substring match, N_R, the likelihood of a repeated substring occurring randomly increases. This consequently increases the fraction of repeats in a given sequence, decreasing the number of valid overlap end sites and subsequently the number of potentially valid overlaps. Increasing BUFFER_SIZE similarly decreases the number of valid overlap end sites, decreasing the size of the valid overlap data set. Increasing the permitted Levenshtein distance between approximate repeats, L_D, will increase the length of an average repeat, again decreasing the size of the valid overlap data set. These trends are illustrated in Figure 3-4.

It is critical to choose appropriate values for L_D, N_R and *buffer_size*. The reasons for this are two-fold. First, if these parameters are too strict we will over-constrain our search space, resulting in a null solution set. Second, if these parameters are too lax, our resulting search space may contain primer overlaps that will produce mis-assemblies if included in the final solution. One can derive the statistical dependence of the overlap search set on these three parameters

[20] On the order of 20-50 nucleotides.

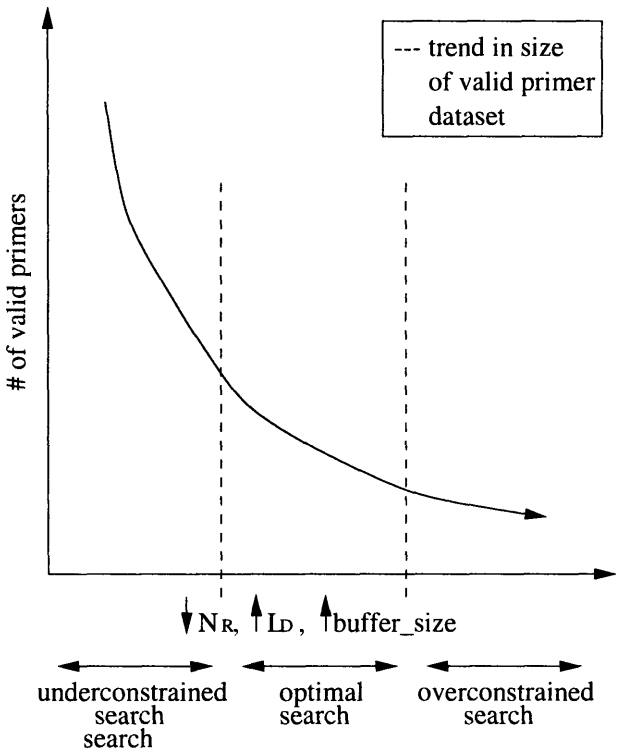


Figure 3-4: A plot of the theoretical tradeoffs for TVO size versus repeat detection thresholds.

by taking a large number of sequences[21], systematically varying L_D, N_R and *buffer_size* and recording the size of the valid primer overlap array that results for some standardized overlap T_M criterion. Such an analysis would characterize the upper bound on repeat detection, i.e. how strict our parameters can be before over-constraining the search space. Unfortunately, characterising the lower bound on repeat detection is not as easy. The only way to determine at which point our parameters are lax enough to produce solution arrays which will mis-assemble is to find those solutions; manufacture the arrays and determine if mis-assembly occurs. To properly characterize the lower bound, one would need to take thousands, if not hundreds of thousands, of data points – amounting to decades of laboratory work.[22]

[21] A good source of sequences is the NCBI GenBank: http://www.ncbi.nlm.nih.gov/Genbank/index.html

[22] This is yet another reason why having a quantitative model of mis-annealing events is so important. Having one accurate, central model would obviate the need to manually characterize the performance of different primer design regimes. This would, in turn, accelerate the rate of development of intelligent primer design tools.

Whenever possible, a user should choose values for L_D, N_R and *buffer_size* that match with his, or his discipline's, current understanding of mis-annealing kinetics.

Chapter 4

A Systematic Approach to Primer Design

We have already presented the background and factors one must consider when attempting to design technique-compatible DNA oligonucleotides. In this section, we address these considerations from a high-level perspective and present a coherent primer design strategy.

Although we have already introduced many of the following terms, a quick redefinition of terms should make reading this section a little easier. Additionally, we assume that the reader has some basic familiarity with artificial intelligence search techniques.[1]

- **Primer** : A short piece of DNA. Primers can either be "lead strand" or "lag strand" primers. Similarly, two primers can either be "same-strand" primers or "opposite-strand" primers.

- **Overlap, Primer Overlap** : The sub-sequence defined by the overlapping regions of two complementary opposite-strand primers.

- **Order** : The rate at which a data set or algorithm grows. Notation is: **O**().

[1]For a good reference, see [22].

- **Primer Data Set** : The set of all potential primers for a given DNA sequence and design regime.

- **Primer Array** : A complete collection of primers which can be used to construct a DNA sequence.

- **Primer Position** : A given primer's location, or index, in the primer array.

- **Solution Set** : The set of all possible primer arrays, or the set of all *potential* solutions. Not all potential solutions are *valid* solutions.

- **Solution** : A primer array from the solution set which satisfies all of the user's requirements. The goal for any design tool is to provide the best, if not optimal, solution. By definition, a solution is valid.

- **Restriction** : A limitation or requirement from a user.

- **Constraint** : An absolute requirement on a program variable or state, such as a solution. Used to determine the validity of a variable or state.

- **Heuristic** : A ranking or preference criterion.

- **Conflict** : Two primer overlaps, or two same-strand primers, which span the same nucleotides are said to conflict.

- **Gap** : Any portion of the target DNA sequence which is not spanned by a valid primer overlap.

4.1 Primer Design Regimes

There are two general approaches to primer design for whole gene synthesis. These are: fixed-length oligonucleotide design, and sequence feature driven design. Fixed-length oligonucleotide design is very simplistic; the choice of one primer defines the final primer array and the set of all potential arrays is well-defined and finite in size.

In fact, if the target oligonucleotide length is N, and the length of the total sequence is S, then the size of the primer set is $\mathbf{O}(S)$ and the size of the solution set[2] is $\mathbf{O}(N)$.

In contrast, feature-driven design first implies that one can have oligos of varying lengths. If the median target primer lenth is M and the difference between the maximum and minimum primer length is D, this added complexity increases the order of the primer data set from $\mathbf{O}(S)$ to $\mathbf{O}(2DS)$. While this increase seems manageable, the size of the solution set increases to $\mathbf{O}(\frac{S}{M}^D)$ which is significantly larger than we can reasonably deal with. It is necessary to build a set of constraints based on relevant sequence features, such as the location of repeat sequences and domain melting temperature, to traverse the primer set, preemptively decrease the size of the solution set, and find a solution in a reasonable amount of time.

4.1.1 Fixed-Length Oligonucleotide Design

If one is restricted to using oligos of some fixed length, very little "design" is required. The choice of one primer defines all remaining primers in the final array, and there are $\mathbf{O}(N)$ choices available for each primer position. Therefore, a brute force approach can be used to find all possible primer sets. These sets can be evaluated and ranked according to some user-defined heuristic, sorted and presented to the user.

The fixed-length approach to oligonucleotide design prevails in the currently available primer design tools, such as Gene2Oligo and [CHURCH]. It is attractive because the finite size of the solution set enables one to quickly and exhaustively find the "best" solution which satisfies the user's constraints. Uniform length primers with uniform length overlaps are also more likely to have similar melting temperatures, as T_M is dependent on sequence length. [SANTALUCIA]. The disadvantage is that the fixed-length method may miss valid solutions to the primer design problem because it is so highly restricted.[3]

For a specific discussion of the exhaustive methods used to construct the primer

[2] A solution is an array of primers from the primer set which comprise the target strand

[3] Perhaps there are two repeat sequences which short enough that one *could* buffer them inside a primer overlap. If the two repeats are separated such that it's impossible to buffer both of them at the same time, the fixed-length method will find no solutions.

set and array set, and the heuristic methods used to evaluate the possible solutions, please refer to Chapter 5.

4.1.2 Feature-Driven Oligonucleotide Design

Feature-driven oligonucleotide design (FD) is the clever use of DNA sequence features to limit the size of the data set one must search to find a final solution. The most distinctive feature of FD design is that one is not designing *primers* so much as designing *primer overlaps.*[4] Primers and primer overlaps are defined and illustrated in Figure 1-1.

Feature-driven design first involves identifying those physical features of the target DNA strand which may adversely impact oligo purification or assembly. Relevant features include: repeated sub-sequences; long strings of tandem nucleotide repeats; sub-sequences with a melting temperature which is too high or two low; etc.[5] Once these features are identified, the DNA sequence is annotated with the feature information. The second step in FD design is to convert the user's requirements into a set of constraints which can be used to determine whether a given overlap is *valid.* These constraints are then applied to the annotated DNA sequence to prune the set of all *possible* primer overlaps into the set of all *valid* primer overlaps. Eliminating invalid options from a data set early on improves the efficiency of any search run on that data set. We preemptively save time and computation by not evaluating any solutions which contain invalid primers. [22]

The user requirements and identified sequence features are then used to inform an algorithm which searches the set of valid overlaps with the goal of constructing a solution array. The restrictions on a solution array are simple: it must consist of a set of valid primers which exactly reconstruct the target DNA sequence without gaps or overlapping same-strand primers.[6] This is a classic example of a *Constraint*

[4] A given primer consists of two adjacent primer overlaps. Please refer to Chapter 5 for more in-depth discussion.

[5] The set of features which are relevant is dependant on the user's requirements.

[6] As we are designing primer overlaps, the constraints on the array are that the overlaps must exactly cover the target DNA sequence without any gaps or redundant coverage.

Satisfaction Problem (CSP). Artificial Intelligence (AI) research already provides a number[7] of search techniques for finding "optimal" solutions to CSPs. [22]

Our approach is to assign each valid primer overlap a heuristic value based on its physical characteristics[8] This set of valid, heuristically ranked overlaps is then searched a CSP technique called *constraint propagation with forward checking.* [9]

The advantage of feature-driven design is that, assuming a user has some flexibility in the length of his primers, it has a significantly higher chance of finding an optimal primer array. Although there is substantial work involved a implementing a constraint satisfaction system, the beauty of CSP programming is that one only needs to build the system once. If a user needs to adjust the search – perhaps he only needs to avoid hairpin sequences or knows in advance that his target gene contains no repeated subsequences – then the constraint and heuristic definitions can be updated to indicate what kind of solution the system is trying to find (and how it will find the solution) without necessitating a change in the underlying program structure.

Requirements for Feature-Driven Primer Design

We now know enough to outline the required elements of a feature-driven primer design tool:

1. Definitions of requirements and sequence features which a user may want applied to their design situation.

2. Methods to process the user input into a set of heuristics and constraints.

3. Methods to identify and evaluate DNA sequence features according to user input.

4. A data structure to hold all relevant information about the DNA sequence and those identified features.

[7]For an in-depth discussion of AI search techniques and solving CSPs, see [22].

[8]The heuristic value indicates how suitable a given primer overlap is for a user's indicated purification and assembly protocols. A higher heuristic value reflects that a given primer is more suitable.

[9]This technique is an example of an "optimal" search technique.

5. Methods to parse the annotated data structure.

6. A method which applies the constraints to the data structure to generate the set of valid primers.

7. A constraint-satisfaction system which uses the previously created set of constraints and heuristics to search the generated set of primers for all possible solutions.

One must first construct a minimal representation of the DNA sequence along with feature annotations. This annotated data structure can then be used to find a final primer array for WGS by solving a search/constraint satisfaction problem.

User requirements, such as avoiding primers and primer overlaps with a T_M outside the acceptable range, or overlaps which do not properly buffer repeat sequences, are applied as unary constraints to prune the set of possible primers down to the set of valid primer *overlaps*. This valid overlap set is then traversed by a backtracking constraint propagation algorithm which uses nucleotide locations in the target strand as the variables and the set of primer overlaps which span that base position as the values.[10] Once a nucleotide has been accounted for in the solution array[11], all overlaps which conflict with that choice are eliminated and we check to make sure that the constraint propagation did not result in any one nucleotide having a null set of spanning overlaps. The next base position to evaluate is chosen using the *minimum remaining values* (MRV) heuristic. This is graphically depicted in Figure 4-1.

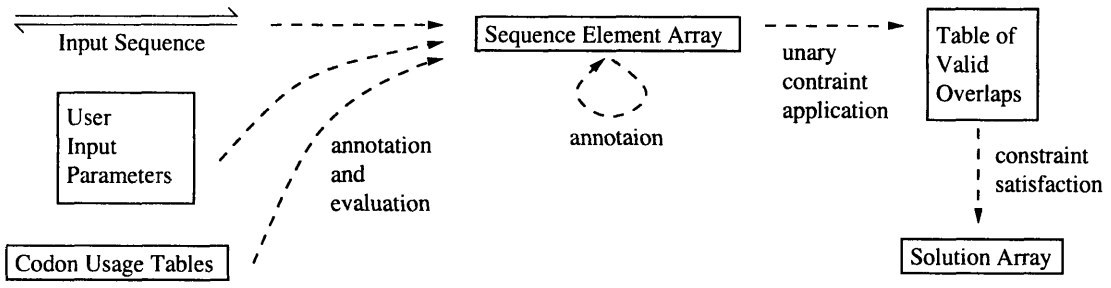


Figure 4-1: high level overview depiction.

[10]My use of 'variables', 'values', 'constraints' and 'heuristics' is consistent with the use in [22].
[11]This is to say, once a primer overlap has been chosen which includes that nucleotide position.

Current AI techniques provide many variations and extentions for solving the above constraint propagation problem. In the remainder of this document, however, any reference to "feature driven primer design" or "FD design" refers to the system just described.

I discuss the in-depth details of the data representations and related methods in Chapter 5.1, the CSP algorithms in Chapter, and all discussion specific to a programming implementation can be found in Chapter 6.

4.2 Sequence Optimisation

Sequence optimisation through codon choice and substitution is a very powerful tool for synthetic biologists. A researcher may already have a target DNA sequence, but may need to "adjust" that sequence so he can manufacture it using primer assembly methods. It is also common that a researcher has isolated an amino-acid sequence for a protein from one organism, but needs to reverse-engineer that AA-sequence and design a DNA sequence which is both manufacturable and compatible with a different organism.[12] Codon substitution is also a powerful tool for designing optimal WGS primers.

As discussed in the biology background of this thesis (Chapter 2), the amino acid code is redundant. The existence of multiple codons per amino acid allows one to have a significant number of DNA sequence variations which all result in the same target protein.[13] In the scope of this thesis, sequence optimisation exploits codon substitution to improve the physical characteristics of a DNA sequence for gene assembly protocols. By limiting the available "choices" to those codons which are frequently used in the target organism, this optimisation technique simultaneously

[12]The work of Cohen and Boyer [8] in recombinant genetic techniques proved that genes could be cloned from one organism into another while retaining biological function. The expression of a gene can be improved and controlled, however, if the DNA sequence for that gene is re-tooled or "optimised" for the target organism.

[13]This is a zeroth-order statement. First, the DNA sequence to be optimised must be in an Open Reading Frame. An ORF is a sequence intended to be transcribed into mRNA which is then translated into an amino-acid peptide chain. The mRNA sequence, and thus the DNA sequence, has direct and secondary effects on peptide production rates and kinetics. [25]

ensures that the resulting DNA sequence is compatible with the target organism.

The goal of sequence optimisation, in the context of primer design, is to homogenize primer melting temperatures and minimize the presence of repeat and hairpin sequences. This is a seemingly straightforward bioengineering problem that is very difficult to solve in practice.

Recall that there are, on average, 2.5 codons per amino acid. For a target DNA sequence which is N nucleotides long, there exist $2.5^{\frac{N}{3}}$ possible permutations[14]; an exhaustive method would require evaluating all of them, if only to eliminate those substitutions which would fail to improve or worsen the situation with respect to WGS primer design. Such an analysis technique is impossible to implement, but the power of codon substitution as a design method still remains and must be addressed.

Luckily, many of the same techniques that we have applied to the primer design problem, specifically our feature-driven primer design strategy, can also be used to address the sequence optimisation problem.

4.2.1 Limiting the Optimisation Space

There are a number of ways to limit the optimisation space and make it feasible to include codon substitution in our "bag of tricks" for WGS primer design. Each of these approaches require some knowledge of the target DNA sequence, and often the same information is required for feature-driven primer design.

Do not consider those codons which do not need to be optimised. The DNA sequence to be manufactured may not contain an Open Reading Frame, and will therefore not contain any codons. If a user has designed a gene with secondary mRNA structures in mind, he may indicate that some regions or all of the nucleotide sequence should not be changed in the process of designing WGS primers. In both of these instances, by decreasing the number of nucleotide triplets which are eligible for substitution we decrease the space of the optimisation problem. Additionally, if

[14]For a typical N of 2000, there are $1.07x10^{266}$ permutations. This number is large enough to overflow variables in most programming languages, and evaluating that many potential sequences would take *millennia*.

the feature-driven design strategy succeeds in finding a solution primer array for a given sequence – *even though that sequence contains repeats and problematic thermal domains* – then the best optimisation strategy is to do nothing.[15] The FD design strategy is intended to be strict enough that any solution found will satisfy all the constraints of a user's oligo purification protocols and WGS assembly methods. Optimisation via codon substitution should only be used as a last resort if the FD design strategy fails[16], or as an attempt to guarantee in advance that a solution will be found[17].

4.2.2 Optimisation via Local Search Algorithms

In the case of either pre- or post-processing optimisation, the optimisation space is *still* too large to systematically search.[18] The goal of the optimisation is to objectively "improve" the DNA sequence enough to allow the FD design strategy to find a solution. Hill-climbing search, a non-optimal local search algorithm, is a simple and sufficient approach for codon optimisation.

Hill-climbing search begins with a large number of possible states, in our case the $2.5^{\frac{N}{3}}$ potential codon permutations. One state is assigned, either at random or according to some heuristic algorithm. If the change "improves" the situation, the previous state is thrown away. If the change "worsens" the situation, the previous state is retained and that specific change is thrown away or removed from the list of future possible permutations. Hill-climbing search iterates until it reaches a local maximum in whatever criterion defines "improvement". It is not an optimal search, so it is not guaranteed to find an *optimal* solution, but it *is* a complete search, which means it is guaranteed to find a solution provided one exists. [22]

Pre- and post-optimisation regimes both use the annotated DNA sequence information from the feature-driven design strategy, hill climb search and similar heuristics

[15] In the vernacular, *"If the code compiles, ship it."*

[16] Post-processing.

[17] Pre-processing.

[18] Here, systematically implies that the search "remembers" where it's been, to allow exhaustive investigation of all possible permutations without re-covering territory. Recall that the number of codon permutations is prohibitively large.

to decide if a particular optimisation attempt was an "improvement." The difference in the regimes lies in how far each attempts to optimise the sequence. Post-optimisation will terminate at the first iteration where the FD strategy produces a solution. Pre-optimisation, however, will continue until a local maximum in "goodness" for the DNA sequence is achieved and only *then* will it run through the FD design procedure and check to see if a solution exists.

Both of these optimisation regimes, along with specific details of feature-driven design, associated data structures and methods, are discussed in greater detail in Chapters 5 and 6.

4.3 User Requirements and Input

The user's requirements and input parameters motivated our interpretation of the WGS primer design problem as a constraint satisfaction and search problem. Please revisit Chapter 3 for all background discussion and a complete listing of the user's requirements and input variables.

4.4 Integrated Primer Design Approach

At the risk of being pedantic, cell biology is highly complex; any phenomenon is invariably an exception to some rule. As a result, the most important aspect of our approach to WGS primer design is that we abstract the details of what makes a primer suitable from the actual process of finding suitable primers.

Below is a summary of our integrated design approach, presented in order of execution. Each item represents a distinct step in the process flow; these steps are elaborated upon in Chapter 5. An example walk-through of our primer design approach, following this outline, is presented in Chapter 6. Implementation-level details of any named algorithms are listed in Chapter 7.[19]

[19]A note on typography: data structures are in normal case, except when defined, functions and named algorithms are printed in SMALLCAPS. Variable names are italicized.

1. **User Input Processing**

   From the user's input, use the primer requirements to find the primer overlap requirements. Then, set the variables which describe the size range of valid primer overlaps, the acceptable range of primer T_M variations, repeat tolerances, hairpin primer tolerances, etc. These variables represent the design constraints. If the user did not choose pre-optimisation, build the **Sequence Element Array** (SEA), from the input nucleotide sequence.

   In the Sequence Element Array, process the user's input and set the annotation data for open reading frames and no-optimisation regions. Using the **Codon Usage Table** (CUT) for the host organism, assign values in the Sequence Element Array annotating codons, and amino acids.

2. **Pre-Optimisation**

   Our pre-optimisation strategy can only be applied to amino-acid sequences. If a user provides a nucleotide sequence with ORFs, use the Codon Usage Table to translate from a nucleotide sequence to an amino acid sequence for each ORF, and pre-optimise the ORFs in aggregate. Identify repeated amino acid sequences and construct the **Identified AA Repeat Table** (IART). Using the IART, assign each repeat a different codon usage pattern and assign codons randomly for those amino acids which are not in a repeat sequence. Construct the Sequence Element Array using these codon assignments for all optimisable nucleotides in ORFs and the user's input for all remaining nucleotides. Flag those nucleotides which were in an amino-acid level repeat as being non-optimisable. At this point, one can proceed to feature identification or one can choose to apply hill-climbing search techniques to improve the T_M characteristic of those optimisable portions of the Sequence Element Array.

3. **Feature Identification**

   Run those algorithms which identify the relevant features of the DNA sequence. Annotate T_M domain information in the Sequence Element Array. Construct

the **Identified Nucleotide Repeat Table** (INRT) in the same fashion as the IART.[20]

4. **Unary Constraint Propagation: Repeated Sub-Sequences**

   Flag all entries in the Sequence Element Array which are also listed in the Identified Nucleotide Repeat Table as being in a repeat. Flag all entries in the SEA which are in a repeat or within *buffer_size* of a repeat as being invalid overlap end sites. Flag all remaining entries as valid end sites.

5. **Unary Constraint Propagation: Overlap T_M and Length**

   Apply the following constraints to the completely annotated SEA to build the **Valid Overlap Table** (TVO): the outside/end nucleotides of valid overlaps must be valid end sites; valid overlaps must meet a user's length criteria; valid overlaps must satisfy T_M criteria. Each entry in the Valid Overlap Table contains the terminating indices which define that overlap along with a value representing how much that overlap's melting temperature varies from the user's target mean.

   Once we have generated the Valid Overlap Table we no longer need to consider any physical characteristics of the DNA sequence in order to generate a solution. The variance of overlap T_M, included in the Valid Overlap Table, is retained only as a choice heuristic.[21] Strictly speaking, even this heuristic is unnecessary, as the unilateral constraint that an overlap must satisfy a user's melting temperature and length requirements in order to be considered valid guarantees that every entry in the Valid Overlap Table is compatible with WGS assembly.

6. **Constraint Satisfaction Over the Set of Valid Overlaps**

   From the Valid Overlap Table we build a linked-list data structure called the

[20] Recall that the functions STRSTR and LEVENSHTEIN are general string manipulation functions. They work equally well for finding repeats in either amino acid strings or nucleotide sequences, providing we use the single-letter amino acid code.

[21] Overlaps with a lower variance in T_M are more desirable.

Nucleotide Overlap Assignment List (NOAL). The entries in the NOAL correspond to the nucleotide indices of the target DNA sequence or the Sequence Element Array. As an example; the zeroth entry of the NOAL points to an array of all the valid overlaps which span the zeroth nucleotide of the target sequence. The entries of the overlap sub-array are ordered by the previously discussed T_M heuristic.

Using the VOT and the NOAL we can treat the task of finding a set of valid overlaps which completely and exactly spans the target DNA sequence as a constraint satisfaction problem. In this constraint satisfaction problem, the *variables* are the nucleotides and and the *values* are those overlaps which span the nucleotides Assigning a value to one variable, such as the 0^{th} nucleotide, will assign that same value to other variables, such as the 1^{st}, 2^{nd} and 3^{rd} nucleotides.

As previously discussed, the constraints which we must satisfy in order to find a solution set are: no overlap conflicts may exist[22]; no gaps may exist; no hairpin structures can exist between two adjacent primers[23]. The detailed execution of the CSP problem is described in Chapter 5. At its completion, our constraint satisfaction algorithm will either return an ordered array of valid primer overlaps which span the target sequence or it will return a null array if no solution exists.

7. **Post-Optimisation**

In the event that a solution does not exist, we post-optimise the nucleotide sequence via codon substitution. First, we must revisit the Sequence Element Array. Recall that all nucleotides in the SEA have been annotated with melting temperature information and that we have derived a **Thermally Biased Codon Usage Table** (TBCUT) for the host organism. We further annotate the optimisable codons in the SEA with an ordered list of possible codon substitutions which will "improve" the T_M characteristic of the sequence, based on the

[22]No one nucleotide can be spanned by more than one overlap.
[23]This is subject to the user's discretion

context of the codons in the SEA and the TBCUT. Our post-optimisation strategy does not attempt to "improve" the repeat characteristic of the sequence, however.

Using the amino acids as variables and the possible codon substitutions as values, we then use a hill-climbing search technique to find a solution in the following manner:[24]

- Save the old instance of the SEA along with the number of entries in the Valid Overlap Table.
- Substitute one codon. Update the codon substitution annotation data in the SEA for that codon and its two neighbors.
- Run the new SEA through **Step 3**, **Step 4** and **Step 5**.
- Compare the new number of entries in the VALID OVERLAP TABLE to the old value. If the number of entries *increased*, then the optimisation attempt "improved" the sequence. Throw away the old SEA and proceed to **Step 6**. If the number of entries *decreased*, then the optimisation attempt was not advantageous. Revert back to the old saved SEA and remove the attempted optimisation from the SEA annotation data.
- Iterate until a solution is found or until no more codon substitution options exist.

8. **Failure Mode**

   In the event that all primer design attempts fail to return a solution, the user must be notified. The program should, at a minimum, display the target sequence with all thermal domains and identified repeats flagged. It is then the responsibility of the user to analyse which properties of the DNA sequence or to adjust their input constraints.

9. **Success Mode**

[24] Detailed discussion of this procedure can be found in Chapter 5.

If the program finds a valid primer overlap array, the last step is to construct the solution primer array. A primer solution array contains both lead- and lag-strand primers, so the overlap solution will produce two primer solutions. The choice heuristic for this final decision is to select whichever array has the more uniform *primer* melting temperature characteristic.

Each step will be discussed in detail, with necessary data structures in Chapters 5 and 6. Details of the specific algorithms involved, such as TEMPEVAL, THERMALMAP, etc, can also be found in Chapter 5.

Chapter 5

Details of WGS Primer Design

As described in Chapter 4, our approach to WGS primer design systematically abstracts out the user's requirements and the physical features of the DNA sequence from the problem of actually finding a solution primer array. In this chapter, we present the implementation-level details of that abstraction; including data structures, relevant global variables, algorithms and the design program's process flow.

It is worth noting that, due to the inadequacies of our chosen programming language[1], we are not presenting a verified and tested implementation with supporting source code. Instead, this chapter contains an implementation-level description of our software design as a modular program framework,[2] sufficient to allow a reader to re-implement our work in a suitable language of his choice.

Finally, this section assumes that the reader is reasonably familiar with object oriented programming techniques, artificial intelligence search techniques and recursive and iterative styles of function design.

5.1 Data Structures

In all of our approaches for solving the primer design problem, we focused on keeping all of the necessary data structures small and manageable with fast access times.

[1] PHP, while convenient, is completely unsuitable for larger programming projects.

[2] That is, we present our design, data abstraction and algorithms in the form of pseudocode.

The space of any problem involving DNA or computational genomics can grow at a fantastic rate, and inappropriate or negligently designed data structures or algorithms can make a problem impossible to compute. One of the appealing traits of PHP was that its arrays and lists are actually hash tables which one can access in constant time. It also has very efficient built-in memory management because it is intended to write web server scripts. However, PHP is not a strongly-typed language, its namespace is non-existent, and its syntax is inconsistent. All of these traits make PHP completely inappropriate for large object-oriented programs.

As a direct result of the difficulty involved in building robust systems in PHP, our data structures are very straightforward. the **Codon Usage Table** (CUT) and the **Thermally Biased Codon Usage Table** (TBCUT) contain all of the data which pertains to the host organism. The **Sequence Element Array** (SEA) contains all of the information about the target DNA sequence, including annotation information for codon substitution and building the **Table of Valid Overlaps** (TVO). Auxillary feature annotation information, specific to repeat subsequences, is stored in either the **Identified AA Repeat Table** (IART) or the **Identified Nucleotide Repeat Table** (INRT). The solution to the overlap constraint satisfaction problem is stored in the **Solution Overlap Array** (SOA), and then is converted into the SOLUTION PRIMER ARRAY (SPA). Additional data, such as the user's input options, are stored in auxillary variables which do not contribute significantly to the memory overhead of our primer design software.

For all of these data structures, with the exception of the Table of Valid Overlaps, the order of growth is either $\mathbf{O}(N)$[3] or static. This is a vast improvement over other primer design tools, such as deCODE biostructures Inc.'s Gene Composer software. [25] Gene Composer begins its search process by "randomly dividing the duplex sequence into an enormous number of possible overlapping sets of oligos," which are then, "checked to see if they happen to meet defined criteria for oligo length limits and overlap length limits." [25]. Recall our discussion in Chapter 4.1 where we analysed the order of growth of the possible primer set and the possible search space. Gene

[3]Where N is the length of the target DNA sequence.

Composer begins by building a the entire possible space, a data set that grows exponentially with the target sequence length and the allowed variation in primer length! Another available software tool, Gene2Oligo, sensibly applies melting temperature constraints early on to build a tree of valid primers that is directly analagous to our Table of Valid Overlaps. Gene2Oligo then traverses this tree to find a solution set using a depth-first search with backup. [21] Our strategy deviates from Gene2Oligo's in that we include feature identification from the Sequence Element Array as additional constraints for building the Table of Valid Overlaps. Then, rather than using a depth-first search with backup, we use constraint propagation with forward checking in order to build our solution set.

5.1.1 Sequence Element Array

The Sequence Element Array should properly be an array of **Sequence Element** (SE) objects. Figure 5-1 contains an example instantiation of a Sequence Element, including descriptions of the object parameters.

5.1.2 Identified Repeat Tables

Both the amino acid and nucleotide table of repeats are constructed by the same algorithm, FINDREPEATS.[5] The only difference between the two is that we build an Identified AA Repeat Table by calling FINDREPEATS with an amino-acid sequence, and the Identified Nucleotide Repeat Table by calling FINDREPEATS with a nucleotide sequence.

Both the IART and the INRT are hash tables indexed by the original matching subsequence, with one entry for each instance of a particular repeated subsequence. Each entry contains the start and end indices of that repeat, along with the repeat itself. One can use this table for two ends; it can be used simply to flag entries in the Sequence Element Array as being invalid overlap end sites, or it can be used to direct and inform an algorithm which removes repeated subsequences via codon

[5]The details of the FINDREPEATS algorithm can be found in Chapter 5.

Type	Name	Value	Description & Accessor Methods
`int`	*Sequence_Index*	7	The index of that sequence element in the target DNA sequence.
`char`	*Base*	t	The base/nucleotide for that sequence element: A, T, C, G
`bool`	*Repeat?*	0	Indicates whether the sequence element is included in a repeat sub-sequence. This is used to constrain the number of valid overlap end sites.
`int`	*Repeat_ID*	-	If the sequence element is included in a repeat sub-sequence, this indicates which repeated sub-sequence. This parameter is used when reporting the physical annotation data for the DNA sequence to the user.
`bool`	*Valid?*	0	Indicates whether the sequence element is a valid overlap end site. Any sequence element which is inside a repeat, or which is within BUFFER_SIZE of a repeat is invalid: **Valid? = 0**
`bool`	*ORF?*	1	Indicates whether the nucleotide is in an open-reading frame.
`char`	*AA*	V	If a sequence element is in an ORF, this specifices which amino acid it is in: A, R, G, L, V, M, P... This is relevant for sequence optimisation.
`int`	*Codon_Index*	1	If a sequence element is in an ORF, this specifies whether this sequence element is in the 1^{st}, 2^{nd} or 3^{rd} codon position. This is relevant for sequence optimsation.
`int`	*Codon_ID*	0	For the amino acid designated by **AA**, this identifies which codon in the Codon Usage Table is specified by this sequence element and the other two nucleotides in the codon. This is relevant for sequence optimisation.
`bool`	*Opt?*	1	Indicates whether the nucleotide is eligible for optimisation.
`array of int`	*Substitution*	{2, 0, 3}	An ordered array of the preferential codon substitutions for this sequence element. Numbers indicate the index of the codon in the Codon Usage Table, while the order is determined by a look-up on Thermally Biased Codon Usage Table.[4]
`int`	*SE_Temp*	37 °C	Indicates the average T_M of the sequence element as determined by the windowing function, THERMALMAP. *SE_Temp*, along with the user's constraints on T_M is used to construct *Substitution*.

Figure 5-1: An example instantiation of Sequence Element in the Sequence Element Array

substitution.[6]

Figure 5-2 illustrates the kinds of repeats that one would find by instantiating the FINDREPEATS function with an amino acid sequence, an N_R of four and a threshold L_D of zero. It is appropriate to stress the fact that this data structure is more impor-

[6]We only propose methods to optimise out nucleotide sequence repeats at the amino acid level using the IART, although it is not infeasable to optimise out repeats at the nucleotide sequence level.

Hash Index	*Repeat Instance*	*Start Index*	*End Index*	*Sequence*
PAVL	1	8	14	GPAVLIM
	2	47	53	PAVLI
	3	72	78	GPAVLI
QRTI	1	23	26	QRTI
	2	81	84	QRTI

Figure 5-2: An example instantiation of the Table of Identified AA Repeats

tant than the specific algorithm which builds it. Any particular implementation of the FINDREPEATS algorithm could be better or worse at actually identifying repeats which physically interfere with a given primer assembly process. Regardless of how well our repeat identification model approximates the biology, if the table of identified repeats exists in this or some similar form, we can use it to faithfully eliminate invalid overlaps from our search space. The extensible nature of our approach to primer design is improving the individual algorithms and models as our understanding of DNA hybridization kinetics improves. Our data abstraction is meant as a persistent framework to prevent the primer design problem from degenerating into a mess of complex and often inter-dependent modeling problems.

5.1.3 Table of Valid Overlaps

Once the Sequence Element Array has been fully annotated to indicate which elements are valid overlap end sites, we can build the Table of Valid Overlaps. The TVO is constructed by iteratively traversing the SEA, and creating an entry for each overlap which begins and ends on valid sites and conforms to melting temperature constraints. Each entry contains the valid overlap's start index, end index and predicted melting temperature, as illustrated by Figure 5-3. All sequence information is omitted from the valid overlap table. This is for for two reasons: the sequence information is not needed to solve the constraint satisfaction problem of building an overlap solution array; including the sequence information would only increase the memory overhead involved in storing and parsing the TVO.

Once we have constructed the complete TVO for a given sequence and set of user constraints, we need to construct an equivalent data structure which is compatible

Start Index	*End Index*	*Predicted* T_M
0	15	37 °C
	17	38 °C
1	15	36 °C
	17	37 °C
	18	38 °C

Figure 5-3: An example instantiation of the Table of Valid Overlaps

with our CSP algorithms. Recall that in our constraint satisfaction problem, the variables are the nucleotide indices and the values are those overlaps which span the nucleotides. An ideal CSP overlap data structure would be a table or linked-list such as the one depicted in Figure 5-4. The index of the data structure would be a given nucleotides index in the target DNA sequence or the SEA. The entry for a given index would be a list of the valid spanning overlaps ordered according to the melting temperature heuristic.[7]

Nucleotide Index	*Ordered List of Spanning Overlaps*, (start, end)
0	{(0, 15), (0, 17)}
1	{(1, 17), (1, 18), (1, 15)}

Figure 5-4: An example instantiation of the Nucleotide Overlap Assignment List

5.1.4 Solution Arrays

The Solution Overlap Array and its derivative, the Solution Primer Array are both very simple data structures. Their exact structure will depend on the language of implementation and the specific user interface, but we provide representative examples.

The Solution Overlap Array is just an ordered list of the spanning overlaps from the Table of Valid Overlaps. Its structure is demonstrated Figure 5-5.

Overlap_0	Overlap_0	Overlap_0	...	Overlap_S
$(start_0, end_0)$	$(start_1, end_1)$	$(start_2, end_2)$	...	$(start_S, end_S)$
(0, 15)	(16, 34)	(35, 54)	...	(2436, N-1)

Figure 5-5: An example instantiation of the Solution Overlap Array

[7]Such data structures are trivial to implement in SCHEME, LISP or ML; these are conveniently the languages of choice for building constraint satisfaction systems.

The Solution Primer Array is a three-dimensional array of primer sequences organised by lead strand and lag strand. Its structure is illustrated in Figure 5-6. It is constructed by parsing both the SOA and the SEA to extract the sequence information for the solution overlaps. Because a primer is simply two adjacent overlaps, any given solution overlap array will generate two primer solution arrays.[8] Note that the end primers represent a special case because at least two primers will only have one component overlap. In these instances, the user should provide a "dummy sequence" which will be removed when the WGS assembly product is amplified and purified.

	Primers	P_0	P_1	P_2	...	P_{n-2}	P_{n-1}	PCR Primers
Sol 1	Lead	$O_0 + O_1$	$O_2 + O_3$	$O_4 + O_5$	...	$O_{m-3} + O_{m-2}$	O_{m-1} + d_s	$O_0 + O_1$
	Lag	d_s + O_0	$O_1 + O_2$	$O_3 + O_4$	...	$O_{m-4} + O_{m-3}$	$O_{m-2} + O_{m-1}$	$O_{m-2} + O_{m-1}$
Sol 2	Lead	d_s + O_0	$O_1 + O_2$	$O_3 + O_4$	...	$O_{m-4} + O_{m-3}$	$O_{m-2} + O_{m-1}$	$O_0 + O_1$
	Lag	$O_0 + O_1$	$O_2 + O_3$	$O_4 + O_5$	...	$O_{m-3} + O_{m-2}$	O_{m-1} + d_s	$O_{m-2} + O_{m-1}$

Figure 5-6: An example instantiation of the Solution Primer Array.

In the Figure 5-6: P is a primer and there are (n) primers in the solution; O is an overlap and there are (m) overlaps in a solution. The subscripts indicate the index of each element.[9] If two overlaps are summed together, $O_0 + O_1$, it means that the resulting primer is the concatenation of the sequences for those two overlaps. It is assumed that overlap sequences correspond to subsequences on the leading, or coding strand as read from 5'→3'. The lag strand is typically the reverse complement, as read from 5'→'3. In the above table, a sum surmounted by a bar, $O_1 \bar{+} O_2$, represents the *complement*[10] of the concatenation of the two lead strand overlap sequences as read from 3'→5'. *d_s* indicates a dummy sequence.

5.2 User Input and Constraint Variables

All of the user inputs are propagated into either data structures, such as the Sequence Element Array, or variables which define the constraints on our search algorithms.

[8]This is the 3rd dimension.

[9]Recall that the first element of an array is indexed by a 0

[10]A↔T, C↔G

5.2.1 List of User Input Variables

As described in Chapter 3.3, the list of user inputs is summarised in Figure 5-7.

Type	Name	Description
`string`	*target_sequence*	The input target sequence.
`string`	*target_host*	The target host organism.
`string`	*target_host*	The target host organism.
`int`	*primer_length_min*	The minimum primer length.
`int`	*primer_length_max*	The maximum primer length.
`int`	*primer_length_fixed*	The length for fixed primers.
`int`	*primer_mean_*T_M	Target primer melting temperature.
`int`	*primer_*T_M*_+range*	Allowed positive T_M variation.
`int`	*primer_*T_M*_-range*	Allowed negative T_M variation.
`int`	N_R	Minimum length of a repeat.
`int`	L_D	Maximum variation allowed between two instances of one repeat.
`int`	*buffer_size*	Number of nucleotides flanking an indentified repeat which must be flagged as invalid overlap end sites.
`array of int`	*orf_range*	Indices for all nucleotides which are in Open Reading Frames.
`array of int`	*noopt_range*	Indices for all nucleotides which are not eligable for optimisation.

Figure 5-7: User Input Variables

5.2.2 Constraint Variables

Having the list of input parameters for primer design and knowing the physical restrictions on primer design from Chapter 3, we must compile the set of constraints which the program will use to construct an optimal set of primers for PCA assembly. Many of these constraints come directly from the user. The remainder, which we must compute, are shown in Figure 5-8.

5.3 Sequence Feature Identification Methods

Sequence feature identification is ubiquitous in our approach to designing primers. We use the same family of algorithms to identify relevant features in the DNA sequence, to determine if a given primer meets the user's criteria and to construct the heuristics

Type	Name	Description
int	*overlap_length_min*	The minimum overlap length, defined to be half of the minimum primer length, rounded up.
int	*overlap_length_max*	The maximum primer length, defined to be half of the maximum primer length, rounded down.
int	*overlap_mean_T_M*	Target overlap melting temperature.
int	*overlap_T_M_+range*	Allowable T_M variation.
int	*primer_T_M_-range*	Allowable negative T_M variation.

Figure 5-8: Constraint Variables

which inform our optimisation procedures. This section summarises all of our feature identification methods along with how and when they are applied in the primer design process.

5.3.1 Thermal Evaluation Methods

Provided that we have a good physical model for predicting the melting temperature of an arbitrary DNA sequence,[11] our required thermal evaluation methods are very straightforward.[12] In order to design primers around thermal features and optimise the DNA sequence on the basis of thermal features, we require methods to:

- Determine the melting temperature of an arbitrary duplex sequence. All other evaluation methods will use this method. – (TEMPEVAL: *represents our basic physical model of DNA duplex formation thermodynamics.*)

- Evaluate the melting temperature of potential primer or primer overlaps and determine if they are valid according to a user's T_M restrictions for purification and assembly. – (CHECKTEMP: *evaluates a constraint for feature-driven primer design, used to build the* TVO*, given an annotated* SEA.)

- Analyze primer solution arrays and rank them in order of "most optimal". This is done by evaulating the melting temperature of all primer or primer overlaps

[11] Deterministic models do exist for this. We're using the salt-corrected, nearest-neighbors model for duplex formation. [23]

[12] The ability to predict T_M is synonymous with being able to evaluate some sequence against another on the basis of T_M.

in a solution array, then finding the expected value and variance of T_M for that set. The "best" array will have the lowest variance around the user's target T_M. – (SOLUTIONRANK: *computes a statistical heuristic for ordering potential solution arrays.*)

- Analyze an overall sequence to build a "map" of melting temperature domains. This allows the program to limit optimisation to "problem domains"[13] and informs the program as to whether the T_M of the "problem domain" is too high or too low. – (THERMALMAP: *computes a heuristic for targeted optimisation, used to annotate the entries in the* SEA *accordingly.*)

- Analyze the codon usage table for a target organism and rank all codons by their context and contribution to a sequence's melting temperature.[14] This "thermally ranked" codon usage table is required by the optimisation algorithms to choose an appropriate substitution. – (CODONRANK: *computes a choice heuristic for codon substitution which can be used to instantiate Substitution array for elements in the* SEA.)

Specific algorithms for TEMPEVAL, CHECKTEMP, SOLUTIONRANK, THERMALMAP and CODONRANK are presented in Chapter 5.

5.3.2 Evaluation Methods for Repeats

We approach repeat sub-sequences in the same manner as other existing primer design tools; we look for exact sub-string matches in our target DNA sequence and expand those matches until a user-defined homology criterion is exceeded. The difference in our approach is that other primer design tools use BLAST [21] [25] to locate repeated sub-sequences while we using the string manipulation tools LEVENSHTEIN and STRSTR. The rationalisation and trade-offs involved in using string manipulation tools was discussed in Chapter 3.

[13] Targeted optimisation improves an algorithm's efficiency and success rate.

[14] An example of such a table for *eschericia coli* can be found in [APPENDIX FOO].

It is possible to identify repeated sub-sequences while lacking the ability to evaulate repeats according to their contribution to mis-assembly in whole gene synthesis. From observations that DNA can form mismatched hybrid duplexes under a range of melting temperatures, we assume that exactly repeated and approximately repeated subsequences can lead to mis-annealing events. There exists no qualitative or quantitative model for predicting whether a given mis-annealing event might occur for a user's predicted assembly protocol, however. This forces us to create methods of evaluating repeats are which are both simplistic and pessimistic.

An additional level of complexity is added to the repeat problem because, while melting temperature is a local phenomenon, repeated sub-sequences are a global phenomenon. We could attempt to predict what changes might improve some repeats at the cost of creating others, but recall the prohibitive size of the optimisation space. There is no good way to predict how codon substitutions will improve or worsen repeat conditions, and without the ability to evaluate the contribution of any given repeat to the likelihood of a WGS assembly failure we can't even make informed decisions about which repeats we should optimise.

As a result, we limit the methods for identifying and evaluating repeated subsequences to:

- Locate subsequences which adhere to the user defined repeat detection criteria of minimum match window, N_R, and Levenshtein distance, L_D. – (FINDREPEATS: *builds an identified repeat table, then sets the Repeat? parameter for all entries in the* SEA *which are contained in the* IRT *to 'True.'*

- Apply FINDREPEATS to an amino acid sequence. Parse the resulting IART and assign each entry in the IART a different codon usage pattern. – (OPTOUT: *constructs a* SEA *where amino acid sequence repeats do not tranlate into nucleotide sequence repeats*

- Check an arbitrary sequence for hairpins according to a user's indicated criteria for the number of binding bases, N_H, and the number of bases in the loop, N_O.

– (CHECKHAIRPINS: *a constraint for choosing overlaps for the primer solution array; two adjacent overlaps may not contain a hairpin sequence.*

- Return the maximum-length repeat in the INRT and the overall fraction of the target DNA sequence which is in a repeat region. – (REPEATCHECK: *an arbitrary heuristic for estimating the repeat characteristic of a given DNA sequence.*

5.3.3 Identifying Open Reading Frames

Demarcation of Open Reading Frames (ORFs) is the responsibility of the user. Identified ORFs which are not subsequently labeled as being non-optimisable are considered eligable for optimisation. This is problematic if a user constructs a gene with secondary structures and incorrectly assumes that his target gene contains no ORFs.

ORFs must begin with a valid methionine codon, end with a valid stop codon, and contain a multiple of three nucleotides. Identifying ORFs to the rest of the primer design program only require a process which updates the *ORF?* parameter for the objects in the Sequence Element Array to 'True.'

5.3.4 Identifying Codons

Codons are identified by parsing the SEA for all elements which have *ORF?* flagged. By iterating through the sub-array of sequence elements corresponding to each ORF, one can look up each three-base window in the Codon Usage Table, then annotate the *AA*, *Codon_Index* and *Codon_ID* parameters for the appropriate Sequence Elements.

5.3.5 No-Optimisation Regions

It is the responsibility of the user to indicate any and all no-optimisation regions. The Sequence Elements in the Sequence Element Array which correspond to nucleotides in the indicated no-optimisation regions should have their *Opt?* parameter set to 'False.'

5.4 Parsing the Annotated SEA

Parsing the SEA is as straightforward as building appropriate accessor functions for the Sequence Element object and then iterating through the array of SE objects.

5.5 Pre-Optimisation Strategy

As stated earlier, the nucleotide sequence optimisation search space is too large to allow us to evaulate all states and choose the best one as our starting point for primer design. However, DNA has two potential levels of representation: a nucleotide sequence and a corresponding amino acid sequence. The amino acid sequence represents a static and immutable design goal, a protein. The amino acid search space is therefore static,[15] which makes it feasible to pre-optimise a sequence by making intelligent codon assignments.

Our pre-optimisation strategy is as follows:

1. From the user's input, pull together an array of the optimisable codons and translate from nucleotide to amino acid sequence. Alternately, the user may provide an amino acid sequence as the input.

2. Use FINDREPEATS to find *exact repeats* in the amino acid sequence and build the corresponding IART. Do this by instantiating FINDREPEATS with a a threshold of zero ($L_D = 0$) and a buffer of zero (*buffer_size* $= 0$). We consider exact matches only to keep the optimisation strategy simple.

3. For each instance of a particular repeat in the IART, derive a different codon usage pattern. An example of this is illustrated in [FIGURE FOO]. For non-repeat amino acids, assign codons randomly from the codon usage table.

4. Propagate the codon assignments into the Sequence Element Array. Flag any nucleotides which were previously in an amino-acid level repeat as now being non-optimisable.

[15] This is not a protein design tool; we are restricted to nucleotide substitutions which preserve the amino acid sequence.

5. Apply THERMALMAP to evaluate the melting temperature characteristic of the sequence. Annotate the SEA.

6. Now that all codons in the optimisable regions have context,[16] update the *Substitution* array for all optimisable elements in the SEA.

7. Apply a hill-climbing search on those codons which are still optimisable. Use THERMALMAP and the sequence's T_M characteristic as the heuristic.

5.6 Constructing the Table of Valid Overlaps

We construct the TVO in the following manner:

1. Run FINDREPEATS on the nucleotide sequence with the user-specified values for N_R and L_D.

2. For every nucleotide index which is included in an identified repeat, change the *Repeat?* parameter for the corresponding Sequence Element to 'True'.

3. Traverse the Sequence Element Array. Set the parameter *Valid?* = 'False' for every element that has *Repeat?* = 'True', or that is within *buffer_size* bases of a repeat element.

4. Beginning with the 0^{th} element of the SEA, check for valid end sites that are a valid overlap distance away from the start element. For all found pairs, evaluate the melting temperature of the overlap and compare it to the acceptable range. If the potential overlap satisfies the T_M constraint, add it to the TVO.

5.7 The Primer Constraint Satisfaction Problem

After constructing the Table of Valid Overlaps, the primer design collapses to finding a set of overlaps from the TVO which uniquely spans the target sequence. We approach

[16] Context refers both to the flanking nucleotides and a codon's thermal characteristic with respect to the user's target T_M.

this by treating the sequence indices as variables and the overlaps in the TVO as values. A solution exists once each variable is assigned exactly one value.

5.7.1 Summary of Constraints

The constraints on the primer design problem are as follows:

1. **No variable may be assigned more than one value.**

   This is equivalent to declaring that we can not have any conflicting primer overlaps in our solution array.

2. **Every variable must be assigned at least one value.**

   This is equivalent to declaring that we can not have any gaps in our solution array.

3. **No two adjacent primer overlaps may form a hairpin structure.**

5.7.2 The Minimum Remaining Values (MRV) Heuristic

Rather than assigning values to variables at random, we preferentially choose to evaluate the most constrained variable first. This ensures that if we're going to find out that a given TVO produces no solution, we'll find out sooner rather than later. [22]

5.7.3 Constraint Propagation with Forward Checking

The algorithm for constraint propagation with forward checking is as follows:[17]

1. **Choose a variable with the lowest number of assigned values.**

   In our case, parse the TVO or the nucleotide-indexed version of the TVO and select a sequence index that is spanned by the fewest number of overlaps.

[17]This is straight out of [22]

2. **Assign that variable the best possible value.**

   If our theoretical nucleotide is spanned by two overlaps, we will select the overlap with a T_M that is closest to the target T_M. This assigns values to all of the nucleotides spanned by that overlap, not just the nucleotide we chose to evaluate.

3. **Propagate the constraints.**

   Now that we've made an assignment, the second overlap that spans our theoretical nucleotide represents a *conflict*, as described in the previous section. We eliminate that second overlap from the TVO. Additionally, we check to see there are any overlaps in the TVO who's assignment would create a *gap* in the solution array. If any such overlaps exist, they are eliminated from the TVO.

4. **Check to see if a solution still exists.**

   If the entire target sequence is no longer spanned by the assignments in the solution array and the remaining overlaps in the TVO, then the previous assignment in **Step 2** was a poor choice.

5.8 Post-Optimisation Strategy: Local Search

Post-processing with search starts with the first failed attempt at feature-driven design and attempts to "fix" the original sequence through codon optimsation. Because the feature driven primer design process annotates the DNA sequence with relevant feature information,[18] we potentially save the work required to identify the "problem regions" in the target sequence.

An FD design attempt could fail for one of two reasons: either the target sequence was problematic from a thermal standpoint or the target sequence contained repeats which the FD design tool could not cope with.[19] These two situations require very

[18]Relevant feature information includes regions with low and high melting temperature and bases which are part of repeated subsequences.

[19]Our FD design strategy can not deal with repeats that are longer than some length defined, in part. by the minimum overlap between opposite-strand primers.

different optimisation strategies.

5.8.1 Optimising for T_M

The first reason for an FD primer design attempt to fail is that the target sequence contains long[20] regions with a T_M outside an acceptable range, or otherwise problematic thermal domains. We can correct this using condon subsitution in the following manner.

- Find and flag the "problematic" triplets/codons. A triplet is problematic if two or more of its bases are flagged as being in a domain where T_M is too high or too low (as determined by a user's restrictions). To limit the optimisation space, only problematic triplets are eligiable for codon substitution.

- Using the thermally ranked codon usage table for the host organism, identify an alternate codon to each problematic triplet (whenever possible) which would bias the melting temperature in the correct direction.

- Make one change and evaluate whether the T_M characteristic of the strand improved. Do this by re-running the feature identification algorithms.

- If the T_M characteristic did not improve, make a different codon substitution.

- Once the T_M characteristic improves, re-run the FD design process and check to see if a solution now exists.

- Iterate until a solution is found.

This represents a greedy local search strategy known as a "hill climb search". [22] Because we only look for improvement in the melting temperature characteristic, our optimisations could be creating problematic repeats.

[20] "Long" implies a region that is longer than the maximum length for a primer.

5.8.2 Removing Repeated Subsequences

If the FD design attempt fails due to the presence of repeated subsequences, the failure mode is likely that one or more repeats are longer than a maximum acceptable length. One can try to remove repeats in much the same way as removing problematic thermal regions.

- Find and flag the repeat codons. A codon is considered to be part of a repeat if two or more of its bases are flagged as being in a repeated subsequence. To limit the optimisation space, only repeat codons are eligiable for substitution.
- Using the thermally ranked codon usage table for the host organism, identify those alternate codons to each repeated triplet (whenever possible) which might properly bias the melting temperature.
- Make one change and evaluate whether the repeat characteristic of the overall improved[21]. Do this by re-running the feature identification algorithms.
- If the repeat characteristic did not improve, make a different codon substitution.
- Once the repeat characteristic improves sufficiently[22], re-run the FD design process and check to see if a solution now exists.
- Iterate until a solution is found.

This represents a greedy local search strategy known as a "hill climb search". [22] Because we only look for improvement in the repeats characteristic, our optimisations could be creating problematic melting temperature domains.

5.8.3 A Combined Heuristic for Post-Optimisation Search

It should now be obvious that we can not optimise a sequence for melting temperature independent from repeated subsequence. We could combine the two optimisation

[21] A sequence's characeristic improves if the fraction found in repeats decreases while the longest repeat is shorter than the maximum acceptable repeat length

[22] "Sufficient" improvement should also confirm that no repeats exist which are longer than the maximum acceptable length.

strategies can by merging the thermal and repeat improvement criteria into one overal improvement heuristic. The downside to this tack is that such heuristics are very difficult to construct. To build a suitable heuristic, one needs the answers to questions such as, "Is it preferable to improve repeats at the cost of T_M and in what situations is that the case?" Often, questions like this can't be answered until one already has a working search system, along with a large test data set and time enough to evaluate each heuristic option.[23]

There exists an alternate heuristic for improvement which doesn't require one to track either the thermal or repeat characteristics of the optimised DNA sequence. This heuristic is simply the number of valid primers found by the feature-driven design tool. If a codon substitution results in a higher number of valid primer overlaps found by the feature-driven design protocol, then that substitution was advantageous and should be retained.

A simple algorithm for this is:

- Find and flag all codons which are eligable for substitution. Based on the T_M information for the codon, indicate the preferred thermal bias (if any).
- Using the thermally ranked codon usage table for the host organism, identify the valid substitutions which satisfy the above thermal bias constraint.
- Change one codon at random. Run the feature-driven design preprocessor and count the number of valid overlaps in the solution search database.
- If the number of valid overlaps decreased, revert the current codon and flag it so that particular substitution no longer considered a valid option.
- If the number of valid overlaps did not change, make a different codon substitution.
- If the number of valid overlaps increased, run the remainder of the FD design process and see if a valid solution now exists.

[23]This sort of Catch 22 is the bane if artificial intelligence.

- Iterate until a solution is found.

5.8.4 Drawbacks to Post-Optimisation

The downside to post-optimisation using a hill-climbing search is that it isn't an optimal search strategy; its probability of success depends largely on the order in which it substitutes codons. One could inform the search strategy by ranking eligable codons, but that requires constructing a heuristic to prioritize all potential optimisations. An improperly constructed choice heuristic could result in the optimisation attempt removing repeats but making the T_M characteristic worse, or visa versa. If an optimisation attempt fails, it is not proof that a solution does not exist; it only demonstrates that the strategy was not good enough to find a solution.

This approach to post-optimisation is also an iterative search technique with significant computational overhead. In order to determine whether an optimisation attempt was succesful, one needs to re-annotate the DNA sequence, build the table of valid primers and compare the size of the table to the previous iteration.

Chapter 6

Discussion and Conclusions

Biology is not a trivial subject. Because data collection is so time consuming, every attempt to arrive at a more concise computational model improves the our chances of making real advances in the field. In my case, I have tried to break down the brutally complex problem of primer design, and I believe if nothing else, I have made the problem itself a little more clear.

Future developments on this thesis include implementing the sytem in a programming language such as SCHEME or PERL. Additionally, I hope my treatment of the material may motivate enterprising individuals to solve the mystery of mismatched duplex hybridization kinetics.

Appendix A

Auxilliary Tables and Equations

$$\Delta G^{\circ}(\text{total}) = \Sigma_i n_i \Delta G^{\circ}(i) + \Delta G^{\circ}(\text{init w/ term G·C}) + \Delta G^{\circ}(\text{init w/ term A·T}) + \Delta G^{\circ}(\text{sym})$$

$$T_M = \Delta H^{\circ} / (\Delta S^{\circ} + R \ln C_T)$$

$$\Delta S^{\circ}(\text{oligomer, } [\text{Na}^+]) = \Delta S^{\circ}(\text{unified oligomer, 1 M NaCl}) + 0.368 x N x \ln[\text{Na}^+]$$

Table A.1: Existing gene design software with primer design support: adapted from [25]

.

Year	Software Name	Paper Reference
1991	PINCERS	Tamura, T.; et. al. *Biotechniques,* **1991**, *10*, 782-4.
1998	CalcGene	Hale, R.S.; et. al. *Protein Expr Purif,* **1998**, *12*, 18508.
1999	COD OP	Withers-Martinez, C.; et. al. *Protein Eng*, **1999**, *12*, 1113-20.
2002	DNA Works	Hoover, D.M.; et. al. *Nucleic Acids Res,* **2002**, *30*, e43.
2004	Gene2Oligo	[21]
2004	DNA2.0	Gustafsson, C.; et. al. *Trends Biotechnol*, **2004**, *22*, 346-53.
2004	GeMS	Kodumal, S.J.; et. al. *Proc Natl Acad Sci USA*, **2004**, *101*, 15573-8 Epub2(
2005	Gene Composer	[25]

Table A.2: Table of Unified Nearest-Neighbors parameters, adapted from [23].

Sequence	$\Delta G°$	$\Delta H°$	$\Delta S°$
AA/TT	-1.00	-7.9	-22.2
AT/TA	-0.88	-7.2	-20.4
TA/AT	-0.58	-7.2	-21.3
CA/GT	-1.45	-8.5	-22.7
GT/CA	-1.44	-8.4	-22.4
CT/GA	-1.28	-7.8	-21.0
GA/CT	-1.30	-8.2	-22.2
CG/GC	-2.17	-10.6	-27.2
GC/CG	-2.24	-9.8	-24.4
GG/CC	-1.84	-8.0	-19.9

Table A.3: Codon Usage Table for *Escherichia coli.* Low-frequency codons have been omitted.

Amino Acid	codon fraction values
Gly	GGA 0.15, GGG 0.16, GGU 0.34, GGC 0.35
Pro	CCC 0.14, CCU 0.19, CCA 0.21, CCG 0.47
Ala	GCU 0.19, GCA 0.24, GCC 0.26, GCG 0.31
Val	GUA 0.17, GUC 0.20, GUU 0.29, GUG 0.34
Leu	CUU 0.12, UUG 0.13, UUA 0.15, CUG 0.46
Ile	AUC 0.37, AUU 0.49
Met	AUG 1.00
Cys	UGU 0.47, UGC 0.53
Phe	UUC 0.41, UUU 0.59
Tyr	UAC 0.40, UAU 0.60
Trp	UGG 1.00
His	CAC 0.41, CAU 0.59
Lys	AAG 0.27, AAA 0.73
Arg	CGG 0.12, CGU 0.34, CGC 0.34
Gln	CAA 0.34, CAG 0.66
Asn	AAC 0.48, AAU 0.52
Glu	GAG 0.33, GAA 0.67
Asp	GAC 0.36, GAU 0.64
Ser	UCG 0.13, UCC 0.14, UCA 0.15, UCU 0.17, AGU 0.17, AGC 0.23
Thr	ACA 0.19, ACU 0.19, ACG 0.24, ACC 0.38

Table A.4: A Portion of the Thermally Biased Codon Usage Table for *Escherichia coli*. ΔG values were calculated using the Unified Nearest-Neighbors parameters in Table A.1.

Amino Acid	A—T	T—A	A—A	···	G—G
Arg	CGT $\Delta G = -6.05$ CGG $\Delta G = -6.89$ CGC $\Delta G = -7.13$	CGT $\Delta G = -5.49$ CGG $\Delta G = -6.61$ CGC $\Delta G = -7.16$	CGT $\Delta G = -5.63$ CGG $\Delta G = -6.75$ CGC $\Delta G = -7.30$	···	CGT $\Delta G = -7.30$ CGG $\Delta G = -8.09$ CGC $\Delta G = -8.82$
His	CAT $\Delta G = -5.05$ CAC $\Delta G = -5.61$	CAT $\Delta G = -5.08$ CAC $\Delta G = -5.64$	CAT $\Delta G = -5.22$ CAC $\Delta G = -5.78$	···	CAT $\Delta G = -6.02$ CAC $\Delta G = -7.30$
Met	ATG $\Delta G = -4.77$	ATG $\Delta G = -4.21$	ATG $\Delta G = -4.63$	···	ATG $\Delta G = -5.47$
⋮	⋮	⋮	⋮	⋮	⋮

Bibliography

[1] T. Anada, T. Arisawa, Y. Ozaki, T. Takarada, Y. Katayama, and M. Maeda. The separation of oligodeoxynucleotides having a single-base difference by affinity capillary electrophoresis using oligodeoxynucleotide-polyacrylamide conjugate. *Electrophoresis*, 23(14):2267–2273, July 2002.

[2] T. Anada, M. Ogawa, H. Yokomizo, Y. Ozaki, T. Takarada, Y. Katayama, and M. Maeda. Oligodeoxynucleotide-modified capillary for electrophoretic separation of single-stranded DNAs with a single-base difference. *Analytical Sciences*, 19:73–77, January 2003.

[3] L. Au, F. Yang, W. Yang, S. Lo, and C. Kao. Gene synthesis by a LCR-based approach: High-level production of leptin-l54 using synthetic gene in *escherichia coli*. *Biochemical and Biophysical Research Communications*, 248:200–203, July 1998.

[4] Jeremy M. Berg, John L. Tymocko, and Lubert Stryker. *Biochemistry*. W. H. Freeman and Company, New York, 2002.

[5] S. Bommarito, N. Peyret, and J. SantaLucia, Jr. Thermodynamic parameters for DNA sequences with dangling ends. *Nucleic Acids Research*, 28(9):1929–1934, May 2000.

[6] D. Buchanan, E. Jameson, J. Perlette, A. Malik, and R. Kennedy. Effect of buffer, electric field, and separation time on detection of aptamer-ligand complexes for affinity probe capillary electrophoresis. *Electrophoresis*, 24(9):1375–1382, May 2003.

[7] G. Church and T. Knight. Synthetic biology. electronic mail correspondence, February 2004.

[8] S. Cohen, A. Chang, H. Boyer, and R. Helling. Construction of biologically functional bacterial plasmids *in vitro*. *Proceedings of the National Academy of Sciences of the United States of America*, 70(11):3240–3244, November 1973.

[9] F. Eckstein, editor. *Oligonucleotides and Analogues*. Oxford University Press, New York, 1991.

[10] R. Frank, D. Müller, and C. Wolff. Identification and suppression of secondary structures formed from deoxy-oligonucleotides during electrophoresis in denaturing polyacrylamide-gels. *Nucleic Acids Research*, 9(19):4967–4979, October 1981.

[11] C. Heller and J. Viovy. *Brief Report:* Electrophoretic separation of oligonucleotides in replenishable polyacrylamide-filled capillaries. *Applied and Theoretical Electrophoresis*, 4:39–41, 1994.

[12] S. Ikuta, K. Takagi, R. Wallace, and K. Itakura. Dissociation kinetics of 19 base paired oligonucleotide-DNA duplexes containing different single-mismatched base pairs. *Nucleic Acids Research*, 15(2):797–811, January 1987.

[13] K. Ketomäki, H. Hakala, O. Kuronen, and H. Lönnberg. Hybridization properties of support-bound oligonucleotides: The effect of the site of immobilization on the stability and selectivity of duplex formation. *Bioconjugate Chemistry*, 14(4):811–816, July–August 2003.

[14] T. Knight and A. Wozniak. Synthetic biology urop. notes from conversations with TK, April 2004.

[15] http://www.merriampark.com/ld.htm.

[16] P. Levison, S. Badger, J. Dennis, P. Hathi, M. Davies, I. Bruce, and D. Schimkat. Recent developments of magnetic beads for use in nucleic acid purification. *Journal of Chromatography A*, 816:107–111, August 1998.

[17] U. Maskos and E. Southern. A novel method for parallel analysis of multiple mutations in multiple samples. *Nucleic Acids Research*, 21(9):2269–2270, May 1993.

[18] http://ncbi.nlm.nih.gov/.

[19] Y. Ozaki, T. Ihara, Y. KKatayama, and M. Maeda. Affinity capillary electrophoresis using DNA conjugates. *Nucleic Acids Symposium Series*, (37):235–236, 1997.

[20] Jerome K. Percus. *Mathematics of Genome Analysis.* Cambridge University Press, Cambridge, United Kingdom, 2002.

[21] J. Rouillard, W. Lee, G. Truan, X. Gao, X. Zhou, and E. Gulari. Gene2Oligo: oligonucleotide design for *in vitro* gene synthesis. *Nucleic Acids Research*, 32:W176–W180, July 2004.

[22] Stuart J. Russell and Peter Norvig, editors. *Artificial Intelligence: A Modern Approach.* Pearson Education, Inc., Upper Saddle River, New Jersey 07458, 2003.

[23] J. SantaLucia, Jr. A unified view of polymer, dumbbell, and oligonucleotide DNA nearest-neighbor thermodynamics. *Proceedings of the National Academy of Sciences of the United States of America*, 95(4):1460–1465, February 1998.

[24] N.V. Sorokin, Chechetkin V.R., Livshits M.A., Pan'kov S.V., Donnikov M.Y., Gryadunov D.A., Lapa S.A., and Zasedatelev A.S. Discrimination between perfect and mismatched duplexes with oligonucleotide gel microchips; role of thermodynamic and kinetic effects during hybridization. *J Biomol Struct Dyn.*, 22(6):725–734, June 2005.

[25] L. Stewart and A. Burgin. Whole gene synthesis: A gene-o-matic future. Whole Gene Synthesis Review Article for FDDD 2005, 2005.

[26] J. Tian, H. Gong, N. Sheng, X. Zhou, E. Gulari, X. Gao, and G. Church. Accurate multiplex gene synthesis from programmable DNA microchips. *Nature*, 432(7020):1050–1054, December 2004.

[27] http://www.kazusa.or.jp/codon/.