Training Finite State Classifiers to Improve PCR Primer Design

Daniel A. Ashlock  
Mathematics Department  
Bioinformatics Program  
Iowa State University,  
Ames, Iowa 50011  
danwell@iastate.edu

Kenneth M. Bryden  
Mechanical Engineering Department  
Iowa State University,  
Ames, Iowa 50011  
kmbryden@iastate.edu

Steven A. Corns  
Mechanical Engineering Department  
Iowa State University,  
Ames, Iowa 50011  
scorns@iastate.edu

Patrick S. Schnable  
Agronomy Department  
Iowa State University,  
Ames, Iowa, 50011  
schnable@iastate.edu

Tsui-Jung Wen  
Agronomy Department  
Iowa State University,  
Ames, Iowa, 50011  
tjwen@iastate.edu

Abstract

We present results on training finite state machines as classifiers for polymerase chain reaction primers. The goal is to decrease the number of primers that fail to amplify correctly. Finite state classifiers are trained with a novel evolutionary algorithm that uses an incremental fitness reward system and multi-population hybridization. The system presented here creates a post-production add-on to a standard primer picking program intended to compensate for organism and lab specific factors.

1 Introduction

The design of polymerase chain reaction (PCR) primers is a well-studied problem when considered in terms of the biophysics of DNA. Software for picking primers that amplify known sequences of DNA, such as the Primer 3 program from the NCSA Biology Workbench, are available for free download. Designing primers that not only have good biophysical properties but also take into account idiosyncratic features of particular organisms, labs, and
technicians is a more difficult. It is probably an impractical goal for a general purpose primer design tool to incorporate such features. This study presents a method for creating project-specific post processing tools for primer picking that can partially compensate for organism and lab specific factors affecting the success of PCR primer amplification. Creation of these tools requires that sufficient primers have been tested within the context of a project to provide a set of training data. With a large set of primers available as training data, an evolutionary algorithm uses the success and failure data for primers to train finite state classifiers (FSCs) that predict if a given primer pair will or will not succeed in amplification. We developed these tools as part of a genomics and genetic mapping project in *zea mays* (corn). The training data used in an effort of this kind are noisy for several reasons. Perfectly good primers are sometimes given bad scores because of errors made while running a PCR amplification. Since primers work in pairs, a good primer may end up being scored as a bad primer because of flaws in its partner. Finally, the process of scoring primers into the categories “worked” and “did not work” is itself somewhat subjective.

### 1.1 What are PCR primers?

PCR primers are short sequences of DNA used in a biochemical reaction that amplifies other DNA. PCR amplification of DNA underlies a multitude of technologies from forensic DNA fingerprinting to techniques for finding analogs for a known gene in one organism in other organisms (gene fishing). DNA is normally in the form of a double strand with the opposite sides of the strand composed of complementary bases of DNA. When heated the strands fall apart. When subsequently cooled the strands rejoin (anneal) to strands that complement them. The key to PCR amplification is that smaller segments of DNA anneal faster than longer ones and there is an enzyme (a DNA polymerase) that will extend a partial double strand of DNA as shown in Figure 1. In a PCR amplification, the primers are typically chains of 17-25 DNA bases, much smaller than the target DNA to be amplified. This means that, if the primers complement part of the target DNA, they will often anneal earlier than other DNA in the solution. After annealing, the part of the target DNA where the primer landed is double-stranded but the remainder of the target DNA is single-stranded. At this point the DNA polymerase goes to work extending the double strand and making a complementary copy of one strand of the target DNA.

If we had only one primer present in the reaction then it would make complementary copies of only one of the two strands of the target DNA. This would lead to one copy per strand of original target DNA per heating and cooling cycle. Since the strand serving as a template is not copied, but rather keeps having complements of itself produced, the result is linear amplification. This is why PCR amplification uses two primers, one for each strand of the target. As the PCR amplification reaction runs through multiple heating and cooling cycles, each of which roughly doubles the number of copies of DNA between two paired primers, geometric amplification takes place.

A good pair of PCR primers must have roughly the same melting temperature for the reaction to proceed efficiently. The most important factors controlling melting temperature are primer length and base composition. There are four DNA bases, G, C, A, and T. They complement one another with G binding to C and A to T. The G-C bond is stronger than the A-T bond and so, in general, sequences with higher GC content have higher melting temperatures. There are also nearest-neighbor interactions that affect melting temperature. A primer-picking program will examine a given stretch of DNA for a pair of primers flanking
CCAGTGTTACTAGGCTACTACTGCGACTACG

GGTCACAATGATCGATGATGACGCTGATGC

GGTCAC==>>

CCAGTGTTACTAGGCTACTGCGACTACG

GGTCACAATGATCGATGATGACGCTGATGC

1.2 The Problem

If PCR primer design following the principles outlined above, is already algorithmic, then what remains to be done? A key point is that a primer must not anneal in more than one place. If it does, then more DNA than the target may be amplified. When we attempt to amplify DNA we typically do not start with a pure sample of the target stretch of DNA. We may be amplifying DNA derived from blood found under the fingernails of a crime victim or DNA purified from corn seedlings. In either case, the entire genome of an organism is present when we begin the PCR reaction. The DNA sequence of a primer that is supposed to amplify a particular target may, because of duplication of sequence in the genome, appear multiple times and so confound the PCR reaction. If DNA in an organism were picked at random, having a primer appear more than once would be a shocking statistical anomaly. Evolution, however, functions in many cases by duplicating genetic material and then modifying one of the copies. Because of this, many long sequences are duplicated multiple times in a genome. Unless a primer design program has access to the entire genome of a creature it cannot compensate for this source of bad PCR reactions. Corn does not have a sequenced genome and it also has a large number of transposable elements. Sometimes called “jumping genes”, these DNA sequences copy themselves, functioning as a kind of genetic parasite, and act to make the problem of long, repeated sequences acute.

In a long-term project in which hundreds or thousands of PCR primers are designed for a single organism, machine learning can be used to detect such confounding sequence redundancy. This forms the primary justification for our attempts to train FSCs to pick from among multiple primer pairs designed for a given target. In addition to the relatively obvious problem of redundant annealing sites for a primer, PCR reactions can fail for other reasons. These include unsuspected biochemistry, incorrect lab technique, or variations in equipment or reagents. An attempt to discover which classes of primer work in the context
of a particular project may allow some of the lab-specific factors that interfere with efficient PCR to be detected.

# 2 Representation and Fitness

Two key features of any evolutionary-computation-based machine learning system are the data structure holding the putative problem solutions to be evolved and the fitness function. Our data structure is a finite state machine with 64 states and transitions driven by DNA bases. Finite state machines are a standard representation for diverse tasks in evolutionary computation [4, 6, 9, 5, 10]. Figure 2 shows part of such a machine. The states of the FSC have three possible types: ? (don’t know), + (good primer), and - (bad primer). These state labels are used to permit the finite state machine to function as a classifier, though not in the usual manner, as we will see when the fitness function is specified. Our choice of a finite state device has a useful feature for training on PCR primer data. The primers are of variable length and so some representations, e.g. neural nets, require missing variables to be filled in or the variable length DNA strand to be reduced in some way, to a fixed number of variables. With a finite state device we simply drive the device with the DNA data until we reach its end.

For training a population of FSCs we used a collection of 2000 PCR primer pairs, 1000 of which amplified correctly and 1000 of which either failed to amplify at all or apparently amplified multiple targets. The fitness of an FSC on a training set of primers is computed as follows. Each PCR primer is run through the FSC. As the machine passes through each state, values from Table 1 are summed. These numbers represent complete neutrality to
all factors except “it works” and “it doesn’t work.” Fitness for an FSC is summed over all
primers in the training set.

<table>
<thead>
<tr>
<th>Primer is</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>Good</td>
<td>+</td>
</tr>
<tr>
<td>Bad</td>
<td>-1</td>
</tr>
</tbody>
</table>

Table 1: Incremental payoffs per state for FSCs.

This fitness function rewards the FSC incrementally, after each state transition. Scoring
FSCs on their accuracy of their final state and no other only was attempted in a preliminary
study and worked badly. When only the final state of the FSC (+, -) was used in scoring,
machines that refused to classify most primers (in which states labeled with ? formed a
majority) were a common outcome. Our hypothesis is that the incremental reward acts to
smooth the fitness landscape, in essence simplifying the evolutionary search problem. In
addition, the incremental fitness permits the FSCs to be indecisive. Large positive scores
indicate a primer that the FSC classifies as quite likely to work correctly, large negative
scores are votes of no confidence in a primer. Scores near zero, however, indicate either
ignorance (the primer is of a type not encountered before) or confusion (primers with good
and bad sequence features). For picking new primers, we picked the highest scoring pair that
both returned positive scores.

3 Experimental Design

A population of 600 FSCs evolved for 1000 generations were used when training classifiers.
The FSCs are initialized uniformly at random, filling in both transitions and state labels at
random. Our model of evolution is single tournament selection with tournament size four.
The population is randomly shuffled into groups of four. The two most fit FSCs reproduce
and replace the two least fit in each group. Reproduction treats the string of states in an
FSC as a linear chromosome. The two FSCs reproducing are copied, the copies undergo two-
point crossover, and then each copy is subjected to one mutation. The mutation modifies
the initial state of the FSC 10% of the time, randomly picks a new destination for one of
the transitions 30% of the time, and modifies the label \{+, -, ?\} on a state 60% of the
time. During crossover the initial state of the FSC moves with the first state. One hundred
simulations with distinct starting populations were performed, saving the best FSC from
each simulation.

The best-of-simulation FSCs are used to initialize additional sets of simulations which
we term hybridizations. For other instances of hybridization in the context of evolutionary
computation see [1, 2]. These simulations are identical to the first set except that 100
members of the initial random population are replaced with the best-of-run FSCs from the
first 100 simulations. The other members of these initial populations are generated uniformly
at random.
### Table 2: Predictions versus truth results for the most fit FSC located during the first set of simulations.

<table>
<thead>
<tr>
<th>Training data</th>
<th>Crossvalidation Set</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Prediction</strong></td>
<td><strong>Prediction</strong></td>
</tr>
<tr>
<td>+  -  ?</td>
<td>+  -  ?</td>
</tr>
<tr>
<td><strong>Good</strong></td>
<td>Good</td>
</tr>
<tr>
<td>666 256 78</td>
<td>115 106 29</td>
</tr>
<tr>
<td><strong>Bad</strong></td>
<td>Bad</td>
</tr>
<tr>
<td>287 659 54</td>
<td>96 125 29</td>
</tr>
</tbody>
</table>

### Table 3: Predictions versus truth results for the most fit FSC located during the second set of simulations, the first set of hybridization.

<table>
<thead>
<tr>
<th>Training data</th>
<th>Crossvalidation Set</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Prediction</strong></td>
<td><strong>Prediction</strong></td>
</tr>
<tr>
<td>+  -  ?</td>
<td>+  -  ?</td>
</tr>
<tr>
<td><strong>Good</strong></td>
<td>Good</td>
</tr>
<tr>
<td>626 337 37</td>
<td>104 137 9</td>
</tr>
<tr>
<td><strong>Bad</strong></td>
<td>Bad</td>
</tr>
<tr>
<td>181 802 17</td>
<td>79 161 10</td>
</tr>
</tbody>
</table>

### Table 4: Predictions versus truth results for the most fit FSC located during the third set of simulations, the second set of hybridization.

<table>
<thead>
<tr>
<th>Training data</th>
<th>Crossvalidation Set</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Prediction</strong></td>
<td><strong>Prediction</strong></td>
</tr>
<tr>
<td>+  -  ?</td>
<td>+  -  ?</td>
</tr>
<tr>
<td><strong>Good</strong></td>
<td>Good</td>
</tr>
<tr>
<td>636 364 0</td>
<td>127 107 16</td>
</tr>
<tr>
<td><strong>Bad</strong></td>
<td>Bad</td>
</tr>
<tr>
<td>171 829 0</td>
<td>95 132 23</td>
</tr>
</tbody>
</table>
Figure 3: Histograms of the distribution of fitnesses. Top to bottom, for the first set of simulations, for the first set of hybridizations, and for the second set of hybridizations.
4 Results

In order to get a classification of a primer as good or bad we count the number of + or − states encountered as the FSC is driven by the primer. If the number of + and − states tie, we fail to classify the primer. Otherwise, the more common result is taken to be the good/bad prediction. For the initial set of simulations the best classifier achieved 1325 correct classifications, 543 incorrect classifications, and had 132 failures to classify on the training data. On a 500 element reserved cross-validation set the classifier achieved 240 correct classifications, 202 incorrect classifications, and 58 failures to classify. The breakdown of correct and incorrect predications and failures to classify are given in Table 2. This level of performance, if continued in subsequent performance, represents a nontrivial savings (roughly $4.50 per bad primer avoided). An analysis of potential economic impact appears in the discussion section.

![Figure 4: Best fitness in the population as a function of generations for the four runs (20, 22, 39, and 58) that produced exceptional fitness with a typical run for comparison purposes.](image)

The distribution of final best-of-run fitnesses, given as the top histogram in Figure 3, shows that four populations broke out far ahead of the pack. The plot of best fitnesses in each generation for these four runs and for an additional more typical run are given in Figure 4. These four exceptional runs do not show sudden jumps in fitness; rather they exhibit steady progress at a greater rate then the other 96 runs. This suggests that the fitness landscape on which the search is taking place contains long uphill paths that are not difficult to miss.

Examination of the saved FSCs showed that the majority of populations use more ? states than those found by the four high-performing populations. This suggests that the superior populations have recognized some sort of pattern in the data that escapes the others. This phenomenon of “pack leaders” is what motivated the use of hybridization.

The first set of hybridization studies produced improvements in both fitness, shown in Figure 3, and classification performance, with classification on the cross-validation set passing 50% correct. Table 3 gives a breakdown of the performance of the best classifier located in the first set of hybridization runs.

The behavior of fitness was, not surprisingly, quite different during the hybridization
runs. The simulation that yielded the best classifier is plotted in Figure 5. Best fitness  
remains flat for roughly 50 generation and then begins a slow, steady upward progress.  
The commencement of this progress coincides with a cessation of a steep slope in the mean fitness  
plot as the 500 random classifiers placed in the initial population are eliminated by crossover  
and selection and as the population settles around some consensus genotype.

Figure 5: Best and mean fitness as a function of generations for the run in the  
first set of hybridization studies that yielded the best classifier.

The best-of-run fitness distribution for the first set of hybridization studies also had some  
high-end outliers. An additional hybridization study was run and the resulting fitnesses are  
shown in Figure 3. Table 4 shows the classification scores on the training and cross validation  
data. These data indicate that the best classifier in the third hybridization is over-trained  
on the data. Its classification of the cross-validation data has become worse.

5 Discussion

The training data used in this study were drawn from an ongoing genetic mapping project  
in the Schnable Lab at Iowa State University. Primers are used in pairs to amplify parts of  
the corn genome that exhibit size polymorphisms as a means of tracing the co-inheritance of  
genes. The primers are designed and used in pairs. Failures can result from flaws in one of  
the primers, in both, or in a mismatch between the two primers. This means that some of  
the “bad” primers in the training data are in fact primers that might work just fine with a  
different partner. In addition PCR reactions sometimes go awry for reasons unrelated to the  
primers themselves. Equipment failure, operator failure, or over-age reagents can all derail  
a reaction being run with good primers. This means that our positive examples are almost  
all correct while our negative examples are partly wrong.

5.1 Economic Impact

Since many potential targets for mapping are available and, typically, several primer pairs  
can be designed for each target, rejecting good primers is a less important problem than
failing to reject bad primers. The gain from employing the best classifier found is given by computing the bad primers avoided. To estimate this gain the following assumptions are used;

(i) The failure statistics for multiple pairs of primers designed for the same target are the same.

(ii) All primers classified as other than good will be redesigned.

(iii) The behavior of the best classifier (given in Table 3) on the cross-validation data is representative.

20.8% of all primers are correctly accepted as good, 15.8% are incorrectly accepted as good. The remaining primers are not used and their targets are put back into the primer design pipeline to have new primers designed. After the redesign, we will thus should see a 7% drop in the number of bad primers actually accepted. Given that 8,276 primers failed to work in the course of the first phase of the mapping project, about 580 bad primer pairs, 1160 primers, or $5,200 would have been saved. Once the machine learning system has been built, the cost of running these checks is a nominal, requiring a simple modification of the primer design pipeline.

5.2 Incremental Fitness

The fitness function reported here incrementally rewards a given FSC, permitting the classifier to have multiple opinions in the course of processing a primer. This was done to compensate for poor performance when the FSCs were scored only on a final call on each primer. We conjecture that the effect of the incremental reward is to smooth the fitness surface. The shift of incremental fitness did have the effect of eliminating a large local optima in which the FSCs refused to classify many of the primers in the training data.

For some of the simulations, we studied a trace of mean fitness and examined the classification behavior of FSCs in those populations. Evolution of the examined populations had the following points in common;

• Initial populations, which essentially guess randomly, receive a fitness of near zero with the reward numbers given in Table 1.

• Refusal to classify is a common behavior as it yields an adequate fitness and is, perhaps, one of the most crossover stable types of genome.

• When incremental fitness is used the populations break out of the refusal to classify state. Subsequent fitness progress does not seem to follow an obvious pattern. This suggests that the training data contain many patterns which yield, via order of discovery, different paths of increasing fitness.

The improvement in performance forms a post-hoc justification of the incremental fitness function but it is the case that we are using the incremental fitness as a surrogate for our true target: correct classification. An earlier study [3] was performed that used incremental fitness for half of each of the simulation and then shifted to the ratio of correct to incorrect classifications. The results were intermediate in quality between the initial and hybridized
runs reported in that study. Subsequent hybridization of those populations was not as useful, with final classification accuracy of hybridized populations being similar.

The numbers in Table 1 used to compute incremental fitness were chosen to match the neutral character of the training data with its equal number of good and bad examples. Since rejecting negative primers is more important for the system’s potential economic impact, resetting the numbers to emphasize correct results on bad primers is a definite next step.

6 Conclusions

The primary goal of this study was to improve primer design performance by using machine learning as post-processor to capture features of primer performance not related directly to the DNA biophysics already embedded in primer-picking packages. The technique yields improved performance. A study is currently underway to assess the impact of the FSCs on actual wet-lab performance and more definitive claims of success wait on these wet-lab results.

The novel features of this study, aside from its main goal, are the use of a per-state incremental fitness function and the use of hybridization. This fitness function was developed in response to poor performance of standard fitness functions (per test case performance) on the good/bad primer prediction problem. Hybridization has been used by one of the authors before [1, 2] on substantially different problems with positive results. Other researchers are encouraged to try hybridization studies on their specific problems. The goal of such studies, beyond improving performance on particular problems, would be to start to characterize the border between problems where hybridization is valuable and problems where it is not valuable.

The notion of building blocks [7, 8], sub-structures of members of an evolving population that can be mixed and matched by crossover to enhance evolutionary search, is a perennial topic in evolutionary computation research. It is intuitive that hybridization should function well when building blocks are available in a representation. It is less intuitive that hybridization should fail in the absence of building blocks, as crossover with structures evolved in a different population might serve as a useful macro-mutation. We suggest that hybridization studies may be able to help identify systems in which building blocks exist. This is an area for future study.

7 Next Steps

The clearest areas for additional work on this project are the fitness function and studying how to perform hybridization effectively.

7.1 Improving the Fitness Function

The opinion of one of our FSCs after it has seen only a few bases of a primer is probably worthless. In fact many of the FSCs we found have the ? state label on many of the states encountered early during the processing of many primers. Following this notion, it might improve performance to weight incremental rewards gained later in the processing of a primer higher than those gained earlier. A preliminary study of this technique yields mixed results, but there are many possible schemes for increasing the weight of reward as time goes by.
We also reported in Section 5 that a preliminary study of shifting from incremental reward fitness to the ratio of correct to incorrect predictions was promising, but, after hybridization, did not enhance final best performance. This preliminary study looked at only one way to shift from incremental reward to prediction accuracy. Since prediction accuracy is what we actually desire, schemes for involving prediction accuracy more directly in assessment of the FSCs merit additional study.

At present the incremental reward for states labeled with a ? is zero. A small study of the results of setting the reward for ? states to -1 substantially degraded performance. The logic behind that experiment was that a small negative score for ? states would encourage machines to make some guess about almost every primer. It did have this effect, but the FSCs made more new bad guesses than new good guesses. Since some ? states are probably required early in the processing of a primer, it may be that such negative rewards should be phased in over the course of the examination of a primer.

As noted in Section 5, primer pairs work or fail together. This study, however, trains on the individual primers. Feeding both primers through the FSC and using the resulting fitness was tried and worked badly. In spite of this, schemes for testing the primers as a pair should be considered for future studies.

7.2 Understanding Hybridization

Hybridization, the seeding of a set of simulations with best-of-run structures from another collection of simulations, can help. In this study we have verified that hybridization improves both fitness and prediction accuracy for the PCR primer FSCs. In Reference [2] it was demonstrated that hybridization helped in code induction for the Tartarus [11] problem. In Reference [1] hybridization was critical in evolving simple optical character recognition systems. While we have examples to show that hybridization can help, we still lack the answers to at least two questions. What sort of problems will hybridization help with? What sort of schedule of stopping and starting simulations with transfer of best-of-run structures to new starting populations is good?

The first question is open ended and quite difficult to answer. We lack a good taxonomy of problems and so at present are left to pile up examples. Clearly problems where there is a unique best answer that is easy to find will not benefit from hybridization. We conjecture that if a given representation of solutions for a problem exhibits a high level of epistasis then hybridization will not help.

The second question, that of finding effective hybridization schedules, is likely to have task-specific answers. It is clear that if multiple hybridizations are to be performed, some evolution must happen before re-hybridization takes place. Hybridization can be viewed as a more extreme version of the selection technique used in “Island Genetic Algorithms” [12]. It would not be difficult to test different hybridization schedules for particular problems. This is a part of the authors ongoing research.

7.3 Detecting Critical Patterns

If our FSCs are detecting some systematic flaws in PCR primers that are not apparent to the code in standard primer design packages, then those patterns are themselves of interest. We give two speculative procedures for locating and understanding these patterns.
• Systematic enumeration of bad patterns could be performed by tracing those paths through the evolved FSCs that lead to large negative scores in multiple classifiers. A potential problem with this approach is that many of the patterns located might not be good primers according to standard primer design software and might not even appear in corn (or any organism).

• A modified systematic enumeration in which known DNA from corn is fed through the FSCs in primer-sized chunks and then the aggregate score from several FSCs are plotted along the length of the DNA to see what features are detected by relatively high and low scores. It would not be surprising if known or unknown transposable elements were detected in this fashion.

8 Acknowledgments

This research was supported by a competitive grant from the National Science Foundation Plant Genome Program (award number: DBI-9975868) to Patrick S. Schnable, Daniel Ashlock, and others. We would like to thank the members of the Schnable Lab for their support of this work and the members of the Complex Adaptive Systems program at Iowa State for helpful discussions.

References


