Automata & DNA Computing Project Report

Logan Lopez
loganlopez@mail.usf.edu
University of South Florida
Tampa, Florida

Autumn Chavez
amchavez@mail.usf.edu
University of South Florida
Tampa, Florida

Carlos Leon
cleon@mail.usf.edu
University of South Florida
Tampa, Florida

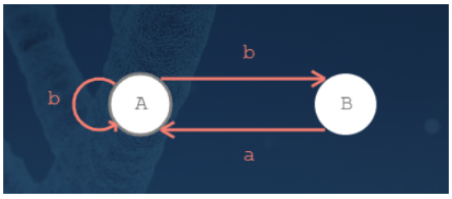
ACM Reference Format:
Logan Lopez, Autumn Chavez, and Carlos Leon. 2019. Automata & DNA Computing Project Report. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The goal for this project was to produce a visually appealing model. The model was to show the translation given a selected input string from the alphabet into a strand of DNA. Then show for each transition which enzymes are used animated in sequence. Lastly, the final generated strand or the failed word and state of the automaton. for the following language:

$$L = \{(b + ba)* : w \in a, b*\}$$

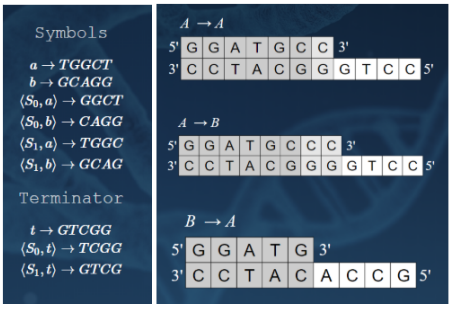
We implemented the Ben Shapiro model, a two state finite state automata using DNA computing. This example is modelled off of the language $L = \{(b + ba)*\}$.



An input string is processed into DNA using set transition rules, spacers, and a terminator. Acceptance or rejection of the input string is determined by taking the created DNA strand and using repeating cycles of the Ligase and FokI enzymes. These enzymes either ligate the input words or cleave the obtained strand, which means the next input word is eliminated. If this process leads to the termination sequence, see Figure 2 and 3, then the input is accepted.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn



Overall, our implementation shows the process of evaluating a DNA strand given an input value for the two state finite automata and expands on how it is rejected or accepted for the language. This project was tested using various input strings and provided correct results.

Input	Expected	Result
bab	Accepted	Accepted
baba	Accepted	Accepted
b	Accepted	Accepted
bb	Accepted	Accepted
bba	Accepted	Accepted
babbab	Accepted	Accepted
a	Rejected	Rejected
aa	Rejected	Rejected
aab	Rejected	Rejected
aba	Rejected	Rejected
aaba	Rejected	Rejected
baa	Rejected	Rejected

The Ben Shapiro model of DNA computing has some advantages when compared to traditional DNA sequencing. Mainly in that, that the automaton sequences the relevant DNA/RNA without the need of any super specialized computing power. It does, however, have some downsides. Its lack of complexity, means that for a practical experiment more than one automaton is required. For example, Ben Shapiro did an experiment to detect prostate cancer genes in strand of mRNA using an automaton and had to use three automatons. Furthermore, he his automaton had to do something upon rejection - in this case release a drug suppressant. This is an important point, because the final state of the automaton needs to be checked in order to get the diagnosis, but doing so will require the automaton to release a drug into a system that is detectable. Another downside, is that the transitions are all predetermined in their DNA forms, thus will fail if anything can interfere with a specific binding.

There have been advances in the field of DNA computing in the last two decades beyond that of the simple Ben Shapiro Finite State Machine, however, given the lack of Chemistry and Biology knowledge among our peers, we felt this was a better visual representation. The advances, to describe briefly, push DNA computing from 2 states to 3, use more hardware enzymes, and allow for greater interaction between the enzymes. The latest advancement push DNA computing into Push Down Automata territory. While this may be a great leap, it does rely on a specific sort of DNA where the strands at both ends have been bound together. Furthermore, the stack certainly does not have infinite space, nor can it be accessed without cost.