darwin

Scalable Version Control of Genomic Data

Vanderbilt Software

Introduction

Software

The synthetic biology workflow currently iterates through newer genomic designs by editing DNA text directly. DNA editing software has become a highly graphical display tool which heavily interacts with its user, offering advanced capabilities such as integrated BLAST requests and a huge variety of error-checking and annotation capabilities. Despite all these invaluable features, the text editor has still broken no new ground in tracking changes made to the DNA. No integrated software package exists to track changes in a distributed, secure, and efficient manner. darwin is a software package to document changes to DNA which allows for easy, standardized, and collaborative editing on the genome scale.

Background

Coding

Programmers need to track even minute changes to their code exhaustively, especially as the project scales up. Typical tools used for this process include git and svn, which create 'diffs" which capture differences between files and send them over a network to collaborators.

However, no tool currently exists to support genome-scale changes in a useful way. Programming version control tools can be used to cover DNA data, but they aren't used to working with that sort of input and are extraordinarily inefficient. darwin is a way around that. It uses version control tools as a backend, so it takes advantage of their strength. But it preprocesses the DNA data to optimize these tested and proven methods of version control, producing a much more efficient and secure tracking system built from the ground up to be able to scale to genome-size files. darwin is completely open-source, so any security or optimization issues can be identified and solved immediately to help all users of the tool. It's also agnostic to the type of version control backend used, so it can be put into place on a massive variety of systems.

Biology

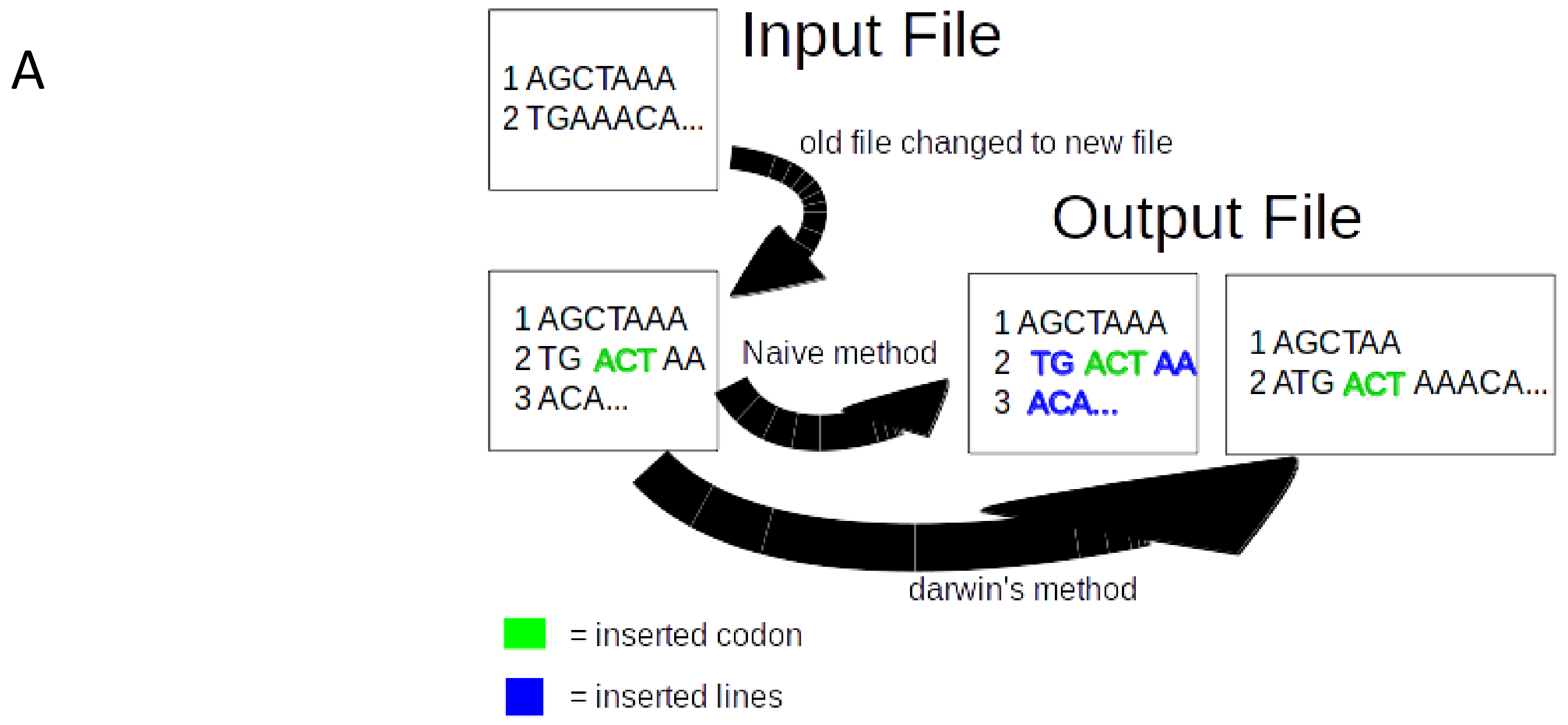
The basis for the workings of any organism is rooted in its proteins as provided for in its genome. Every single protein can be traced back to an open reading frame, a set of base pairs defined by a start and stop sequence that evenly divides into codons (sets of 3 base pairs). Every codon corresponds to an amino acid and through the processes of transcription and translation, proteins are manufactured in the cell.

While DNA is a relatively stable molecule, different things can occur causing a mutation in the genetic code. There are several different types of mutations which are defined by how they affect protein translation downstream, specifically insertions, deletions, and substitutions. These can either be synonymous meaning that although there is a changes to base pair, the codon still codes for the same amino acid so there is no detectable change, or nonsynonymous which may affect the makeup of the protein. Lastly, there are frameshift mutations which generate an entirely different open reading frame.

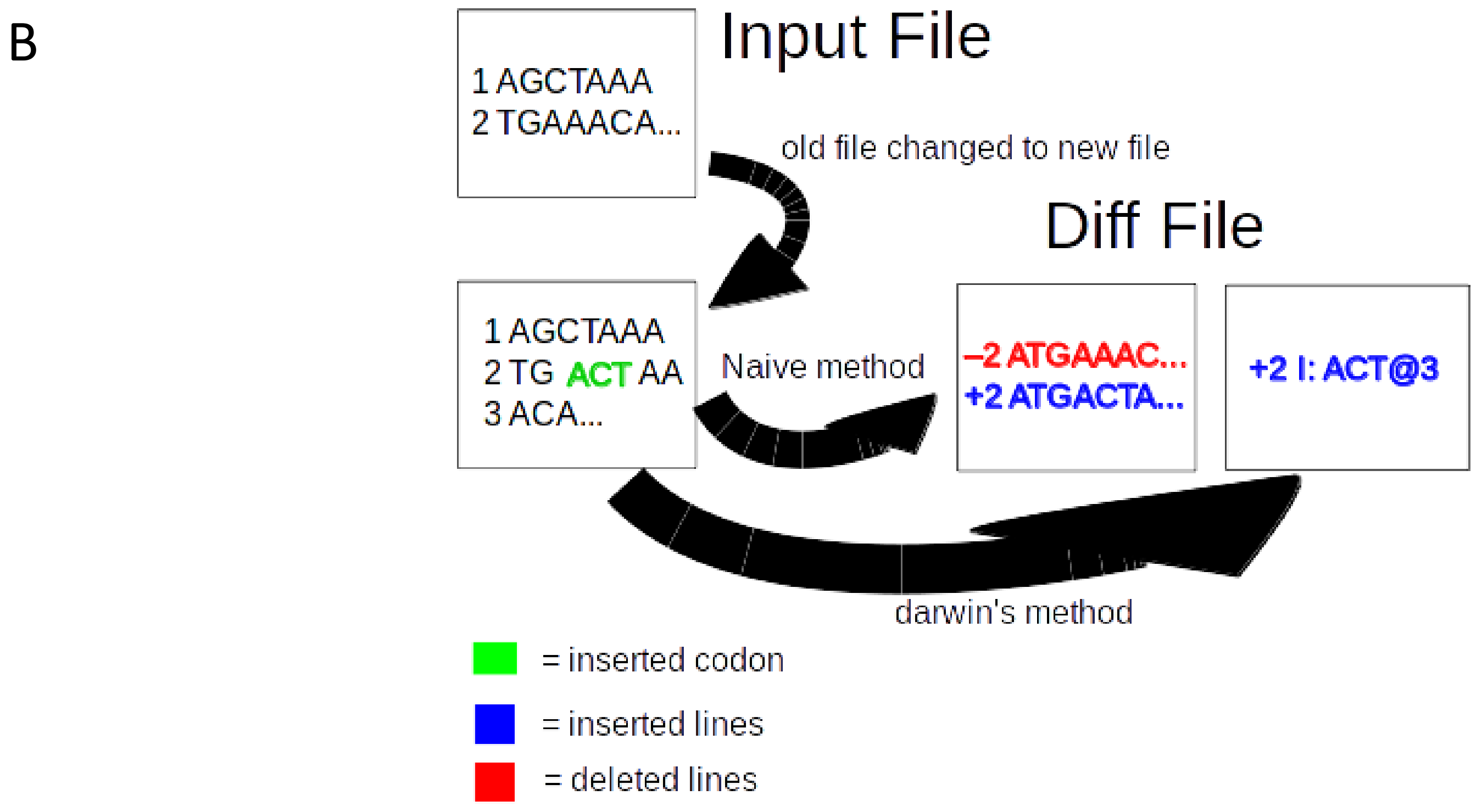
Algorithm

darwin's Novel Approach

git, svn, and other version control systems focus on differences between lines. Since most DNA file formats split DNA to fixed-length lines, many lines are changed at once, for example, when inserting a single new line. darwin does away with that by producing a formatted file representing each ORF on its own line of text, making each edit only modify a single line of the output text.



Genes can be very long. To combat this, darwin will sample a section of every newly inserted ORF and compare it to nearby ORFs; if the new ORF is similar to another ORF, it is counted as "edited," and darwin only records the character-by-character changes required to transform the old ORF into the new ORF.



Finally, darwin uses concurrency to help speed up the process. File I/O is typically extremely slow, much slower than processing a block of file data already in memory. Splitting the processing concurrently helps to open up that speed bottleneck.

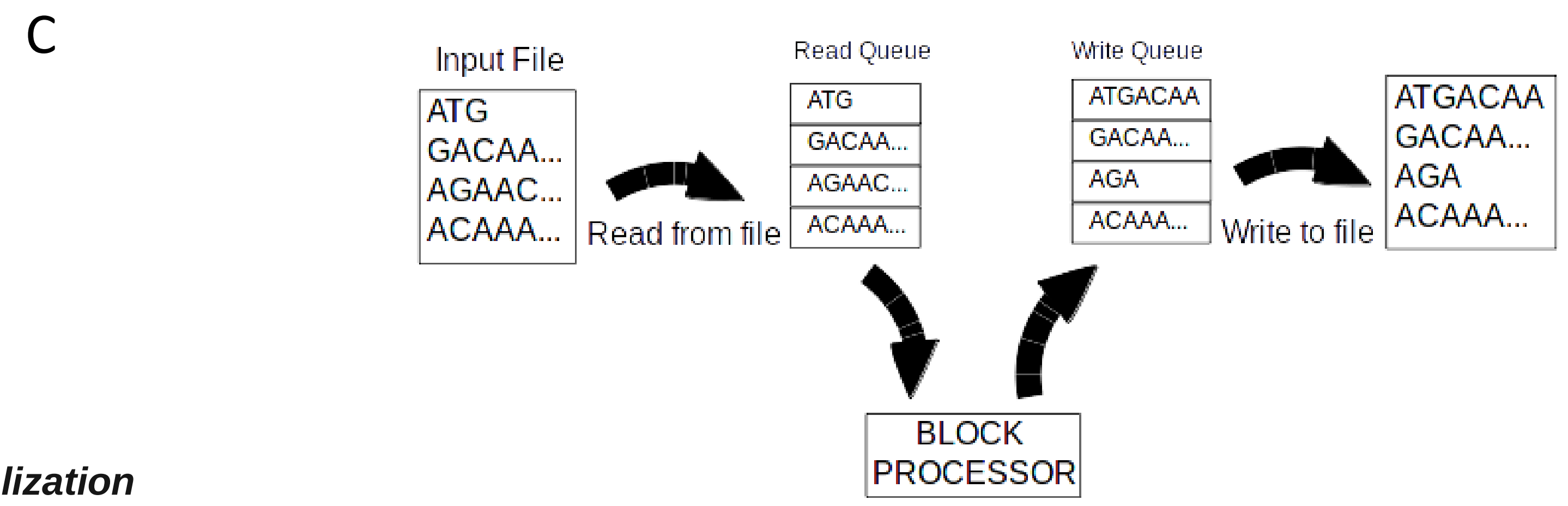


Figure 1. Code Visualization
(A)darwin eliminates extra lines in the output file
(B)darwin's unique method of parsing ORF
(C)Representation of darwin's block processor increasing processing speed

Analysis

Results

We produced software which implemented these algorithms on specially cleaned input files and ran them through a typical version control system. We performed experimentation on transformations between iterations 3-6 of our wetware team's yeast plasmids. As a control, we compared darwin's output running through the git version control system to simple processing with git (vanilla_git). We expected darwin to run faster and show fewer changed lines in the diff output than vanilla_git.

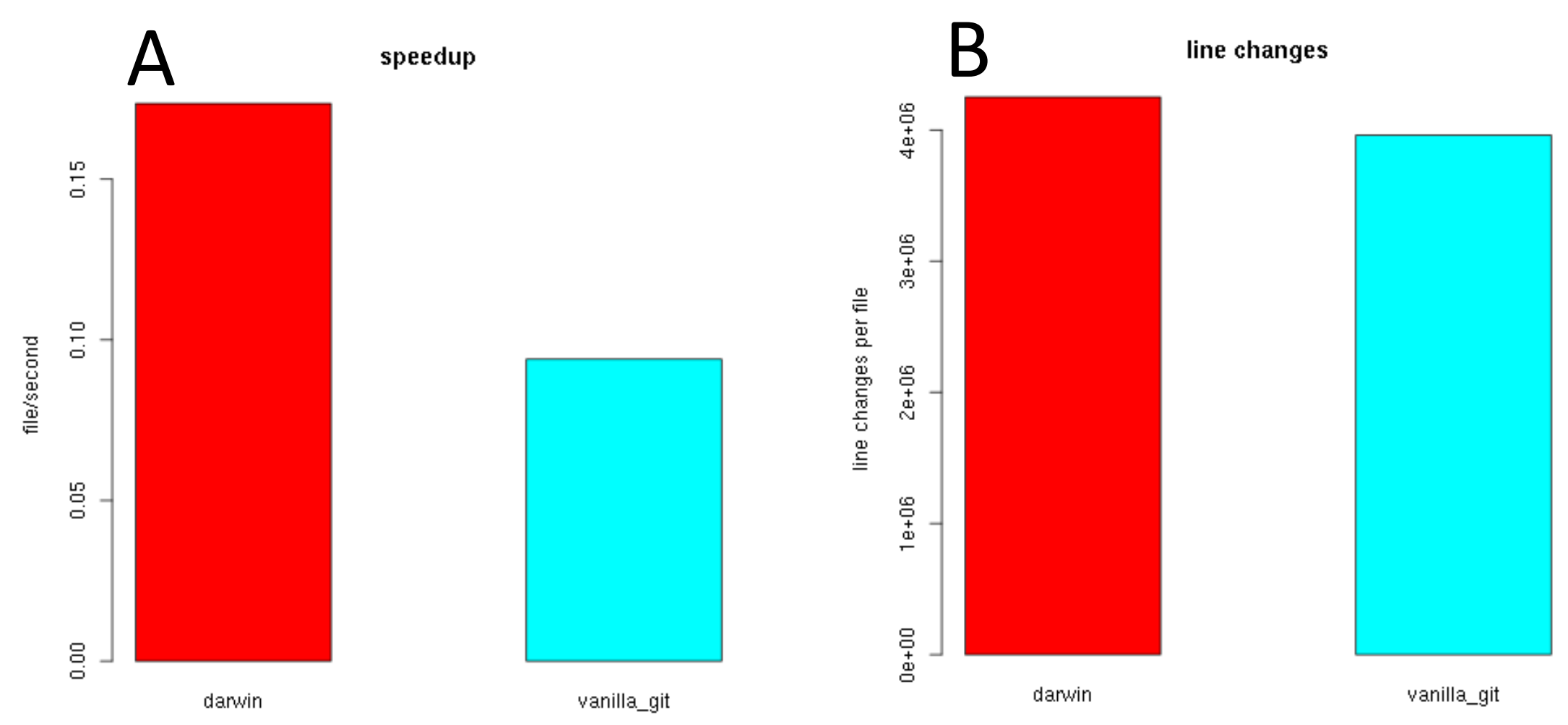


Figure 2. Quantitative Evaluation of Code
(A)Speed of file processing in darwin vs. vanilla_git
(B)Line changes in darwin vs. vanilla_git

As expected, darwin's preprocessing produced a significantly faster diff, with an average speed of 5.763 seconds per file, as opposed to 10.640 seconds for the vanilla_git. However, against our assumptions, that performance did not seem to rely upon the reduction of changed lines reported by git, since darwin actually produced more changed lines. More research is required to find the correlation between the number of changed lines reported by git and the number of lines actually changed in the file.

Future Research

Future work mostly revolves around bringing darwin from a standalone diff-producing executable to an integrated system with a GUI usable by novices. The major challenge with that is parsing the plethora of different file types used to represent DNA although the implementation of other formats such as SBOL, ApE, etc., could be done easily by extracting the actual genetic sequence from the file.

In addition, the current algorithms are unable to deal with introns of any sort in the input DNA sequence, and will attempt to split the file into ORFs regardless. The ability to correctly identify non-coding regions like this would create a far more valuable piece of software, able to deal with prokaryotes and eukaryotes alike.

References/Acknowledgments

VANDERBILT
School of Engineering

- Vanderbilt Department of Electrical Engineering and Computer Science
- Dr. Jules White