A group is a fundamental mathematical object that consists of a set and an operation - like the integers and addition - which satisfies certain basic properties. Mathematicians who study groups often want to carry out various computations in these groups, and if a group is set up in a certain way then some computations become easier. One condition which ensures that certain calculations are easy in groups is called automaticity, and we studied a generalization of this property called graph automaticity.

In order to prove that a group is automatic or graph automatic, we must first understand the “generators” of the group. Generators of a group are fundamental components that build all of the elements of the group. For instance, 1 is a generator for the integers, since by adding (and subtracting) 1, we can build any number in the integers. Using the generators, we must develop what is called a “normal form” for group elements. This means developing a standard way to write any element in terms of generators. We must then “build” simple machines, often used in theoretical computer science, called finite state automata (FSA), which can do simple computations, to compare normal forms. Further, we must be able to build finite state automata which can tell the difference between two normal forms that differ by a generator - for instance, in the integers, this means recognizing two numbers that differ by 1. For automatic groups these normal forms must be expressed in terms of group generators, while for graph automatic groups we are allowed to use any set of symbols to express the normal forms.

The initial goal of this project was to show that the a collection of groups denoted $\Gamma_3(n)$, an example of a Diestel-Leader group, is graph automatic. This is a collection of groups because for every integer $n$ we get a different group. To do this, we analyzed group elements, which look like $2 \times 2$ matrices whose entries are polynomials. Breaking down the problem, we realized that the crux of the problem was instructing a finite state automaton to recognize a polynomial (ie something like $t + 3t$, where $t$ is a variable) which corresponds to a group element. We were able to construct a normal form and a Finite State Automaton which could do this. However, because of the complicated multiplication in the group, we were unable to extend this result to creating FSA which could distinguish two group elements that differ by 1. For automatic groups these normal forms must be expressed in terms of group generators, while for graph automatic groups we are allowed to use any set of symbols to express the normal forms.

We were, however, able to prove that two related collections of groups are graph automatic. This is especially interesting because the related groups are infinitely generated, that is, require an infinite number of building blocks to create all group element. Not many results are known in the literature about infinitely generated graph automatic groups. Our deep understanding of the normal form in $\Gamma_3(n)$ led us to a related normal form for these two new groups. We studied a group we refer to as $H_n$, which is a subgroup of $\Gamma_3(n)$ (a subgroup is a group that lives within another group; even numbers, for example, are a subgroup of the integers) and a group $G_n$, which can be thought of as an expansion of $H_n$. We were able to construct a unique normal form for both of these groups, and all of the necessary finite state automata to prove that these two groups are graph automatic.

It was interesting that both the subgroup $H_n$, which can be thought of as “smaller” than $\Gamma_3(n)$, since it is contained in it, and the group $G_n$, which is in some senses “bigger” than $\Gamma_3(n)$ were both graph automatic, but that similar techniques could not be used to prove that $\Gamma_3(n)$ was graph automatic. Our proofs extended to show two additional families of related infinitely generated groups were also graph automatic; in these groups the polynomial entries in the matrices were allowed to have a different set of coefficients than in $\Gamma_3(n)$. We still hope to show that $\Gamma_3(n)$ is graph automatic but that will require the development of a different normal form. Our work did not show that $\Gamma_3(n)$ is not graph automatic. Hopefully the insights gained from the normal forms we investigated in this project will lead to future work in this direction.

Faculty Mentor: Jennifer Taback

Funded by the Maine Space Grant Consortium and the National Science Foundation