Project Specification
Automata that move both ways

James Parkinson
2005821

March 14, 2023

1 Abstract

Usual finite automata move their head in one direction and thus examine the input word only once. How much power would they gain if the input head could move both ways? Surprisingly, these two-way machines can still only recognize regular languages, the same as DFA. During this project I will research various types of two-way automata and design and develop software to simulate and work with them.

2 Problem

Normal finite state automata work by reading an input word one character at a time, and either accept or reject based on what state the automata is in after the last input character is read. Two way finite automata however, can go back and forth through the input word multiple times until it either accepts or rejects.

In this project, I will mainly focus on a specific type of two way automata called Transducers, which introduce the idea of an output word. At each transition, transducers output 0 or more characters in the output alphabet (which could be different to the input alphabet) and move left or right over the input word. These transducers describe what are known as *regular functions* and there are very few software implementations available to simulate and work with these transducers. Specifically, I will work towards creating software that can produce the conjunction of two such transducers and explore how the number of states grows when this happens.

3 Objectives

The first part of this project will focus on understanding various proofs that describe properties of these two way automata including some constructions, such as converting from 2DFA to 1DFA, and implementing them in software to build up a repository of functions to work with and simulate two way automata, working towards understanding and implementing the proof for conjoining two transducers.

The second part of the project will be using the software I have developed to investigate how much more complex transducers get when a conjunction is performed, and to figure out what properties affect this increase in complexity.

4 Technical Considerations

For the software development aspect of this project, I could choose many different programming languages as the vast majority are capable of implementing these simulation algorithms. However, there are a few features that I am prioritising when choosing a language:

- Speed - As I will be simulating automata, some of the constructions I will be implementing have an exponential increase in the number of states, so I will need to be able to do a lot of processing very quickly to compensate for this, so the performance of a language is quite important.

- Support - During development it will be very useful for there to be a lot of reliable libraries / packages I can make use of so I can focus on implementing the constructions I have studied and not spend too much time making things like data structures from scratch.

- Readability - Since the constructions I am implementing will be quite complicated, the readability of my code is quite important as the clearer the code is, the easier it is to understand what's going on.

Considering these aspects, I will now go through a few languages that I have considered and say whether or not they would be a good fit for this project.

- Python is a very readable language, and since it is so popular there are a vast amount of libraries and packages available. However python is let down by its performance, and so it would not be suitable for this project as it would take too long to process the exponential growth in states for large automata.

- Java has better performance than python and also has a large amount of packages, however it is let down in the readability department as there is a lot of boiler plate code required.

- c++ has the best performance of the three languages by far, as it is a lower level language and has many built in optimisations. c++ also has a lot of packages available to make use of and it is a lot more readable than Java.

From these comparisons I think it is very clear that c++ would be the best language for me to use during the project as it meets all three of my criteria, and it also has extra features that would be useful to me such as easier memory management.

During the project I will be making use of github for version control, as it is used a lot in industry, and allows for recovery of past code as well as serving as a backup in case anything happens to the storage device in my physical machine.

5 Timetable

Week	**Task**
	Term 1
W1	Beginning the project
W2	*Project Specification*
W3	Gather resources
W4-W6	Reading and understanding proofs
W7	Begin software implementations
W8	*Progress Report*
W9-10	Software implementations
	Christmas holiday
	Term 2
W1-W2	Software implementations
W3-W7	Investigating complexity
W8-10	Project presentation / Final report write up
	Easter holiday
	Term 3
W1	Final report

During the christmas and easter holidays I will also be allocating some time to work on the project, however I cannot say how much time I will be able to allocate so it is likely that the vast majority of work will be completed during term time.

6 Ethical Considerations

There are no ethical considerations necessary for this project.

Week	**Task**
Term 1	
W1	Beginning the project
W2	*Project Specification*
W3	Gather resources
W4-W6	Reading and understanding proofs
W7	Begin software implementations
W8	*Progress Report*
W9-10	Software implementations
Christmas holiday	
Term 2	
W1-W6	Further research about 2DFT composition
W7-W8	Begin implementing 2DFT composition
W9-10	Project presentation / Final report write up
Easter holiday	
Term 3	
W1	Final report