Progress Report
Automata that move both ways

James Parkinson
2005821

March 14, 2023

1 Introduction

Standard finite automata (such as DFA and NFA) move over the input word in a single direction, hence only reading it once, and these automata are known to only recognise regular languages. This project will study a different class of finite automata called two way finite automata (such as 2DFA and 2NFA) which can move in both directions over the input word, allowing for multiple passes over it. Surprisingly, these two way automata still only recognise regular languages which means there exists conversions between two way automata and one way automata, which will be implemented in code as a starting point for this project.

However, the main goal of this project is to produce software that performs the conjunction of two way deterministic finite transducers (2DFT), which are a type of two way automata that output a string of characters (or the empty string) at each transition that form an output word once the automata terminates. This results in the simulation of a partial function $f : \Sigma^* \to \Gamma^*$ where Σ and Γ are the input and output alphabets of the transducer respectively. More specifically, if we have a transducer A that simulates a function $f : \Sigma^* \to \Gamma^*$ and a transducer B that simulates a function $g : \Gamma^* \to \Delta^*$, the software would take A and B as input and return a new transducer C that simulates the composition $g \circ f : \Sigma^* \to \Delta^*$.

2 Background research

2.1 Academic resources

Many sources have been identified that will be useful for this project, some of the key resources have been listed in the references section of this report, and a brief description of each one will be given here.

[1] is a book called An Automata Toolbox by Mikołaj Bojańczyk and Wojciech Czerwiński, which contains a vast amount of information about automata theory. The part of the book that is most important for this project is chapter 13.3 "Deterministic two-way transducers" which includes a very clear definition of deterministic two-way transducers, as well as a proof that shows 2DFTs are closed under composition which is what this project is trying to reproduce in software.

[2] is a paper titled "Serial composition of 2-way finite-state transducers and simple programs on strings" by M. P. Chytil and V. Jákl that provides another proof for closure under composition for 2DFTs.

[3] is a paper titled "The Reduction of Two-Way Automata To One-Way Automata" by J.C. Shepherdson that provides a proof that for every 2DFA there exists an equivalent 1DFA and provides a construction that can be used to create a non-minimal one-way automata given a two-way automata.

[4] is a paper titled "A note on the reduction of two-way automata to one-way automata" by Moshe Y. Vardi that provides an alternate proof for reducing two way automata to one way automata using subset construction, instead of crossing sequence analysis as used in [3]

2.2 Pre-existing software

During the research phase multiple software solutions for finite state automata were found, however all of them seemed to only focus on one way state machines such as DFA, NFA and PDA. None of the pre-existing software that was found included any implementation of two way automata such as 2DFA, 2NFA and 2DFT, so as this project aims to implement some functionality for two way automata including 2DFA and 2DFT it will be providing functionality that is not currently available.

3 Current state of the project

As shown in section 5 of the project specification, the first 6 weeks of term was spent gathering resources and reading through proofs in order to understand them for when the software implementation begins in week 7. Then from week 7 to 10 the software implementation started, which would continue into term 2.

The project progressed as planned during the first 6 weeks, plenty of resources were gathered and the proofs had been read through multiple times as I tried to fully understand them. Then during the start of week 7 I set up the git repository and started to write the first few lines of code.

However, I fell ill at the end of week 7 and didn't recover fully until the middle of week 9. This greatly affected my ability to work on the project as I was bedridden for the majority of week 8 which also impacted my other responsibilities such as other coursework and society exec obligations. When I recovered enough to start working again I had to first catch up with the backlog of everything I had missed, which delayed me getting back to the project even more than I first expected.

To make up for this lost time, I am going to modify the timetable I gave in the specification by adding on two extra weeks for implementing software and reducing the amount of time spent investigating complexity. The full updated timetable for term 2 will be shown in the next section of the report.

4 Timetable for Term 2

As mentioned before, I have updated the timetable for term 2 to add an extra two weeks for software development by reducing the time allocated to investigating the increase in complexity after these functions have been performed.

Week	**Task**
Christmas holiday	
Term 2	
W1-W2	Implement 2DFA to 1DFA
W3-W5	Implement 2DFT composition
W6-W7	Investigating Complexity
W8-10	Project presentation / Final report write up
Easter holiday	
Term 3	
W1	Final report

No tasks have been assigned during the christmas and easter holidays since I cannot guarantee how much available time I will have, however I will still be working on the project to some degree during these times.

5 Project Management + Reflection

During term 1, I allocated parts of my timetable to work solely on the project as I found that having a regular set time every week to work on the project really helped me to stay on track. The project has gone okay so far, although falling ill made me lose a lot of time and pushed back where I wanted to be by the end of this term by a lot, so I am going to make up for that lost time during term 2 and also over the christmas holidays as much as I can.

References

[1] M. Bojańczyk and W. Czerwiński. *An Automata Toolbox*. 2017.

[2] M. P. Chytil and V. Jákl. "Serial composition of 2-way finite-state transducers and simple programs on strings". In: (1977), pp. 135–147.

[3] J. C. Shepherdson. "The Reduction of Two-Way Automata to One-Way Automata". In: *IBM Journal of Research and Development* 3.2 (1959), pp. 198–200. DOI: 10.1147/rd.32.0198.

[4] Moshe Y. Vardi. "A note on the reduction of two-way automata to one-way automata". In: *Information Processing Letters* 30.5 (1989), pp. 261–264. ISSN: 0020-0190. DOI: https://doi.org/10.1016/0020-0190(89)90205-6. URL: https://www.sciencedirect.com/science/article/pii/0020019089902056.

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	Implement 2DFT composition
W9-10	Project presentation / Final report write up
	Easter holiday
	Term 3
W1	Final report