CPSC-406 Report

Marc Nathaniel Domingo
Chapman University

May 18, 2023

Abstract

The purpose of this course is to understand how the Theory of Algorithms coincide with the application of functions and algorithms in Software Engineering. In particular, this course places an emphasis on the concepts of Automata, Logic, and Concurrency in regard to algorithms in our eventual application of these topics to a final project.

Contents

1 Introduction

1.1 Week 1

During this week, the overall outline and "roadmap" of topics and project deadlines were covered during class, as well as an introduction to the concept of Definitive Finite Automata (DFA).

1.2 Week 2

Lectures covered Definitive Finite Automata (DFA) and Non-Definitive Finite Automata (NFA) in more detail, as well as how to convert NFA to DFA.

1.3 Week 3

During this week, the lectures covered the concepts of querying a database, as well as algorithms for Unification and Resolution.

1.4 Week 4

During this week, lectures covered the concept of Distributed Hash Tables (DHT) in the application of Decentralized Systems.

1.5 Week 5

During this week, the lectures covered the concepts of Turing Machines and their relation to Polynomial and Non-Polynomial Time Problems.

1.6 Week 6

During this week, the lectures reviewed Satisfiability as covered during our lectures on Non-Polynomial Time Problems in their application to a Sudoku Solver.

1.7 Week 7

During this week, the lectures covered the concept of Distributed Systems in terms of how to create a Logical Clock.

1.8 Week 8

During this week, the lectures covered the concept of Big O-complexity in terms of certain sorting algorithms such as bubble sort, in addition to an introduction to Complexity Theory.

1.9 Week 9

During this week, the lectures covered the concept of model checking in addition to how the Needham-Schroeder Key Protocol can be modeled through Spin to witness how messages can be intercepted.

1.10 Week 10

During this week, the lectures covered the concept of Temporal Logic and how it can be applied to solve Satisfiability problems.

1.11 Week 11

During this week, the lectures covered the concepts of Composing Automata in addition to writing Regular Expressions based off of Non-Definitive Automata (NFA).

1.12 Week 12

During this week, the lectures covered the concepts of Atomicity, Sequential Consistency, and Weak Memory in the context of a multi-threaded program in addition to what Safetiness, Liveliness, and Fairness are defined as within the context of a system.

1.13 Week 13

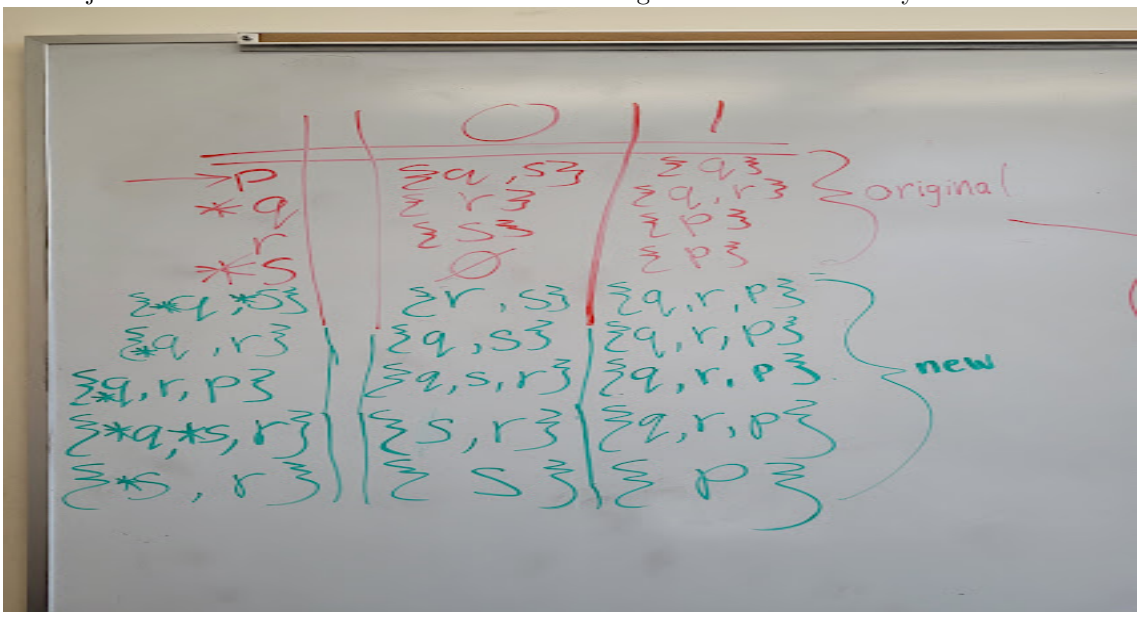
During this week, the lectures covered the concepts of the logic and concurrency behind the world wide web (Www) and goes over what the Semantic Web is and what is possible to decentralize over the internet.

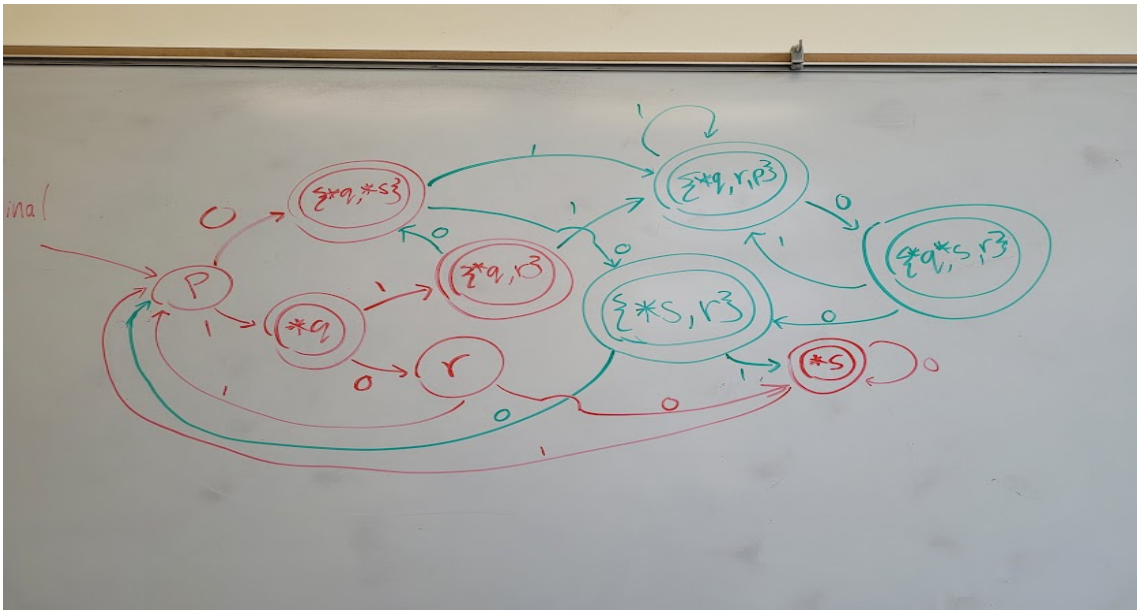
2 Homework

This section contains solutions to homework.

2.1 Week 2

The objective of this week's homework is to convert a given NFA to a DFA by hand.





2.2 Week 3

The objective of this week's homework is to solve Unification Algorithm problems in addition to practicing drawing an SLD-tree solution to a program given a particular input.

1. $f(x, f(x,Y)) \stackrel{?}{=} f(f(Y,a), f(U,b))$

1. $X \stackrel{?}{=} f(Y,a)$

$\sigma_1 = [f(Y,a)/X]$

2. $f(X,Y) \stackrel{?}{=} f(U,b)$

3. $X \stackrel{?}{=} U$

$X\sigma_1 \stackrel{?}{=} U \rightarrow f(Y,a) \stackrel{?}{=} U$

$\sigma_3 = [f(Y,a)/U]$

4. $Y \stackrel{?}{=} b$

$\sigma_4 = [b/Y]$

$\sigma_2 = \sigma_3 \circ \sigma_4 = [f(Y,a)/U, b/Y]$

$\sigma = \sigma_1 \circ \sigma_2 = [f(Y,a)/X, f(Y,a)/U, b/Y]$

$MGU\ \sigma = [f(b,a)/X, f(b,a)/U, b/Y]$

2. $f(g(U), f(X,Y)) \stackrel{?}{=} f(X, f(Y,U))$

1. $g(U) \stackrel{?}{=} X$

$\sigma_1 = [g(U)/X]$

2. $f(X,Y) \stackrel{?}{=} f(Y,U)$

3. $X \stackrel{?}{=} Y$

$X\sigma_1 \stackrel{?}{=} Y \rightarrow g(U) \stackrel{?}{=} Y$

$\sigma_3 = [g(U)/Y]$

4. $Y \stackrel{?}{=} U$

$Y\sigma_3 \stackrel{?}{=} U \rightarrow g(U) \stackrel{?}{=} U$ Fail Check Occurs

3. $h(U, f(g(V), W), g(W)) \stackrel{?}{=} h(f(X,b), U, Z)$

1. $U \stackrel{?}{=} f(X,b)$

$\sigma_1 = [f(X,b)/U]$

2. $f(g(V),W) \stackrel{?}{=} U$

$f(g(V),W) \stackrel{?}{=} U\sigma_1 \rightarrow f(g(V),W) \stackrel{?}{=} f(X,b)$

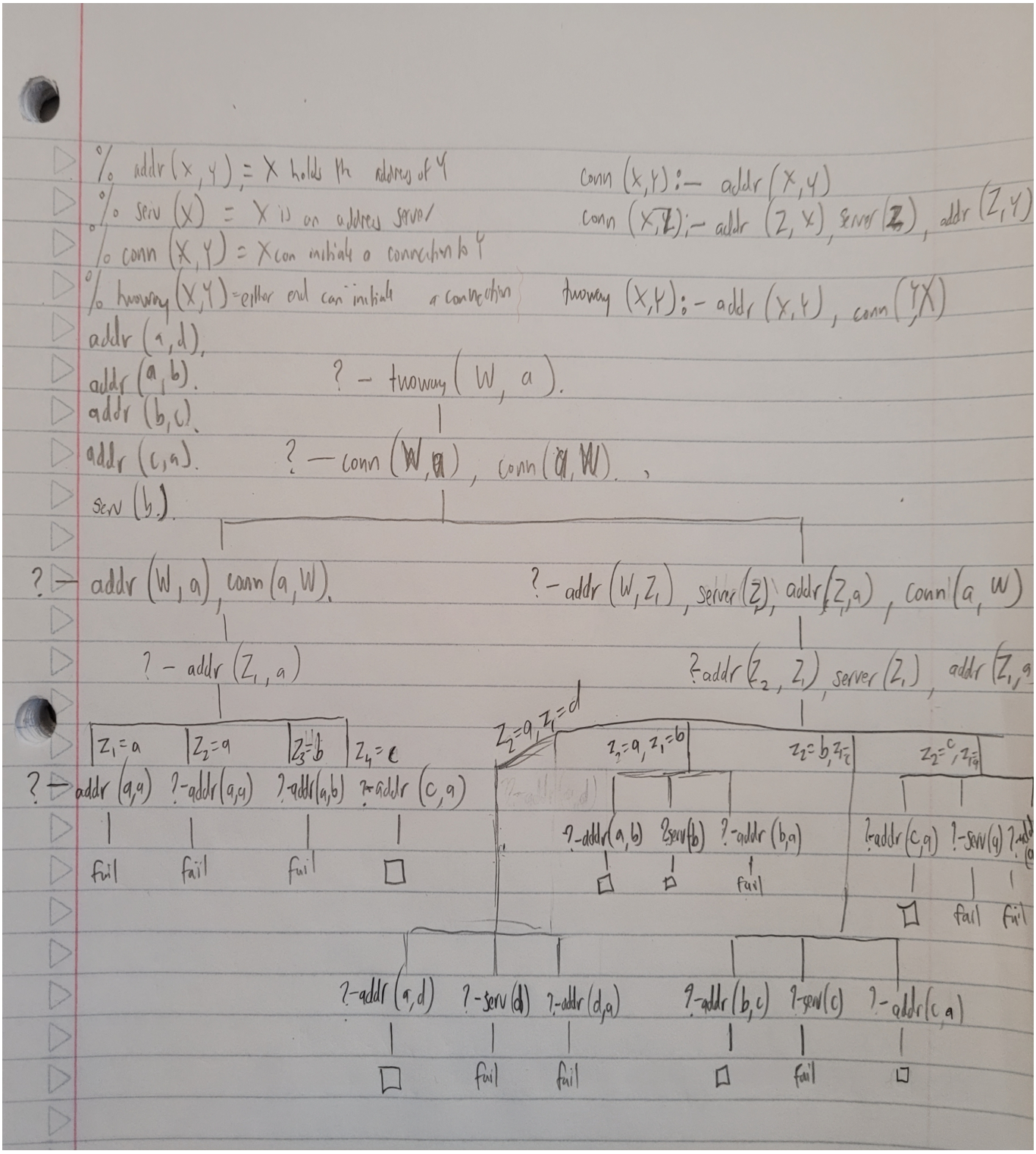
3. $g(V) \stackrel{?}{=} X \rightarrow \sigma_3 = [g(V)/X]$

4. $W \stackrel{?}{=} b \rightarrow \sigma_4 = [b/W]$ $\sigma_2 = \sigma_3 \circ \sigma_4 = [g(U)/X, b/W]$

5. $g(W) \stackrel{?}{=} Z \Rightarrow \sigma_5 = [g(W)/Z]$

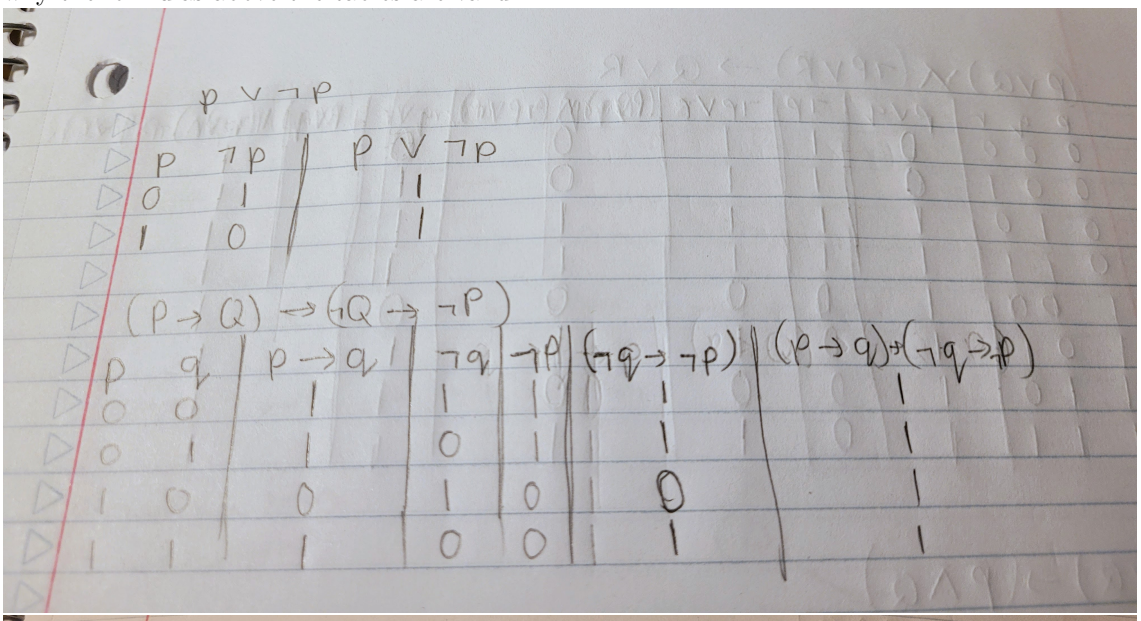
$MGU\ \sigma = \sigma_1 \circ \sigma_2 \circ \sigma_5 = [f(X,b)/U, g(U)/X, b/W, g(W)/Z]$

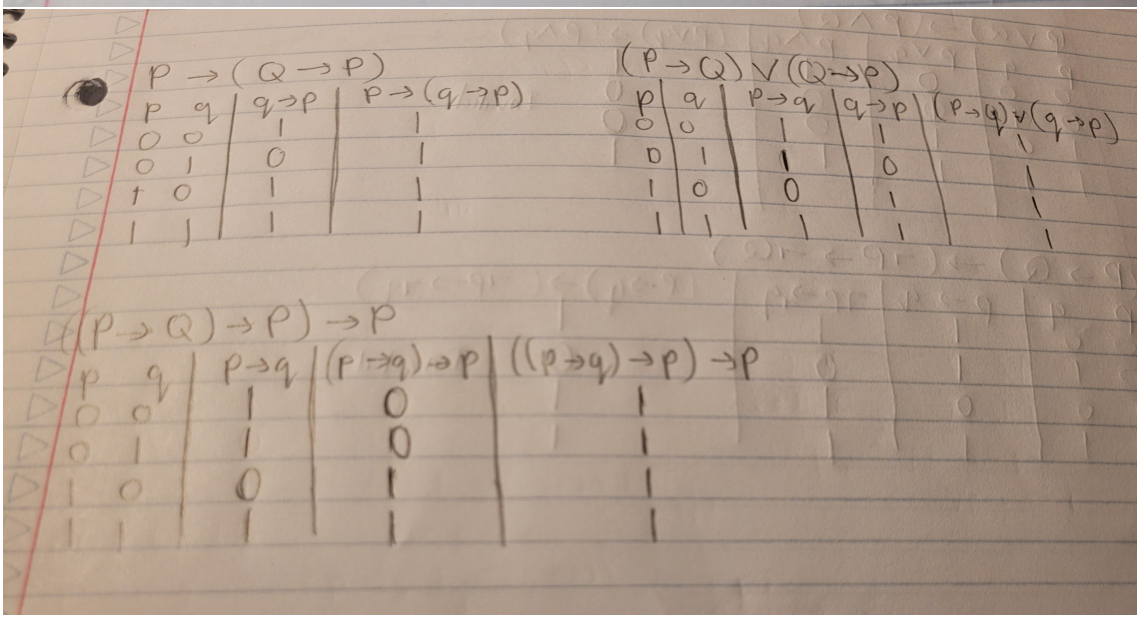
$MGU_\sigma = [f(g(V),b)/U, g(V)/X, b/W, g(W)/Z]$

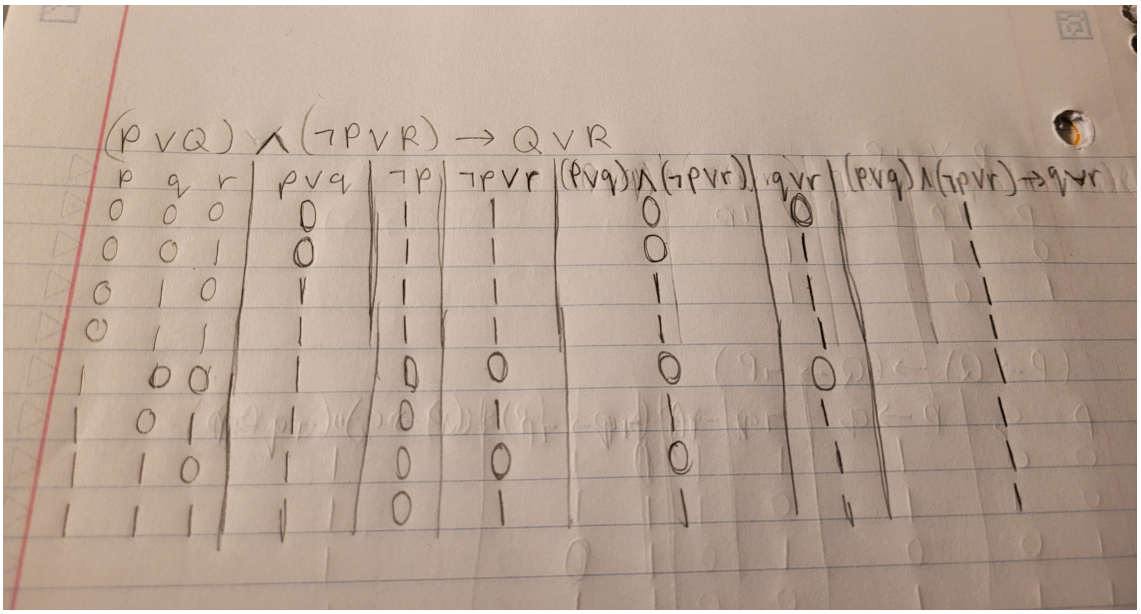

% addr(X,Y) = X holds the address of Y
% serv(X) = X is an address server
% conn(X,Y) = X can initiate a connection to Y
% twoway(X,Y) = either end can initiate a connection
addr(a,d).
addr(a,b).
addr(b,c).
addr(c,a).
serv(b).
conn(X,Y) :- addr(X,Y)
conn(X,Y) :- addr(Z,X), serv(Z), addr(Z,Y)
twoway(X,Y) :- addr(X,Y), conn(Y,X)
?- twoway(W,a).
?- conn(W,a), conn(a,W).
?- addr(W,a), conn(a,W).
?- addr(W,Z1), serv(Z1), addr(Z1,a), conn(a,W)
?- addr(Z1,a)
?- addr(Z2,Z1), serv(Z1), addr(Z1,
Z1=a
Z2=a
Z3=b
Z4=c
?- addr(a,a)
?- addr(a,a)
?- addr(a,b)
?- addr(c,a)
fail
fail
fail
Z1=a, Z2=d
Z2=a, Z1=b
Z2=b, Z1=c
Z2=c, Z1=a
?- addr(a,b)
?- serv(b)
?- addr(b,a)
fail
?- addr(c,a)
?- serv(a)
fail
fail
?- addr(a,d)
?- serv(d)
?- addr(d,a)
?- addr(b,c)
?- serv(c)
?- addr(c,a)
fail
fail
fail

2.3 Week 6

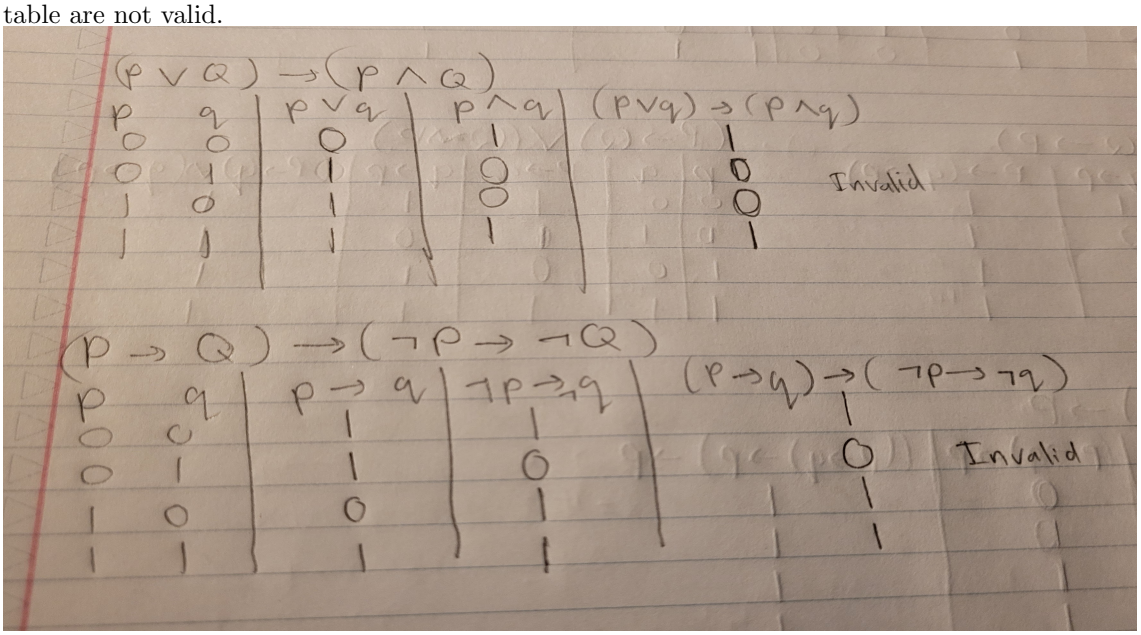
The objective of this week's homework was to create Indirect Truth Tables to understand Satisfiability in terms of Propositional Logic. For the following tables, we used the Indirect Truth Table method to prove why the formulas above the tables are valid.







For the following tables, the Indirect Truth Table method was used to prove why the formulas above the table are not valid.



2.4 Week 9

The objective of this week's homework is to explore exercises relating to the Needham-Schroeder Public-Key Protocol using Spin models.

One such example is to go through the program **ns.pml** and define what the meaning behind the propositions *success*, *AliceBob*, and *BobAlice*.

- **success** is a propositional variable that is true if and only if the first protocol (Alice) has successfully sent a reply to the second protocol (Bob), and if the second protocol (Bob) had sent a reply to the first protocol's (Alice) initial message.

- **aliceBob** is a propositional variable that is true if and only if the partner of the first protocol, Alice, is Bob.
- **bobAlice** is a propositional variable that is true if and only if the partner of the second protocol, Bob, is Alice.

The next exercise required determining which of the two formulas at the bottom of the code would violate the protocol when executed, as well as to explain the nature of the protocol based on that violation.

- The violation was caused by

$$ltl\{[](success\&\&aliceBob \rightarrow aliceBob)\}$$

- As the only difference between the two lines is the placement of **aliceBob** and **bobAlice** are swapped, this implies that the protocol is hard-coded such that the protocol does not get violated if Alice replies to Bob's reply, but in the event that the roles are reversed, the protocol is susceptible to intruders.

The next exercise of the homework involved looking through the execution sequence output from the protocol violation.

- The execution sequence ended up being 83 lines long, which matched up with the protocol violation message, with an additional line representing the attempt at the next check.

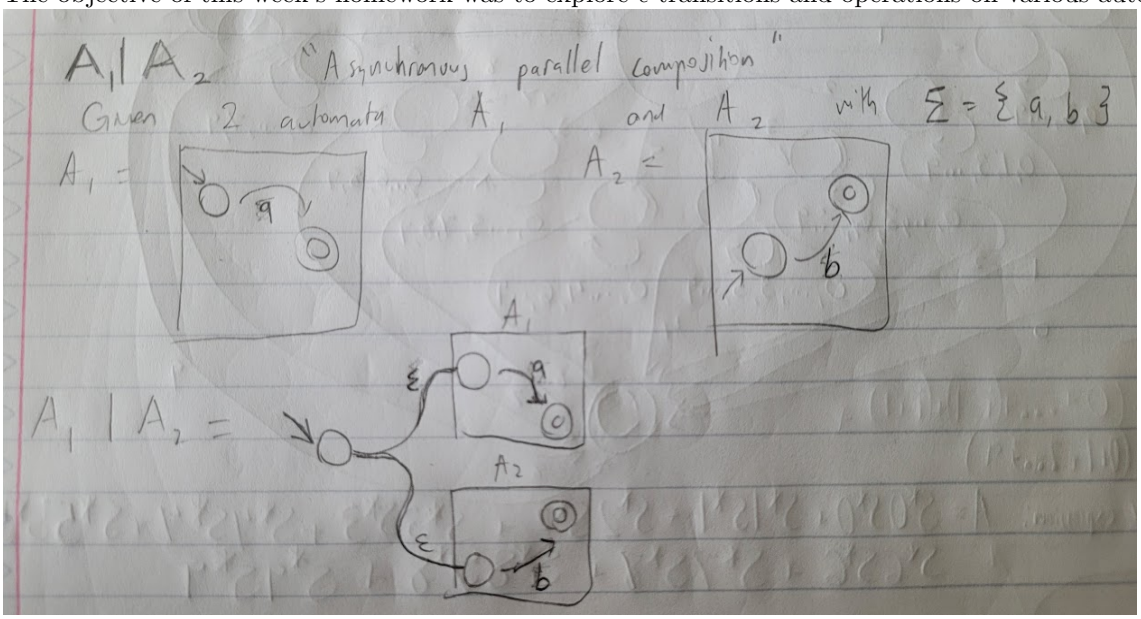
The last exercise involved creating a Message Sequence Chart (MSC) that represents a successful attack on the protocol.

```
6:   proc  0 (Alice:1) ns.pml:77 (state 12)  [network!msg1,partnerA,data.key,data.d1,data.d2]
 7:   proc  2 (Intruder:1) ns.pml:144 (state 1)      [network?msg,_,data.key,data.d1,data.d2]
23:   proc  2 (Intruder:1) ns.pml:201 (state 64)
[network!msg,rcpt,data.key,data.d1,data.d2]
24:   proc  1 (Bob:1) ns.pml:104 (state 1)   [network?msg1,bob,data.key,data.d1,data.d2]
34:   proc  1 (Bob:1) ns.pml:123 (state 17)
[network!msg2,partnerB,data.key,data.d1,data.d2]
35:   proc  2 (Intruder:1) ns.pml:144 (state 1)      [network?msg,_,data.key,data.d1,data.d2]
45:   proc  2 (Intruder:1) ns.pml:180 (state 37)
[network!msg,rcpt,intercepted.key,intercepted.d1,intercepted.d2]
46:   proc  0 (Alice:1) ns.pml:80 (state 13)  [network?msg2,alice,data.key,data.d1,data.d2]
52:   proc  0 (Alice:1) ns.pml:93 (state 20)
[network!msg3,partnerA,data.key,data.d1,data.d2]
53:   proc  2 (Intruder:1) ns.pml:144 (state 1)      [network?msg,_,data.key,data.d1,data.d2]
69:   proc  2 (Intruder:1) ns.pml:201 (state 64)
[network!msg,rcpt,data.key,data.d1,data.d2]
70:   proc  1 (Bob:1) ns.pml:126 (state 18)   [network?msg3,bob,data.key,data.d1,data.d2]
```

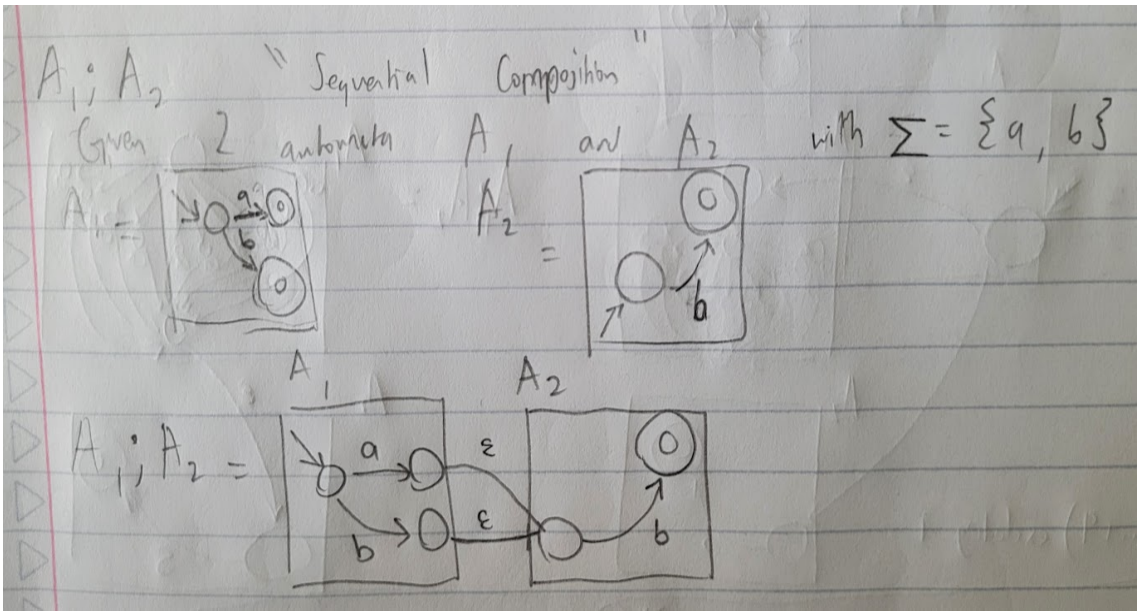
- The MSC represents a successful attack on the protocol as the intruder has managed to message both Alice and Bob before the other is able to receive and reply to the message from a proper protocol.

2.5 Week 11

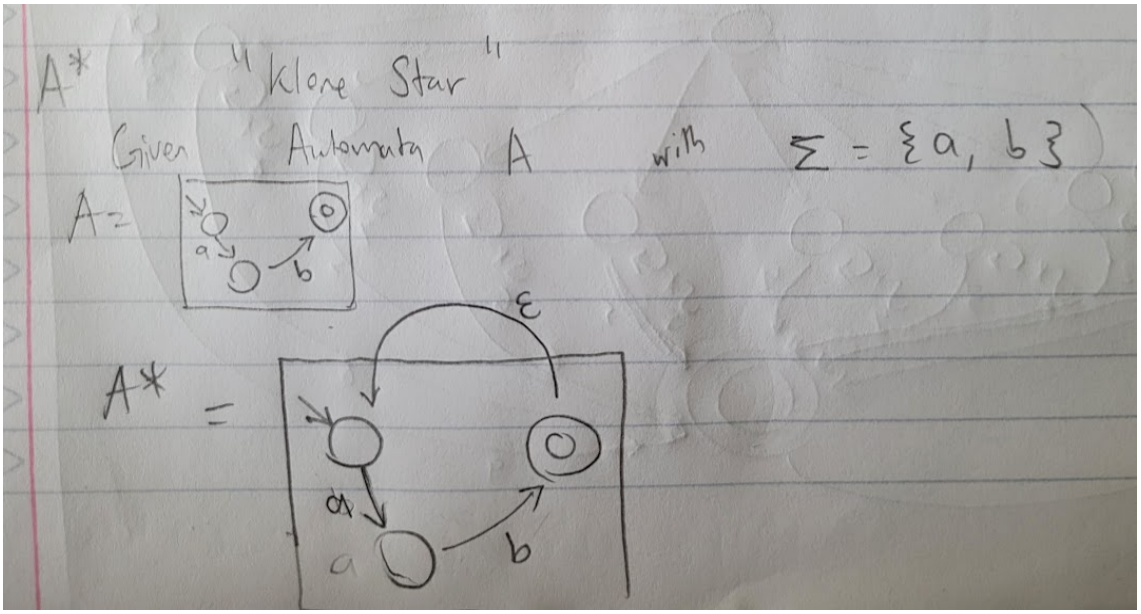
The objective of this week's homework was to explore ϵ-transitions and operations on various automata.



- Asynchronous Parallel Composition accepts languages that are capable of operating in parallel to each other, utilizing a set of strings that consists of the combined alphabet that each language accepts.
- The above example would have a regular expression of $A_1|A_2 = \epsilon a + \epsilon b$



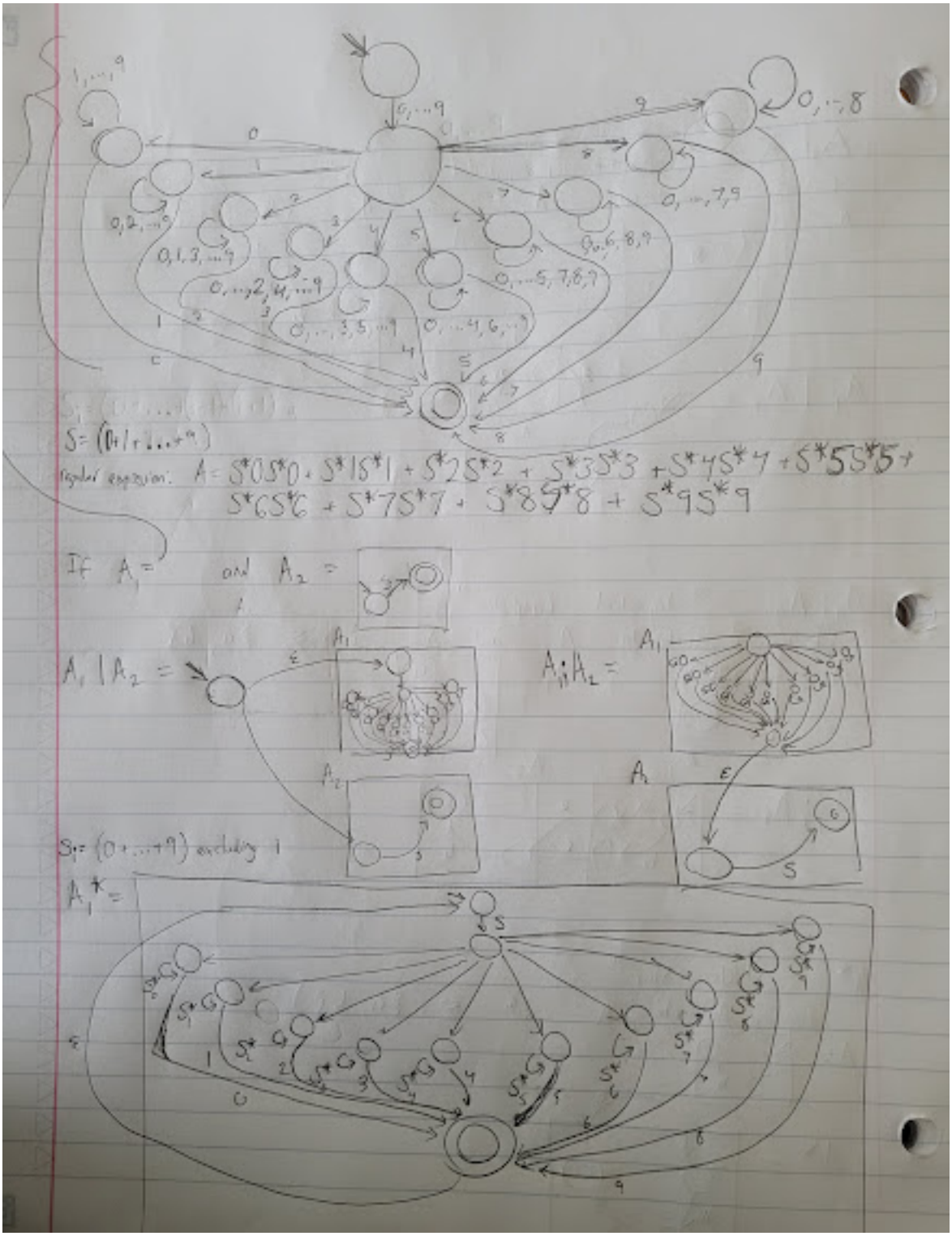
- Sequential Composition accepts languages that are capable of running in a sequential sequence, that is the output of one language is used as the starting point of another language, utilizing a set of strings that both languages share.
- The above example would have a regular expression of $A_1; A_2 = (a + b)\epsilon b$



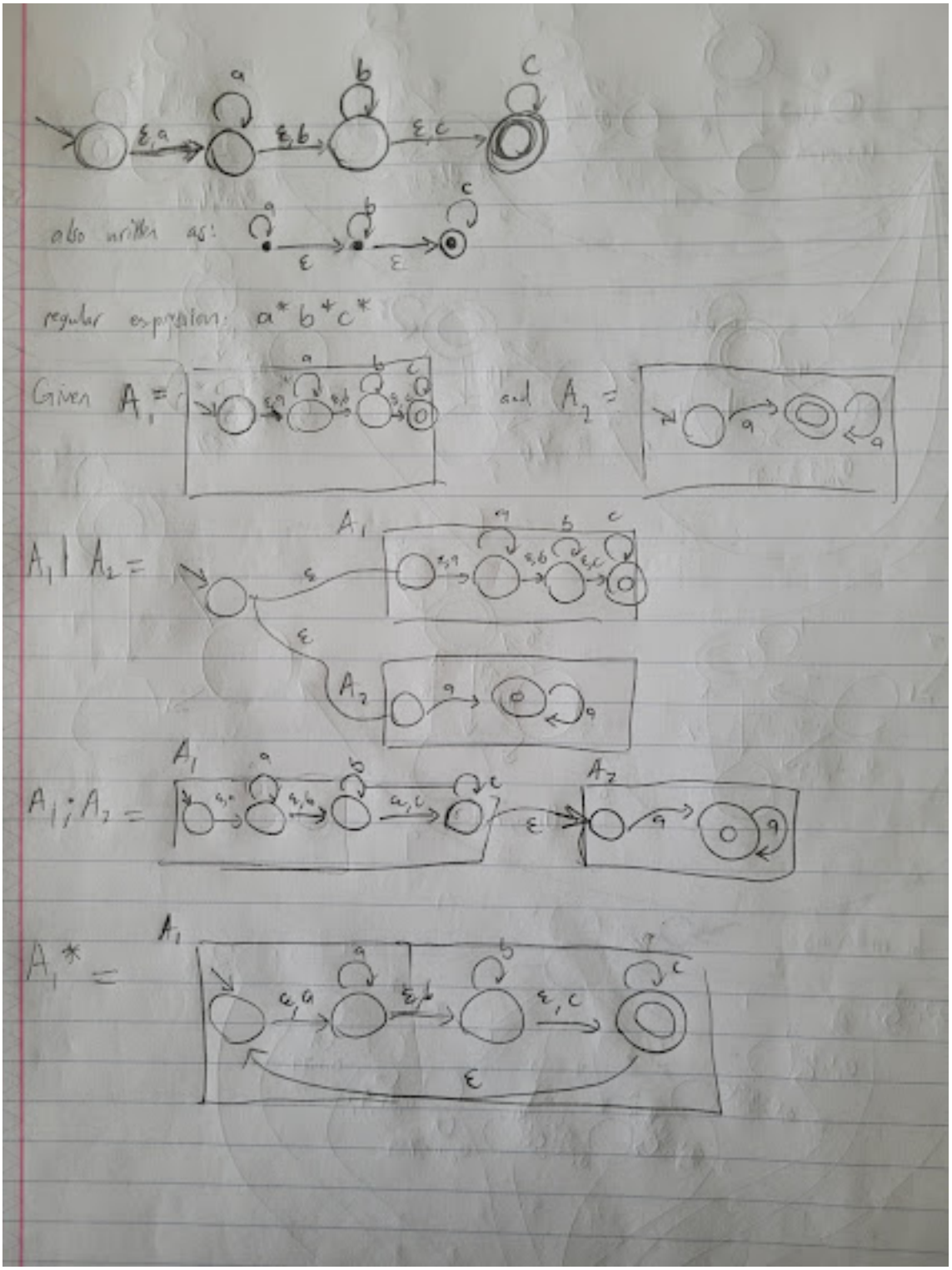
- Kleene Star accepts languages that are capable of infinitely repeating, starting back at the machine's initial state.
- The above example would have a regular expression of $A^* = ab\epsilon^*$

The next section of the homework consisted of finding the DFA and regular expressions of examples from the textbook, as well as finding the ϵ-NFA if the above operations were applied to the examples.

- The set of strings over the alphabet { 0, 1, ... 9 } such that the digit has appeared before.



- The set of strings consisting of zero or more a's followed by zero or more b's, followed by zero or more c's.


a
b
c
ε, a
ε, b
ε, c
also written as:
ε
ε
regular expression: a* b* c*
Given A1 =
and A2 =
A1 | A2 =
A1
A2
A1 ; A2 =
A1*

2.6 Week 12

The objective of this week's homework was to explore the concepts of Mutual Exclusion with and without Atomicity in addition to the concepts of Sequential Consistency in Weak Memory through the use of online multi-threading Java Programs.

Homework Question 1: Have a look at peterson.py. What program behavior do you expect? Is your expectation confirmed when you run the program?

- Based on the program itself, I expect the program to attempt to run both threads a total of 10000 times. The execution of each thread would iterate the "counter" of the program each time the thread is run, so I would expect the print statement at the end of the program to note that the program's counter ended at 20000. This expectation was confirmed after running the program.

Homework Question 2: Analyse the Java program peterson in the same way as you analysed the Python program in the previous exercise. Make sure to run the Java program on your local machine. What observations do you make?

- Based on the program itself, I expect that the program, similar to the python program from the previous question, would attempt to run both threads a total of 10000 times as both the python and java programs share a similar code for what is defined as the "critical_section" in the python file and what is defined as a "process" in the java file. During the execution of each thread, the program would iterate the "counter" variable after each thread has completed their process, and because both threads are running the "process" function that iterates 10000 times, I would expect the print statement at the end of the program to note that the counter ended at 10000. This expectation was confirmed after running the program.
- In terms of the program's behavior when executed, the java threading program appeared to run slower than the python program.

Homework Question 3: Explain why the outcome $\mathbf{a} = \mathbf{0}$, $\mathbf{b} = \mathbf{0}$ is not sequentially consistent, but the other three outcomes are.

- The outcome $\mathbf{a} = \mathbf{0}$, $\mathbf{b} = \mathbf{0}$ is not sequentially consistent within the context of the threading model as defined by memoryModel as there exists no currently existing iteration within the program that would compute and read both **a** and **b** as **0** before they are overwritten by the values of **x** and **y**. Currently, getting both **a** and **b** as **0** could be possible without the resetting of all variables at the end of *runTest()*.

Homework Question 4: Report the results you get from running memoryModelWithStats on your local machine. Include the specs of your processor, in particular the number of cores. If you can find out something about the caches, add this as well.

- Processor Information:
  - Name: Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz 2.71 GHz
  - Base Speed: 2.71 GHz
  - Number of Sockets: 1
  - Number of Cores: 2
  - L1 Cache: 128 KB
  - L2 Cache: 512 KB
  - L3 Cache: 3.0 MB
- Results from memoryModelWithStats :

- (0, 0): 0
- (0, 1): 538
- (1, 0): 461
- (1, 1): 1

Homework Question 5: You can force sequential consistency of memoryModelWithStats by declaring certain variables volatile. In general, declaring variables as volatile comes at cost in execution time, so we want to use this sparingly. Which variables must be declare as volatile to ensure sequential consistency? Measure and report the effect that volatile has on your run time (I use java Main — gnomon).

- In the memoryModelWithStats, the variables that needed to be declared as volatile in order to ensure sequential consistency were the x and y variables, as those variables are shared, used, and updated across both created threads. By declaring them as volatile, this helps ensure sequential consistency as declaring them as volatile applies the "lock" that the previous programs were missing, protecting them during the "critical section" of each respective thread.
- Setting the variables x and y had affected the runtime of the program as the original program had an initial runtime of 9.875962 seconds, whereas the modified code had a runtime of 11.0930762 seconds. As I had difficulty installing gnomon, I used the Instant and Duration Java packages in order to record the respective runtimes of the program.

3 Paper

For this report's paper, students were either given the option to write a technical paper or a creative writing paper revolving around a particular currently existing algorithm. The creative writing option, which I had chosen for this report, requires the student to write a story about the world after the internet has suddenly stopped working, and how a currently existing algorithm has been adapted to either fill the gap that the internet had left or has improved upon the already existing legacy that the internet has had. In addition to this, the creative writing paper also required a scientific commentary on how said framework has been used to develop the world to the current way that it is by the government and local communities.

For this paper, I chose to write about the Serval Project.

A Brief History of the Fallout from the Internet's Disappearance and the Rise of the Meshnet

Necessity is the mother of invention.
Our need will be the real creator.

Proverbial phrases that describes how the driving force behind human innovation is ultimately driven by the demand for or perceived absence of something that is viewed as necessary to absolve or remedy. Phrases from the past that, while one has been attributed to multiple origins such as Aesop's Fables and the other being attributed to a translation of the philosopher Plato's work, still remain relevant to the present day. Phrases that have rung true ever since the internet ceased to exist.

Though there have been multiple accounts as to what had gone wrong during the past to cause such a catastrophic phenomenon, self-proclaimed "internet historians", communication companies in the private sector, and investigative governmental agencies around the world have yet to come to a consensus regarding the origins of the internet's sudden cessation of existence. While the claims made by said "internet historians" were often of questionable credibility and validity, many have speculated that there had been some truth to the notion that the complexity behind the causes of the internet's disappearance was muddled and obscured due to organizational politics, pressure from bureaucratic interference conflicting with standard

procedures and protocols, and efforts from both private sector communication companies and local and global-scale governments in order to cover-up faults [MMD] or to spread blame for the disappearance so thin in order to diffuse accountability [OST].

In contrast, one thing that is generally agreed upon is the fact that the loss of the internet has had a large impact from the individual scale all the way to the global scale, primarily due to the fact that the internet was one of the most convenient ways in which communication across the globe was made possible.

At the time, one of the groups most heavily impacted by the recent loss of connectivity across the globe were small businesses. Many small businesses had relied on the internet for various reasons, with advertising, communicating with potential customers for orders that can range from orders originating in the same general location to international orders originating from another country [IIM] and potential feedback on products, or their very store being hosted online via their own website or some other online platform. A similar issue arose in a multitude of software companies as many employees had either worked remotely or were outsourced from across the globe. As businesses and firms had at the time still primarily relied on the use of emails to communicate with and contact employees, customers, and business partners [SSB], this had caused a great strain on the global economy at the time and created an outcry for the necessity of a replacement for the absence of the internet.

As the proverbs attributed to Aesop's Fables and the works of Plato have echoed throughout human history, the necessity for something to fill the massive communication gap that the internet had left behind had become the driving force behind the innovation for said replacement.

The records of what is currently known as the "Meshnet" tend to associate its roots with a project called the "Serval Project" [TSP]. The Serval Project itself was an application that attempted to utilize what its creators had coined as "Wireless Mesh Networks," a decentralized form of wireless network that does not rely on preexisting architecture, which the project had described as network devices that had existed during the era in which the internet was still around. These networks had enabled mobile devices such as phones to be able to directly connect to other mobile devices on the mesh network through the use of specific radio frequencies, which at the time was significant as it enabled secure and encrypted communication during times of emergency such as earthquakes and had been used in smaller communities to organize against their government [TSM]. Though the project's capabilities were not tested beyond calls and text messages within small communities during the testing of these mesh networks, as the framework of the project had utilized mesh networks that were capable of connecting to other isolated mesh networks across the globe through the use of internet protocols used in the past such as IPv4 [TSP], the very ability to these decentralized networks to others across the world without the need of centralized architecture had provided proof that replacing the communication gap that the internet had left behind was a possibility.

In the past, governments and larger corporations have often foolishly ignored the technological projects of hobbyists and amateurs that have eventually led to what would eventually become integrated into regular use, such as the ability to send emails over the internet [PTD]. However, with the advent of the economic crisis that arose as a result of the internet's sudden disappearance, these entities did not make the same mistake when further testing of the Serval Project and efforts to connect dispersed mesh networks during the peak of the internet cessation crisis. Governments had worked with companies and their respective branches in other countries through agencies such as the United States' Advanced Research Projects Agency (ARPA), which had an integral role in the funding of the creation of an early form of the internet that had primarily been used for military and government purposes [HSF], in order to be able to fund and expand these efforts. The involvement of local and international governments had initially been met with some opposition by advocates for the Serval Project to remain in its decentralized roots, and while their concerns had been acknowledged by those involved in the expansion and evolution of the Serval Project for global use, these concerns had ultimately been washed away due to the fervent need for a return to being able to access the

level of communication and interconnectedness that was made possible by the internet.

Although the internet and the ability to connect to it had disappeared altogether, very early frameworks of the internet and its various communication protocols such as TCP and UDP (to name a few examples) [HSF] were eventually implemented in the routing algorithms used to navigate devices in the Serval Project's ever-growing mesh network. This was one of the major steps taken towards the Meshnet's eventual replacement of the original capabilities provided by the internet. Although it took several months to be able to implement ways for previously internet-reliant devices to install the software necessary to connect to the mesh network without a physical connection to a mesh network port, was a step towards the recovery and eventual surpassing of the communicative properties of the internet.

4 Conclusions

(approx 400 words) A critical reflection on the content of the course. Step back from the technical details. How does the course fit into the wider world of software engineering? What did you find most interesting or useful? What improvements would you suggest?

Overall, Algorithm Analysis relates to the wider world of software engineering as it deals with the categorization, classification, management, and understanding of the various structures associated with the different algorithms that we often take for granted. One such example of said algorithms taken for granted are the protocols used for communication and how they map sender to recipient addresses, verify whether a message had been successfully sent to the intended recipient or not and to ensure protection against attacks to access information not intended for the intruder to access. In addition to this, the Algorithm Analysis course had briefly touched the concepts of time and space complexity, which are important factors as a software engineer to consider when deciding how to structure programs, deciding what algorithms to include, and when considering the limitations of the system that the client and/or intended end user has access to. In terms of the course itself, Algorithm Analysis' nature as an experimental course due to a change in direction in terms of what was to be expected and what was to be delivered by the end of the course led to the overall course having both elements that I found interesting and found helpful in terms of actually learning and applying the concepts learned during the lectures as well as elements that I came into conflict with due to the very experimental nature of how the course was offered this year. As for what I enjoyed, I had found the more involved homework assignments towards the end of the semester, specifically the homework assignments that required students to analyze the Needham-Schroeder Public-Key Protocol, to explore epsilon-transitions and operations on automata, and the assignment to explore Mutual Exclusion, Atomicity, and Sequential Consistency with Weak Memory using Java multi-threading programs. In addition to the more involved homework assignments, I had also particularly enjoyed the Group Project as the structure of requiring students to attend Office Hours during the pre-MVP demonstration of the project had encouraged students to hold each other accountable and to make sure that everyone pulls their own weight in addition to helping promote software engineering skills in a team environment. As for an aspect of Algorithm Analysis that I would change, if the individual paper would remain a requirement for the Final Report in future semesters of Algorithm Analysis, the actual prompt for the individual paper should be provided to students during the initial course overview lecture as only having a month to work on a potentially content-heavy paper was not easy to balance alongside other Finals and Final Projects required by other classes in addition to the Group Project for the class. In addition to this, I would also suggest having a similar requirement for students to check-in at Office Hours regarding the individual paper in order to ensure that the student remains on-track and understands what is required of the paper.

References

[ALG] Algorithm Analysis, Chapman University, 2023.

[MMD] Man-made disasters: Why Technology and Organizations (Sometimes) Fail, N. Pidgeon and M. O'Leary, 2000.

[OST] Organizational Silence and Hidden Threats to Patient Safety, K. Henriksen and E. Dayton, 2006.

[TSM] The Serval Mesh: A Platform for Resilient Communications in Disaster & Crisis, P. Gardener-Stephen et al., 2013.

[TSP] The Serval Project: Practical Wireless Ad-Hoc Mobile Telecommunications, P. Gardener-Stephen et al., 2011.

[IIM] Internet-enabled International Marketing: A Small Business Network Perspective

[SSB] An exploratory study of small business Internet commerce issues, S. Poon and P. MC Swatman, 1997.

[PTD] The Politics of Technological Development: comparing the history of radio to the history of the internet, The University of Vermont, n.d.

[HSF] History, structure, and function of the internet, J. Glowniak, 1998.