CPSC-354 Report

Jonathan Karam
Chapman University

September 15, 2025

Abstract

This report is a collection of notes, discussions, and homework solutions for CPSC 354. Homework 1 discusses the MIU puzzle and proves why `MI` cannot become `MU`. Homework 2 introduces abstract rewriting systems (ARS), includes TikZ diagrams, and classifies systems by termination, confluence, and unique normal forms. Homework 3 studies a rewriting system over $\{a, b\}$ and explores its equivalence classes and termination.

Contents

1 Introduction

This report documents my learning week by week. Each section includes my notes and discussions of the lecture topics, followed by my homework solutions. The purpose is to show understanding, practice writing in LaTeX, and explain the material in my own words.

2 Week 1: The MIU Puzzle

2.1 Notes and Discussion

The first week introduced the MIU puzzle, created by Douglas Hofstadter. It is a formal system that begins with the word `MI` and has four rules. The puzzle asks whether it is possible to reach `MU`. This is important because it shows how formal systems can be analyzed with invariants, a key concept in logic and computer science.

2.2 Homework

Rules:

1. If a string ends with `I`, add a `U`.
2. If a string starts with `M`, double the part after `M`.
3. Replace `III` with `U`.
4. Remove `UU`.

Why `MI` cannot become `MU`: Let $\#I(w)$ be the number of `I`'s in w. Initially $\#I(\texttt{MI}) = 1$. Each rule preserves $\#I(w) \pmod 3$:

- Rule 1: no effect.
- Rule 2: doubles the count, $1 \mapsto 2$, never $0 \pmod 3$.
- Rule 3: subtracts 3, same remainder.
- Rule 4: no effect.

Thus $\#I(w) \pmod 3$ is invariant. Since `MU` has $\#I = 0$, it cannot be reached.

Deeper explanation: The invariant argument proves that all reachable words have I-count $\equiv 1$ or $2 \pmod 3$, never 0. The first letter M never disappears, so all reachable words start with M. Another way: define a homomorphism $h(M) = 0, h(U) = 0, h(I) = 1 \pmod 3$. Every rule preserves h, so $h(\texttt{MI}) = 1$ and $h(\texttt{MU}) = 0$. Contradiction.

3 Week 2: Abstract Rewriting Systems

3.1 Notes and Discussion

This week focused on abstract rewriting systems . An ARS consists of a set A and a relation R. We care about whether rewriting always stops, whether different rewrite paths can rejoin, and whether normal forms are unique. These ideas are important in programming languages, compilers, and theorem provers.

3.2 Homework

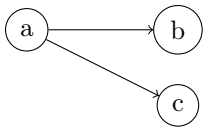
We drew diagrams and classified ARSs.

Examples

1: $A = \{\}, R = \{\}$ (empty system)

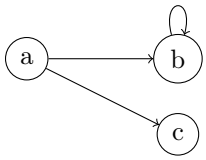
2: $A = \{a\}, R = \{\}$

3: $A = \{a\}, R = \{(a, a)\}$

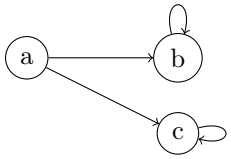


4: $A = \{a, b, c\}, R = \{(a, b), (a, c)\}$

5: $A = \{a, b\}, R = \{(a, a), (a, b)\}$



6: $A = \{a, b, c\}, R = \{(a, b), (b, b), (a, c)\}$



7: $A = \{a, b, c\}, R = \{(a, b), (b, b), (a, c), (c, c)\}$

Classification Table

ARS	$Terminating$	$Confluent$	UN	$Reason$
1	Yes	Yes	Yes	$Emptysystem$
2	Yes	Yes	Yes	$aisnormal$
3	No	Yes	Yes	$a \to aloop$
4	Yes	No	No	$a \to borc, twonormals$
5	No	Yes	Yes	$alljoinatb$
6	No	No	Yes	$a \to b(loop)orc(nf)$
7	No	No	Yes	$nonfs, vacuousUN$

4 Week 3: Strings over {a,b}

4.1 Notes and Discussion

We studied rewriting rules over alphabet $\{a, b\}$. This shows how simple systems can classify strings into equivalence classes. We also saw how orienting rules one way can make a system terminating without changing equivalence.

4.2 Homework 3

Exercise 5 rules:

$$ab \to ba, \quad ba \to ab, \quad aa \to \varepsilon, \quad b \to \varepsilon$$

Examples: `abba` $\to$ `aa` $\to \varepsilon$. `bababa` $\to$ `aaa` $\to$ `a`.

Not terminating because swaps loop. Equivalence classes: even number of a's $\to \varepsilon$, odd $\to a$. Normal forms: ε, a. Fix: orient $ba \to ab$ only.

Exercise 5b rules:

$$ab \leftrightarrow ba, \quad aa \to a, \quad b \to \varepsilon$$

Examples: `abba` $\to$ `a`. `bababa` $\to$ `a`.

Equivalence: all b vanish, runs of a collapse to one a. Classes: $\{\varepsilon, a\}$. Fix: orient swaps, keep $aa \to a$ and $b \to \varepsilon$.

5 Conclusion

Week 1: Learned about invariants and why MU is impossible. **Week 2:** Saw ARS diagrams and how to classify systems. **Week 3:** Explored rewriting with $\{a, b\}$, equivalence classes, and how termination can be fixed.

References

[BLA] Author, LaTeX Overview, Publisher, Year.