CPSC-354 Report

Mitchell Toney
Chapman University

September 14, 2025

Abstract

Contents

1 Introduction

2 Week by Week

2.1 Week 1: HW1

The MU puzzle is a puzzle created by Douglas Hofstadter. It consists of four rules that can be applied to a string MI.

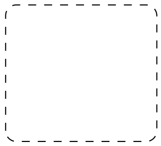
1. $xI \to xIU$
2. $Mx \to Mxx$
3. $xIIIy \to xUy$
4. $xUUy \to xy$

When first approaching this puzzle, the first strategy that came to mind was to take advantage of rule number 2 to keep duplicating the I's until there is a multiple of three, then using rules 3 and 4 to get rid of the I's and leave a remaining U.

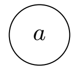
The issue with this is that $2^n \bmod 3$ will never equal 0, it infinitely cycles between equaling 1 and 2, and without being able to get rid of all the I's, which would require them being a multiple of 3, you will never be able to get MU.

Thus, the puzzle is not solvable.

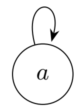
2.2 Week 2: HW2



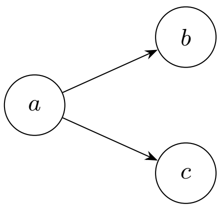
1. $A = \varnothing,\ R = \varnothing$



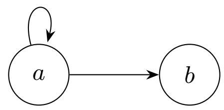
2. $A = \{a\},\ R = \varnothing$



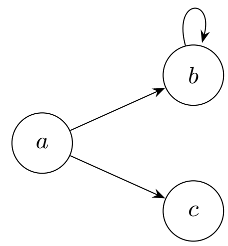
3. $A = \{a\},\ R = \{(a, a)\}$



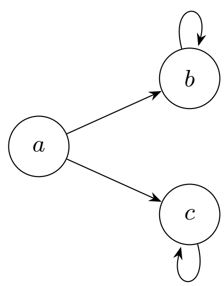
4. $A = \{a, b, c\}, \quad R = \{(a, b), (a, c)\}$



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



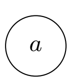
6. $A = \{a, b, c\}, \quad R = \{(a, b), (b, b), (a, c)\}$



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

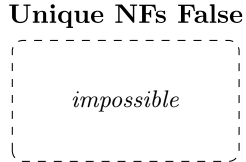
#	Terminating	Confluent	Unique NFs
1	Yes	Yes	Yes
2	Yes	Yes	Yes
3	No	Yes	No
4	Yes	No	No
5	No	Yes	Yes
6	No	No	No
7	No	No	No

Confluent True, Terminating True, Unique NFs True



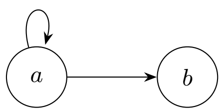
$A = \{a\},\ R = \varnothing$

Confluent True, Terminating True, Unique NFs False



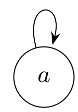
(no ARS exists)

Confluent True, Terminating False, Unique NFs True



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

Confluent True, Terminating False, Unique NFs False



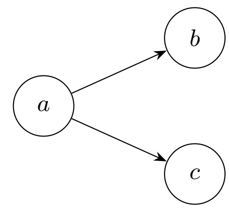
$A = \{a\},\ R = \{(a, a)\}$

Confluent False, Terminating True, Unique NFs True

impossible

(no ARS exists)

Confluent False, Terminating True, Unique NFs False



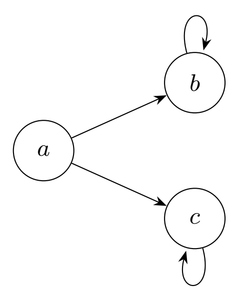
$A = \{a, b, c\},\ R = \{(a, b), (a, c)\}$

Confluent False, Terminating False, Unique NFs True

impossible

(no ARS exists)

Confluent False, Terminating False, Unique NFs False



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

2.3 Week 3: HW3

2.3.1 Exercise 5

Consider rewrite rules:

$$\begin{aligned} ab &\to ba \\ ba &\to ab \\ aa &\to \\ b &\to \end{aligned}$$

Example Reductions

Reducing `abba`:

$$\begin{aligned} abba &\to baba \quad (\text{using } ab \to ba) \\ baba &\to bbaa \quad (\text{using } ba \to ab) \\ bbaa &\to baa \quad (\text{using } b \to \varepsilon) \\ baa &\to aba \quad (\text{using } ba \to ab) \\ aba &\to baa \quad (\text{using } ab \to ba) \end{aligned}$$

There is an infinite loop between `aba` and `baa`.

Reducing `bababa`:

$$\begin{aligned} bababa &\to ababab \quad (\text{using } ba \to ab) \\ ababab &\to baabab \quad (\text{using } ab \to ba) \\ baabab &\to ababab \quad (\text{using } ba \to ab) \end{aligned}$$

This is an infinite loop between `ababab` and `baabab`.

Why the ARS is not terminating The ARS is not terminating because the rules $ab \to ba$ and $ba \to ab$ create infinite cycles. These rules allow us to swap adjacent a and b characters indefinitely, leading to non-terminating reduction sequences.

Non-equivalent strings Two strings that are not equivalent: `a` and `aa`.

The string `a` cannot be reduced further, while `aa` reduces to nothing using the rule $aa \to$. Since $nothing \neq a$, these strings are in different equivalence classes.

Equivalence classes The equivalence relation $\leftrightarrow^*$ has infinitely many equivalence classes. Each equivalence class can be characterized by the number of a's modulo 2 and the number of b's modulo 1.

The equivalence classes are:

- $[\varepsilon]$: strings with even number of a's and no b's
- $[a]$: strings with odd number of a's and no b's
- $[b]$: strings with any number of a's and at least one b

The normal forms are: ε, a, and b.

Modifying the ARS to be terminating To make the ARS terminating without changing equivalence classes, we can remove the symmetric rules and keep only one direction:

$$\begin{aligned} ba &\to ab \\ aa &\to \varepsilon \\ b &\to \varepsilon \end{aligned}$$

This eliminates the infinite cycles while preserving the same equivalence relation.

2.3.2 Semantic question

Parity of a's: "Does this string contain an odd number of a's?"
Answer: Yes if the normal form is a, No if the normal form is ε.

Exercise 5b

Consider rewrite rules:

$$\begin{aligned} ab &\to ba \\ ba &\to ab \\ aa &\to a \\ b &\to \end{aligned}$$

Reducing `abba`:

$$abba \to baba$$
$$baba \to abba$$

Reducing `bababa`:

$$bababa \to ababab$$
$$ababab \to bababa$$

Why not terminating. The symmetric swaps $ab \leftrightarrow ba$ allow infinite rewriting.

Non-equivalent strings. a and ε are not equivalent.

Equivalence classes. Exactly two: $[\varepsilon]$ (no a's; all b's delete) and $[a]$ (at least one a; since $aa \sim a$). Normal forms (under a terminating orientation): ε and a.

Terminating variant.

$$ba \to ab$$
$$aa \to a$$
$$b \to$$

This gives a complete semantics to the ARS: it computes the invariant for any input string.

3 Essay

4 Evidence of Participation

5 Conclusion

References

[BLA] Author, Title, Publisher, Year.