

CPSC-354 Report

Ethan Tapia
Chapman University

September 15, 2025

Abstract

Contents

1	Introduction	1
2	Week by Week	1
2.1	Week 1	1
2.2	Week 2	2
2.3	Week 3	4
3	Evidence of Participation	5
4	Conclusion	5

1 Introduction

2 Week by Week

2.1 Week 1

Lecture Summary

We introduced *formal systems* and worked with Hofstadter’s MIU-system as a rule-based rewriting game. Alphabet: $\Sigma = \{M, I, U\}$. Axiom (start string): MI . Production rules:

(R1) If a string ends in I , append U : $xI \Rightarrow xIU$.

(R2) If a string is Mx , duplicate x : $Mx \Rightarrow Mxx$.

(R3) Replace any III by U : $xIIIy \Rightarrow xUy$.

(R4) Delete any UU : $xUUy \Rightarrow xy$.

Key idea: reason about *invariants* that rules preserve, instead of searching blindly through derivations.

Homework: The MU-puzzle

Definition 2.1 (I-count and residue). For a string w , let $\#_I(w)$ be the number of I ’s in w , and define the residue

$$\varphi(w) = \#_I(w) \bmod 3 \in \{0, 1, 2\}.$$

Lemma 2.2 (Effect of each rule on $\#_I$). *For any string w :*

1. **(R1)** and **(R4)** do not change $\#_I$.
2. **(R2)** doubles the number of I 's after the initial M , so φ is multiplied by 2 modulo 3.
3. **(R3)** decreases $\#_I$ by 3, so φ is unchanged.

Proposition 2.3 (Invariant modulo 3). *Every string derivable from MI has $\varphi \in \{1, 2\}$. In particular, no derivable string has $\varphi = 0$.*

Proof. We use induction on the length of a derivation from MI .

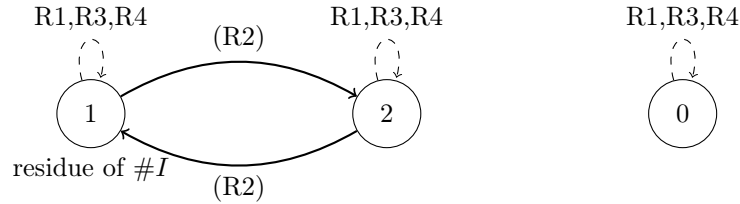
Base. $\varphi(MI) = 1$.

Step. Assume $\varphi \in \{1, 2\}$ for some derivable w . By Lemma 2.2, rules (R1), (R3), and (R4) keep φ unchanged, and rule (R2) maps $1 \leftrightarrow 2$ modulo 3. None of these operations yields 0 from a value in $\{1, 2\}$. Therefore the next string also has $\varphi \in \{1, 2\}$. \square

Theorem 2.4 (MU is unreachable). *MU cannot be derived from MI in the MIU-system.*

Proof. MU contains zero I 's, hence $\varphi(MU) = 0$. By Proposition 2.3, every derivable string has residue 1 or 2. Thus MU is not derivable. \square

Conclusion. Starting from MI we can toggle the residue $1 \leftrightarrow 2$ with (R2) and otherwise keep it fixed with (R1), (R3), (R4). We never reach residue 0, so no sequence of legal rule applications yields MU.



Question: If the MU-puzzle shows that some goals are unreachable due to invariants (like the mod-3 property of I 's), how does this idea connect to undecidability in programming languages?

2.2 Week 2

Lecture Summary

We introduced *Abstract Reduction Systems (ARS)*: a pair (A, R) with one-step reduction $R \subseteq A \times A$. Key notions: reducible/normal form, joinability, confluence, termination, and unique normal forms.

Homework Part 2: The 8 Combinations

We provide an example ARS for each combination of (confluent, terminating, unique NFs). If a row is impossible, we explain why.

Confluent	Terminating	Unique NFs	Example
True	True	True	$A = \{a\}, R = \emptyset$ (Fig. 1)
True	True	False	<i>Impossible</i>
True	False	True	$A = \{a, b\}, R = \{(a, a), (a, b)\}$ (Fig. 2)
True	False	False	$A = \{a\}, R = \{(a, a)\}$ (Fig. 3)
False	True	True	<i>Impossible</i>
False	True	False	$A = \{a, b, c\}, R = \{(a, b), (a, c)\}$ (Fig. 4)
False	False	True	<i>Impossible</i>
False	False	False	$A = \{a, b, c\}, R = \{(a, b), (a, c), (b, b), (c, c)\}$ (Fig. 5)

Why some rows are impossible. If an ARS has unique normal forms, it must be confluent. If an ARS is both confluent and terminating, then every element reduces to a unique normal form. Therefore the rows (T, T, F), (F, T, T), and (F, F, T) cannot occur.



Figure 1: Combination (True, True, True). Terminating, confluent, unique NF.



Figure 2: Combination (True, False, True). Non-terminating, confluent, unique NF b .

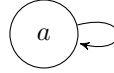


Figure 3: Combination (True, False, False). Non-terminating, confluent, no normal form.

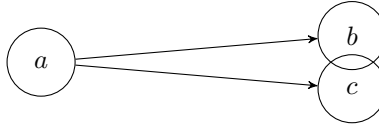


Figure 4: Combination (False, True, False). Terminating, not confluent; two distinct normal forms b, c are not joinable.

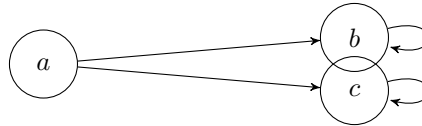


Figure 5: Combination (False, False, False). Non-terminating (loops), not confluent, no unique normal forms.

Conclusion. The MU-puzzle illustrates how invariants prove impossibility in a formal system. The ARS framework provides the general language to study rewrite systems via termination, confluence, and normal forms. The 8-combination analysis shows which behaviors are possible and which are structurally impossible.

Question: Could there be a general framework that unifies invariants with confluence and termination, so that impossibility and determinism appear as two sides of the same rewriting theory?

2.3 Week 3

Lecture Summary

TBD

Homework 3

Exercise 5 Consider an ARS with $[A = a, b^* = \varepsilon, a, b, aa, ab, ba, bb, aaa, \dots]$ and rewrite rules $[ab \rightarrow ba, \quad ba \rightarrow ab, \quad aa \rightarrow \varepsilon, \quad b \rightarrow \varepsilon.]$

Reduce some example strings such as *abba* and *bababa*.

$$abba \rightarrow aa \rightarrow \varepsilon, \quad bababa \rightarrow aaa \rightarrow a. \quad (1)$$

Find two strings that are not equivalent. How many non-equivalent strings can you find? ε a

These have different normal forms and cannot be transformed into each other.

How many equivalence classes does $\xleftrightarrow{*}$ have? What are the normal forms? There are two equivalence classes:

- (a) Strings whose normal form is ε ,
- (b) Strings whose normal form is a .

The class is determined by the parity of the number of a 's in the string.

Can you modify the ARS so that it becomes terminating without changing its equivalence classes? Yes. Remove one of the first two rules. They only permute a and b and do not affect equivalence classes, but having both makes the system non-terminating.

Question. If I remove all the b 's from a string, does the remaining word reduce to a or to ε ?" This can be answered using the ARS because $b \rightarrow \varepsilon$ always deletes b 's, and the final result depends only on whether the number of a 's left is odd or even. Odd $\mapsto a$, even $\mapsto \varepsilon$.

Exercise 5b Now replace the rule $aa \rightarrow \varepsilon$ with $aa \rightarrow a$.

1. Reduce some example strings such as *abba* and *bababa*.

$$abba \rightarrow aa \rightarrow a, \quad bababa \rightarrow aaa \rightarrow aa \rightarrow a. \quad (2)$$

2. Find two strings that are not equivalent.

- ε
- a

How many equivalence classes are there? What are the normal forms? There are two equivalence classes:

- (a) Strings with no a 's ; \rightarrow ; normal form ε ,
- (b) Strings with at least one a ; \rightarrow ; normal form a .

Modify the ARS to make it terminating. As above, remove one of the two swapping rules $ab \leftrightarrow ba$.

Question: Is the system confluent? That is, if a string can be reduced in two different ways, do the reductions always lead to the same normal form?

3 Evidence of Participation

4 Conclusion

References

[BLA] Author, [Title](#), Publisher, Year.