The Word Problem for Semigroups

1 Semigroup Equations

In these notes we consider equations which hold for algebras with an associative binary operation. Such algebras are called *semigroups*. For example, the real numbers under the operation of addition, or the operation of multiplication, are a semigroup. However, they are not a semigroup under subtraction since subtraction is not associative, that is,

$$(i - j) - k = i - (j - k),$$

is not always true. Similarly, the integers, or for that matter, the $n \times n$ matrices over the integers, under either addition or multiplication, form semigroups. Notice that the last example, the $n \times n$ matrices over the integers under multiplication is an example of a *noncommutative* semigroup, that is,

$$M \cdot N = N \cdot M$$

is not always true for square integer matrices M and N.

The most basic example of a semigroup is the set, Σ^*, of strings over an alphabet, Σ, where the binary operation is concatenation. In fact, we take the associativity of concatenation so much for granted that we typically write xyz for the concatenation of $x, y, z \in \Sigma^*$, without bothering to indicate whether they were concatenated as $(x \circ y) \circ z$ or as $x \circ (y \circ z)$, since of course it doesn't make any difference. When Σ^* is regarded as a semigroup under concatenation, it is traditional to refer to the strings in Σ^* as *words*.

To illustrate another situation where semigroups arise, consider the eight ways of moving a square in place: the four rotations of the square by $n\pi/2$ radians clockwise for $n = 0, 1, 2, 3$, and the four reflections about its axes—horizontal, vertical, upper left to lower right diagonal, and lower left to upper right diagonal. A natural binary operation applies to these motions, namely, "do the first motion, then do the second." It's easy enough to check that these eight are the only possible ways of moving the square in place, so the result of performing two consecutive motions is the same as some single one of the eight motions. These eight motions of the square are the elements of a semigroup, $\mathcal{A}$, known as the *Rigid Automorphisms of the Square*.

For example, suppose `c` means rotating the square $\pi/2$ radians **c**lockwise, and `v` represents reflecting it about its **v**ertical axis. Then `cv` denotes first rotating $\pi/2$ radians clockwise and then reflecting about the vertical. The net effect of doing the motions `c` followed by `v` is the same as doing single the reflection, `u`, about the **u**pper left to lower right diagonal. So we can say

$$\texttt{cv} = \texttt{u}.$$

Similarly, $((\mathtt{vc})\mathtt{c})\mathtt{c}$ denotes reflecting about the vertical and then rotating clockwise three times. Of course we would usually describe this sequence of motions simply with the word $\mathtt{vccc}$. We can omit the parentheses because the "do this, do that" operation, is obviously associative. As a matter of fact, $\mathtt{vccc}$ also has the same effect as $\mathtt{u}$, so we also have

$$\mathtt{vccc} = \mathtt{cv}. \tag{1}$$

One of the automorphisms in $\mathcal{A}$ is, $\mathtt{z}$, the "rotation" by $0\pi/2 =$ **z**ero radians. That is, $\mathtt{z}$ has no effect. So for any symbol, $s \in \{\mathtt{c}, \mathtt{v}, \mathtt{z}\}$, the word equations

$$\mathtt{z}s = s \text{ and } s\mathtt{z} = s \tag{2}$$

are true for $\mathcal{A}$.[1] It is also true that [2]

$$\mathtt{cccc} = \mathtt{z} \text{ and } \mathtt{vv} = \mathtt{z}. \tag{3}$$

We can use these equations to deduce many new ones. For example,

$$(\mathtt{cv})\mathtt{c} = (\mathtt{vccc})\mathtt{c} = \mathtt{v}(\mathtt{cccc}) = \mathtt{vz} = \mathtt{v},$$

so we conclude that

$$\mathtt{cvc} = \mathtt{v}. \tag{4}$$

In fact, it is not hard to show that the equations (1)–(3) are a *complete* set of axioms for proving word equations over the semigroup $\mathcal{A}$. Namely, *every* equation between words over the alphabet $\{\mathtt{c}, \mathtt{v}, \mathtt{z}\}$ which is true in $\mathcal{A}$ follows by repeatedly substituting occurrences of one side of an equation (1)–(3) by the other side.

More precisely, for any set, $\mathcal{E}$, of word equations, we say a word w results from word v by a single *left-right $\mathcal{E}$-substitution*, in symbols,

$$v \underset{\mathcal{E}}{\longrightarrow} w,$$

iff there are strings x, y and an equation of the form $l = r$ in $\mathcal{E}$ such that

$$v = xly \text{ and } w = xry.$$

We say w results from v by an *undirected $\mathcal{E}$*-substitution, in symbols,

$$v \underset{\mathcal{E}}{\longleftrightarrow} w,$$

iff either $v \underset{\mathcal{E}}{\longrightarrow} w$ or $w \underset{\mathcal{E}}{\longrightarrow} v$.

We say w results from v by left-right $\mathcal{E}$-substitutions, in symbols,

$$v \underset{\mathcal{E}}{\longrightarrow}^{*} w,$$

[1] Of course the equations (2) also hold for any *word* $x \in \{\mathtt{c}, \mathtt{v}, \mathtt{z}\}^*$ in place of the symbol s, but if we replaced s by x, then (2) would describe an infinite number of word equations instead of just six.

[2] The equations (3) imply that $\mathcal{A}$ satisfies the additional property that every element has a two-sided inverse, so $\mathcal{A}$ is in fact what is called a *group*.

iff w can be obtained from v by a series of zero or more single left-right $\mathcal{E}$-substitutions. Formally, there is a sequence of $n \geq 0$ words $u_0, u_1, \ldots, u_n$ such that v is identical to u_0, w is identical to u_n, and

$$u_0 \underset{\mathcal{E}}{\longrightarrow} u_1 \underset{\mathcal{E}}{\longrightarrow} \cdots \underset{\mathcal{E}}{\longrightarrow} u_n.$$

Finally, we say that v and w are *provably equal from* $\mathcal{E}$, in symbols

$$v \underset{\mathcal{E}}{\longleftrightarrow}^* w,$$

iff there is a sequence of words $u_0, u_1, \ldots, u_n$ such that v is identical to u_0, w is identical to u_n, and

$$u_0 \underset{\mathcal{E}}{\longleftrightarrow} u_1 \underset{\mathcal{E}}{\longleftrightarrow} \cdots \underset{\mathcal{E}}{\longleftrightarrow} u_n.$$

Problem 1. Let $\mathcal{E}_\mathcal{A}$ be the equations (1)–(3), and call the following eight words the *canonical* words for $\mathcal{A}$:

$$\mathtt{z}, \mathtt{c}, \mathtt{cc}, \mathtt{ccc}, \mathtt{v}, \mathtt{vc}, \mathtt{vcc}, \mathtt{vccc}.$$

(a) Show that for every word $v \in \{\mathtt{c}, \mathtt{v}, \mathtt{z}\}^*$, there is a canonical word w such that $v \underset{\mathcal{E}_\mathcal{A}}{\longleftrightarrow}^* w$.

(b) Using the fact that there are only eight rigid automorphisms of the square, conclude from part (a) that two words over $\{\mathtt{c}, \mathtt{v}, \mathtt{z}\}$ denote the same automorphisms iff they are $\mathcal{E}_\mathcal{A}$-provably equal to the same canonical word.

(c) Conclude that the *true* word equations of $\mathcal{A}$ are half-decidable.

(d) Prove that they are actually decidable.

2 The Word Problem

The example in the previous section illustrates a typical situation in semigroup theory: we are given some finite set $\mathcal{E}$ of word equations, and we want to know whether two words can be proved equal using these equations. This problem is called the Semigroup Word Problem. More precisely, we define the *Semigroup Word Problem*, SWP, to be the set of triples $(\mathcal{E}, v, w)$ such that $v \underset{\mathcal{E}}{\longleftrightarrow}^* w$.

For example, our proof of equation (4) implies that

$$(\{\mathtt{cv} = \mathtt{vccc}, \mathtt{cccc} = \mathtt{z}, \mathtt{vz} = \mathtt{v}\}, \mathtt{cvc}, \mathtt{v}) \in SWP.$$

But the equations used to prove $\mathtt{cvc} = \mathtt{v}$ are not enough to prove the equation $\mathtt{vv} = \mathtt{z}$, even though $\mathtt{vv} = \mathtt{z}$ is true over $\mathcal{A}$. That is,

Claim 2.1.

$$(\{\mathtt{cv} = \mathtt{vccc}, \mathtt{cccc} = \mathtt{z}, \mathtt{vz} = \mathtt{v}\}, \mathtt{vv}, \mathtt{z}) \notin SWP.$$

Problem 2. Prove Claim 2.1.

Problem 3. Explain why SWP is half-decidable.

The main result we want to establish is:

Theorem 2.2. *The Semigroup Word Problem is not decidable.*

The proof will follow from showing how, given any 2-counter machine, C, to construct a set $\mathcal{E}_C$ of word equations and two words, w_C and v_C, such that

$$C \text{ halts iff } (\mathcal{E}_C, v_C, w_C) \in SWP.$$

In other words, the proof will show that $HALT_{\mathsf{2CM}} \leq_{\mathrm{m}} SWP$. Since we know that the halting problem for 2-counter machines is undecidable, Theorem 2.2 will follow.

Actually, instead of the halting problem for 2-CM's, it will be technically convenient to use the *clean-halting* problem, $CLNHALT_{\mathsf{2CM}}$. A 2-CM, C, is said to *cleanly halt* if, started at instruction number zero with empty counters, it halts by transferring to the smallest possible halting line number with its counters empty again. That is, after starting in configuration $(0,0,0)$, it eventually halts in configuration $(n,0,0)$, where n is the number of instructions in the 2-CM.

Problem 4. Prove that $HALT_{\mathsf{2CM}} \leq_{\mathrm{m}} CLNHALT_{\mathsf{2CM}}$.

The idea behind the proof is to represent every configuration of a 2-CM, C, as a word over a certain alphabet. Specifically, let n be the number of instructions in C. The alphabet will be the integers from 0 to n, along with two additional symbols $\mathtt{a}$ and $\mathtt{b}$. A configuration (k,l,m) of C will be represented by the word

$$w_{(k,l,m)} ::= \mathtt{b}\overbrace{\mathtt{aa}\ldots\mathtt{a}}^{l} k \overbrace{\mathtt{a}\ldots\mathtt{a}}^{m}\mathtt{b}.$$

We will also construct some word equations $\mathcal{E}_C$ corresponding to the instructions of C. These equations will have the following key property:

Lemma 2.3 (Left-right Simulation).

$$w_{(k,l,m)} \xrightarrow[\mathcal{E}_C]{} w_{(k',l',m')} \textit{ iff } \textit{step}_C(k,l,m) = (k',l',m'). \tag{5}$$

From this Lemma we can immediately conclude that

$$C \text{ cleanly halts iff } w_{(0,0,0)} \xrightarrow[\mathcal{E}_C]{}{}^{*} w_{(n,0,0)},$$

where n is the length (number of instructions) of C.

The equations $\mathcal{E}_C$ are obtained directly from the instructions in C. Namely, depending on the instruction on line i, we include in $\mathcal{E}_C$ one or two equations of the forms given in Figure 1.

inc1	i = a i+1
inc2	i = i+1 a
dec1	a i = i+1 b i = b i+1
dec2	i a = i+1 i b = i+1 b
ifr1 j k	b i = b j a i = a k
ifr2 j k	i b = j b i a = k a

Figure 1: From 2-CM Instructions to Equations

0: ifr1 8 1
1: dec1
2: ifr1 9 3
3: dec1
4: ifr1 10 5
5: dec1
6: inc2
7: ifr1 8 1

Figure 2: C_{div3} sets Counter2 to $\lfloor(\text{Counter1})/3\rfloor$

$$
\begin{aligned}
\mathtt{b}0 &= \mathtt{b}8,\\
\mathtt{a}0 &= \mathtt{a}1,\\
\mathtt{a}1 &= 2,\\
\mathtt{b}1 &= \mathtt{b}2,\\
\mathtt{b}2 &= \mathtt{b}9,\\
\mathtt{a}2 &= \mathtt{a}3,\\
\mathtt{a}3 &= 4,\\
\mathtt{b}3 &= \mathtt{b}4,\\
\mathtt{b}4 &= \mathtt{b}10,\\
\mathtt{a}4 &= \mathtt{a}5,\\
\mathtt{a}5 &= 6,\\
\mathtt{b}5 &= \mathtt{b}6,\\
6 &= 7\mathtt{a},\\
\mathtt{b}7 &= \mathtt{b}8,\\
\mathtt{a}7 &= \mathtt{a}1.
\end{aligned}
$$

Figure 3: Simulating Equations for C_{div3}

For example, let C_{div3} be the 2-CM in Figure 2. Then the resulting equations $\mathcal{E}_{C_{\mathsf{div3}}}$ would be the ones in Figure 3.

It is easy to verify that for any C, at most one rule in $\mathcal{E}_C$ applies to any configuration word. The Left-right Simulation Lemma then follows directly from the definitions of $\mathcal{E}_C$ and step_C.

Problem 5. Exhibit the first dozen configuration words obtained from $w_{(0,7,0)}$ by doing left-right $\mathcal{E}_{C_{\mathsf{div3}}}$-substitutions. Indicate exactly which substitution occurred at each step.

3 Undirected Substitutions

The Left-right Simulation Lemma reveals that substituting lefthand sides of equations in $\mathcal{E}_C$ by righthand sides corresponds exactly to steps performed by C. But $\mathcal{E}_C$-provability also allows substituting righthand sides for lefthand sides, so we must examine whether there now could be an unintended way to get from one configuration word to another.

Since C is deterministic, a given configuration of C has a unique *next* configuration. The problem is that it may not have a unique *preceding* configuration. Consequently a configuration word may substitute right-to-left in more than one way. For example, suppose the zeroth and first instructions of C were both (`ifr1 2 2`). Then configurations $(0, k, m)$ and $(1, k, m)$ both lead in one step to configuration $(2, k, m)$. This implies that $w_{(2,k,m)}$ could substitute right-to-left into both $w_{(0,k,m)}$ and $w_{(1,k,m)}$. Therefore it is $\mathcal{E}_C$-provable that $w_{(0,k,m)} = w_{(1,k,m)}$, even though neither is obtainable from the other by left-right $\mathcal{E}_C$-substitutions. Nevertheless, provability among configuration words remains closely related to going from one configuration word to another strictly by left-right substitutions.

Lemma 3.1 (Undirected Simulation). *Let w be a configuration word of C.*

1. *If $v \underset{\mathcal{E}_C}{\longrightarrow} w$, then v is also a configuration word of C.*
2. *If $v \underset{\mathcal{E}_C}{\longleftrightarrow}{}^{*} w$, then there is an configuration word, u, such that $v \underset{\mathcal{E}_C}{\longrightarrow}{}^{*} u$ and $w \underset{\mathcal{E}_C}{\longrightarrow}{}^{*} u$.*

Proof. Part 1 follows from the definitions of configuration word and $\mathcal{E}_C$, as the reader may verify.

To prove Part 2, assume $v \underset{\mathcal{E}_C}{\longleftrightarrow}{}^{*} w$. Consider the *shortest* sequence of words $u_0, u_1, \ldots, u_n$ such that v is identical to u_0, w is identical to u_n, and

$$u_0 \underset{\mathcal{E}_C}{\longleftrightarrow} u_1 \underset{\mathcal{E}_C}{\longleftrightarrow} \cdots \underset{\mathcal{E}_C}{\longleftrightarrow} u_n.$$

By Part 1, each u_i is a configuration word. Suppose there are three consecutive words u_i, u_{i+1}, u_{i+2} which are obtained by a right-left substitution followed by left-right substitution. That is,

$$u_{i+1} \underset{\mathcal{E}_C}{\longrightarrow} u_i \text{ and } u_{i+1} \underset{\mathcal{E}_C}{\longrightarrow} u_{i+2}.$$

Now by the Left-right Simulation Lemma 2.3, u_i is the configuration word corresponding to one step of C from the u_{i+1} configuration—and so is u_{i+2}. But since C is deterministic, the u_{i+1} configuration of C uniquely determines the next configuration, so it must be that the u_i and u_{i+2}

configurations are the same, which is to say that u_i and u_{i+2} are the same word! Thus the two steps deriving u_{i+2} from u_{i+1} from u_i are actually deriving u_i from itself. This means that there is a shorter sequence of undirected substitutions from v to w obtained by removing u_{i+1} and u_{i+2} from the sequence, contradicting the fact that the original sequence was the shortest.

So in the shortest sequence of undirected substitutions from v to w, the substitutions from v must begin with (zero or more) left-right substitutions to some word u, and then be followed by (zero or more) right-left substitutions from u to w. □

Corollary 3.2. *Let w be a configuration word which represents a halting configuration of C. Then for all words v,*

$$v \underset{\mathcal{E}_C}{\longleftrightarrow}^* w \text{ implies } v \underset{\mathcal{E}_C}{\longrightarrow}^* w$$

Proof. By the Undirected Simulation Lemma 3.1, there is a series of $\mathcal{E}_C$-substitutions turning v into w in which the only right-left substitutions are at the end. But since w represents a halting configuration, it follows by the Left-right Simulation Lemma 2.3 that w no left-right substitution can start with w, which is to say that no right-left substitution can end with w. So the substitutions turning v into w must consist solely of left-right substitutions, *viz.*, $v \underset{\mathcal{E}_C}{\longrightarrow}^* w$. □

So we conclude that the equation $w_{(0,0,0)} = w_{(n,0,0)}$ is provable from the equations $\mathcal{E}_C$ iff C cleanly halts. In other words,

$$C \in CLNHALT_{\mathsf{2CM}} \text{ iff } (\mathcal{E}_C, w_{(0,0,0)}, w_{(n,0,0)}) \in SWP,$$

which shows that

Corollary 3.3. $CLNHALT_{\mathsf{2CM}} \leq_{\mathrm{m}} SWP$.

This is what was needed to complete the proof of Theorem 2.2.

Problem 6. In proving the undecidability of the Semigroup Word Problem, we used equations over an $n+3$ symbol alphabet $\{\mathtt{b}, \mathtt{a}, 0, \ldots, n\}$ to simulate a 2-Counter Machine of length n. Let 2-SWP be the SWP for words over a *two symbol* alphabet. Show that 2-SWP is undecidable.