Examining the Class of Formal Languages which are Expressible via Word Equations

Matthew Konefal

Loughborough University

February 2023

Recap: Word Equations

- A fundamental object within Combinatorics on Words
- Have the form $u = v$, where u and v are words comprised of
  - constants, taken from a finite alphabet $\Sigma = \{a, b, ...\}$.
  - variables, taken from a finite alphabet $\Xi = \{X, Y, ...\}$.
- Examples:

$$XabYc = ZXcY \qquad Y = XX$$

Recap: Word Equations

- Solution: an assignment h to the variables which makes both sides identical.
- Examples:

$$XabYc = ZXcY$$

$$\text{e.g.,} \qquad X \to bab, \qquad Y \to cc, \qquad Z \to ba.$$

Recap: Word Equations

- Solution: an assignment h to the variables which makes both sides identical.
- Examples:

$$bababccc = bababccc$$

$$\text{e.g.,} \qquad X \to bab, \qquad Y \to cc, \qquad Z \to ba.$$

Recap: Word Equations

- Solution: an assignment h to the variables which makes both sides identical.
- Examples:

$$bababccc = bababccc$$

$$\text{e.g.,} \qquad X \to bab, \qquad Y \to cc, \qquad Z \to ba.$$

$$Y = XX$$

$$\text{e.g.,} \qquad X \to ba, \qquad Y \to baba = (ba)^2.$$

Recap: Word Equations

- Solution: an assignment h to the variables which makes both sides identical.
- Examples:

$$bababccc = bababccc$$

$$\text{e.g.,} \qquad X \to bab, \qquad Y \to cc, \qquad Z \to ba.$$

$$baba = baba$$

$$\text{e.g.,} \qquad X \to ba, \qquad Y \to baba = (ba)^2.$$

Recap: Word Equations

- Solution: an assignment h to the variables which makes both sides identical.
- Examples:

$$babab c cc = babab c cc$$

$$\text{e.g.,} \qquad X \to bab, \qquad Y \to cc, \qquad Z \to ba.$$

$$Y = XX$$

$$\text{e.g.,} \qquad X \to ba, \qquad Y \to baba = (ba)^2.$$

$$X \to w, \qquad Y \to w^2, \qquad \text{for any } w \in \Sigma^*$$

Expressing Formal Languages via Word Equations

$$Y = XX$$

- Fix $h(Y)$.
- Then we can complete h to a solution if and only if $h(Y)$ is a square.
- Idea: given a word equation e and a variable Y, the set of possible $h(Y)$ occurring in solutions h is a formal language L.
- We say "e expresses L via its variable Y".

e.g., What language does $Yaa = X$ expresses via its variable X?

Expressing Formal Languages via Word Equations

$$Y = XX$$

- Fix $h(Y)$.
- Then we can complete h to a solution if and only if $h(Y)$ is a square.
- Idea: given a word equation e and a variable Y, the set of possible $h(Y)$ occurring in solutions h is a formal language L.
- We say "e expresses L via its variable Y".

  e.g., $Yaa = X$ expresses $L(\Sigma^* aa)$ via its variable X.

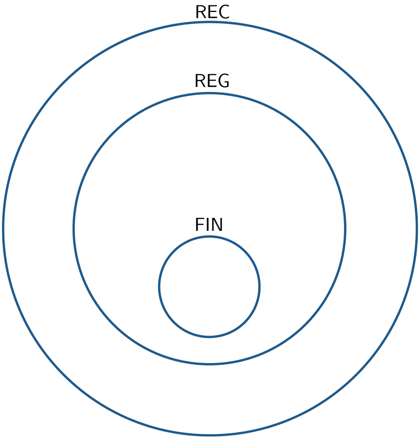
The Class WE

- WE: the class of languages "expressible" by word equations in this way.
- Not very well understood.
- Closure properties are unusual:

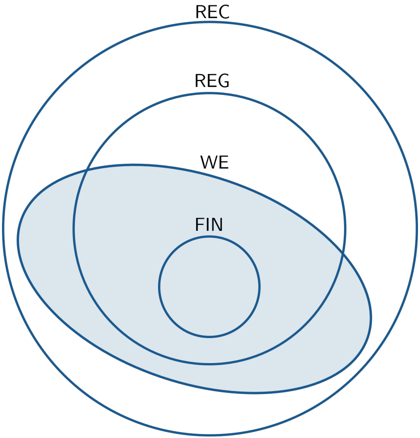
Theorem (Karhumäki et al. 2000)

WE is closed under union, concatenation, and finite perturbation. WE is not closed under complementation or Kleene star, but if $|L| = 1$, $L^* \in WE$.

WE and Other Classes



WE and Other Classes



- Also: every pattern language is expressible.

What Have I Been Doing?

Theorem (Day et al. 2022)

It is undecidable whether the language expressed via a variable in a word equation is regular.

I have been investigating the “reverse” statement:

Conjecture

It is decidable whether a regular language is in WE.

- I have introduced some necessary and sufficient conditions for a language to be expressible.
- closure of WE under union and concatenation $\Rightarrow$ Focus on submonoid languages $X^* \subset \Sigma^*$.

X is a Set of Powers of a Single Word

Let $X = \{a^2, a^3\}$. Is X^* expressible?

X is a Set of Powers of a Single Word

Let $X = \{a^2, a^3\}$. Is X^* expressible?

$X^* = \{a\}^* \backslash \{a\}$, so yes.

X is a Set of Powers of a Single Word

Let $X = \{a^2, a^3\}$. Is X^* expressible?

$X^* = \{a\}^* \setminus \{a\}$, so yes.
More generally,

Proposition

Let $w \in \Sigma^+$, and $\mathscr{E} \subseteq \mathbb{N}$. Let $X = \{w^i : i \in \mathscr{E}\}$. Then $X^* \in WE$.

Proof idea: All sufficiently large powers of w must be in X^*. So X^* is obtained from w^* by removing finitely many words. Expressibility then follows from the closure properties.

A Tool for Showing Inexpressibility

Kahrhumäki et al. have introduced a “pumping lemma”, which can be used to prove inexpressibility for a language L.

1. Assume to the contrary that $L \in WE$,
2. Choose a “good” factorisation scheme $\mathscr{F}$,
3. Pick some $w \in L$ with lots of distinct $\mathscr{F}$-factors,
4. Kahrhumäki : some $\mathscr{F}$-factors of w can be replaced with any word, and membership of L is preserved.
5. Replace those $\mathscr{F}$-factors to yield a word $w' \notin L$: a contradiction.

A Useful Factorisation Scheme

Definition

Let $h \in \Sigma^+$. To obtain the $\mathscr{F}_h$-factorisation of $w \in \Sigma^*$, we split w to the left of each occurrence of the factor h in w, (even for overlapping occurrences of h).

A Useful Factorisation Scheme

Examples of $\mathscr{F}_{aa}$-factorisation:

$$abaaabbaab, \qquad aaaaa, \qquad bbbbbba.$$

A Useful Factorisation Scheme

Examples of $\mathscr{F}_{aa}$-factorisation:

$$ab|a|aabb|aab, \qquad aaaaa, \qquad bbbbbba.$$

A Useful Factorisation Scheme

Examples of $\mathscr{F}_{aa}$-factorisation:

$$ab|a|aabb|aab, \qquad a|a|a|aa, \qquad bbbbbba.$$

A Useful Factorisation Scheme

Examples of $\mathscr{F}_{aa}$-factorisation:

$$ab|a|aabb|aab, \qquad a|a|a|aa, \qquad bbbbbba.$$

Lemma

For any h, $\mathscr{F}_h$-factorisation is suitable for use in the Karhumäki result.

Thinness

A thin language is one having a "forbidden" factor z.
e.g.,

$$L = \{a^n b^n : n \in \mathbb{N}\}$$

is thin (we can take $z = ba$).

$$L = \{aa, bbab, bbabb\}^*$$

is thin (we can take $z = b^5$).

Expressibility for Thin Submonoids

First main result:

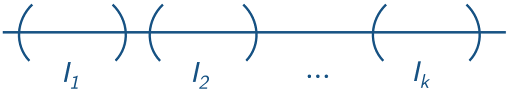
Theorem

A thin submonoid X^* is expressible if and only the words in X are pairwise commutative.

Proof idea: We showed ($\Leftarrow$) earlier. For ($\Rightarrow$), we prove the contrapositive using $\mathscr{F}_h$-factorisation in the Karhumäki tool.

A Useful Idea from the Proof

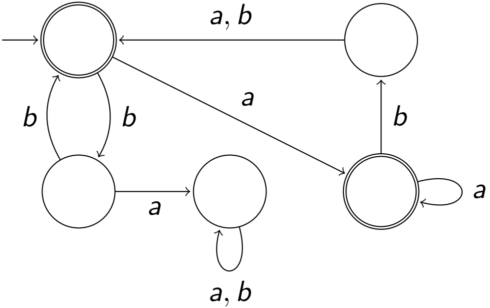
- We need to demonstrate that our constructed word w has k distinct $\mathscr{F}_h$-factors.
- To do this, we introduce disjoint intervals $I_1, I_2, \ldots, I_k \subset \mathbb{R}$.



- We then argue that, for each I_j, w has an $\mathscr{F}_h$-factor whose length lies in I_j.
- Thus w has at least k distinct $\mathscr{F}_h$-factors.

An Example

Let $\Sigma = \{a, b\}$, and let $X = \{a, aba, bb\}$. Is $X^* \in WE$?



ababa always takes the automaton to its sink state. So *ababa* is a forbidden factor for X^*; X^* is thin. The elements of X are not pairwise commutative. So from our result, $X^* \notin WE$.

Well-Formed Regular Expressions

- We call a regular expression "well-formed" if it is equal to $\emptyset$, or avoids the symbol $\emptyset$.
- It is elementary to rewrite any regular expression so that it is well-formed.
- The "well-formed" regular expressions are completely "additive".
- i.e., we cannot have "destructive" sub-expressions $L\emptyset \equiv \emptyset$.

Expressibility for Thin Regular Languages

Second main result:

Theorem

Let L be regular and thin. Let R be any well-formed regular expression for L. Then $L \in WE$ if and only if, for every sub-expression Y^* of R, the words of $L(Y)$ are pairwise commutative.

- The proof extends that of the previous result.
- Notice: R can be (pretty much) any regular expression for L.
- Corollary: Expressibility is decidable for thin regular languages.

An Example

- Let $\Sigma = \{a, b, c\}$, and consider

$$L = \{w : \text{ any even positions in } w \text{ must contain } b\}.$$

- L is not a submonoid: $a \in L$, but $aa \notin L$.
- L is thin: a forbidden factor is $z = aa$.
- A regular expression for L is $R := (\Sigma b)^*(\Sigma|\varepsilon)$.
- In R, the Kleene star is applied to the non-commutative set $\{ab, bb, cb\}$.
- Second main result $\Rightarrow L \notin WE$.

Motivation

- L is called dense if it is not thin; this case is less straightforward.
- Idea: at the last step of the Karhumäki tool, we need to be more thoughtful about what we “pump in”.
- Solution: refine the $\mathscr{F}_h$ technique to guarantee that we pump in only one place.
- This lets us investigate some dense submonoids X^*.

Third Main Result

Theorem

Let $X \subset \Sigma^+$ contain distinct words w_1, w_2 such that

- *no proper left-factor of w_1 or w_2 is in X,*
- *no proper right-factor of w_1 or w_2 is in X,*
- *no words in X have w_1 or w_2 as a proper left-factor,*
- *no words in X have w_1 or w_2 as a proper right-factor.*

Then $X^* \in WE$ if and only if $\Sigma \subseteq X$.

Proof idea: Use w in the Karhumäki tool made up of just w_1 and w_2. We can pump some $z \notin X^*$ into w in exactly one place using our refined method. From the premises, we can "strip" everything off the pumped word to conclude that $z \in X^*$: a contradiction.

Third Main Result: Corollaries

This result allows us to characterise the expressibility of X^* in following cases:

- the lengths of words in X have a common divisor $p > 1$,
- X is bifix, (i.e., no word in X is a left- nor a right-factor of any other),
- X is a power of a "code",

$\vdots$

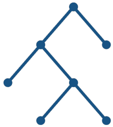
An Example

Let $\Sigma = \{a, b\}$, and $X = \{aaa, aab, ab, ba, bb\}$. Is $X^* \in WE$?

- Take $w_1 = aaa$, $w_2 = ba$.
- The proper left- and right-factors of w_1 and w_2 are aa, a, and b, none of which are in X.
- No word in X has w_1 or w_2 as a proper left- or right-factor.
- $\Sigma \not\subseteq X$.
- Third main result $\Rightarrow X^* \notin WE$.

X Obtained From a Tree

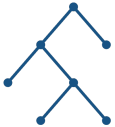
Let $|\Sigma| = n$. An n-ary rooted tree $\mathscr{T}$ represents a set $X \subset \Sigma^+$ according to the possible "downward" paths in $\mathscr{T}$. For instance,



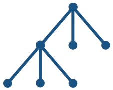
$X = \{aa, aba, abb, b\}$,

X Obtained From a Tree

Let $|\Sigma| = n$. An n-ary rooted tree $\mathscr{T}$ represents a set $X \subset \Sigma^+$ according to the possible "downward" paths in $\mathscr{T}$. For instance,

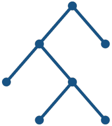


$X = \{aa, aba, abb, b\}$,

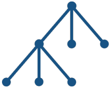


X Obtained From a Tree

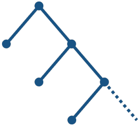
Let $|\Sigma| = n$. An n-ary rooted tree $\mathscr{T}$ represents a set $X \subset \Sigma^+$ according to the possible "downward" paths in $\mathscr{T}$. For instance,



$X = \{aa, aba, abb, b\}$,

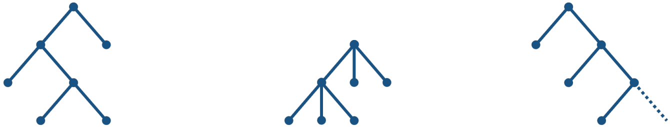


$X = \{aa, ab, ac, b, c\}$,



X Obtained From a Tree

Let $|\Sigma| = n$. An n-ary rooted tree $\mathscr{T}$ represents a set $X \subset \Sigma^+$ according to the possible "downward" paths in $\mathscr{T}$. For instance,

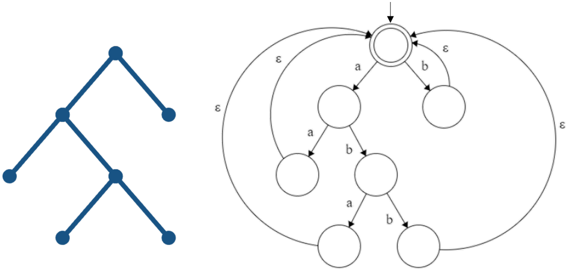


$X = \{aa, aba, abb, b\}$, $\quad X = \{aa, ab, ac, b, c\}$, $\quad X = \{b^i a : i \in \mathbb{N}\}$.

- The sets representable in this way are called "maximal prefix".
- Given a tree $\mathscr{T}$, we want to know whether the corresponding (dense) submonoid X^* is expressible.

Synchronised Trees

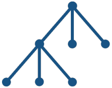
Given a maximal prefix set X, and its tree $\mathscr{T}$, we can interpret $\mathscr{T}$ as an automaton $\mathscr{A}$ for X^*, e.g.,



If $\mathscr{A}$ is synchronised, we call X synchronised too.

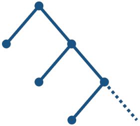
e.g., the above X is synchronised; a synchronising word is $z = bb$.

Synchronised Trees



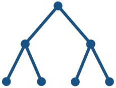
Synchronised

$z = b$



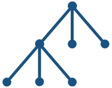
Synchronised

$z = a$

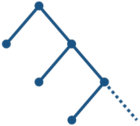


Not Synchronised

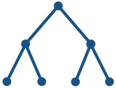
Synchronised Trees



Synchronised
$z = b$



Synchronised
$z = a$



Not Synchronised

If X is regular and synchronised, there is a technique we can apply to obtain a "nice" regular expression for X^*:

A "Nice" Regular Expression for X^*

- Choose a finite set Z of words that synchronise the tree automaton of X to its root node.
- Idea: In words from X^*, we can isolate the last occurrence of any $z \in Z$.

A "Nice" Regular Expression for X^*

- Choose a finite set Z of words that synchronise the tree automaton of X to its root node.
- Idea: In words from X^*, we can isolate the last occurrence of any $z \in Z$.
- Observation: Any word may precede this last occurrence.

A "Nice" Regular Expression for X^*

- Choose a finite set Z of words that synchronise the tree automaton of X to its root node.
- Idea: In words from X^*, we can isolate the last occurrence of any $z \in Z$.
- Observation: Any word may precede this last occurrence.
- Observation: We "catch up" with the X-factorisation after reading this last occurrence.

A "Nice" Regular Expression for X^*

- Choose a finite set Z of words that synchronise the tree automaton of X to its root node.
- Idea: In words from X^*, we can isolate the last occurrence of any $z \in Z$.
- Observation: Any word may precede this last occurrence.
- Observation: We "catch up" with the X-factorisation after reading this last occurrence.
- Compute regular expressions R_Z for Z and R' for $X^* \cap \overline{L_Z}$, using standard techniques.
- A regular expression for X^* is then

$$R := (\Sigma^* R_Z | \varepsilon) R'$$

Why is this Technique Helpful?

$$R = (\Sigma^* R_Z | \varepsilon) R'$$

- $L(R') = X^* \cap \overline{L_Z}$ is regular and thin.
- By our earlier result, it is easy to check whether $L(R') \in WE$.
- If $L(R') \in WE$, then the closure properties give $X^* = L(R) \in WE$.

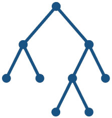
Why is this Technique Helpful?

$$R = (\Sigma^* R_Z | \varepsilon) R'$$

- $L(R') = X^* \cap \overline{L_Z}$ is regular and thin.
- By our earlier result, it is easy to check whether $L(R') \in WE$.
- If $L(R') \in WE$, then the closure properties give $X^* = L(R) \in WE$.
- Unfortunately, if $L(R') \notin WE$, it is still possible that $X^* = L(R) \in WE$.

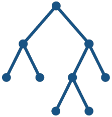
Some Applications

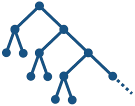
$$R = (\Sigma^* R_Z | \varepsilon) R'$$



Some Applications

$$R = (\Sigma^* R_Z|\varepsilon)R'$$

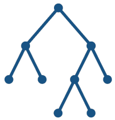




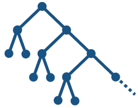
$$X = \{aa, ab, baa, bab, bb\}$$
$$Z = \{baa, bab\}$$
$$L(R') = X^* \cap \overline{L_Z}$$
$$R' = (aa)^*(ab|\varepsilon)(bb)^*$$
$$\Rightarrow L(R') \text{ expressible}$$
$$\Rightarrow X^* \text{ expressible}$$

Some Applications

$$R = (\Sigma^* R_Z | \varepsilon) R'$$



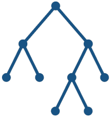
$$X = \{aa, ab, baa, bab, bb\}$$
$$Z = \{baa, bab\}$$
$$L(R') = X^* \cap \overline{L_Z}$$
$$R' = (aa)^*(ab|\varepsilon)(bb)^*$$
$$\Rightarrow L(R') \text{ expressible}$$
$$\Rightarrow X^* \text{ expressible}$$



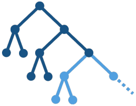
$$X = L(b^* a(a|b))$$
$$Z = \{baa, bab\}$$
$$L(R') = X^* \cap \overline{L_Z}$$
$$R' = (aa)^*(ab|\varepsilon)$$
$$\Rightarrow L(R') \text{ expressible}$$
$$\Rightarrow X^* \text{ expressible}$$

Some Applications

$$R = (\Sigma^* R_Z | \varepsilon) R'$$



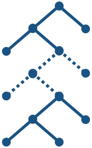
$$X = \{aa, ab, baa, bab, bb\}$$
$$Z = \{baa, bab\}$$
$$L(R') = X^* \cap \overline{L_Z}$$
$$R' = (aa)^*(ab|\varepsilon)(bb)^*$$
$$\Rightarrow L(R') \text{ expressible}$$
$$\Rightarrow X^* \text{ expressible}$$



$$X = L(b^* a(a|b))$$
$$Z = \{baa, bab\}$$
$$L(R') = X^* \cap \overline{L_Z}$$
$$R' = (aa)^*(ab|\varepsilon)$$
$$\Rightarrow L(R') \text{ expressible}$$
$$\Rightarrow X^* \text{ expressible}$$

- The left tree here is a 'pruned' version of the right tree. Performing this pruning at any depth yields an expressible submonoid.

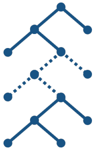
Some Applications: Families of Trees



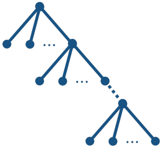
(1-parameter family)

X^* is always expressible

Some Applications: Families of Trees



(1-parameter family)
X^* is always expressible



(2-parameter family)
X^* is always expressible

Fourth Main Result

Proposition

Let X be maximal prefix and not synchronised. Then every regular expression for X^* has a sub-expression Y^* for which both

- *$L(Y)$ is not pairwise commutative, and*
- *$Y \neq \Sigma$.*

Proof idea: by contradiction

- We suspect that the above conclusion implies $X^* \notin WE$.
- Certainly, if we wanted to show expressibility for such an X^*, we could not use the same technique we have been using in the synchronised case.

Two Open Problems

Open Problem

Let $L \in REG$. Suppose that every regular expression for L has a sub-expression Y^* for which

- *$L(Y)$ is not pairwise commutative, and*
- *$Y \neq \Sigma$.*

Does this imply that $L \notin WE$?

Open Problem

Given $L \in REG$, is the property "$L \in WE$" decidable?

We suspect that in both cases, the answer is "yes".

Thank You!