Decomposition and synchronization of finite codes

Dominique Perrin

LIGM, Université Paris-Est

Joint work with Andrew Ryzhikov

10 June 2019

Codes

A *word* over alphabet Σ is a finite sequence of elements (called *letters*) from Σ. Examples: $abba$, $abcda$.

Definition

A set X of words is called a *code* if no word can be represented as a concatenation of words in X in two different ways.

Examples: $\{a, aba\}$, ba^*a are codes. The set $\{ab, ba, aba\}$ is not: we have $(aba)(ba) = (ab)(aba)$.

Codes

A *word* over alphabet Σ is a finite sequence of elements (called *letters*) from Σ. Examples: $abba, abcda$.

Definition

A set X of words is called a *code* if no word can be represented as a concatenation of words in X in two different ways.

Examples: $\{a, aba\}$, ba^*a are codes. The set $\{ab, ba, aba\}$ is not: we have $(aba)(ba) = (ab)(aba)$.

Synchronizing codes

Definition

A word w is called *synchronizing* for a code X if for any words u, v such that $uwv \in X^*$ we have $uw, wv \in X^*$. A code having a synchronizing word is also called *synchronizing*.

This means that once we see ww in the sequence of codewords, we can partition the decoding into two independent parts before the second w and after the first w regardless the context.

Example: the code $\{aa, ab, ba, bb\}$ is not synchronizing. Indeed, assume that there exists a synchronizing word w for it. Suppose that it's of even length. Then taking u, v of odd length implies $uwv \in X^*$, but uw, wv are of odd length and thus are not in X^*. The same for odd length.

Synchronizing codes

Definition

A word w is called *synchronizing* for a code X if for any words u, v such that $uwv \in X^*$ we have $uw, wv \in X^*$. A code having a synchronizing word is also called *synchronizing*.

This means that once we see ww in the sequence of codewords, we can partition the decoding into two independent parts before the second w and after the first w regardless the context.

Example: the code $\{aa, ab, ba, bb\}$ is not synchronizing. Indeed, assume that there exists a synchronizing word w for it. Suppose that it's of even length. Then taking u, v of odd length implies $uwv \in X^*$, but uw, wv are of odd length and thus are not in X^*. The same for odd length.

Synchronizing codes

Example: $\{a, aba\}$ is synchronizing.
Example: Take the code $\{a, baab\}$.

baab a a baab a a baab baab a
ba a baab a a baab a a bbaaba
ba abaabaabaa baab baab a

The word $baabbaab$ is synchronizing: after reading it, only one interpretation is possible.

Synchronizing codes

Example: $\{a, aba\}$ is synchronizing.
Example: Take the code $\{a, baab\}$.

baab a a baab a a baab baab a
ba a baab a a baab a a bbaaba
ba abaabaabaa baab baab a

The word $baabbaab$ is synchronizing: after reading it, only one interpretation is possible.

Prefix codes

Definition

A code is called *prefix* if no its codeword is a prefix of another codeword.

To a finite prefix code X one can naturally assign its literal automaton $A = (Q, \Sigma, \delta)$, which is a partial DFA. The set Q of states is the set of proper prefixes of words in X. The transition function is defined as

$$\delta(q, x) = \begin{cases} qx & \text{if } qx \text{ is a proper prefix of a codeword,} \\ \varepsilon & \text{if } qx \in X. \end{cases}$$

The number of states of a literal decoder does not exceed the total length of all words in the code.

Prefix codes

Definition

A code is called *prefix* if no its codeword is a prefix of another codeword.

To a finite prefix code X one can naturally assign its literal automaton $A = (Q, \Sigma, \delta)$, which is a partial DFA. The set Q of states is the set of proper prefixes of words in X. The transition function is defined as

$$\delta(q, x) = \begin{cases} qx & \text{if } qx \text{ is a proper prefix of a codeword,} \\ \varepsilon & \text{if } qx \in X. \end{cases}$$

The number of states of a literal decoder does not exceed the total length of all words in the code.

Prefix codes

Definition

A code is called *prefix* if no its codeword is a prefix of another codeword.

To a finite prefix code X one can naturally assign its literal automaton $A = (Q, \Sigma, \delta)$, which is a partial DFA. The set Q of states is the set of proper prefixes of words in X. The transition function is defined as

$$\delta(q, x) = \begin{cases} qx & \text{if } qx \text{ is a proper prefix of a codeword,} \\ \varepsilon & \text{if } qx \in X. \end{cases}$$

The number of states of a literal decoder does not exceed the total length of all words in the code.

Synchronizing automata

Definition

A word $w \in \Sigma^*$ is called *synchronizing* for a partial DFA $A = (Q, \Sigma, \delta)$ if the image of the set Q under the mapping defined by w in A has size exactly one.

A finite prefix code is synchronizing if and only if its literal automaton is synchronizing.

One-word codes

Our goal is to study (small) finite synchronizing codes.

Proposition

A one-word code $X = \{x\}$ is synchronizing if and only if x is primitive. If it is synchronizing, x is a synchronizing word for it.

One-word codes

Our goal is to study (small) finite synchronizing codes.

Proposition

A one-word code $X = \{x\}$ is synchronizing if and only if x is primitive. If it is synchronizing, x is a synchronizing word for it.

One-word codes

A word is called *primitive* if it is not a power of a shorter word.

Theorem (Weinbaum)

Each primitive word w has a conjugate $w' = uv$ such that u and w are unique factors of the circular word of w.

Corollary

Let A be the literal automaton of a synchronizing one-word code $X = \{x\}$. Then there exists a synchronizing word of length at most $\frac{|x|}{2}$ for A, and this bound is optimal.

Lower bound is provided by $\{a^k b a^{k+1} b\}$.

One-word codes

A word is called *primitive* if it is not a power of a shorter word.

Theorem (Weinbaum)

Each primitive word w has a conjugate $w' = uv$ such that u and w are unique factors of the circular word of w.

Corollary

Let A be the literal automaton of a synchronizing one-word code $X = \{x\}$. Then there exists a synchronizing word of length at most $\frac{|x|}{2}$ for A, and this bound is optimal.

Lower bound is provided by $\{a^k ba^{k+1} b\}$.

Compositions of codes

To study prefix codes of more than one word we need more powerful tools.
Let Y, Z be codes over the alphabets Σ_Y, Σ_Z, where Z is a finite code.
If $|\Sigma_Y| = |Z|$, there exists a bijection $\beta : \Sigma_Y \to Z$.
Using it, one can construct the code X which is the *composition* $Y \circ_\beta Z$ of the codes Y and Z by taking $X = \{\beta(y) \mid y \in Y\}$. Here $\beta(y) = \beta(y_1) \ldots \beta(y_n)$ for $y = y_1 \ldots y_n$ with $y_1, \ldots, y_n \in \Sigma_Y$.

Compositions of codes

To study prefix codes of more than one word we need more powerful tools.
Let Y, Z be codes over the alphabets Σ_Y, Σ_Z, where Z is a finite code.
If $|\Sigma_Y| = |Z|$, there exists a bijection $\beta : \Sigma_Y \to Z$.
Using it, one can construct the code X which is the *composition* $Y \circ_\beta Z$ of the codes Y and Z by taking $X = \{\beta(y) \mid y \in Y\}$. Here $\beta(y) = \beta(y_1) \ldots \beta(y_n)$ for $y = y_1 \ldots y_n$ with $y_1, \ldots, y_n \in \Sigma_Y$.

Complete codes

A word w is called a *factor* of w' if there exist words u, v such that $uwv = w'$. By X^* we denote the set of words which are concatenations of words from X.

Definition

A set of words X is called *complete* if every word is a factor of some word in X^*. Otherwise we call it *non-complete*, and the word which is not a factor is called *mortal*.

For example, $\{aa, ab, b\}$ is complete, while $\{aa, b\}$ is not since bab is not a factor of a concatenation of codewords.

Theorem (Schützenberger, 1955)

For a recognizable code, to be complete is equivalent to be a maximal by inclusion code.

Complete codes

A word w is called a *factor* of w' if there exist words u, v such that $uwv = w'$. By X^* we denote the set of words which are concatenations of words from X.

Definition

A set of words X is called *complete* if every word is a factor of some word in X^*. Otherwise we call it *non-complete*, and the word which is not a factor is called *mortal*.

For example, $\{aa, ab, b\}$ is complete, while $\{aa, b\}$ is not since bab is not a factor of a concatenation of codewords.

Theorem (Schützenberger, 1955)

For a recognizable code, to be complete is equivalent to be a maximal by inclusion code.

Complete codes

A word w is called a *factor* of w' if there exist words u, v such that $uwv = w'$. By X^* we denote the set of words which are concatenations of words from X.

Definition

A set of words X is called *complete* if every word is a factor of some word in X^*. Otherwise we call it *non-complete*, and the word which is not a factor is called *mortal*.

For example, $\{aa, ab, b\}$ is complete, while $\{aa, b\}$ is not since bab is not a factor of a concatenation of codewords.

Theorem (Schützenberger, 1955)

For a recognizable code, to be complete is equivalent to be a maximal by inclusion code.

Maximal decomposition

Theorem

Every prefix code is a composition of a complete code and a synchronizing code.

Thus, we can restrict to studying finite complete prefix codes.

Corollary

Every prefix code of $2, 3, 5, 6$ words is synchronizing.

Corollary

Every finite prefix code consisting only of primitive words is synchronizing.

Maximal decomposition

Theorem

Every prefix code is a composition of a complete code and a synchronizing code.

Thus, we can restrict to studying finite complete prefix codes.

Corollary

Every prefix code of $2, 3, 5, 6$ words is synchronizing.

Corollary

Every finite prefix code consisting only of primitive words is synchronizing.

Maximal decomposition

Theorem (P., Ryzhikov)

Provided an automaton A recognizing a prefix code X, the maximal decomposition of this code can be computed in polynomial time.

For a maximal decomposition $X = Y \circ Z$, Z is recognized by the automaton obtained by minimization of A with all states final. An automaton recognizing Y can be then recovered in a simple way.

Maximal decomposition

Theorem (P., Ryzhikov)

Provided an automaton A recognizing a prefix code X, the maximal decomposition of this code can be computed in polynomial time.

For a maximal decomposition $X = Y \circ Z$, Z is recognized by the automaton obtained by minimization of A with all states final. An automaton recognizing Y can be then recovered in a simple way.

General codes

The *combinatorial rank* of a set X of words is the smallest number k such that X is a subset of C^* for a set C of words, where C has cardinality k.

Theorem (Ryzhikov, 2019)

Every finite code such that its combinatorial rank equals its cardinality is synchronizing.

Corollary

Every two-word code is synchronizing.

Conjecture (Ryzhikov, 2019)

For every two-word code X there is a synchronizing word in X^2.

General codes

The *combinatorial rank* of a set X of words is the smallest number k such that X is a subset of C^* for a set C of words, where C has cardinality k.

Theorem (Ryzhikov, 2019)

Every finite code such that its combinatorial rank equals its cardinality is synchronizing.

Corollary

Every two-word code is synchronizing.

Conjecture (Ryzhikov, 2019)

For every two-word code X there is a synchronizing word in X^2.

General codes

Conjecture (P., Ryzhikov)

Every finite code is a composition of a complete and a synchronizing one.

That would imply in particular that every three-word code is synchronizing.

A words is called *unbordered* if none of its prefixes equals to its suffix.

Proposition (Ryzhikov)

Let $X = \{x, y, z\}$ be a code such that $|x| \geq |y| \geq |z|$ and x, y are unbordered. Then X is synchronizing.

General codes

Conjecture (P., Ryzhikov)

Every finite code is a composition of a complete and a synchronizing one.

That would imply in particular that every three-word code is synchronizing.
A words is called *unbordered* if none of its prefixes equals to its suffix.

Proposition (Ryzhikov)

Let $X = \{x, y, z\}$ be a code such that $|x| \geq |y| \geq |z|$ and x, y are unbordered. Then X is synchronizing.

Thank you! Any questions?