Theory of Computer Science

C4. Regular Languages: Pumping Lemma, Closure Properties and Decidability

Gabriele Röger

University of Basel

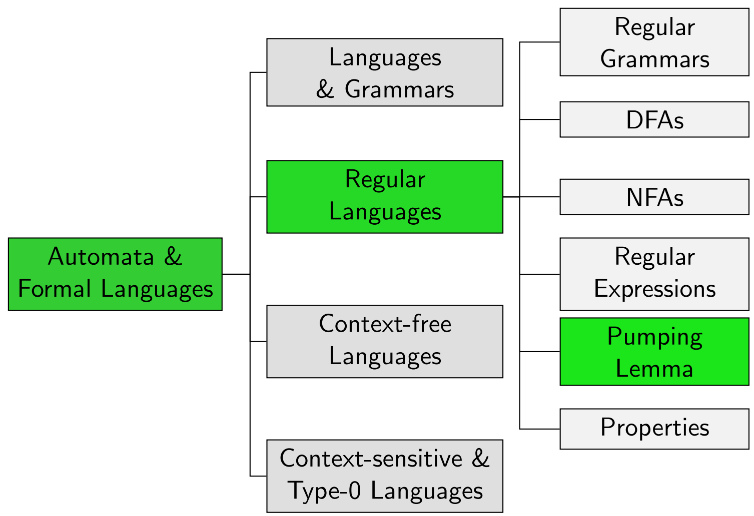
March 30, 2020

Theory of Computer Science

March 30, 2020 — C4. Regular Languages: Pumping Lemma, Closure Properties and Decidability

C4.1 Pumping Lemma

Overview



Pumping Lemma: Motivation

You can show that a language is regular by specifying an appropriate grammar, finite automaton, or regular expression. How can you you show that a language is not regular?

- Direct proof that no regular grammar exists that generates the language $\rightsquigarrow$ difficult in general
- Pumping lemma: use a necessary property that holds for all regular languages.

Picture courtesy of imagerymajestic / FreeDigitalPhotos.net

Pumping Lemma

Theorem (Pumping Lemma)

Let L be a regular language. Then there is an $n \in \mathbb{N}$ (a pumping number for L) such that all words $x \in L$ with $|x| \geq n$ can be split into $x = uvw$ with the following properties:

1. $|v| \geq 1$,
2. $|uv| \leq n$, *and*
3. $uv^iw \in L$ *for all* $i = 0, 1, 2, \ldots$.

Question: what if L is finite?

Pumping Lemma: Proof

Theorem (Pumping Lemma)

Let L be a regular language. Then there is an $n \in \mathbb{N}$ (a pumping number for L) such that all words $x \in L$ with $|x| \geq n$ can be split into $x = uvw$ with the following properties:

1. $|v| \geq 1$,
2. $|uv| \leq n$, *and*
3. $uv^iw \in L$ *for all* $i = 0, 1, 2, \ldots$.

Proof.

For regular L there exists a DFA $M = \langle Q, \Sigma, \delta, q_0, E \rangle$ with $\mathcal{L}(M) = L$. We show that $n = |Q|$ has the desired properties.

Consider an arbitrary $x \in \mathcal{L}(M)$ with length $|x| \geq |Q|$. Including the start state, M visits $|x| + 1$ states while reading x. Because of $|x| \geq |Q|$ at least one state has to be visited twice. ...

Pumping Lemma: Proof

Theorem (Pumping Lemma)

Let L be a regular language. Then there is an $n \in \mathbb{N}$ (a pumping number for L) such that all words $x \in L$ with $|x| \geq n$ can be split into $x = uvw$ with the following properties:

1. $|v| \geq 1$,
2. $|uv| \leq n$, *and*
3. $uv^iw \in L$ *for all* $i = 0, 1, 2, \ldots$.

Proof (continued).

Choose a split $x = uvw$ so M is in the same state after reading u and after reading uv. Obviously, we can choose the split in a way that $|v| \geq 1$ and $|uv| \leq |Q|$ are satisfied. ...

Pumping Lemma: Proof

Theorem (Pumping Lemma)

Let L be a regular language. Then there is an $n \in \mathbb{N}$ (a pumping number for L) such that all words $x \in L$ with $|x| \geq n$ can be split into $x = uvw$ with the following properties:

1. $|v| \geq 1$,
2. $|uv| \leq n$, and
3. $uv^i w \in L$ for all $i = 0, 1, 2, \ldots$.

Proof (continued).

The word v corresponds to a loop in the DFA after reading u and can thus be repeated arbitrarily often. Every subsequent continuation with w ends in the same end state as reading x. Therefore $uv^i w \in \mathcal{L}(M) = L$ is satisfied for all $i = 0, 1, 2, \ldots$. □

Pumping Lemma: Application

Using the pumping lemma (PL):

Proof of Nonregularity

- If L is regular, then the pumping lemma holds for L.
- By contraposition: if the PL does not hold for L, then L cannot be regular.
- That is: if there is no $n \in \mathbb{N}$ with the properties of the PL, then L cannot be regular.

Pumping Lemma: Caveat

Caveat:

The pumping lemma is a necessary condition for a language to be regular, but not a sufficient one.

- ⇝ there are languages that satisfy the pumping lemma conditions but are not regular
- ⇝ for such languages, other methods are needed to show that they are not regular (e.g., the Myhill-Nerode theorem)

Pumping Lemma: Example

Example

The language $L = \{\mathrm{a}^n\mathrm{b}^n \mid n \in \mathbb{N}\}$ is not regular.

Proof.

Assume L is regular. Then let p be a pumping number for L.

The word $x = \mathrm{a}^p\mathrm{b}^p$ is in L and has length $\geq p$.
Let $x = uvw$ be a split with the properties of the PL.

Then the word $x' = uv^2w$ is also in L. Since $|uv| \leq p$, uv consists only of symbols a and $x' = \mathrm{a}^{|u|}\mathrm{a}^{2|v|}\mathrm{a}^{p-|uv|}\mathrm{b}^p = \mathrm{a}^{p+|v|}\mathrm{b}^p$.

Since $|v| \geq 1$ it follows that $p + |v| \neq p$ and thus $x' \notin L$.

This is a contradiction to the PL. ⇝ L is not regular. □

Pumping Lemma: Another Example I

Example

The language $L = \{\mathtt{a}\mathtt{b}^n\mathtt{a}\mathtt{c}^{n+2} \mid n \in \mathbb{N}\}$ is not regular.

Proof.

Assume L is regular. Then let p be a pumping number for L.

The word $x = \mathtt{a}\mathtt{b}^p\mathtt{a}\mathtt{c}^{p+2}$ is in L and has length $\geq p$.
Let $x = uvw$ be a split with the properties of the PL.

From $|uv| \leq p$ and $|v| \geq 1$ we know that uv consists of one a followed by at most $p - 1$ bs.

We distinguish two cases, $|u| = 0$ and $|u| > 0$. ...

Pumping Lemma: Another Example II

Example

The language $L = \{\mathtt{a}\mathtt{b}^n\mathtt{a}\mathtt{c}^{n+2} \mid n \in \mathbb{N}\}$ is not regular.

Proof (continued).

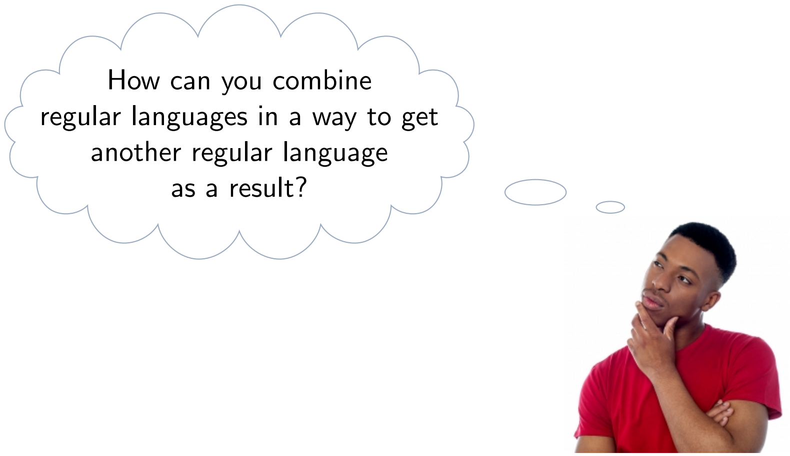
If $|u| = 0$, then word v starts with an a.
Hence, $uv^0w = \mathtt{b}^{p-|v|+1}\mathtt{a}\mathtt{c}^{p+2}$ does not start with symbol a and is therefore not in L. This is a contradiction to the PL.

If $|u| > 0$, then word v consists only of bs.
Consider $uv^0w = \mathtt{a}\mathtt{b}^{p-|v|}\mathtt{a}\mathtt{c}^{p+2}$. As $|v| \geq 1$, this word does not contain two more cs than bs and is therefore not in language L.
This is a contradiction to the PL.

We have in all cases a contradiction to the PL.
$\rightsquigarrow$ L is not regular. □

C4.2 Closure Properties

Closure Properties



Picture courtesy of stockimages / FreeDigitalPhotos.net

Closure Properties: Operations

Let L and L' be regular languages over Σ and Σ', respectively.

We consider the following operations:

- union $L \cup L' = \{w \mid w \in L \text{ or } w \in L'\}$ over $\Sigma \cup \Sigma'$
- intersection $L \cap L' = \{w \mid w \in L \text{ and } w \in L'\}$ over $\Sigma \cap \Sigma'$
- complement $\bar{L} = \{w \in \Sigma^* \mid w \notin L\}$ over Σ
- concatenation $LL' = \{uv \mid u \in L \text{ and } v \in L'\}$ over $\Sigma \cup \Sigma'$
  - special case: $L^n = L^{n-1}L$, where $L^0 = \{\varepsilon\}$
  - also called product
- star $L^* = \bigcup_{k \geq 0} L^k$ over Σ

German: Abschlusseigenschaften, Vereinigung, Schnitt, Komplement, Produkt, Stern

Closure Properties

Definition (Closure)

Let $\mathcal{K}$ be a class of languages.

Then $\mathcal{K}$ is closed...

- ...under union if $L, L' \in \mathcal{K}$ implies $L \cup L' \in \mathcal{K}$
- ...under intersection if $L, L' \in \mathcal{K}$ implies $L \cap L' \in \mathcal{K}$
- ...under complement if $L \in \mathcal{K}$ implies $\bar{L} \in \mathcal{K}$
- ...under concatenation if $L, L' \in \mathcal{K}$ implies $LL' \in \mathcal{K}$
- ...under star if $L \in \mathcal{K}$ implies $L^* \in \mathcal{K}$

German: Abgeschlossenheit, $\mathcal{K}$ ist abgeschlossen unter Vereinigung (Schnitt, Komplement, Produkt, Stern)

Closure Properties of Regular Languages

Theorem

The regular languages are closed under:

- *union*
- *intersection*
- *complement*
- *concatenation*
- *star*

Closure Properties

Proof.

Closure under union, concatenation, and star follows because for regular expressions α and β, the expressions $(\alpha|\beta)$, $(\alpha\beta)$ and (α^*) describe the corresponding languages.

Complement: Let $M = \langle Q, \Sigma, \delta, q_0, E\rangle$ be a DFA with $\mathcal{L}(M) = L$. Then $M' = \langle Q, \Sigma, \delta, q_0, Q \setminus E\rangle$ is a DFA with $\mathcal{L}(M') = \bar{L}$.

Intersection: Let $M_1 = \langle Q_1, \Sigma_1, \delta_1, q_{01}, E_1\rangle$ and $M_2 = \langle Q_2, \Sigma_2, \delta_2, q_{02}, E_2\rangle$ be DFAs. The product automaton

$$M = \langle Q_1 \times Q_2, \Sigma_1 \cap \Sigma_2, \delta, \langle q_{01}, q_{02}\rangle, E_1 \times E_2\rangle$$

$$\text{with } \delta(\langle q_1, q_2\rangle, a) = \langle \delta_1(q_1, a), \delta_2(q_2, a)\rangle$$

accepts $\mathcal{L}(M) = \mathcal{L}(M_1) \cap \mathcal{L}(M_2)$. □

German: Kreuzproduktautomat

C4.3 Decidability

Decision Problems and Decidability (1)

"Intuitive Definition:" Decision Problem, Decidability

A decision problem is an algorithmic problem where
- for a given input
- an algorithm determines if the input has a given property
- and then produces the output "yes" or "no" accordingly.

A decision problem is decidable if an algorithm for it (that always gives the correct answer) exists.

German: Entscheidungsproblem, Eingabe, Eigenschaft, Ausgabe, entscheidbar

Note: "exists" $\neq$ "is known"

Decision Problems and Decidability (2)

Notes:
- not a formal definition: we did not formally define "algorithm", "input", "output" etc. (which is not trivial)
- lack of a formal definition makes it difficult to prove that something is not decidable

$\rightsquigarrow$ studied thoroughly in the next part of the course

Decision Problems: Example

For now we describe decision problems in a semi-formal "given" / "question" way:

Example (Emptiness Problem for Regular Languages)

The emptiness problem $\mathrm{P}_\emptyset$ for regular languages is the following problem:

Given: regular grammar G
Question: Is $\mathcal{L}(G) = \emptyset$?

German: Leerheitsproblem

Word Problem

Definition (Word Problem for Regular Languages)

The word problem $\mathrm{P}_{\in}$ for regular languages is:

Given: regular grammar G with alphabet Σ and word $w \in \Sigma^*$

Question: Is $w \in \mathcal{L}(G)$?

German: Wortproblem (für reguläre Sprachen)

Decidability: Word Problem

Theorem

The word problem for regular languages is decidable.

Proof.

Construct a DFA M with $\mathcal{L}(M) = \mathcal{L}(G)$.
(The proofs in Chapter C2 describe a possible method.)

Simulate M on input w. The simulation ends after $|w|$ steps.

The DFA M is an end state after this iff $w \in \mathcal{L}(G)$.
Print "yes" or "no" accordingly. □

Emptiness Problem

Definition (Emptiness Problem for Regular Languages)

The emptiness problem $\mathrm{P}_{\emptyset}$ for regular languages is:

Given: regular grammar G

Question: Is $\mathcal{L}(G) = \emptyset$?

German: Leerheitsproblem

Decidability: Emptiness Problem

Theorem

The emptiness problem for regular languages is decidable.

Proof.

Construct a DFA M with $\mathcal{L}(M) = \mathcal{L}(G)$.

We have $\mathcal{L}(G) = \emptyset$ iff in the transition diagram of M there is no path from the start state to any end state.

This can be checked with standard graph algorithms (e.g., breadth-first search). □

Finiteness Problem

Definition (Finiteness Problem for Regular Languages)

The finiteness problem P_∞ for regular languages is:

Given: regular grammar G
Question: Is $|\mathcal{L}(G)| < \infty$?

German: Endlichkeitsproblem

Decidability: Finiteness Problem

Theorem

The finiteness problem for regular languages is decidable.

Proof.

Construct a DFA M with $\mathcal{L}(M) = \mathcal{L}(G)$.

We have $|\mathcal{L}(G)| = \infty$ iff in the transition diagram of M there is a cycle that is reachable from the start state and from which an end state can be reached.

This can be checked with standard graph algorithms. □

Intersection Problem

Definition (Intersection Problem for Regular Languages)

The intersection problem $\mathrm{P}_\cap$ for regular languages is:

Given: regular grammars G and G'
Question: Is $\mathcal{L}(G) \cap \mathcal{L}(G') = \emptyset$?

German: Schnittproblem

Decidability: Intersection Problem

Theorem

The intersection problem for regular languages is decidable.

Proof.

Using the closure of regular languages under intersection, we can construct (e.g., by converting to DFAs, constructing the product automaton, then converting back to a grammar) a grammar G'' with $\mathcal{L}(G'') = \mathcal{L}(G) \cap \mathcal{L}(G')$ and use the algorithm for the emptiness problem $\mathrm{P}_\emptyset$. □

Equivalence Problem

Definition (Equivalence Problem for Regular Languages)

The equivalence problem $\mathrm{P}_=$ for regular languages is:

Given: regular grammars G and G'
Question: Is $\mathcal{L}(G) = \mathcal{L}(G')$?

German: Äquivalenzproblem

Decidability: Equivalence Problem

Theorem

The equivalence problem for regular languages is decidable.

Proof.

In general for languages L and L', we have

$$L = L' \text{ iff } (L \cap \bar{L}') \cup (\bar{L} \cap L') = \emptyset.$$

The regular languages are closed under intersection, union and complement, and we know algorithms for these operations.

We can therefore construct a grammar for $(L \cap \bar{L}') \cup (\bar{L} \cap L')$ and use the algorithm for the emptiness problem $\mathrm{P}_\emptyset$. □

Summary

- The pumping lemma can be used to show that a language is not regular.
- The regular languages are closed under all usual operations (union, intersection, complement, concatenation, star).
- All usual decision problems (word problem, emptiness, finiteness, intersection, equivalence) are decidable for regular languages.