Summary last week

- diagonal language d is $\{x \in \{0,1\}^* \mid M_x \text{ accepts } x\}$
- diagonalising away: $cd = \{0,1\}^* - d$ distinct from all languages accepted by TMs
- hence membership problem MP $:= \{M\#x \mid M \text{ accepts } x\}$ not recursive
- similarity proofs HP (previous lecture), MP non-recursive: diagonalising away

Summary last week

- diagonal language d is $\{x \in \{0,1\}^* \mid M_x \text{ accepts } x\}$
- diagonalising away: $cd = \{0,1\}^* - d$ distinct from all languages accepted by TMs
- hence membership problem MP $:= \{M\#x \mid M \text{ accepts } x\}$ not recursive
- similarity proofs HP (previous lecture), MP non-recursive: diagonalising away
- r.e. languages closed under union, intersection, but not complement, difference
- recursive languages closed under union, intersection, complement, difference

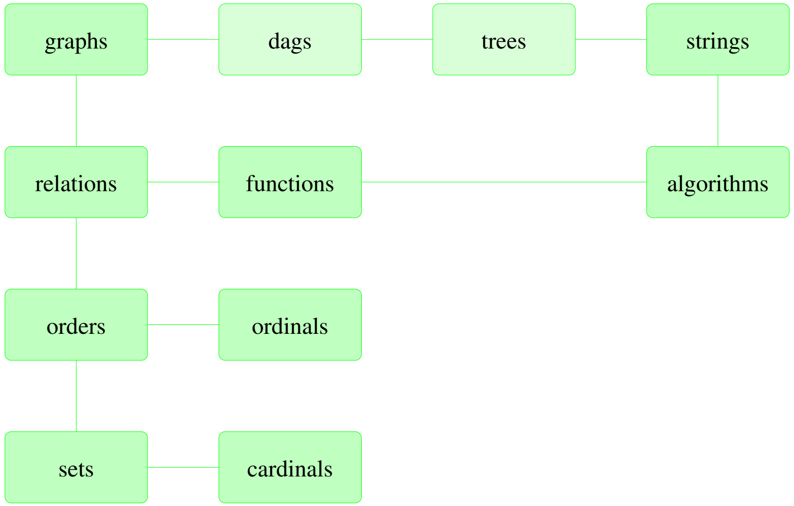
Summary last week

- diagonal language d is $\{x \in \{0,1\}^* \mid M_x \text{ accepts } x\}$
- diagonalising away: $cd = \{0,1\}^* - d$ distinct from all languages accepted by TMs
- hence membership problem MP $:= \{M\#x \mid M \text{ accepts } x\}$ not recursive
- similarity proofs HP (previous lecture), MP non-recursive: diagonalising away
- r.e. languages closed under union, intersection, but not complement, difference
- recursive languages closed under union, intersection, complement, difference
- f is reduction from L to L' if f computable and $\forall x$, $x \in L$ iff $f(x) \in L'$
- $L \leq L'$, L reducible to L', if there exists reduction f from L to L'
- if L non-recursive and $L \leq L'$ then L' is non-recursive
- MP $\leq$ HP and HP $\leq$ MP

Course themes

- directed and undirected graphs
- relations and functions
- orders and induction
- trees and dags
- finite and infinite counting
- elementary number theory
- Turing machines, algorithms, and complexity
- decidable and undecidable problem

Discrete structures



Regular languages

Question

What languages can be accepted for machines more restricted than TMs?

Regular languages

We consider finite automata. These accept regular languages, and will show these are recursive, but not necessarily the other way around,

relevance of regular languages

- software for designing and testing of digital circuits
- software components of compiler, e.g. for lexical analysis:
- software for searching in long texts
- software to verify all kinds of systems having a finite number of states
- components of computer games (computer-controlled non-player-character)

Deterministic finite automata (DFAs)

Example

$\emptyset$ and the set of all strings are regular, as are all finite languages.

Deterministic finite automata (DFAs)

Example

$\emptyset$ and the set of all strings are regular, as are all finite languages.

Definition

A DFA is a 5-tuple $A = (Q, \Sigma, \delta, s, F)$ with

1. Q a finite set of states
2. Σ a finite set of input symbols, (Σ is called the input alphabet)
3. $\delta : Q \times \Sigma \to Q$ the transition function
4. $s \in Q$, the start or initial state
5. $F \subseteq Q$ a finite set of accepting or final states

Deterministic finite automata (DFAs)

Example

$\emptyset$ and the set of all strings are regular, as are all finite languages.

Definition

A DFA is a 5-tuple $A = (Q, \Sigma, \delta, s, F)$ with

1. Q a finite set of states
2. Σ a finite set of input symbols, (Σ is called the input alphabet)
3. $\delta \colon Q \times \Sigma \to Q$ the transition function
4. $s \in Q$, the start or initial state
5. $F \subseteq Q$ a finite set of accepting or final states

Beware: δ must be defined, for all possible inputs

Transition table

	$a_1 \in \Sigma$	$a_2 \in \Sigma$	$\cdots$
$q_1 \in Q$	$\delta(q_1, a_1)$	$\delta(q_1, a_2)$	$\cdots$
$q_2 \in Q$	$\delta(q_2, a_1)$		
$\vdots$	$\vdots$		

Transition table

	$a_1 \in \Sigma$	$a_2 \in \Sigma$	$\cdots$
$q_1 \in Q$	$\delta(q_1, a_1)$	$\delta(q_1, a_2)$	$\cdots$
$q_2 \in Q$	$\delta(q_2, a_1)$		
$\vdots$	$\vdots$		

Transition graph

For a DFA $A = (Q, \Sigma, \delta, s, F)$, its (directed) transition graph with initial state d and final states F where:

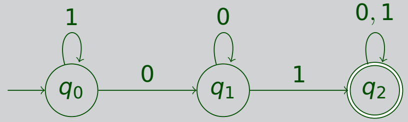
1. the states are the nodes
2. the edges E are
$$(p, q) \qquad p, q \in Q \text{ and } \exists a \in \Sigma \text{ with } \delta(p, a) = q$$
3. the edges are labelled by symbols by a function $b \colon E \to \Sigma$ defined by
$$(p, q) \mapsto a \qquad \text{if } \delta(p, a) = q$$

Example

The DFA $A = (\{q_0, q_1, q_2\}, \{0, 1\}, \delta, q_0, \{q_2\})$ with transition table

	0	1
$\to q_0$	q_1	q_0
q_1	q_1	q_2
$*q_2$	q_2	q_2

has the following transition graph:



Definition (extending the transition function)

Let δ be a transition function. The extended transition function $\hat{\delta}\colon Q \times \Sigma^* \to Q$ is inductively defined by:

$$\hat{\delta}(q, \epsilon) := q$$
$$\hat{\delta}(q, xa) := \delta(\hat{\delta}(q, x), a) \qquad x \in \Sigma^*,\ a \in \Sigma$$

Definition (extending the transition function)

Let δ be a transition function. The extended transition function $\hat{\delta}\colon Q \times \Sigma^* \to Q$ is inductively defined by:

$$\hat{\delta}(q, \epsilon) := q$$
$$\hat{\delta}(q, xa) := \delta(\hat{\delta}(q, x), a) \qquad x \in \Sigma^*,\ a \in \Sigma$$

Definition

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA; the language $\mathsf{L}(A)$ accepted by A is:

$$\mathsf{L}(A) := \{x \in \Sigma^* \mid \hat{\delta}(q_0, x) \in F\}$$

Definition (extending the transition function)

Let δ be a transition function. The extended transition function $\hat{\delta}\colon Q \times \Sigma^* \to Q$ is inductively defined by:

$$\hat{\delta}(q, \epsilon) := q$$
$$\hat{\delta}(q, xa) := \delta(\hat{\delta}(q, x), a) \qquad x \in \Sigma^*,\ a \in \Sigma$$

Definition

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA; the language $\mathsf{L}(A)$ accepted by A is:

$$\mathsf{L}(A) := \{x \in \Sigma^* \mid \hat{\delta}(q_0, x) \in F\}$$

Example

For the DFA A above, $\hat{\delta}(q_0, 0010) = q_2$
$\hat{\delta}(q_0, 0010)$ is computed recursively as follows:

- $\hat{\delta}(q_0, 0010) = \delta(\hat{\delta}(q_0, 001), 0) = \delta(q_2, 0) = q_2$
- $\hat{\delta}(q_0, 001) = \delta(\hat{\delta}(q_0, 00), 1) = \delta(q_1, 1) = q_2$
- $\hat{\delta}(q_0, 00) = \delta(\hat{\delta}(q_0, 0), 0) = \delta(q_1, 0) = q_1$
- $\hat{\delta}(q_0, 0) = \delta(\hat{\delta}(q_0, \epsilon), 0) = \delta(q_0, 0) = q_1$

Example

For the DFA A above, $\hat{\delta}(q_0, 0010) = q_2$
$\hat{\delta}(q_0, 0010)$ is computed recursively as follows:

- $\hat{\delta}(q_0, 0010) = \delta(\hat{\delta}(q_0, 001), 0) = \delta(q_2, 0) = q_2$
- $\hat{\delta}(q_0, 001) = \delta(\hat{\delta}(q_0, 00), 1) = \delta(q_1, 1) = q_2$
- $\hat{\delta}(q_0, 00) = \delta(\hat{\delta}(q_0, 0), 0) = \delta(q_1, 0) = q_1$
- $\hat{\delta}(q_0, 0) = \delta(\hat{\delta}(q_0, \epsilon), 0) = \delta(q_0, 0) = q_1$

Example

For the DFA A, we have $\mathrm{L}(A) = \{x01y \mid x, y \in \Sigma^*\}$. The language $\mathrm{L}(A)$ is the set of all words in which 01 occurs somewhere (or rather of words not of the form: a number of 1s followed by a number of 0s)

Example

For the DFA A above, $\hat{\delta}(q_0, 0010) = q_2$
$\hat{\delta}(q_0, 0010)$ is computed recursively as follows:

- $\hat{\delta}(q_0, 0010) = \delta(\hat{\delta}(q_0, 001), 0) = \delta(q_2, 0) = q_2$
- $\hat{\delta}(q_0, 001) = \delta(\hat{\delta}(q_0, 00), 1) = \delta(q_1, 1) = q_2$
- $\hat{\delta}(q_0, 00) = \delta(\hat{\delta}(q_0, 0), 0) = \delta(q_1, 0) = q_1$
- $\hat{\delta}(q_0, 0) = \delta(\hat{\delta}(q_0, \epsilon), 0) = \delta(q_0, 0) = q_1$

Example

For the DFA A, we have $\mathrm{L}(A) = \{x01y \mid x, y \in \Sigma^*\}$. The language $\mathrm{L}(A)$ is the set of all words in which 01 occurs somewhere (or rather of words not of the form: a number of 1s followed by a number of 0s)

Definition

A formal language L is regular, if $\exists$ DFA A, such that $\mathrm{L}(A) = L$

Closedness of the regular languages

Theorem

] Let L, M be regular languages (over the alphabet Σ). Then

1. *the complement $\sim L$ is regular*
2. *the intersection $L \cap M$ is regular*
3. *the union $L \cup M$ ist regular*
4. *the set difference $L \setminus M$ ist regular*

Sketch.

- swap accept/not-accept states
- pair of states; (q, q') accept if q and q' accept
- pair of states: (q, q') accept if q or q' accept
- $L \setminus M = L \cap \sim M$ and previous items

Limitations of finite automata

Example

Consider the language

$$B = \{\mathrm{a}^n\mathrm{b}^n \mid n \geqslant 0\} = \{\epsilon, \mathrm{ab}, \mathrm{aabb}, \mathrm{aaabbb}, \dots\}$$

The language B is not regular (note that it is recursive)

Limitations of finite automata

Example

Consider the language

$$B = \{a^n b^n \mid n \geqslant 0\} = \{\epsilon, ab, aabb, aaabbb, \dots\}$$

The language B is not regular (note that it is recursive)

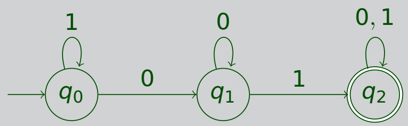
Example

Consider the language

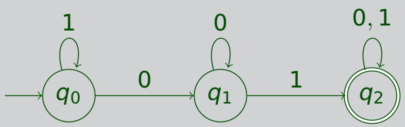
$$C = \{0^{2^n} \mid n \geqslant 0\} = \{0, 00, 0000, 00000000, \dots\}$$

The language C is not regular

Example



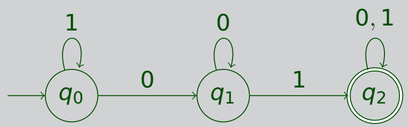
Example



Question

What can we say about the states the automaton goes 'through' to accept the word $w = 0000110$?

Example



Question

What can we say about the states the automaton goes 'through' to accept the word $w = 0000110$?

Answer

since $\ell(w) = 7 > 3 = |Q|$ the automaton must go through some state at least twice; the automaton cycles

Theorem (Pumping lemma)

Let L be a regular language over Σ. Then there exists *a number $n \in \mathbb{N}$, such that for* all *words $w \in L$ of length at least n ($\ell(w) \geq n$), there* exist *words $x, y, z \in \Sigma^*$ such that $w = xyz$ and*

- $y \neq \epsilon$;
- $\ell(xy) \leqslant n$; and
- for all $k \geqslant 0$, $x(y)^k z \in L$.

Theorem (Pumping lemma)

Let L be a regular language over Σ. Then there exists *a number $n \in \mathbb{N}$, such that for* all *words $w \in L$ of length at least n ($\ell(w) \geq n$), there* exist *words $x, y, z \in \Sigma^*$ such that $w = xyz$ and*

- $y \neq \epsilon$;
- $\ell(xy) \leqslant n$; and
- for all $k \geqslant 0$, $x(y)^k z \in L$.

Proof.

- Assume L is regular. Then there exists a DFA $A = (Q, \Sigma, \delta, s, F)$ such that $L = \mathrm{L}(A)$

Theorem (Pumping lemma)

Let L be a regular language over Σ. Then there exists *a number $n \in \mathbb{N}$, such that for* all *words $w \in L$ of length at least n ($\ell(w) \geq n$), there* exist *words $x, y, z \in \Sigma^*$ such that $w = xyz$ and*

- $y \neq \epsilon$;
- $\ell(xy) \leqslant n$; and
- for all $k \geqslant 0$, $x(y)^k z \in L$.

Proof.

- Assume L is regular. Then there exists a DFA $A = (Q, \Sigma, \delta, s, F)$ such that $L = \mathrm{L}(A)$
- Let $\#(Q) = n$ and

$$w = w_1 \cdots w_m \in L$$

with $w_1, \ldots, w_m \in \Sigma$ and $m \geq n$

Proof. (continued).

- define $p_l := \hat{\delta}(s, w_1 \cdots w_l)$;
  note that for $l = 0$, $w_1 \cdots w_l = \epsilon$ and hence $p_0 = s$

Proof. (continued).

- define $p_l := \hat{\delta}(s, w_1 \cdots w_l)$;
  note that for $l = 0$, $w_1 \cdots w_l = \epsilon$ and hence $p_0 = s$
- by the pigeon hole principle, there are $i, j \in \{0, \ldots, n\}$ such that $i < j$ and $p_i = p_j$:
  w has $\geq n + 1$ prefixes, but A has only n states

Proof. (continued).

- define $p_l := \hat{\delta}(s, w_1 \cdots w_l)$;
  note that for $l = 0$, $w_1 \cdots w_l = \epsilon$ and hence $p_0 = s$
- by the pigeon hole principle, there are $i, j \in \{0, \ldots, n\}$ such that $i < j$ and $p_i = p_j$:
  w has $\geq n + 1$ prefixes, but A has only n states
- we decompose w

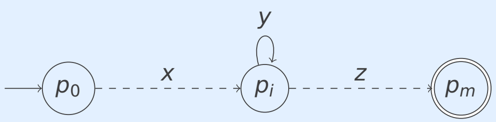
$$\underbrace{w_1 \cdots w_i}_{x} \quad \underbrace{w_{i+1} \cdots w_j}_{y \neq \epsilon} \quad \underbrace{w_{j+1} \cdots w_m}_{z}$$

Proof. (continued).

- define $p_l := \hat{\delta}(s, w_1 \cdots w_l)$;
  note that for $l = 0$, $w_1 \cdots w_l = \epsilon$ and hence $p_0 = s$
- by the pigeon hole principle, there are $i, j \in \{0, \ldots, n\}$ such that $i < j$ and $p_i = p_j$:
  w has $\geq n + 1$ prefixes, but A has only n states
- we decompose w

$$\underbrace{w_1 \cdots w_i}_{x} \quad \underbrace{w_{i+1} \cdots w_j}_{y \neq \epsilon} \quad \underbrace{w_{j+1} \cdots w_m}_{z}$$

- the situation can be depicted as:

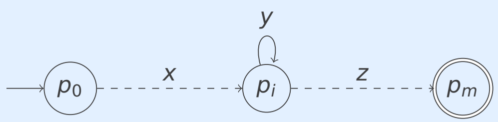


Proof. (continued).

- define $p_l := \hat{\delta}(s, w_1 \cdots w_l)$;
  note that for $l = 0$, $w_1 \cdots w_l = \epsilon$ and hence $p_0 = s$
- by the pigeon hole principle, there are $i, j \in \{0, \ldots, n\}$ such that $i < j$ and $p_i = p_j$:
  w has $\geq n + 1$ prefixes, but A has only n states
- we decompose w

$$\underbrace{w_1 \cdots w_i}_{x} \quad \underbrace{w_{i+1} \cdots w_j}_{y \neq \epsilon} \quad \underbrace{w_{j+1} \cdots w_m}_{z}$$

- the situation can be depicted as:



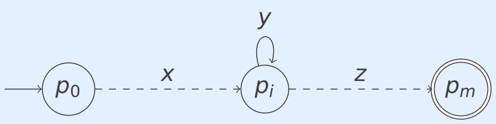
- to accept the word $x(y)^k z$, the automaton runs k times along the path connecting p_i to p_j

Proof. (continued).

- define $p_l := \hat{\delta}(s, w_1 \cdots w_l)$;
  note that for $l = 0$, $w_1 \cdots w_l = \epsilon$ and hence $p_0 = s$
- by the pigeon hole principle, there are $i, j \in \{0, \ldots, n\}$ such that $i < j$ and $p_i = p_j$: w has $\geq n + 1$ prefixes, but A has only n states
- we decompose w

$$\underbrace{w_1 \cdots w_i}_{x} \quad \underbrace{w_{i+1} \cdots w_j}_{y \neq \epsilon} \quad \underbrace{w_{j+1} \cdots w_m}_{z}$$

- the situation can be depicted as:



- to accept the word $x(y)^k z$, the automaton runs k times along the path connecting p_i to p_j ∎

Application of the pumping lemma

Theorem (Application (1))

Let L be a formal language over Σ such that:

- *for all $n \in \mathbb{N}$ there exists a word $w \in L$ with $\ell(w) \geqslant n$ such that*
- *for all $x, y, z \in \Sigma^*$ with $w = xyz$, $y \neq \epsilon$ and $\ell(xy) \leq n$, there exists a $k \in \mathbb{N}$ with $x(y)^k z \notin L$*

Then L is not regular. ∎

Application of the pumping lemma

Theorem (Application (1))

Let L be a formal language over Σ such that:

- *for all $n \in \mathbb{N}$ there exists a word $w \in L$ with $\ell(w) \geqslant n$ such that*
- *for all $x, y, z \in \Sigma^*$ with $w = xyz$, $y \neq \epsilon$ and $\ell(xy) \leq n$, there exists a $k \in \mathbb{N}$ with $x(y)^k z \notin L$*

Then L is not regular. ∎

Example (1)

Let $\Sigma = \{1\}$; then

$$D = \{w \in \Sigma^* \mid \ell(w) \text{is a prime number}\}$$

not regular

Example (2)

We show that for D we have

- for all $n \in \mathbb{N}$ there exists a word $w \in L$ with $\ell(w) \geqslant n$ such that
- for all $x, y, z \in \Sigma^*$ with $w = xyz$, $y \neq \epsilon$ and $\ell(xy) \leq n$ there exists $k \in \mathbb{N}$ mit $x(y)^k z \notin L$

Example (2)

We show that for D we have

- for all $n \in \mathbb{N}$ there exists a word $w \in L$ with $\ell(w) \geqslant n$ such that
- for all $x, y, z \in \Sigma^*$ with $w = xyz$, $y \neq \epsilon$ and $\ell(xy) \leq n$ there exists $k \in \mathbb{N}$ mit $x(y)^k z \notin L$

We choose $w = 1^p$, where p is a prime number great than or equal to $n + 2$; hence $w \in L$ and $\ell(w) = p \geqslant n + 2 \geqslant n$.

Example (2)

We show that for D we have

- for all $n \in \mathbb{N}$ there exists a word $w \in L$ with $\ell(w) \geqslant n$ such that
- for all $x, y, z \in \Sigma^*$ with $w = xyz$, $y \neq \epsilon$ and $\ell(xy) \leq n$ there exists $k \in \mathbb{N}$ mit $x(y)^k z \notin L$

We choose $w = 1^p$, where p is a prime number great than or equal to $n + 2$; hence $w \in L$ and $\ell(w) = p \geqslant n + 2 \geqslant n$.

Let x, y, z be arbitrary words such that $w = xyz$, $\ell(xy) \leqslant n$ and $y \neq \epsilon$.

Example (2)

We show that for D we have

- for all $n \in \mathbb{N}$ there exists a word $w \in L$ with $\ell(w) \geqslant n$ such that
- for all $x, y, z \in \Sigma^*$ with $w = xyz$, $y \neq \epsilon$ and $\ell(xy) \leq n$ there exists $k \in \mathbb{N}$ mit $x(y)^k z \notin L$

We choose $w = 1^p$, where p is a prime number great than or equal to $n + 2$; hence $w \in L$ and $\ell(w) = p \geqslant n + 2 \geqslant n$.

Let x, y, z be arbitrary words such that $w = xyz$, $\ell(xy) \leqslant n$ and $y \neq \epsilon$.

Set $m := \ell(y)$; We choose $k := \ell(xz) = p - m$. Consider

$$v := x(y)^{(p-m)} z$$

Example (2)

We show that for D we have

- for all $n \in \mathbb{N}$ there exists a word $w \in L$ with $\ell(w) \geqslant n$ such that
- for all $x, y, z \in \Sigma^*$ with $w = xyz$, $y \neq \epsilon$ and $\ell(xy) \leq n$ there exists $k \in \mathbb{N}$ mit $x(y)^k z \notin L$

We choose $w = 1^p$, where p is a prime number great than or equal to $n + 2$; hence $w \in L$ and $\ell(w) = p \geqslant n + 2 \geqslant n$.

Let x, y, z be arbitrary words such that $w = xyz$, $\ell(xy) \leqslant n$ and $y \neq \epsilon$.

Set $m := \ell(y)$; We choose $k := \ell(xz) = p - m$. Consider

$$v := x(y)^{(p-m)} z$$

Then $v \notin L$, since

$$\ell(v) = \ell(x(y)^{(p-m)} z) = (p - m) + m \cdot (p - m) = (p - m) \cdot (m + 1)\,.$$

That is, $\ell(v)$ is not a prime number, if $(p - m) > 1$ and $(m + 1) > 1$

Example

The language

$$E = \{w \in \Sigma^* \mid w \text{ contains as many 0s as 1s }\}$$

is not regular:

Example

The language

$$E = \{w \in \Sigma^* \mid w \text{ contains as many 0s as 1s }\}$$

is not regular:

1. Applying the pumping lemma becomes easy if we can find a “pumpable’ subword comprising only 0s

Example

The language

$$E = \{w \in \Sigma^* \mid w \text{ contains as many 0s as 1s }\}$$

is not regular:

1. Applying the pumping lemma becomes easy if we can find a “pumpable’ subword comprising only 0s
2. We choose the word $w := 0^n1^n \in E$

Example

The language

$$E = \{w \in \Sigma^* \mid w \text{ contains as many 0s as 1s }\}$$

is not regular:

1. Applying the pumping lemma becomes easy if we can find a “pumpable’ subword comprising only 0s
2. We choose the word $w := 0^n1^n \in E$
3. Consider all decompositions of w into x, y and z such that $\ell(xy) \leq n$ and $y \neq \epsilon$

Example

The language

$$E = \{w \in \Sigma^* \mid w \text{ contains as many 0s as 1s }\}$$

is not regular:

1. Applying the pumping lemma becomes easy if we can find a “pumpable’ subword comprising only 0s
2. We choose the word $w := 0^n 1^n \in E$
3. Consider all decompositions of w into x, y and z such that $\ell(xy) \leq n$ and $y \neq \epsilon$
4. We then must have $x = 0^i$, $y = 0^j$, $j \neq 0$ and $i + j \leqslant n$

Example

The language

$$E = \{w \in \Sigma^* \mid w \text{ contains as many 0s as 1s }\}$$

is not regular:

1. Applying the pumping lemma becomes easy if we can find a “pumpable’ subword comprising only 0s
2. We choose the word $w := 0^n 1^n \in E$
3. Consider all decompositions of w into x, y and z such that $\ell(xy) \leq n$ and $y \neq \epsilon$
4. We then must have $x = 0^i$, $y = 0^j$, $j \neq 0$ and $i + j \leqslant n$
5. choosing $k = 0$

Example

The language

$$E = \{w \in \Sigma^* \mid w \text{ contains as many 0s as 1s }\}$$

is not regular:

1. Applying the pumping lemma becomes easy if we can find a “pumpable’ subword comprising only 0s
2. We choose the word $w := 0^n 1^n \in E$
3. Consider all decompositions of w into x, y and z such that $\ell(xy) \leq n$ and $y \neq \epsilon$
4. We then must have $x = 0^i$, $y = 0^j$, $j \neq 0$ and $i + j \leqslant n$
5. choosing $k = 0$

we have $x(y)^0 z \notin E$, so the conditions of the pumping lemma are satisfied, hence L is not regular