Regular Languages and Expressions

Lecture 2

Thursday, August 25, 2022

LaTeXed: October 13, 2022 14:18

2.1

Regular Languages

Regular Languages

A class of simple but useful languages.

The set of regular languages over some alphabet Σ is defined inductively as:

1. $\emptyset$ is a regular language.
2. $\{\epsilon\}$ is a regular language.
3. $\{a\}$ is a regular language for each $a \in \Sigma$. Interpreting a as string of length 1.
4. If L_1, L_2 are regular then $L_1 \cup L_2$ is regular.
5. If L_1, L_2 are regular then $L_1 L_2$ is regular.
6. If L is regular, then $L^* = \cup_{n \geq 0} L^n$ is regular.
   The $\cdot^*$ operator name is **Kleene star**.
7. If L is regular, then so is $\overline{L} = \Sigma^* \setminus L$.

Regular languages are closed under operations of union, concatenation and Kleene star.

Regular Languages

A class of simple but useful languages.

The set of regular languages over some alphabet Σ is defined inductively as:

1. $\emptyset$ is a regular language.
2. $\{\epsilon\}$ is a regular language.
3. $\{a\}$ is a regular language for each $a \in \Sigma$. Interpreting a as string of length 1.
4. If L_1, L_2 are regular then $L_1 \cup L_2$ is regular.
5. If L_1, L_2 are regular then $L_1 L_2$ is regular.
6. If L is regular, then $L^* = \cup_{n \geq 0} L^n$ is regular.
   The $\cdot^*$ operator name is **Kleene star**.
7. If L is regular, then so is $\overline{L} = \Sigma^* \setminus L$.

Regular languages are closed under operations of union, concatenation and Kleene star.

Regular Languages

A class of simple but useful languages.
The set of regular languages over some alphabet Σ is defined inductively as:

1. $\emptyset$ is a regular language.
2. $\{\epsilon\}$ is a regular language.
3. $\{a\}$ is a regular language for each $a \in \Sigma$. Interpreting a as string of length 1.
4. If L_1, L_2 are regular then $L_1 \cup L_2$ is regular.
5. If L_1, L_2 are regular then $L_1 L_2$ is regular.
6. If L is regular, then $L^* = \cup_{n \geq 0} L^n$ is regular.
   The $\cdot^*$ operator name is **Kleene star**.
7. If L is regular, then so is $\overline{L} = \Sigma^* \setminus L$.

Regular languages are closed under operations of union, concatenation and Kleene star.

Regular Languages

A class of simple but useful languages.

The set of regular languages over some alphabet Σ is defined inductively as:

1. $\emptyset$ is a regular language.
2. $\{\epsilon\}$ is a regular language.
3. $\{a\}$ is a regular language for each $a \in \Sigma$. Interpreting a as string of length 1.
4. If L_1, L_2 are regular then $L_1 \cup L_2$ is regular.
5. If L_1, L_2 are regular then $L_1 L_2$ is regular.
6. If L is regular, then $L^* = \cup_{n \geq 0} L^n$ is regular.
   The $\cdot^*$ operator name is **Kleene star**.
7. If L is regular, then so is $\overline{L} = \Sigma^* \setminus L$.

Regular languages are closed under operations of union, concatenation and Kleene star.

Regular Languages

A class of simple but useful languages.

The set of regular languages over some alphabet Σ is defined inductively as:

1. $\emptyset$ is a regular language.
2. $\{\epsilon\}$ is a regular language.
3. $\{a\}$ is a regular language for each $a \in \Sigma$. Interpreting a as string of length 1.
4. If L_1, L_2 are regular then $L_1 \cup L_2$ is regular.
5. If L_1, L_2 are regular then $L_1 L_2$ is regular.
6. If L is regular, then $L^* = \cup_{n \geq 0} L^n$ is regular.
   The $\cdot^*$ operator name is **Kleene star**.
7. If L is regular, then so is $\overline{L} = \Sigma^* \setminus L$.

Regular languages are closed under operations of union, concatenation and Kleene star.

Regular Languages

A class of simple but useful languages.

The set of regular languages over some alphabet Σ is defined inductively as:

1. $\emptyset$ is a regular language.
2. $\{\epsilon\}$ is a regular language.
3. $\{a\}$ is a regular language for each $a \in \Sigma$. Interpreting a as string of length 1.
4. If L_1, L_2 are regular then $L_1 \cup L_2$ is regular.
5. If L_1, L_2 are regular then $L_1 L_2$ is regular.
6. If L is regular, then $L^* = \cup_{n \geq 0} L^n$ is regular.
   The $\cdot^*$ operator name is **Kleene star**.
7. If L is regular, then so is $\overline{L} = \Sigma^* \setminus L$.

Regular languages are closed under operations of union, concatenation and Kleene star.

Regular Languages

Have basic operations to build regular languages.
Important: Any language generated by a finite sequence of such operations is regular.

Lemma 2.1.

Let $L_1, L_2, \ldots,$ be regular languages over alphabet Σ. Then the language $\cup_{i=1}^{\infty} L_i$ is not necessarily regular.

Some simple regular languages

Lemma 2.2.

If $\mathbf{w}$ *is a string then* $\mathbf{L} = \{\mathbf{w}\}$ *is regular.*

Example: $\{\mathbf{aba}\}$ or $\{\mathbf{abbabbab}\}$. Why?

Lemma 2.3.

Every finite language $\mathbf{L}$ *is regular.*

Examples: $\mathbf{L} = \{\mathbf{a}, \mathbf{abaab}, \mathbf{aba}\}$. $\mathbf{L} = \{\mathbf{w} \mid |\mathbf{w}| \leq \mathbf{100}\}$. Why?

Some simple regular languages

Lemma 2.2.

If $\mathbf{w}$ *is a string then* $\mathbf{L} = \{\mathbf{w}\}$ *is regular.*

Example: $\{\mathbf{aba}\}$ or $\{\mathbf{abbabbab}\}$. Why?

Lemma 2.3.

Every finite language $\mathbf{L}$ *is regular.*

Examples: $\mathbf{L} = \{\mathbf{a}, \mathbf{abaab}, \mathbf{aba}\}$. $\mathbf{L} = \{\mathbf{w} \mid |\mathbf{w}| \leq \mathbf{100}\}$. Why?

More Examples

- $\{\mathbf{w} \mid \mathbf{w}$ is a keyword in Python program$\}$
- $\{\mathbf{w} \mid \mathbf{w}$ is a valid date of the form mm/dd/yy$\}$
- $\{\mathbf{w} \mid \mathbf{w}$ describes a valid Roman numeral$\}$
  $\{\mathbf{I}, \mathbf{II}, \mathbf{III}, \mathbf{IV}, \mathbf{V}, \mathbf{VI}, \mathbf{VII}, \mathbf{VIII}, \mathbf{IX}, \mathbf{X}, \mathbf{XI}, \ldots\}$.
- $\{\mathbf{w} \mid \mathbf{w}$ contains "CS374" as a substring$\}$.

Review questions

1. $L_1 \subseteq \{0,1\}^*$ be a finite language. L_1 is a set with finite number of strings. T/F?
2. $L_2 = \{0^i \mid i = 0, 1, \ldots, \infty\}$. The language L_2 is regular. T/F?
3. $L_3 = \{0^{2i} \mid i = 0, 1, \ldots, \infty\}$. The language L_3 is regular. T/F?
4. $L_4 = \{0^{17i} \mid i = 0, 1, \ldots, \infty\}$. The language L_4 is regular. T/F?
5. $L_5 = \{0^i \mid i \text{ is } \textbf{not} \text{ divisible by } 17\}$. L_5 is regular. T/F?
6. $L_6 = \{0^i \mid i \text{ is divisible by } 2, 3, \text{ or } 5\}$. L_6 is regular. T/F?
7. $L_7 = \{0^i \mid i \text{ is divisible by } 2, 3, \text{ and } 5\}$. L_7 is regular. T/F?
8. $L_8 = \{0^i \mid i \text{ is divisible by } 2, 3, \text{ but not } 5\}$. L_8 is regular. T/F?
9. $L_9 = \{0^i 1^i \mid i \text{ is divisible by } 2, 3, \text{ but not } 5\}$. L_9 is regular. T/F?
10. $L_{10} = \{w \in \{0,1\}^* \mid w \text{ has at most 374 } 1\text{s}\}$. L_{10} is regular. T/F?

2.1.1
Regular Languages: Review questions

Review questions

1. $\mathbf{L_1} \subseteq \{0,1\}^*$ be a finite language. $\mathbf{L_1}$ is a set with finite number of strings. T/F?
2. $\mathbf{L_2} = \left\{ 0^i \mid i = 0, 1, \ldots, \infty \right\}$. The language $\mathbf{L_2}$ is regular. T/F?
3. $\mathbf{L_3} = \left\{ 0^{2i} \mid i = 0, 1, \ldots, \infty \right\}$. The language $\mathbf{L_3}$ is regular. T/F?
4. $\mathbf{L_4} = \left\{ 0^{17i} \mid i = 0, 1, \ldots, \infty \right\}$. The language $\mathbf{L_4}$ is regular. T/F?
5. $\mathbf{L_5} = \left\{ 0^i \mid i \text{ is } \textbf{not} \text{ divisible by } 17 \right\}$. $\mathbf{L_5}$ is regular. T/F?
6. $\mathbf{L_6} = \left\{ 0^i \mid i \text{ is divisible by } 2, 3, \text{ or } 5 \right\}$. $\mathbf{L_6}$ is regular. T/F?
7. $\mathbf{L_7} = \left\{ 0^i \mid i \text{ is divisible by } 2, 3, \text{ and } 5 \right\}$. $\mathbf{L_7}$ is regular. T/F?
8. $\mathbf{L_8} = \left\{ 0^i \mid i \text{ is divisible by } 2, 3, \text{ but not } 5 \right\}$. $\mathbf{L_8}$ is regular. T/F?
9. $\mathbf{L_9} = \left\{ 0^i 1^i \mid i \text{ is divisible by } 2, 3, \text{ but not } 5 \right\}$. $\mathbf{L_9}$ is regular. T/F?
10. $\mathbf{L_{10}} = \{ w \in \{0,1\}^* \mid w \text{ has at most 374 } 1\text{s} \}$. $\mathbf{L_{10}}$ is regular. T/F?

Review questions

1. $\mathbf{L_1} \subseteq \{0,1\}^*$ be a finite language. $\mathbf{L_1}$ is a set with finite number of strings. T/F?
2. $\mathbf{L_2} = \{0^i \mid i = 0, 1, \dots, \infty\}$. The language $\mathbf{L_2}$ is regular. T/F?
3. $\mathbf{L_3} = \{0^{2i} \mid i = 0, 1, \dots, \infty\}$. The language $\mathbf{L_3}$ is regular. T/F?
4. $\mathbf{L_4} = \{0^{17i} \mid i = 0, 1, \dots, \infty\}$. The language $\mathbf{L_4}$ is regular. T/F?
5. $\mathbf{L_5} = \{0^i \mid i$ is **not** divisible by $17\}$. $\mathbf{L_5}$ is regular. T/F?
6. $\mathbf{L_6} = \{0^i \mid i$ is divisible by $2, 3,$ or $5\}$. $\mathbf{L_6}$ is regular. T/F?
7. $\mathbf{L_7} = \{0^i \mid i$ is divisible by $2, 3,$ and $5\}$. $\mathbf{L_7}$ is regular. T/F?
8. $\mathbf{L_8} = \{0^i \mid i$ is divisible by $2, 3,$ but not $5\}$. $\mathbf{L_8}$ is regular. T/F?
9. $\mathbf{L_9} = \{0^i 1^i \mid i$ is divisible by $2, 3,$ but not $5\}$. $\mathbf{L_9}$ is regular. T/F?
10. $\mathbf{L_{10}} = \{w \in \{0,1\}^* \mid w$ has at most 374 1s$\}$. $\mathbf{L_{10}}$ is regular. T/F?

Review questions

1. $\mathbf{L_1} \subseteq \{0, 1\}^*$ be a finite language. $\mathbf{L_1}$ is a set with finite number of strings. T/F?
2. $\mathbf{L_2} = \{0^i \mid i = 0, 1, \ldots, \infty\}$. The language $\mathbf{L_2}$ is regular. T/F?
3. $\mathbf{L_3} = \{0^{2i} \mid i = 0, 1, \ldots, \infty\}$. The language $\mathbf{L_3}$ is regular. T/F?
4. $\mathbf{L_4} = \{0^{17i} \mid i = 0, 1, \ldots, \infty\}$. The language $\mathbf{L_4}$ is regular. T/F?
5. $\mathbf{L_5} = \{0^i \mid i$ is **not** divisible by $17\}$. $\mathbf{L_5}$ is regular. T/F?
6. $\mathbf{L_6} = \{0^i \mid i$ is divisible by $2, 3,$ or $5\}$. $\mathbf{L_6}$ is regular. T/F?
7. $\mathbf{L_7} = \{0^i \mid i$ is divisible by $2, 3,$ and $5\}$. $\mathbf{L_7}$ is regular. T/F?
8. $\mathbf{L_8} = \{0^i \mid i$ is divisible by $2, 3,$ but not $5\}$. $\mathbf{L_8}$ is regular. T/F?
9. $\mathbf{L_9} = \{0^i1^i \mid i$ is divisible by $2, 3,$ but not $5\}$. $\mathbf{L_9}$ is regular. T/F?
10. $\mathbf{L_{10}} = \{w \in \{0, 1\}^* \mid w$ has at most 374 1s$\}$. $\mathbf{L_{10}}$ is regular. T/F?

Review questions

1. $\mathbf{L_1} \subseteq \{\mathbf{0}, \mathbf{1}\}^*$ be a finite language. $\mathbf{L_1}$ is a set with finite number of strings. T/F?
2. $\mathbf{L_2} = \left\{\mathbf{0}^{\mathbf{i}} \mid \mathbf{i} = \mathbf{0}, \mathbf{1}, \ldots, \infty\right\}$. The language $\mathbf{L_2}$ is regular. T/F?
3. $\mathbf{L_3} = \left\{\mathbf{0}^{\mathbf{2i}} \mid \mathbf{i} = \mathbf{0}, \mathbf{1}, \ldots, \infty\right\}$. The language $\mathbf{L_3}$ is regular. T/F?
4. $\mathbf{L_4} = \left\{\mathbf{0}^{\mathbf{17i}} \mid \mathbf{i} = \mathbf{0}, \mathbf{1}, \ldots, \infty\right\}$. The language $\mathbf{L_4}$ is regular. T/F?
5. $\mathbf{L_5} = \left\{\mathbf{0}^{\mathbf{i}} \mid \mathbf{i} \text{ is } \textbf{not} \text{ divisible by } \mathbf{17}\right\}$. $\mathbf{L_5}$ is regular. T/F?
6. $\mathbf{L_6} = \left\{\mathbf{0}^{\mathbf{i}} \mid \mathbf{i} \text{ is divisible by } \mathbf{2}, \mathbf{3}, \text{or } \mathbf{5}\right\}$. $\mathbf{L_6}$ is regular. T/F?
7. $\mathbf{L_7} = \left\{\mathbf{0}^{\mathbf{i}} \mid \mathbf{i} \text{ is divisible by } \mathbf{2}, \mathbf{3}, \text{and } \mathbf{5}\right\}$. $\mathbf{L_7}$ is regular. T/F?
8. $\mathbf{L_8} = \left\{\mathbf{0}^{\mathbf{i}} \mid \mathbf{i} \text{ is divisible by } \mathbf{2}, \mathbf{3}, \text{but not } \mathbf{5}\right\}$. $\mathbf{L_8}$ is regular. T/F?
9. $\mathbf{L_9} = \left\{\mathbf{0}^{\mathbf{i}}\mathbf{1}^{\mathbf{i}} \mid \mathbf{i} \text{ is divisible by } \mathbf{2}, \mathbf{3}, \text{but not } \mathbf{5}\right\}$. $\mathbf{L_9}$ is regular. T/F?
10. $\mathbf{L_{10}} = \{\mathbf{w} \in \{\mathbf{0}, \mathbf{1}\}^* \mid \mathbf{w} \text{ has at most 374 } \mathbf{1}\text{s}\}$. $\mathbf{L_{10}}$ is regular. T/F?

Review questions

1. $\mathbf{L_1} \subseteq \{0,1\}^*$ be a finite language. $\mathbf{L_1}$ is a set with finite number of strings. T/F?
2. $\mathbf{L_2} = \left\{0^i \mid i = 0, 1, \ldots, \infty\right\}$. The language $\mathbf{L_2}$ is regular. T/F?
3. $\mathbf{L_3} = \left\{0^{2i} \mid i = 0, 1, \ldots, \infty\right\}$. The language $\mathbf{L_3}$ is regular. T/F?
4. $\mathbf{L_4} = \left\{0^{17i} \mid i = 0, 1, \ldots, \infty\right\}$. The language $\mathbf{L_4}$ is regular. T/F?
5. $\mathbf{L_5} = \left\{0^i \mid i \text{ is } \textbf{not} \text{ divisible by } 17\right\}$. $\mathbf{L_5}$ is regular. T/F?
6. $\mathbf{L_6} = \left\{0^i \mid i \text{ is divisible by } 2, 3, \text{ or } 5\right\}$. $\mathbf{L_6}$ is regular. T/F?
7. $\mathbf{L_7} = \left\{0^i \mid i \text{ is divisible by } 2, 3, \text{ and } 5\right\}$. $\mathbf{L_7}$ is regular. T/F?
8. $\mathbf{L_8} = \left\{0^i \mid i \text{ is divisible by } 2, 3, \text{ but not } 5\right\}$. $\mathbf{L_8}$ is regular. T/F?
9. $\mathbf{L_9} = \left\{0^i 1^i \mid i \text{ is divisible by } 2, 3, \text{ but not } 5\right\}$. $\mathbf{L_9}$ is regular. T/F?
10. $\mathbf{L_{10}} = \{w \in \{0,1\}^* \mid w \text{ has at most 374 1s}\}$. $\mathbf{L_{10}}$ is regular. T/F?

Review questions

1. $\mathbf{L_1} \subseteq \{0,1\}^*$ be a finite language. $\mathbf{L_1}$ is a set with finite number of strings. T/F?
2. $\mathbf{L_2} = \{0^i \mid i = 0, 1, \ldots, \infty\}$. The language $\mathbf{L_2}$ is regular. T/F?
3. $\mathbf{L_3} = \{0^{2i} \mid i = 0, 1, \ldots, \infty\}$. The language $\mathbf{L_3}$ is regular. T/F?
4. $\mathbf{L_4} = \{0^{17i} \mid i = 0, 1, \ldots, \infty\}$. The language $\mathbf{L_4}$ is regular. T/F?
5. $\mathbf{L_5} = \{0^i \mid i$ is **not** divisible by $17\}$. $\mathbf{L_5}$ is regular. T/F?
6. $\mathbf{L_6} = \{0^i \mid i$ is divisible by $2, 3,$ or $5\}$. $\mathbf{L_6}$ is regular. T/F?
7. $\mathbf{L_7} = \{0^i \mid i$ is divisible by $2, 3,$ and $5\}$. $\mathbf{L_7}$ is regular. T/F?
8. $\mathbf{L_8} = \{0^i \mid i$ is divisible by $2, 3,$ but not $5\}$. $\mathbf{L_8}$ is regular. T/F?
9. $\mathbf{L_9} = \{0^i1^i \mid i$ is divisible by $2, 3,$ but not $5\}$. $\mathbf{L_9}$ is regular. T/F?
10. $\mathbf{L_{10}} = \{w \in \{0,1\}^* \mid w$ has at most 374 1s$\}$. $\mathbf{L_{10}}$ is regular. T/F?

Review questions

1. $\mathbf{L_1} \subseteq \{0,1\}^*$ be a finite language. $\mathbf{L_1}$ is a set with finite number of strings. T/F?
2. $\mathbf{L_2} = \left\{0^i \mid i = 0, 1, \ldots, \infty\right\}$. The language $\mathbf{L_2}$ is regular. T/F?
3. $\mathbf{L_3} = \left\{0^{2i} \mid i = 0, 1, \ldots, \infty\right\}$. The language $\mathbf{L_3}$ is regular. T/F?
4. $\mathbf{L_4} = \left\{0^{17i} \mid i = 0, 1, \ldots, \infty\right\}$. The language $\mathbf{L_4}$ is regular. T/F?
5. $\mathbf{L_5} = \left\{0^i \mid i \text{ is } \textbf{not} \text{ divisible by } 17\right\}$. $\mathbf{L_5}$ is regular. T/F?
6. $\mathbf{L_6} = \left\{0^i \mid i \text{ is divisible by } 2, 3, \text{ or } 5\right\}$. $\mathbf{L_6}$ is regular. T/F?
7. $\mathbf{L_7} = \left\{0^i \mid i \text{ is divisible by } 2, 3, \text{ and } 5\right\}$. $\mathbf{L_7}$ is regular. T/F?
8. $\mathbf{L_8} = \left\{0^i \mid i \text{ is divisible by } 2, 3, \text{ but not } 5\right\}$. $\mathbf{L_8}$ is regular. T/F?
9. $\mathbf{L_9} = \left\{0^i1^i \mid i \text{ is divisible by } 2, 3, \text{ but not } 5\right\}$. $\mathbf{L_9}$ is regular. T/F?
10. $\mathbf{L_{10}} = \{w \in \{0,1\}^* \mid w \text{ has at most 374 } 1\text{s}\}$. $\mathbf{L_{10}}$ is regular. T/F?

Review questions

1. $\mathbf{L_1} \subseteq \{0,1\}^*$ be a finite language. $\mathbf{L_1}$ is a set with finite number of strings. T/F?
2. $\mathbf{L_2} = \{0^i \mid i = 0, 1, \ldots, \infty\}$. The language $\mathbf{L_2}$ is regular. T/F?
3. $\mathbf{L_3} = \{0^{2i} \mid i = 0, 1, \ldots, \infty\}$. The language $\mathbf{L_3}$ is regular. T/F?
4. $\mathbf{L_4} = \{0^{17i} \mid i = 0, 1, \ldots, \infty\}$. The language $\mathbf{L_4}$ is regular. T/F?
5. $\mathbf{L_5} = \{0^i \mid i$ is **not** divisible by $17\}$. $\mathbf{L_5}$ is regular. T/F?
6. $\mathbf{L_6} = \{0^i \mid i$ is divisible by $2, 3,$ or $5\}$. $\mathbf{L_6}$ is regular. T/F?
7. $\mathbf{L_7} = \{0^i \mid i$ is divisible by $2, 3,$ and $5\}$. $\mathbf{L_7}$ is regular. T/F?
8. $\mathbf{L_8} = \{0^i \mid i$ is divisible by $2, 3,$ but not $5\}$. $\mathbf{L_8}$ is regular. T/F?
9. $\mathbf{L_9} = \{0^i 1^i \mid i$ is divisible by $2, 3,$ but not $5\}$. $\mathbf{L_9}$ is regular. T/F?
10. $\mathbf{L_{10}} = \{w \in \{0,1\}^* \mid w$ has at most 374 1s$\}$. $\mathbf{L_{10}}$ is regular. T/F?

Review questions

1. $\mathbf{L_1} \subseteq \{0,1\}^*$ be a finite language. $\mathbf{L_1}$ is a set with finite number of strings. T/F?
2. $\mathbf{L_2} = \{0^i \mid i = 0, 1, \ldots, \infty\}$. The language $\mathbf{L_2}$ is regular. T/F?
3. $\mathbf{L_3} = \{0^{2i} \mid i = 0, 1, \ldots, \infty\}$. The language $\mathbf{L_3}$ is regular. T/F?
4. $\mathbf{L_4} = \{0^{17i} \mid i = 0, 1, \ldots, \infty\}$. The language $\mathbf{L_4}$ is regular. T/F?
5. $\mathbf{L_5} = \{0^i \mid i$ is **not** divisible by $17\}$. $\mathbf{L_5}$ is regular. T/F?
6. $\mathbf{L_6} = \{0^i \mid i$ is divisible by $2, 3,$ or $5\}$. $\mathbf{L_6}$ is regular. T/F?
7. $\mathbf{L_7} = \{0^i \mid i$ is divisible by $2, 3,$ and $5\}$. $\mathbf{L_7}$ is regular. T/F?
8. $\mathbf{L_8} = \{0^i \mid i$ is divisible by $2, 3,$ but not $5\}$. $\mathbf{L_8}$ is regular. T/F?
9. $\mathbf{L_9} = \{0^i 1^i \mid i$ is divisible by $2, 3,$ but not $5\}$. $\mathbf{L_9}$ is regular. T/F?
10. $\mathbf{L_{10}} = \{w \in \{0,1\}^* \mid w$ has at most 374 1s$\}$. $\mathbf{L_{10}}$ is regular. T/F?

Review questions

1. $\mathbf{L_1} \subseteq \{0,1\}^*$ be a finite language. $\mathbf{L_1}$ is a set with finite number of strings. T/F?
2. $\mathbf{L_2} = \{0^i \mid i = 0, 1, \ldots, \infty\}$. The language $\mathbf{L_2}$ is regular. T/F?
3. $\mathbf{L_3} = \{0^{2i} \mid i = 0, 1, \ldots, \infty\}$. The language $\mathbf{L_3}$ is regular. T/F?
4. $\mathbf{L_4} = \{0^{17i} \mid i = 0, 1, \ldots, \infty\}$. The language $\mathbf{L_4}$ is regular. T/F?
5. $\mathbf{L_5} = \{0^i \mid i$ is **not** divisible by $17\}$. $\mathbf{L_5}$ is regular. T/F?
6. $\mathbf{L_6} = \{0^i \mid i$ is divisible by $2, 3,$ or $5\}$. $\mathbf{L_6}$ is regular. T/F?
7. $\mathbf{L_7} = \{0^i \mid i$ is divisible by $2, 3,$ and $5\}$. $\mathbf{L_7}$ is regular. T/F?
8. $\mathbf{L_8} = \{0^i \mid i$ is divisible by $2, 3,$ but not $5\}$. $\mathbf{L_8}$ is regular. T/F?
9. $\mathbf{L_9} = \{0^i 1^i \mid i$ is divisible by $2, 3,$ but not $5\}$. $\mathbf{L_9}$ is regular. T/F?
10. $\mathbf{L_{10}} = \{w \in \{0,1\}^* \mid w$ has at most 374 1s$\}$. $\mathbf{L_{10}}$ is regular. T/F?

2.2
Regular Expressions

Regular Expressions

A way to denote regular languages

- simple patterns to describe related strings
- useful in
  - text search (editors, Unix/grep, emacs)
  - compilers: lexical analysis
  - compact way to represent interesting/useful languages
  - dates back to 50's: Stephen Kleene
    who has a star names after him.

Inductive Definition

A regular expression $\mathbf{r}$ over an alphabet $\mathbf{\Sigma}$ is one of the following:

Base cases:

- $\emptyset$ denotes the language $\emptyset$
- ϵ denotes the language $\{\epsilon\}$.
- $\mathbf{a}$ denote the language $\{\mathbf{a}\}$.

Inductive cases: If $\mathbf{r}_1$ and $\mathbf{r}_2$ are regular expressions denoting languages $\mathbf{R}_1$ and $\mathbf{R}_2$ respectively then,

- $(\mathbf{r}_1 + \mathbf{r}_2)$ denotes the language $\mathbf{R}_1 \cup \mathbf{R}_2$
- $(\mathbf{r}_1 \bullet \mathbf{r}_2) = \mathbf{r}_1 \bullet \mathbf{r}_2 = (\mathbf{r}_1\mathbf{r}_2)$ denotes the language $\mathbf{R}_1\mathbf{R}_2$
- $(\mathbf{r}_1)^*$ denotes the language $\mathbf{R}_1^*$

Inductive Definition

A regular expression $\mathbf{r}$ over an alphabet $\boldsymbol{\Sigma}$ is one of the following:

Base cases:

- $\emptyset$ denotes the language $\emptyset$
- ϵ denotes the language $\{\epsilon\}$.
- $\mathbf{a}$ denote the language $\{\mathbf{a}\}$.

Inductive cases: If $\mathbf{r_1}$ and $\mathbf{r_2}$ are regular expressions denoting languages $\mathbf{R_1}$ and $\mathbf{R_2}$ respectively then,

- $(\mathbf{r_1} + \mathbf{r_2})$ denotes the language $\mathbf{R_1} \cup \mathbf{R_2}$
- $(\mathbf{r_1} \bullet \mathbf{r_2}) = \mathbf{r_1} \bullet \mathbf{r_2} = (\mathbf{r_1}\mathbf{r_2})$ denotes the language $\mathbf{R_1}\mathbf{R_2}$
- $(\mathbf{r_1})^*$ denotes the language $\mathbf{R_1^*}$

Regular Languages vs Regular Expressions

Regular Languages	**Regular Expressions**
$\emptyset$ regular	$\emptyset$ denotes $\emptyset$
$\{\epsilon\}$ regular	ϵ denotes $\{\epsilon\}$
$\{\mathbf{a}\}$ regular for $\mathbf{a} \in \mathbf{\Sigma}$	$\mathbf{a}$ denote $\{\mathbf{a}\}$
$\mathbf{R_1} \cup \mathbf{R_2}$ regular if both are	$\mathbf{r_1} + \mathbf{r_2}$ denotes $\mathbf{R_1} \cup \mathbf{R_2}$
$\mathbf{R_1 R_2}$ regular if both are	$\mathbf{r_1} \bullet \mathbf{r_2}$ denotes $\mathbf{R_1 R_2}$
$\mathbf{R}^*$ is regular if $\mathbf{R}$ is	$\mathbf{r}^*$ denote $\mathbf{R}^*$

Regular expressions denote regular languages — they explicitly show the operations that were used to form the language

Notation and Parenthesis

- For a regular expression $\mathbf{r}$, $\mathbf{L(r)}$ is the language denoted by $\mathbf{r}$. Multiple regular expressions can denote the same language!
  Example: $\mathbf{(0+1)}$ and $\mathbf{(1+0)}$ denote same language $\mathbf{\{0,1\}}$
- Two regular expressions $\mathbf{r_1}$ and $\mathbf{r_2}$ are equivalent if $\mathbf{L(r_1) = L(r_2)}$.
- Omit parenthesis by adopting precedence order: $*$, concatenate, $+$.
  Example: $\mathbf{r^*s + t = ((r^*)s) + t}$
- Omit parenthesis by associativity of each of these operations.
  Example: $\mathbf{rst = (rs)t = r(st)}$, $\mathbf{r + s + t = r + (s + t) = (r + s) + t}$.
- Superscript $+$. For convenience, define $\mathbf{r^+ = rr^*}$. Hence if $\mathbf{L(r) = R}$ then $\mathbf{L(r^+) = R^+}$.
- Other notation: $\mathbf{r + s}$, $\mathbf{r \cup s}$, $\mathbf{r|s}$ all denote union. $\mathbf{rs}$ is sometimes written as $\mathbf{r \bullet s}$.

Notation and Parenthesis

- For a regular expression $\mathbf{r}$, $\mathbf{L(r)}$ is the language denoted by $\mathbf{r}$. Multiple regular expressions can denote the same language!
  Example: $\mathbf{(0+1)}$ and $\mathbf{(1+0)}$ denote same language $\mathbf{\{0,1\}}$
- Two regular expressions $\mathbf{r_1}$ and $\mathbf{r_2}$ are equivalent if $\mathbf{L(r_1) = L(r_2)}$.
- Omit parenthesis by adopting precedence order: $*$, concatenate, $+$.
  Example: $\mathbf{r^*s + t = ((r^*)s) + t}$
- Omit parenthesis by associativity of each of these operations.
  Example: $\mathbf{rst = (rs)t = r(st)}$, $\mathbf{r + s + t = r + (s + t) = (r + s) + t}$.
- Superscript $+$. For convenience, define $\mathbf{r^+ = rr^*}$. Hence if $\mathbf{L(r) = R}$ then $\mathbf{L(r^+) = R^+}$.
- Other notation: $\mathbf{r+s}$, $\mathbf{r \cup s}$, $\mathbf{r|s}$ all denote union. $\mathbf{rs}$ is sometimes written as $\mathbf{r \bullet s}$.

Notation and Parenthesis

- For a regular expression $\mathbf{r}$, $\mathbf{L(r)}$ is the language denoted by $\mathbf{r}$. Multiple regular expressions can denote the same language!
  Example: $\mathbf{(0+1)}$ and $\mathbf{(1+0)}$ denote same language $\mathbf{\{0,1\}}$
- Two regular expressions $\mathbf{r_1}$ and $\mathbf{r_2}$ are equivalent if $\mathbf{L(r_1) = L(r_2)}$.
- Omit parenthesis by adopting precedence order: $*$, concatenate, $+$.
  Example: $\mathbf{r^*s + t = ((r^*)s) + t}$
- Omit parenthesis by associativity of each of these operations.
  Example: $\mathbf{rst = (rs)t = r(st)}$, $\mathbf{r + s + t = r + (s + t) = (r + s) + t}$.
- Superscript $+$. For convenience, define $\mathbf{r^+ = rr^*}$. Hence if $\mathbf{L(r) = R}$ then $\mathbf{L(r^+) = R^+}$.
- Other notation: $\mathbf{r + s}$, $\mathbf{r \cup s}$, $\mathbf{r|s}$ all denote union. $\mathbf{rs}$ is sometimes written as $\mathbf{r \bullet s}$.

Notation and Parenthesis

- For a regular expression $\mathbf{r}$, $\mathbf{L(r)}$ is the language denoted by $\mathbf{r}$. Multiple regular expressions can denote the same language!
  Example: $\mathbf{(0+1)}$ and $\mathbf{(1+0)}$ denote same language $\mathbf{\{0,1\}}$
- Two regular expressions $\mathbf{r_1}$ and $\mathbf{r_2}$ are equivalent if $\mathbf{L(r_1) = L(r_2)}$.
- Omit parenthesis by adopting precedence order: $*$, concatenate, $+$.
  Example: $\mathbf{r^*s + t = ((r^*)s) + t}$
- Omit parenthesis by associativity of each of these operations.
  Example: $\mathbf{rst = (rs)t = r(st)}$, $\mathbf{r+s+t = r+(s+t) = (r+s)+t}$.
- Superscript $+$. For convenience, define $\mathbf{r^+ = rr^*}$. Hence if $\mathbf{L(r) = R}$ then $\mathbf{L(r^+) = R^+}$.
- Other notation: $\mathbf{r+s}$, $\mathbf{r \cup s}$, $\mathbf{r|s}$ all denote union. $\mathbf{rs}$ is sometimes written as $\mathbf{r \bullet s}$.

Notation and Parenthesis

- For a regular expression $\mathbf{r}$, $\mathbf{L(r)}$ is the language denoted by $\mathbf{r}$. Multiple regular expressions can denote the same language!
  Example: $(0+1)$ and $(1+0)$ denote same language $\{0,1\}$
- Two regular expressions $\mathbf{r_1}$ and $\mathbf{r_2}$ are equivalent if $\mathbf{L(r_1) = L(r_2)}$.
- Omit parenthesis by adopting precedence order: $*$, concatenate, $+$.
  Example: $\mathbf{r^*s + t = ((r^*)s) + t}$
- Omit parenthesis by associativity of each of these operations.
  Example: $\mathbf{rst = (rs)t = r(st)}$, $\mathbf{r + s + t = r + (s + t) = (r + s) + t}$.
- Superscript $+$. For convenience, define $\mathbf{r^+ = rr^*}$. Hence if $\mathbf{L(r) = R}$ then $\mathbf{L(r^+) = R^+}$.
- Other notation: $\mathbf{r + s}$, $\mathbf{r \cup s}$, $\mathbf{r|s}$ all denote union. $\mathbf{rs}$ is sometimes written as $\mathbf{r \bullet s}$.

Notation and Parenthesis

- For a regular expression $\mathbf{r}$, $\mathbf{L(r)}$ is the language denoted by $\mathbf{r}$. Multiple regular expressions can denote the same language!
  Example: $(0 + 1)$ and $(1 + 0)$ denote same language $\{0, 1\}$
- Two regular expressions $\mathbf{r_1}$ and $\mathbf{r_2}$ are equivalent if $\mathbf{L(r_1) = L(r_2)}$.
- Omit parenthesis by adopting precedence order: $*$, concatenate, $+$.
  Example: $r^*s + t = ((r^*)s) + t$
- Omit parenthesis by associativity of each of these operations.
  Example: $rst = (rs)t = r(st)$, $r + s + t = r + (s + t) = (r + s) + t$.
- Superscript $+$. For convenience, define $r^+ = rr^*$. Hence if $L(r) = R$ then $L(r^+) = R^+$.
- Other notation: $r + s$, $r \cup s$, $r|s$ all denote union. rs is sometimes written as $r \bullet s$.

Skills

- Given a language **L** "in mind" (say an English description) we would like to write a regular expression for **L** (if possible)
- Given a regular expression **r** we would like to "understand" **L(r)** (say by giving an English description)

Skills

- Given a language **L** "in mind" (say an English description) we would like to write a regular expression for **L** (if possible)
- Given a regular expression **r** we would like to "understand" **L(r)** (say by giving an English description)

2.2.1
Some examples of regular expressions

Understanding regular expressions

- $(0+1)^*$: set of all strings over $\{0, 1\}$
- $(0+1)^*001(0+1)^*$: strings with 001 as substring
- $0^* + (0^*10^*10^*10^*)^*$: strings with number of 1's divisible by 3
- $\emptyset 0$: $\{\}$
- $(\epsilon+1)(01)^*(\epsilon+0)$: alternating 0s and 1s. Alternatively, no two consecutive 0s and no two consecutive 1s
- $(\epsilon+0)(1+10)^*$: strings without two consecutive 0s.

Understanding regular expressions

- $(0+1)^*$: set of all strings over $\{0,1\}$
- $(0+1)^*001(0+1)^*$: strings with 001 as substring
- $0^* + (0^*10^*10^*10^*)^*$: strings with number of 1's divisible by 3
- $\emptyset 0$: $\{\}$
- $(\epsilon+1)(01)^*(\epsilon+0)$: alternating 0s and 1s. Alternatively, no two consecutive 0s and no two consecutive 1s
- $(\epsilon+0)(1+10)^*$: strings without two consecutive 0s.

Understanding regular expressions

- $(0+1)^*$: set of all strings over $\{0, 1\}$
- $(0+1)^*001(0+1)^*$: strings with 001 as substring
- $0^* + (0^*10^*10^*10^*)^*$: strings with number of 1's divisible by 3
- $\emptyset 0$: $\{\}$
- $(\epsilon + 1)(01)^*(\epsilon + 0)$: alternating 0s and 1s. Alternatively, no two consecutive 0s and no two consecutive 1s
- $(\epsilon + 0)(1 + 10)^*$: strings without two consecutive 0s.

Understanding regular expressions

- $(0+1)^*$: set of all strings over $\{0,1\}$
- $(0+1)^*001(0+1)^*$: strings with 001 as substring
- $0^* + (0^*10^*10^*10^*)^*$: strings with number of 1's divisible by 3
- $\emptyset 0$: $\{\}$
- $(\epsilon+1)(01)^*(\epsilon+0)$: alternating 0s and 1s. Alternatively, no two consecutive 0s and no two consecutive 1s
- $(\epsilon+0)(1+10)^*$: strings without two consecutive 0s.

Understanding regular expressions

- $(0+1)^*$: set of all strings over $\{0,1\}$
- $(0+1)^*001(0+1)^*$: strings with 001 as substring
- $0^* + (0^*10^*10^*10^*)^*$: strings with number of 1's divisible by 3
- $\emptyset 0$: $\{\}$
- $(\epsilon+1)(01)^*(\epsilon+0)$: alternating 0s and 1s. Alternatively, no two consecutive 0s and no two consecutive 1s
- $(\epsilon+0)(1+10)^*$: strings without two consecutive 0s.

Understanding regular expressions

- $(\mathbf{0}+\mathbf{1})^*$: set of all strings over $\{\mathbf{0},\mathbf{1}\}$
- $(\mathbf{0}+\mathbf{1})^*\mathbf{001}(\mathbf{0}+\mathbf{1})^*$: strings with $\mathbf{001}$ as substring
- $\mathbf{0}^* + (\mathbf{0}^*\mathbf{1}\mathbf{0}^*\mathbf{1}\mathbf{0}^*\mathbf{1}\mathbf{0}^*)^*$: strings with number of $\mathbf{1}$'s divisible by $\mathbf{3}$
- $\emptyset\mathbf{0}$: $\{\}$
- $(\epsilon+\mathbf{1})(\mathbf{01})^*(\epsilon+\mathbf{0})$: alternating $\mathbf{0}$s and $\mathbf{1}$s. Alternatively, no two consecutive 0s and no two consecutive 1s
- $(\epsilon+\mathbf{0})(\mathbf{1}+\mathbf{10})^*$: strings without two consecutive 0s.

Understanding regular expressions

- $(0+1)^*$: set of all strings over $\{0, 1\}$
- $(0+1)^*001(0+1)^*$: strings with 001 as substring
- $0^* + (0^*10^*10^*10^*)^*$: strings with number of 1's divisible by 3
- $\emptyset 0$: $\{\}$
- $(\epsilon + 1)(01)^*(\epsilon + 0)$: alternating 0s and 1s. Alternatively, no two consecutive 0s and no two consecutive 1s
- $(\epsilon + 0)(1 + 10)^*$: strings without two consecutive 0s.

Understanding regular expressions

- $(\mathbf{0}+\mathbf{1})^*$: set of all strings over $\{\mathbf{0},\mathbf{1}\}$
- $(\mathbf{0}+\mathbf{1})^*\mathbf{001}(\mathbf{0}+\mathbf{1})^*$: strings with $\mathbf{001}$ as substring
- $\mathbf{0}^*+(\mathbf{0}^*\mathbf{1}\mathbf{0}^*\mathbf{1}\mathbf{0}^*\mathbf{1}\mathbf{0}^*)^*$: strings with number of $\mathbf{1}$'s divisible by $\mathbf{3}$
- $\emptyset\mathbf{0}$: $\{\}$
- $(\epsilon+\mathbf{1})(\mathbf{01})^*(\epsilon+\mathbf{0})$: alternating $\mathbf{0}$s and $\mathbf{1}$s. Alternatively, no two consecutive 0s and no two consecutive 1s
- $(\epsilon+\mathbf{0})(\mathbf{1}+\mathbf{10})^*$: strings without two consecutive 0s.

Understanding regular expressions

- $(\mathbf{0}+\mathbf{1})^*$: set of all strings over $\{\mathbf{0},\mathbf{1}\}$
- $(\mathbf{0}+\mathbf{1})^*\mathbf{001}(\mathbf{0}+\mathbf{1})^*$: strings with $\mathbf{001}$ as substring
- $\mathbf{0}^* + (\mathbf{0}^*\mathbf{1}\mathbf{0}^*\mathbf{1}\mathbf{0}^*\mathbf{1}\mathbf{0}^*)^*$: strings with number of $\mathbf{1}$'s divisible by $\mathbf{3}$
- $\emptyset\mathbf{0}$: $\{\}$
- $(\epsilon+\mathbf{1})(\mathbf{01})^*(\epsilon+\mathbf{0})$: alternating $\mathbf{0}$s and $\mathbf{1}$s. Alternatively, no two consecutive 0s and no two consecutive 1s
- $(\epsilon+\mathbf{0})(\mathbf{1}+\mathbf{10})^*$: strings without two consecutive 0s.

Understanding regular expressions

- $(\mathbf{0}+\mathbf{1})^*$: set of all strings over $\{\mathbf{0},\mathbf{1}\}$
- $(\mathbf{0}+\mathbf{1})^*\mathbf{001}(\mathbf{0}+\mathbf{1})^*$: strings with $\mathbf{001}$ as substring
- $\mathbf{0}^* + (\mathbf{0}^*\mathbf{1}\mathbf{0}^*\mathbf{1}\mathbf{0}^*\mathbf{1}\mathbf{0}^*)^*$: strings with number of $\mathbf{1}$'s divisible by $\mathbf{3}$
- $\emptyset\mathbf{0}$: $\{\}$
- $(\epsilon+\mathbf{1})(\mathbf{01})^*(\epsilon+\mathbf{0})$: alternating $\mathbf{0}$s and $\mathbf{1}$s. Alternatively, no two consecutive 0s and no two consecutive 1s
- $(\epsilon+\mathbf{0})(\mathbf{1}+\mathbf{10})^*$: strings without two consecutive 0s.

Understanding regular expressions

- $(\mathbf{0} + \mathbf{1})^*$: set of all strings over $\{\mathbf{0}, \mathbf{1}\}$
- $(\mathbf{0} + \mathbf{1})^*\mathbf{001}(\mathbf{0} + \mathbf{1})^*$: strings with $\mathbf{001}$ as substring
- $\mathbf{0}^* + (\mathbf{0}^*\mathbf{1}\mathbf{0}^*\mathbf{1}\mathbf{0}^*\mathbf{1}\mathbf{0}^*)^*$: strings with number of $\mathbf{1}$'s divisible by $\mathbf{3}$
- $\emptyset\mathbf{0}$: $\{\}$
- $(\epsilon + \mathbf{1})(\mathbf{01})^*(\epsilon + \mathbf{0})$: alternating $\mathbf{0}$s and $\mathbf{1}$s. Alternatively, no two consecutive 0s and no two consecutive 1s
- $(\epsilon + \mathbf{0})(\mathbf{1} + \mathbf{10})^*$: strings without two consecutive 0s.

Creating regular expressions

- bitstrings with the pattern $\mathbf{001}$ or the pattern $\mathbf{100}$ occurring as a substring
  one answer: $(\mathbf{0}+\mathbf{1})^*\mathbf{001}(\mathbf{0}+\mathbf{1})^* + (\mathbf{0}+\mathbf{1})^*\mathbf{100}(\mathbf{0}+\mathbf{1})^*$
- bitstrings with an even number of $\mathbf{1}$'s
  one answer: $\mathbf{0}^* + (\mathbf{0}^*\mathbf{1}\mathbf{0}^*\mathbf{1}\mathbf{0}^*)^*$
- bitstrings with an odd number of $\mathbf{1}$'s
  one answer: $\mathbf{0}^*\mathbf{1}\mathbf{r}$ where $\mathbf{r}$ is solution to previous part
- bitstrings that do not contain $\mathbf{011}$ as a substring
- Hard: bitstrings with an odd number of 1s and an odd number of 0s.

Creating regular expressions

- bitstrings with the pattern $\mathbf{001}$ or the pattern $\mathbf{100}$ occurring as a substring
  one answer: $\mathbf{(0+1)^*001(0+1)^* + (0+1)^*100(0+1)^*}$
- bitstrings with an even number of $\mathbf{1}$'s
  one answer: $\mathbf{0^* + (0^*10^*10^*)^*}$
- bitstrings with an odd number of $\mathbf{1}$'s
  one answer: $\mathbf{0^*1r}$ where $\mathbf{r}$ is solution to previous part
- bitstrings that do not contain $\mathbf{011}$ as a substring
- Hard: bitstrings with an odd number of 1s and an odd number of 0s.

Creating regular expressions

- bitstrings with the pattern $\mathbf{001}$ or the pattern $\mathbf{100}$ occurring as a substring
  one answer: $\mathbf{(0+1)^*001(0+1)^* + (0+1)^*100(0+1)^*}$
- bitstrings with an even number of $\mathbf{1}$'s
  one answer: $\mathbf{0^* + (0^*10^*10^*)^*}$
- bitstrings with an odd number of $\mathbf{1}$'s
  one answer: $\mathbf{0^*1r}$ where $\mathbf{r}$ is solution to previous part
- bitstrings that do not contain $\mathbf{011}$ as a substring
- Hard: bitstrings with an odd number of 1s and an odd number of 0s.

Creating regular expressions

- bitstrings with the pattern $\mathbf{001}$ or the pattern $\mathbf{100}$ occurring as a substring
  one answer: $\mathbf{(0+1)^*001(0+1)^* + (0+1)^*100(0+1)^*}$
- bitstrings with an even number of $\mathbf{1}$'s
  one answer: $\mathbf{0^* + (0^*10^*10^*)^*}$
- bitstrings with an odd number of $\mathbf{1}$'s
  one answer: $\mathbf{0^*1r}$ where $\mathbf{r}$ is solution to previous part
- bitstrings that do <u>not</u> contain $\mathbf{011}$ as a substring
- Hard: bitstrings with an odd number of 1s <u>and</u> an odd number of 0s.

Creating regular expressions

- bitstrings with the pattern $\mathbf{001}$ or the pattern $\mathbf{100}$ occurring as a substring
  one answer: $\mathbf{(0+1)^*001(0+1)^* + (0+1)^*100(0+1)^*}$
- bitstrings with an even number of $\mathbf{1}$'s
  one answer: $\mathbf{0^* + (0^*10^*10^*)^*}$
- bitstrings with an odd number of $\mathbf{1}$'s
  one answer: $\mathbf{0^*1r}$ where $\mathbf{r}$ is solution to previous part
- bitstrings that do not contain $\mathbf{011}$ as a substring
- Hard: bitstrings with an odd number of 1s and an odd number of 0s.

Creating regular expressions

- bitstrings with the pattern **001** or the pattern **100** occurring as a substring
  one answer: $(0+1)^*001(0+1)^* + (0+1)^*100(0+1)^*$
- bitstrings with an even number of **1**'s
  one answer: $0^* + (0^*10^*10^*)^*$
- bitstrings with an odd number of **1**'s
  one answer: 0^*1r where **r** is solution to previous part
- bitstrings that do not contain **011** as a substring
- Hard: bitstrings with an odd number of 1s and an odd number of 0s.

Creating regular expressions

- bitstrings with the pattern $\mathbf{001}$ or the pattern $\mathbf{100}$ occurring as a substring
  one answer: $\mathbf{(0+1)^*001(0+1)^* + (0+1)^*100(0+1)^*}$
- bitstrings with an even number of $\mathbf{1}$'s
  one answer: $\mathbf{0^* + (0^*10^*10^*)^*}$
- bitstrings with an odd number of $\mathbf{1}$'s
  one answer: $\mathbf{0^*1r}$ where $\mathbf{r}$ is solution to previous part
- bitstrings that do not contain $\mathbf{011}$ as a substring
- Hard: bitstrings with an odd number of 1s and an odd number of 0s.

Creating regular expressions

- bitstrings with the pattern $\mathbf{001}$ or the pattern $\mathbf{100}$ occurring as a substring
  one answer: $\mathbf{(0+1)^*001(0+1)^* + (0+1)^*100(0+1)^*}$
- bitstrings with an even number of $\mathbf{1}$'s
  one answer: $\mathbf{0^* + (0^*10^*10^*)^*}$
- bitstrings with an odd number of $\mathbf{1}$'s
  one answer: $\mathbf{0^*1r}$ where $\mathbf{r}$ is solution to previous part
- bitstrings that do not contain $\mathbf{011}$ as a substring
- Hard: bitstrings with an odd number of 1s and an odd number of 0s.

Bit strings with odd number of **0**s and **1**s

The regular expression is

$$(\mathbf{00} + \mathbf{11})^*(\mathbf{01} + \mathbf{10})\Big(\mathbf{00} + \mathbf{11} + (\mathbf{01} + \mathbf{10})(\mathbf{00} + \mathbf{11})^*(\mathbf{01} + \mathbf{10})\Big)^*$$

(Solved using techniques to be presented in the following lectures...)

Regular expression identities

- $\mathbf{r^*r^* = r^*}$ meaning for any regular expression $\mathbf{r}$, $\mathbf{L(r^*r^*) = L(r^*)}$
- $\mathbf{(r^*)^* = r^*}$
- $\mathbf{rr^* = r^*r}$
- $\mathbf{(rs)^*r = r(sr)^*}$
- $\mathbf{(r + s)^* = (r^*s^*)^* = (r^* + s^*)^* = (r + s^*)^* = \ldots}$

Question: How does on prove an identity?
By induction. On what? Length of $\mathbf{r}$ since $\mathbf{r}$ is a string obtained from specific inductive rules.

Regular expression identities

- $\mathbf{r^*r^* = r^*}$ meaning for any regular expression $\mathbf{r}$, $\mathbf{L(r^*r^*) = L(r^*)}$
- $\mathbf{(r^*)^* = r^*}$
- $\mathbf{rr^* = r^*r}$
- $\mathbf{(rs)^*r = r(sr)^*}$
- $\mathbf{(r+s)^* = (r^*s^*)^* = (r^*+s^*)^* = (r+s^*)^* = \ldots}$

Question: How does on prove an identity?

By induction. On what? Length of $\mathbf{r}$ since $\mathbf{r}$ is a string obtained from specific inductive rules.

Regular expression identities

- $\mathbf{r^*r^* = r^*}$ meaning for any regular expression $\mathbf{r}$, $\mathbf{L(r^*r^*) = L(r^*)}$
- $\mathbf{(r^*)^* = r^*}$
- $\mathbf{rr^* = r^*r}$
- $\mathbf{(rs)^*r = r(sr)^*}$
- $\mathbf{(r+s)^* = (r^*s^*)^* = (r^*+s^*)^* = (r+s^*)^* = \ldots}$

Question: How does on prove an identity?
By induction. On what? Length of $\mathbf{r}$ since $\mathbf{r}$ is a string obtained from specific inductive rules.

Regular expression identities

- $\mathbf{r^*r^* = r^*}$ meaning for any regular expression $\mathbf{r}$, $\mathbf{L(r^*r^*) = L(r^*)}$
- $\mathbf{(r^*)^* = r^*}$
- $\mathbf{rr^* = r^*r}$
- $\mathbf{(rs)^*r = r(sr)^*}$
- $\mathbf{(r + s)^* = (r^*s^*)^* = (r^* + s^*)^* = (r + s^*)^* = \ldots}$

Question: How does on prove an identity?
By induction. On what? Length of $\mathbf{r}$ since $\mathbf{r}$ is a string obtained from specific inductive rules.

2.2.2
An example of a non-regular language

A non-regular language and other closure properties

Consider $\mathbf{L} = \{0^n 1^n \mid n \geq 0\} = \{\epsilon, 01, 0011, 000111, \ldots\}$.

Theorem 2.1.

$\mathbf{L} = \{0^n 1^n \mid n \geq 0\} = \{\epsilon, 01, 0011, 000111, \ldots\}$.
The language $\mathbf{L}$ *is* **not** *a regular language.*

How do we prove it?

Other questions:

- Suppose $\mathbf{R_1}$ is regular and $\mathbf{R_2}$ is regular. Is $\mathbf{R_1} \cap \mathbf{R_2}$ regular?
- Suppose $\mathbf{R_1}$ is regular is $\overline{\mathbf{R_1}}$ (complement of $\mathbf{R_1}$) regular?

A non-regular language and other closure properties

Consider $\mathbf{L} = \{0^n 1^n \mid n \geq 0\} = \{\epsilon, 01, 0011, 000111, \ldots\}$.

Theorem 2.1.

$\mathbf{L} = \{0^n 1^n \mid n \geq 0\} = \{\epsilon, 01, 0011, 000111, \ldots\}$.

The language **L** *is* **not** *a regular language.*

How do we prove it?

Other questions:

- Suppose $\mathbf{R_1}$ is regular and $\mathbf{R_2}$ is regular. Is $\mathbf{R_1} \cap \mathbf{R_2}$ regular?
- Suppose $\mathbf{R_1}$ is regular is $\overline{\mathbf{R_1}}$ (complement of $\mathbf{R_1}$) regular?

A non-regular language and other closure properties

Consider $\mathbf{L} = \{0^n 1^n \mid n \geq 0\} = \{\epsilon, 01, 0011, 000111, \ldots\}$.

Theorem 2.1.

$\mathbf{L} = \{0^n 1^n \mid n \geq 0\} = \{\epsilon, 01, 0011, 000111, \ldots\}$.
The language $\mathbf{L}$ *is* **not** *a regular language.*

How do we prove it?

Other questions:

- Suppose $\mathbf{R_1}$ is regular and $\mathbf{R_2}$ is regular. Is $\mathbf{R_1} \cap \mathbf{R_2}$ regular?
- Suppose $\mathbf{R_1}$ is regular is $\overline{\mathbf{R_1}}$ (complement of $\mathbf{R_1}$) regular?

A non-regular language and other closure properties

Consider $\mathbf{L} = \{0^n 1^n \mid n \geq 0\} = \{\epsilon, 01, 0011, 000111, \ldots\}$.

[redacted]

$\mathbf{L} = \{0^n 1^n \mid n \geq 0\} = \{\epsilon, 01, 0011, 000111, \ldots\}$.

The language **L** *is* **not** *a regular language.*

How do we prove it?

Other questions:

- Suppose $\mathbf{R_1}$ is regular and $\mathbf{R_2}$ is regular. Is $\mathbf{R_1} \cap \mathbf{R_2}$ regular?
- Suppose $\mathbf{R_1}$ is regular is $\overline{\mathbf{R_1}}$ (complement of $\mathbf{R_1}$) regular?

A sketchy proof

[redacted]

$\mathbf{L} = \{0^n 1^n \mid n \geq 0\} = \{\epsilon, 01, 0011, 000111, \ldots\}$.

The language $\mathbf{L}$ *is* **not** *a regular language.*