Example 1.41

- Convert DFA using construction in Proof of Theorem 1.39

- What are the states of DFA $D$?

- What are the start and accept states?

+ Determine $E(q_0)$

- Determine accept states (those with accept state from $N$)

- Determine transitions using union and $E$

- Can remove states with no input
Theorem 1.47: The class of regular languages is closed under the concatenation operation.

Proof Idea:

\[ A \circ B = \{ xy \mid x \in A \text{ and } y \in B \} \]

Theorem 1.49: The class of regular languages is closed under the star operation.

Proof Idea:

\[ A^* = \{ x_1 x_2 \ldots x_k \mid k \geq 0 \text{ and each } x_i \in A \} \]

Theorem 1.45: Regular languages are closed under the union operation.

Proof Idea: We can use NFA to prove this with simpler construction than we did with DFAs.
In arithmetic, we can use operations to build up expressions describing languages. For example:

- \((0 \cup 1)^*\)
- This has a value of \(2^*\)
- \(\times (2 + 3)\)

Similarly, we can use operations to build up expressions describing languages. For example:

- \((5 + 3) \times 4\)

In arithmetic, we can use + and \(\times\) to build up expressions.
Another Example

• (0 ∪ 1)∗

- Starts with the language (0 ∪ 1) and applies the * operation
- Any string of 0's and 1's

If \( \Sigma = \{0, 1\} \), we can write \( \Sigma \) as shorthand for \( (0 \cup 1) \). Since concatenating any number of symbols combining any number of strings of a language is the language consisting of all \( (0 \cup 1) \). So, \( (0 \cup 1) \) means the language \( \{0, 1\} \). Thus, \( (0 \cup 1) \) is shorthand for the sets \( \{0\} \) and \( \{1\} \).
Notes

• Use $L(H)$ to refer to language of a regular expression $H$
• $R+$ shorthand for $R^*$
• Just like in arithmetic, precedence order is: star, concatenation, union
• Parentheses can be omitted
  - Inductive definition
  - Circular definition?
• Difference between $\varepsilon$ and $\emptyset$?

Formal Definition

Definition 1.52: We say that $R$ is a regular expression if $R$ is
1. $a$ for some $a$ in the alphabet
2. $\varepsilon$
3. $\emptyset$
4. $(R_1 \cap R_2)$, where $R_1$ and $R_2$ are regular expressions
5. $(R_1 \cup R_2)$, where $R_1$ and $R_2$ are regular expressions
6. $(R_1)^*$, where $R_1$ is a regular expression

Computation

item 1. $\varepsilon$ represents the language obtained by taking the union of all combinations of $\varepsilon$ and $\emptyset$
item 2. $\emptyset$ represents the language obtained by taking the union of all combinations of $\varepsilon$ and $\emptyset$
item 3. $a$ is the empty language
item 4. $\{a\}$, $\{\varepsilon\}$, respectively
item 5. $\{a\} \cup \{\varepsilon\}$, respectively
item 6. $(R_1)^*$, where $R_1$ is a regular expression

For some $a$ in the alphabet

Definition 1.52: We say that $R$ is a regular expression if $R$ is
Examples

• $0^*10^*$
• $\Sigma^*1\Sigma^*$
• $\Sigma^*001\Sigma^*$
• $1^*(01^+)^*$
• $(\Sigma\Sigma)^*$
• $01 \cup 10$
• $1^*\emptyset$
• $\emptyset^*$

Identities

• $R \cup \emptyset$
• $R \circ \epsilon$
• What about?
  - $R \circ \emptyset$
  - $R \cup \epsilon$
Regular Expression $\rightarrow$ Regular Language

- Proof Idea: build a NFA to recognize the language
- Make a NFA for each of the 6 cases from the formal definition of regular expression

$$R^1 \cdot$$
$$R_1 \circ R_2$$
$$R_1 \cup R_2$$
$$\emptyset = R$$
$$\epsilon = R$$
$$\nu = R$$

Equivalence with Finite Automata

Theorem 1.54: A language is regular if and only if it is described by a regular expression.

In other words, a language can be recognized by a NFA iff it is.

Regular Expression $\leftrightarrow$ Regular Language

$\circ P$ Heeman, 2017
Regular Language $\rightarrow$ Regular Expression

- Proof idea: convert DFA into a regular expression
- Generalized nondeterministic finite automaton (GNFA)

Example $(ab \cup a)^*$

- Convert to NFA

$\{(a \cap qa) - (a \cap qa) - qa - q - \}$

Convert to NFA
Induction

• GNFA (in restricted form) with more than 2 states
  • Proof Idea:
    - Remove a state that is not the start nor accept state
    + Repair the labels on the remaining transitions
  • Proof Idea in more detail:
    - Remove state $q_{rip}$
    - Alter label from $q_i$ to $q_j$
    + Say $R_4$ was label from $q_i$ to $q_j$
    + Say $R_1$ is label from $q_i$ to $q_{rip}$
    + Say $R_2$ is self-loop label on $q_{rip}$
    - New label is $R_4 \cup (R_1)(R_2)^* R_3$

Proof

• Easy to convert DFA into GNFA
  - Make a new start state with epsilon transition to old start state
  - Make a new accept state with epsilon transition from old accept state
  - Add arrows labeled \( \emptyset \) where needed
  - Any arc with multiple labels use union instead
  - No self loops (aren't allowed on start and accept)
  - One will be start and one will be accept
  - Base Case: Just has 2 states
  + Just one transition

In this case all work is done on start and accept.
Example: Convert into a Regular Expression