Computer Science 313

Regular Expressions

Instructor: Wayne Eberly

Department of Computer Science
University of Calgary

Lecture #8

Goal for Today

- ***Regular expressions*** will be both informally and formally defined.
- Part of the relationship between regular expressions and regular languages will be explained and proved.

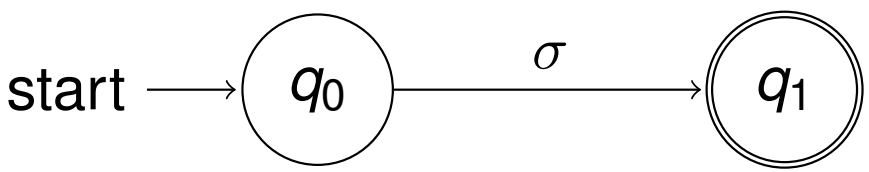
Regular Expressions and Their Languages

Regular Expressions over an alphabet Σ are strings of symbols that include every symbol in Σ, along with a few more special symbols. These — and their languages (all subsets of $\Sigma^\star$ to which they correspond — will now be described.

Regular Expressions and Their Languages

If $\sigma \in \Sigma$ then (the string) σ is also a regular expression (over the alphabet Σ).

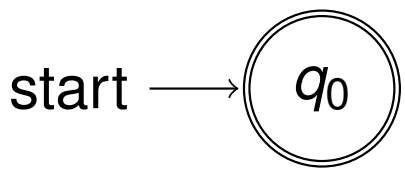
- If $\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$ then each of the following is a regular expression over the alphabet Σ:
  - $\mathtt{a}$
  - $\mathtt{b}$
  - $\mathtt{c}$
- The ***language*** of the regular expression σ (for $\sigma \in \Sigma$) is the set $\{\sigma\}$. This is a regular language since it is the language of the NFA



Regular Expressions and Their Languages

λ is a regular expression (over *every* alphabet Σ).

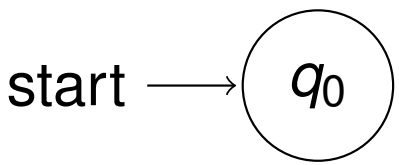
- The ***language*** of the regular expression λ is the set $\{\lambda\}$. This is a regular language since it is the language of the NFA



Regular Expressions and Their Languages

$\emptyset$ is a regular expression (over *every* alphabet Σ).

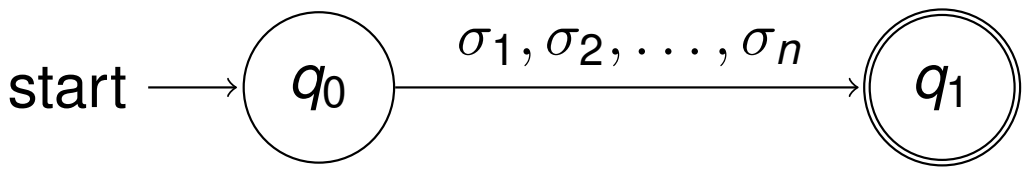
- The ***language*** of the regular expression $\emptyset$ is the empty set $\emptyset$. This is a regular language since it is the language of the NFA



Regular Expressions and Their Languages

Σ a regular expression (over the alphabet Σ).

- The ***language*** of the regular expression Σ is the set Σ — that is, the set of all strings over this alphabet with length one. This is a regular language since it is the language of the NFA that looks like the following, if $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_n\}$.



Regular Expressions and Their Languages

If R_1 and R_2 are regular expressions over the alphabet Σ then the string

$$(R_1 \cup R_2)$$

is a regular expression over the alphabet Σ as well.

- If $\omega = (R_1 \cup R_2)$ then the language $L(\omega)$ of ω is the union of the languages of R_1 and R_2 — that is,

$$L(\omega) = L(R_1) \cup L(R_2).$$

- Recall that the set of regular language is closed under union (Claim #2 from the previous lecture). Thus, if $\omega = (R_1 \cup R_2)$ and the languages $L(R_1)$ and $L(R_2)$ are both regular languages, then $L(\omega)$ is a regular language too.

Regular Expressions and Their Languages

If R_1 and R_2 are regular expressions over the alphabet Σ then the string

$$(R_1 \circ R_2)$$

is a regular expression over the alphabet Σ as well.

- If $\omega = (R_1 \circ R_2)$ then the language $L(\omega)$ of ω is the concatenation of the languages of R_1 and R_2 — that is,

  $$L(\omega) = L(R_1) \circ L(R_2) = \{\omega_1 \cdot \omega_2 \mid \omega_1 \in L(R_1) \text{ and } \omega_2 \in L(R_2)\}.$$

- Recall that the set of regular language is closed under concatenation (Claim #3 from the previous lecture). Thus, if $\omega = (R_1 \circ R_2)$ and the languages $L(R_1)$ and $L(R_2)$ are both regular languages, then $L(\omega)$ is a regular language too.

Regular Expressions and Their Languages

If R_1 is a regular expression over the alphabet Σ then the string

$$(R_1)^\star$$

is a regular expression over the alphabet Σ as well.

- If $\omega = (R_1)^\star$ then the language $L(\omega)$ of ω is the star of the language of R_1 — that is

$$L(\omega) = (L(R_1))^\star = \{\omega_1 \cdot \omega_2 \ldots \omega_k \mid k \geq 0 \text{ and } \omega_1, \omega_2, \ldots, \omega_k \in L(R_1)\}.$$

- Recall that the set of regular language is closed under star (Claim #4 from the previous lecture). Thus, if $\omega = (R_1)^\star$ and the language $L(R_1)$ of R_1 is a regular language, then $L(\omega)$ is a regular language too.

Formal Definition of a Regular Expression

A string of symbols ω is a ***regular expression*** for the alphabet Σ if and only if it can be formed using a finite number of applications of the rules shown on this and the following slide.

1. $\omega = \sigma$, for some symbol $\sigma \in \Sigma$.
2. $\omega = \lambda$.
3. $\omega = \emptyset$.
4. $\omega = \Sigma$.
5. If R_1 and R_2 are both regular expressions over the alphabet Σ and ω is the string $(R_1 \cup R_2)$ then ω is regular expression over the alphabet Σ as well.

Formal Definition of a Regular Expression

6. If R_1 and R_2 are both regular expressions over the alphabet Σ and ω is the string $(R_1 \circ R_2)$ then ω is regular expression over the alphabet Σ as well.
7. If R_1 is a regular expression over the alphabet Σ and ω is the string $(R_1)^\star$ then ω is a regular expression over the alphabet Σ as well.

Formal Definition: The Language of a Regular Expression

If ω is a regular expression for the alphabet Σ then the ***language*** $L(\omega)$ of ω is as shown on the following slides.

1. If $\omega = \sigma$, for $\sigma \in \Sigma$, then $L(\omega) = L(\sigma) = \{\sigma\}$.
2. If $\omega = \lambda$ then $L(\omega) = L(\lambda) = \{\lambda\}$.
3. If $\omega = \emptyset$ then $L(\omega) = L(\emptyset) = \emptyset$.
4. If $\omega = \Sigma$ then $L(\omega) = L(\Sigma) = \Sigma$ (the set of all strings in $\Sigma^\star$ with length one).
5. If ω is the string $(R_1 \cup R_2)$ where R_1 and R_2 are regular expressions over Σ then the language $L(\omega)$ of ω is the set $L(R_1) \cup L(R_2)$.

Formal Definition: The Language of a Regular Expression

6. If ω is the string $(R_1 \circ R_2)$ where R_1 and R_2 are regular expressions over Σ then the language $L(\omega)$ of ω is the set $L(R_1) \circ L(R_2)$.
7. If ω is the string $(R_1)^\star$ where R_1 is a regular expression over Σ then the language $L(\omega)$ of ω is the set $(L(R_1))^\star$.

Regular Expressions and Their Languages

Example: Let $\Sigma = \{0, 1\}$.

1. By rule #4, Σ is a regular language over Σ whose language is $\Sigma = \{0, 1\}$.
2. By rule #7 — and considering the regular expression at line 1 — $(\Sigma)^\star$ is a regular language over Σ whose language is $(\Sigma)^\star = \Sigma^\star$ — the set of all strings over the alphabet Σ.
3. By rule #1, 1 is a regular expression over Σ whose language is $\{1\}$.
4. By rule #6 — and considering the regular expressions at lines 2 and 3 — $((\Sigma)^\star \circ 1)$ is a regular expression over Σ whose language is

$$\Sigma^\star \circ \{1\} = \{\omega \in \Sigma^\star \mid \omega \text{ ends with } 1\}.$$

Regular Expressions and Their Languages

5. By rule #2 λ is a regular expression over Σ whose language is $\{\lambda\}$.
6. By rule #1, 0 is a regular expression over Σ whose language is $\{0\}$.
7. By rule #5 — and considering the regular expressions at lines #5 and #6 — $(\lambda \cup 0)$ is a regular expression over Σ whose language is $\{\lambda\} \cup \{0\} = \{\lambda, 0\}$.
8. By rule #6 – and considering the regular expressions at lines #4 and #7 —

$$(((\Sigma)^\star \circ 1) \circ (\lambda \cup 0))$$

is a regular expression over Σ whose language is

$$\{\omega \in \Sigma^\star \mid \omega \text{ ends with } 1\} \circ \{\lambda, 0\}$$
$$= \{\omega \in \Sigma^\star \mid \text{either } \omega \text{ ends with } 1 \text{ or } \omega \text{ ends with } 10\}.$$

Regular Expressions and Their Languages

9. By rule #6 — and considering the regular expressions at lines #8 and #3 —

$$((((\Sigma)^\star \circ 1) \circ (\lambda \cup 0)) \circ 1)$$

is a regular expression over Σ whose language is

$$\{\omega \in \Sigma^\star \mid \text{either } \omega \text{ ends with } 1 \text{ or } \omega \text{ ends with } 10\} \circ \{1\}$$
$$= \{\omega \in \Sigma^\star \mid \text{either } \omega \text{ ends with } 11 \text{ or } \omega \text{ ends with } 101\}.$$

10. By rule #6 — and considering the regular expressions at lines #9 and #2 —

$$(((((\Sigma)^\star \circ 1) \circ (\lambda \cup 0)) \circ 1) \circ (\Sigma)^\star)$$

is a regular expression over Σ. With a *little* bit of work its language can be shown to be

$$\{\omega \in \Sigma^\star \mid \text{either } 11 \text{ or } 101 \text{ (or both) is a substring of } \omega\}$$

Simplification and Shortcuts

Left and right brackets can be left out.

However — when you are identifying the language of a regular expression, you must now remember that

- The star operation has higher ***precedence*** than the others — so that
  - $R_1 \cup R_2^\star$ has the same language as $(R_1 \cup (R_2)^\star)$ — and *not* generally the same language as $((R_1 \cup R_2))^\star$
  - $R_1 \circ R_2^\star$ has the same language as $(R_1 \circ (R_2)^\star)$ — and *not* generally the same language as $((R_1 \circ R_2))^\star$.
- Concatenation has higher ***precedence*** than union — so that $R_1 \circ R_2 \cup R_3$ has the same language as $((R_1 \circ R_2) \cup R_3)$ — and *not* generally the same language as $(R_1 \circ (R_2 \cup R_3))$.

Simplification and Shortcuts

The $\circ$ symbol can also be left out.

Thus, the regular expression from the previous example

$$(((((\Sigma)^\star \circ 1) \circ (\lambda \cup 0)) \circ 1) \circ (\Sigma)^\star)$$

can now be written more simply as

$$\Sigma^\star 1(\lambda \cup 0)1\Sigma^\star$$

Note that this ***does not*** have the same language as the regular expression

$$\Sigma^\star 1\lambda \cup 01\Sigma^\star$$

Simplifications and Shortcuts

- If R is a regular expression over Σ then R^+ will be used as shorthand for $RR^\star$ (that is, $(R \circ (R)^\star)$). Thus the language of R^+ is

$$\{\omega_1 \cdot \omega_2 \ldots \omega_k \mid k \geq 1 \text{ and } \omega_1, \omega_2, \ldots, \omega_k \in L(R)\}$$

- For each integer $k \geq 0$, R^k will be used as shorthand as the concatenation of k copies of each other — so that R^2 is shorthand for $(R \circ R)$, R^3 is shorthand for $R \circ R \circ R$ (that is, $((R \circ R) \circ R)$), and so on.

  Thus the language of R^k is

$$\{\omega_1 \cdot \omega_2 \ldots \omega_k \mid \omega_1, \omega_2, \ldots, \omega_k \in L(R)\}$$

Equivalence, Part One

Claim: Let Σ be a finite nonempty alphabet and let R be a regular expression over Σ. Then the language $L = L(R) \subseteq \Sigma^\star$ is a regular language.

Sketch of Proof: Without loss of generality we may assume that no "shortcuts" or "simplifications" have been used — because we already know that regular expressions using these always have the same languages as regular expressions that do not use them..

The proof now proceeds using ***mathematical induction*** on the length of the string R. The strong form of mathematical induction will be used. Since every regular expression is a string with length at least one, the case $|R| = 1$ will be considered in the basis.

Equivalence, Part One

Basis: If the length of R is one then one of rules #1–4 were used to define R — so that either

- $R = \sigma$ for some symbol $\sigma \in \Sigma$, and $L(R) = \{\sigma\}$.
- $R = \lambda$, and $L(R) = \{\lambda\}$,
- $R = \emptyset$, and $L(R) = \emptyset$.
- $R = \Sigma$, and $L(R) = \Sigma$.

It has already been confirmed that $L(R)$ is a regular language in each of these cases.

Equivalence, Part One

Let k be an integer such that $k \geq 1$. In order to complete the proof, it is necessary and sufficient to prove the following

> *Inductive Hypothesis:* If R is a regular expression over Σ with length h, for any integer h such that $1 \leq h \leq k$, then $L(R)$ is a regular language.

to prove the following

> *Inductive Claim:* If R is a regular expression over Σ with length $k + 1$, then $L(R)$ is a regular language.

With that noted, let R be a regular expression over Σ with length $k + 1$. Since $k \geq 1$, $k + 1 \geq 2$, and one of rules #5–#7 must have been used to produce R.

Equivalence, Part One

Case: Rule #5 was used to produce R.

- Then R is a string with the form $(R_1 \cup R_2)$ where R_1 and R_2 are also regular expressions over Σ.
- The lengths of R_1 and R_2 must both be between one and $(k+1)-4 = k-3 \leq k$, so it now follows by the *inductive hypothesis* that both $L(R_1)$ and $L(R_2)$ are regular languages.
- Thus $L(R) = L(R_1) \cup L(R_2)$ is also a regular language, since the union of two language is always a regular language — see Claim #2 from the previous set of lecture notes (and its proof).

Equivalence, Part One

Case: Rule #6 was used to produce R.

- Then R is a string with the form $(R_1 \circ R_2)$ where R_1 and R_2 are also regular expressions over Σ.
- The lengths of R_1 and R_2 must both be between one and $(k+1)-4 = k-3 \leq k$, so it now follows by the *inductive hypothesis* that both $L(R_1)$ and $L(R_2)$ are regular languages.
- Thus $L(R) = L(R_1) \circ L(R_2)$ is also a regular language, since the concatenation of two language is always a regular language — see Claim #3 from the previous set of lecture notes (and its proof).

Equivalence, Part One

Case: Rule #7 was used to produce R.

- Then R is a string with the form $(R_1)^\star$ where R_1 is also a regular expression over Σ.
- The length of R_1 is $(k+1)-3 = k-2 \leq k$, so it now follows by the *inductive hypothesis* that $L(R_1)$ is a regular language.
- Thus $L(R) = (L(R_1))^\star$ is a regular language, since the star of a regular language is always a regular language too — see Claim #4 from the previous set of lecture notes (and its proof).

Equivalence, Part One

Conclusion: Since $L(R)$ has been shown to be a regular language in every possible case, this completes the inductive step — and the proof of the claim.

It now follows that the language of every regular expression is a regular language.

Note: The ***proofs*** of the claims in these lecture notes, and the previous notes, are all ***constructive:*** They can be used to discover (and apply) a process to ***construct*** an NFA whose language is $L(R)$ for any given regular expression R.

Furthermore, the NFA will not be very *big:* The number of states in this NFA will never be more than linear in the length of the regular expression R that was used to construct it.

What's Next?

- A constructive proof of the opposite inclusion: Every regular language is the language of some regular expression, as well!