I.5.1.. Regular Grammars and NFAs (13.5)

Proof of Theorem I.5.1.1.

We show that $L(A) = L(G)$:

1. Assume $w = a_1 \cdots a_n \in L(A)$.
   
   Then there exists a sequence of transitions in $A$
   
   $$S = A_0 \xrightarrow{a_1} A_1 \xrightarrow{a_2} A_2 \xrightarrow{a_3} \cdots \xrightarrow{a_{n-1}} A_{n-1} \xrightarrow{a_n} q_F$$

   or
   
   $$S = A_0 \xrightarrow{a_1} A_1 \xrightarrow{a_2} A_2 \xrightarrow{a_3} \cdots \xrightarrow{a_n} A_n \in F$$

   But from this we obtain derivations
   
   $$S = A_0 \rightarrow a_1 A_1 \rightarrow a_1 a_2 A_2 \rightarrow \cdots \rightarrow a_1 a_2 \cdots a_{n-1} A_{n-1} \rightarrow a_1 a_2 \cdots a_{n-1} a_n = w$$

   or
   
   $$S = A_0 \rightarrow a_1 A_1 \rightarrow a_1 a_2 A_2 \rightarrow \cdots \rightarrow a_1 a_2 \cdots a_n A_n \rightarrow a_1 a_2 \cdots a_n = w$$

   So $w \in L(G)$. 

I.5.2. Translating NFAs into Regular Expressions (13.10)

I.5.3. Equivalence Theorem
Proof of Theorem I.5.1.1.

Assume \( w = a_1 \cdots a_n \in L(G) \).
A derivation will have the form
\[
S = A_0 \rightarrow a_1 A_1 \rightarrow a_1 a_2 A_2 \rightarrow \cdots \rightarrow a_1 a_2 \cdots a_{n-1} A_{n-1} \rightarrow a_1 a_2 \cdots a_n = w
\]
or
\[
S = A_0 \rightarrow a_1 A_1 \rightarrow a_1 a_2 A_2 \rightarrow \cdots \rightarrow a_1 a_2 \cdots a_n A_n \rightarrow a_1 a_2 \cdots a_n = w
\]
Then there exists a sequence of transitions in \( A \)
\[
S = A_0 \xrightarrow{a_1} A_1 \xrightarrow{a_2} A_2 \xrightarrow{a_3} \cdots \xrightarrow{a_{n-1}} A_{n-1} \xrightarrow{a_n} q_F
\]
or
\[
S = A_0 \xrightarrow{a_1} A_1 \xrightarrow{a_2} A_2 \xrightarrow{a_3} \cdots \xrightarrow{a_n} A_n \in F
\]
So \( w \in L(A) \).

Example

Before proving Theorem I.5.2.1. we give an example:
Consider the following automaton for the language \( L = \emptyset \).

![Automaton Diagram]

We define regular expressions and simplify them at each intermediate step in order to keep them simple.

From \( A \) to \( E_{\emptyset}^{q, q'} \)

Original automaton:

Let \( L_{\emptyset}^{q, q'} \) be the set of strings which allows us to get from \( q \) to \( q' \) with intermediate states in \( \emptyset \), i.e. without any intermediate states.
We define a regular expression \( E_{\emptyset}^{q, q'} \) s.t. \( L(E_{\emptyset}^{q, q'}) = L_{\emptyset}^{q, q'} \). We can define

- \( E_{q, q'}^{\emptyset} := a_1 | \cdots | a_n \), if \( q \neq q' \) and we have transitions \( q \xrightarrow{a_i} q' \),
- \( E_{q, q'}^{\emptyset} = a_1 | \cdots | a_n | \epsilon \), if \( q = q' \) and we have transitions \( q \xrightarrow{a_i} q' \).
I.5.2. Translating NFAs into Regular Expressions (13.10)

Calculation of $L_{q,q'}^∅$

Original automaton:

From $A$ to $L_{q,q'}^∅$

States with $E_{q,q'}^∅$:

Calculation of $E_{q,q'}^{q_0}$

Let $L_{q,q'}^{q_0}$ be the set of strings which allows us to get from $q$ to $q'$ with intermediate states in $\{q_0\}$.

We define $E_{q,q'}^{q_0}$ s.t. $L(E_{q,q'}^{q_0}) = L_{q,q'}^{q_0}$:

$$E_{q,q'}^{q_0} = E_{q,q'}^∅ | (E_{q,q_0}^∅ E_{q_0,q'}^∅)^*$$
I.5.2. Translating NFAs into Regular Expressions (13.10)

From $E^\emptyset_{q, q'}$ to $E^q_{q, q'}$

States with $E^\emptyset_{q, q'}$:

States with $E^q_{q, q'}$:

Calculation of $E^{q_0, q_1}_{q, q'}$

Let $L_{q, q'}$ be the set of strings which allows us to get from $q$ to $q'$ with intermediate states in $\{q_0, q_1\}$.

We define $E^{q_0, q_1}_{q, q'}$ s.t. $L(E^{q_0, q_1}_{q, q'}) = L_{q, q'}$.

$$E^{q_0, q_1}_{q, q'} = E^{q_0}_{q, q'} \cup (E^{q_0}_{q, q_1} (E^{q_1}_{q_1, q_1})^* E^{q_0}_{q_1, q'}).$$
The Language of $A$: $L(A)$

States with $E_{q_0,q_1}^{q_0,q_1}$:

$\begin{align*}
\AD &\stackrel{1^{*}0(0 \mid 1)^{*}}{\longrightarrow} \\
q_0 &\rightarrow (0 \mid 1)^{*} \\
q_1 &\rightarrow (0 \mid 1)^{*}
\end{align*}$

$\rightarrow L(E_{q_0,q_1}^{q_0,q_1})$ is the set of strings which allows us to get from $q_0$ to $q_1$ using any intermediate states.

$\rightarrow$ The language $L(A)$ is the set of strings which allow us to get from $q_0$ to any accepting state.

$\rightarrow$ In the example there is only one accepting state ($q_1$), so the language accepted by $A$ is the language given by

$$E_{q_0,q_1}^{q_0,q_1} = 1^{*}0(0 \mid 1)^{*}$$

Proof of Theorem I.5.2.1.

Let for states $q$, $q'$ of $A$

$$L(q,q') := \{ w \in T^* \mid q \xrightarrow{w} q' \}$$

We construct for states $q$, $q'$ of $A$ a regular expression $E_{q,q'}$ s.t.

$$L(E_{q,q'}) = L(q,q')$$

If $F = \{q_1, \ldots, q_k\}$ then we obtain

$$L(A) = L_{q_0,q_1} \land \cdots \land L_{q_0,q_k} = L(E_{q_0,q_1} \mid E_{q_0,q_2} \mid \cdots \mid E_{q_0,q_k})$$

(If $F$ is empty, then $L(A) = L(\emptyset)$).

We define regular expressions $E_{q,q'}$ in stages by referring to $E_{q_1,\ldots,q_i}^{q_1,\ldots,q_i}$, s.t.

$$L(E_{q_1,\ldots,q_i}^{q_i,\ldots,q_i}) = L(q_i,q_i)$$

$$:= \{ a_1 \cdot \cdots a_k \in T^* \mid \exists p_i \in \{q_1, \ldots, q_i\}.$$

$$q \xrightarrow{a_1} p_1 \xrightarrow{a_2} p_2 \xrightarrow{q_3} \cdots \xrightarrow{a_{k-1}} p_{k-1} \xrightarrow{a_k} q' \}$$

So $L_{q_i,q_i}^{q_i,\ldots,q_i}$ is the set of words which allow us to get from $q$ to $q'$ by using as intermediate states $q_1, \ldots, q_i$ only.

We define $E_{q_1,\ldots,q_k}^{q_1,\ldots,q_k}$ by induction on $k$.

Then we can define $E_{q,q'} := E_{q,q'}^{q_0,q_1}.$
Proof of Theorem I.5.2.1.

Base case $k = 0$:
Let $a_1, \ldots, a_k$ be the $a_i$ s.t. $q \xrightarrow{a_i} q'$. Then

$$E^0_{q, q'} := \begin{cases} a_1 \ldots a_k & \text{if } q \neq q' \\ a_1 \ldots a_k \epsilon & \text{if } q = q' \end{cases}$$

(in case of $k = 0$ we have $E^0_{q, q'} = \emptyset$ or $= \epsilon$).

So we have

$$q \xrightarrow{v} q_k \xrightarrow{w_1} q_k \xrightarrow{w_2} q_k \xrightarrow{\cdots} q_k \xrightarrow{w_j} q_k \xrightarrow{v'} q'$$

where $j = 0$ is possible, all intermediate transitions avoid $q_k$ and $w = vw_1w_2\cdots w_kv'$.

Induction Step: Assume we have defined $E_{q_1, \ldots, q_k-1}$ for all $p, p' \in Q$.
We define $E_{q, q'}^{q_1, \ldots, q_{k-1}}$.

A transition $q \xrightarrow{w} q'$ which uses only intermediate states $q_1, \ldots, q_k$ can have two forms:

- Either we don’t use $q_k$ as an intermediate state.
  So we have only intermediate states $q_1, \ldots, q_{k-1}$ and have $w \in L_{q, q'}^{q_1, \ldots, q_{k-1}}$.
- Or we reach $q_k$ as an intermediate state. We single out
  - the first part of the transition which doesn’t use state $q_k$ until one reaches for the first time as an intermediate state $q_k$ (note that $q = q_k$ or $q' = q_k$ is possible)
  - the second part where we several times go from $q_k$ to $q_k$ with intermediate states $\neq q_k$,
  - and the last part where we get from $q_k$ to $q'$ without using $q_k$.

In the second part we have

- $v \in L_{q, q'}^{q_1, \ldots, q_{k-1}}$.
- $w_i \in L_{q_k, q_{k-1}}^{q_1, \ldots, q_{k-1}}$.
- $v' \in L_{q_k, q'}^{q_1, \ldots, q_{k-1}}$.
- Therefore $w = vw_1 \cdots w_kv' \in L_{q, q'}^{q_1, \ldots, q_{k-1}}(L_{q_k, q_k}^{q_1, \ldots, q_{k-1}})^* L_{q_k, q'}^{q_1, \ldots, q_{k-1}}$.

Therefore

$$L_{q, q'}^{q_1, \ldots, q_k} \subseteq L_{q, q'}^{q_1, \ldots, q_{k-1}} (L_{q_k, q_k}^{q_1, \ldots, q_{k-1}})^* L_{q_k, q'}^{q_1, \ldots, q_{k-1}}$$

One can see easily as well that for an element $w$ in the right hand side we can derive that $w$ is in the left hand side as well, i.e.

$$L_{q, q'}^{q_1, \ldots, q_k} \supseteq L_{q, q'}^{q_1, \ldots, q_{k-1}} (L_{q_k, q_k}^{q_1, \ldots, q_{k-1}})^* L_{q_k, q'}^{q_1, \ldots, q_{k-1}}$$
I.5.2. Translating NFAs into Regular Expressions (13.10)

Proof of Theorem I.5.2.1.

So

\[ L_{q_1, q'} = L_{q_1, q_k-1} | (L_{q_1, q_k} - 1)(L_{q_1, q_k} - 1) \]

and we can define

\[ E_{q_1, q'} = E_{q_1, q_k-1} | (E_{q_1, q_k} - 1)(E_{q_1, q_k} - 1) \]

We are going to show that

- languages definable by regular expressions,
- languages definable by regular grammars,
- languages definable by NFAs with empty moves,
- languages definable by NFAs,
- languages definable by DFAs

all define the same language.

We already have everything in order to show the above with regular grammars restricted to right-linear grammars.

I.5.3. Equivalence Theorem

Proof of Theorem I.5.3.1.

Lemma I.5.3.2.

Let \( L \) be a language over an alphabet \( T \). The following is equivalent:

1. \( L \) is definable by a regular expression.
2. \( L \) is definable by a right-linear grammar.
3. \( L \) is definable by an NFA with empty moves.
4. \( L \) is definable by an NFA.
5. \( L \) is definable by a DFA.

Furthermore, the corresponding regular expressions, right linear grammars, NFAs with empty moves, NFAs, DFAs can be computed from each other.
Proof of Lemma I.5.3.2.

- (1) → (2) was shown in I.3.1.1. and I.3.2.1.
  - (Finite languages are definable regular grammars. Languages definable by regular grammars are closed under the operations for forming regular expressions).
- (2) → (4) was shown in Theorem I.5.1.1.
  - Right-linear grammars can be simulated by an NFA.
- (4) → (1) was shown in Theorem I.5.2.1.
  - We can determine the language between states of an NFA as a regular expression.
- So (1), (2), (4) are equivalent.

- (3) → (4) was shown in Theorem I.4.4.1.
  - We can omit the empty moves in NFAs with empty moves.
- (4) → (5) was shown in Theorem I.4.5.1.
  - NFAs can be translated into DFAs using as states sets of states.
- (5) → (4) → (3) are trivial.
  - DFAs are special cases of NFAs, NFAs are special cases of NFAs with empty moves.
- So (3), (4), (5) are equivalent.
- So (1), (2), (3), (4), (5) are equivalent.

Equivalence of Left-Linear and Right-Linear Grammars

- It remains to show that left-linear and right-linear grammars are equivalent.
- This is shown as follows:
  - The languages derived by left-linear grammars are $L^R$ for languages derived by right-linear grammars.
  - Regular Expressions are closed under the reverse operation, i.e. if $L$ is definable by a regular expression, so is $L^R$.
  - Therefore as well right-linear grammars are closed under the reverse operation, therefore left-linear and right-linear grammars are equivalent.

Lemma (I.5.3.3.)

1. Let $G$ be a left-linear grammar.
   Then there exist a right-linear grammar $G'$ over the same alphabet s.t. $L(G) = L(G')^R$.
   $G'$ can be computed from $G$.
2. Let $G$ be a right-linear grammar.
   Then there exist a left-linear grammar $G'$ over the same alphabet s.t. $L(G) = L(G')^R$.
   $G'$ can be computed from $G$. 

Right-Linear Languages are the Reverse of Left-Linear Ones
Proof of Lemma I.5.3.3.

We prove only (1), (2) is analogously.

Let $G$ be a left-linear grammar with alphabet $T$, nonterminals $N$, and start symbol $S$.

Let $G'$ be identical to $G$ but with rules $B \rightarrow aC$, $B \rightarrow Ca$ replaced by

$\begin{align*}
B & \rightarrow aC \\
(B, C \in N, a \in T) & \text{ replaced by} \\
B & \rightarrow Ca
\end{align*}$

$G'$ is right-linear. Further it follows immediately for any $w \in (N \cup T)^*$ that $S \Rightarrow G w$ iff $S \Rightarrow G' w^R$.

Therefore

$$L(G') = \{w \in T^* \mid S \Rightarrow G' w\} = \{w^R \in T^* \mid S \Rightarrow G w\} = \{w \in T^* \mid S \Rightarrow G w\}^R = L(G)^R$$

Proof of Lemma I.5.3.4.

(1) We show the existence of $E^R$ by induction on $E$:

- For $E = \emptyset$, $E = \epsilon$ or $E = a$ $L(E)^R = L(E)$, so define $E^R := E$.
- For $E = E_1 \mid E_2$ we have define $E^R = E_1^R \mid E_2^R$.
- For $E = E_1E_2$ define $E^R = E_2^RE_1^R$.
- For $E = E_1^*$ define $E^R = (E_1^R)^*$.

(2) Follows since languages definable by right-linear grammars are exactly the languages definable by regular expressions.
Left-Linear and Right-Linear Grammars are Equivalent

Lemma (I.5.3.5.)

Let $L$ be a language over an alphabet $T$. The following are equivalent:

1. $L = L(G)$ for a left-linear grammar $G$.
2. $L = L(G)$ for a right-linear grammar $G$.

The left-linear and right-linear grammars can be computed from each other.

Proof of Lemma I.5.3.5.

- Assume $L = L(G)$ for a left-linear grammar $G$.
  - Then $L^R = L(G')$ for a right-linear grammar $G'$.
  - Right-linear grammars are closed under $L \mapsto L^R$.
  - Therefore there exists a right-linear grammar $G''$ s.t. $L(G'') = L(G')^R = (L^R)^R = L$.
- Assume $L = L(G)$ for a right-linear grammar $G$.
  - There exists a right-linear grammar $G'$ s.t. $L(G') = L^R$.
  - There exists a left-linear grammar $G''$ s.t. $L(G'') = L(G')^R$.
  - Now $L(G'') = L(G')^R = (L(G))^R = L(G) = L$.

Proof of Theorem I.5.3.1.

By the above.