MTL783: Theory of Computation 1st Semester, 2025-2026

Lecture 5 — 07/08/2025

Lecturer: Prof Minati De *Scribe: Team 5*

Scribed by:

1. Arvind Kumar Sharma (2022MT11268)
2. Krishna Sharma (2022MT11263)
3. Khushank Galgat (2022MT11291)
4. Prateek Mourya (2022MT11937)
5. Soham Sameer Palkar (2022MT61971)

1 Overview

In the last lecture, we discussed regular languages, Nondeterministic Finite Accepters (NFAs), their equivalence with Deterministic Finite Accepters (DFAs), and an example of their interconversion.

In this lecture, we will further explore the equivalence of DFAs and NFAs, present a detailed interconversion procedure with a formal proof, and give additional examples. We will conclude with the formal definition and some examples of regular expressions.

2 Equivalence of Deterministic and Nondeterministic Finite Accepters

Theorem 2.1. *Let L be the language accepted by a nondeterministic finite accepter $M_N = (Q_N, \Sigma, \delta_N, q_0, F_N)$. Then there exists a deterministic finite accepter $M_D = (Q_D, \Sigma, \delta_D, \{q_0\}, F_D)$ such that*

$$L = L(M_D).$$

Proof. Given M_N, we use the procedure `nfa-to-dfa` below to construct the transition graph G_D for M_D. To understand the construction, remember that G_D has to have certain properties. Every vertex must have exactly $|\Sigma|$ outgoing edges, each labeled with a different element of Σ. During the construction, some of the edges may be missing, but the procedure continues until they are all there.

Procedure: nfa-to-dfa

1. Create a graph G_D with vertex $\{q_0\}$. Identify this vertex as the initial vertex.
2. Repeat the following steps until no more edges are missing.

(a) Take any vertex $\{q_i, q_j, \ldots, q_k\}$ of G_D that has no outgoing edge for some $a \in \Sigma$.

(b) Compute $\delta_N^*(q_i, a), \delta_N^*(q_j, a), \ldots, \delta_N^*(q_k, a)$.

(c) If
$$\delta_N^*(q_i, a) \cup \delta_N^*(q_j, a) \cup \cdots \cup \delta_N^*(q_k, a) = \{q_l, q_m, \ldots, q_n\},$$
create a vertex for G_D labeled $\{q_l, q_m, \ldots, q_n\}$ if it does not already exist.

(d) Add to G_D an edge from $\{q_i, q_j, \ldots, q_k\}$ to $\{q_l, q_m, \ldots, q_n\}$ and label it with a.

3. Every state of G_D whose label contains any $q_f \in F_N$ is identified as a final vertex.

4. If M_N accepts λ, the vertex $\{q_0\}$ in G_D is also made a final vertex.

Idea: Each state of M_D corresponds to a subset of states of M_N. The initial state of M_D is $\{q_0\}$. A state of M_D is accepting if and only if its subset contains at least one accepting state of M_N. Transitions in M_D are defined so that reading a symbol a from a subset $S \subseteq Q_N$ leads to the subset of all states reachable in M_N from any state in S by reading a.

Termination: The construction clearly terminates because the number of possible subsets of Q_N is finite — at most $2^{|Q_N|}$. For each subset, we add exactly $|\Sigma|$ outgoing edges. Hence the total number of edges in G_D is at most $2^{|Q_N|} \cdot |\Sigma|$, and the process stops once all are created.

Correctness: We will use the following lemma to prove the correctness of the procedure.

Lemma: Let w be an input string of length n such that:

$$\delta_N^*(q_0, w) = S \quad \Longleftrightarrow \quad \delta_D^*(\{q_0\}, w) = S$$

for some subset $S \subseteq Q_N$. That is, if there exists a w-walk from q_0 to some $q_\ell \in S$ in G_N, then there exists a w-walk from $\{q_0\}$ to S in G_D, and vice versa.

Proof: We prove the lemma by induction on the length of the string (n):

($\Rightarrow$) To prove:
$$\delta_N^*(q_0, w) = S \quad \Rightarrow \quad \delta_D^*(\{q_0\}, w) = S$$

Base case ($n = 0$): When $w = \lambda$, we have $\delta_N^*(q_0, \lambda) = \{q_0\}$ and $\delta_D^*(\{q_0\}, \lambda) = \{q_0\}$ by definition. The statement holds.

Induction hypothesis: Assume the property holds for all strings of length n.

Inductive step: Let $w = va$ where $|v| = n$ and $a \in \Sigma$.

Suppose in M_N there is a computation from q_0 to some q_ℓ labeled w. Then there is an intermediate state q_i reachable from q_0 by v, and a transition from q_i to q_ℓ labeled a. By the inductive hypothesis, in M_D there is a state S containing q_i reachable from $\{q_0\}$ by v. By the construction of δ_D, the a-transition from S leads to exactly the set of states reachable in M_N by reading a from any $q \in S$. Hence q_ℓ is included in $\delta_D^*(\{q_0\}, w)$.

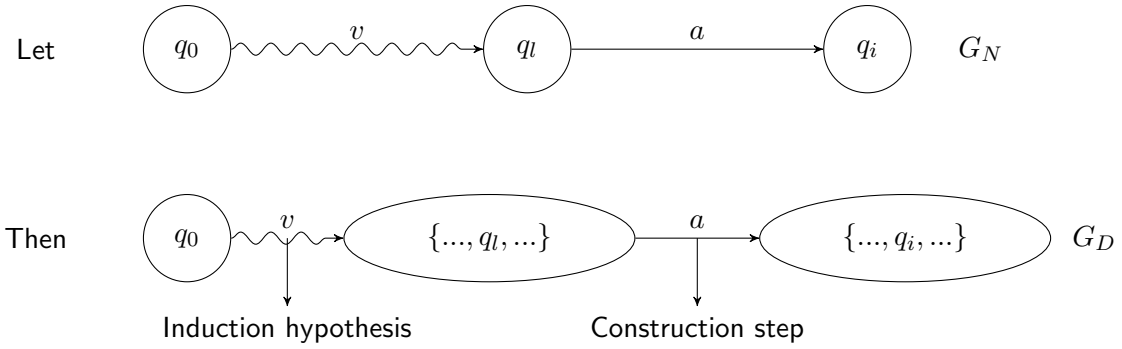
($\Leftarrow$) To prove:
$$\delta_D^*(\{q_0\}, w) = S \quad \Rightarrow \quad \delta_N^*(q_0, w) = S$$

The reverse direction is analogous. Let $\delta_D^*(\{q_0\}, w) = \{q_\ell, q_m, \ldots, q_n\}$ and $\delta_D^*(\{q_0\}, v) = \{q_i, q_j, \ldots, q_k\}$. Then, by the procedure:

$$\delta_N^*(q_i, a) \cup \delta_N^*(q_j, a) \cup \cdots \cup \delta_N^*(q_k, a) = \{q_\ell, q_m, \ldots, q_n\}.$$

At least one among $\delta_N^*(q_i, a), \delta_N^*(q_j, a), \ldots, \delta_N^*(q_k, a)$ must contain q_ℓ. Say $\delta_N^*(q_i, a)$ contains q_ℓ. Then, by the induction hypothesis, there is a v-computation from q_0 to q_i in M_N. Combining the above two statements, we get a w-computation from q_0 to q_ℓ.

Hence, using the above lemma, it is clear that if any string w is accepted by M_N, then it is also accepted by M_D. This is because $\delta_N^*(q_0, w) = q_f \in F_N$. Since $q_f \in S$, we have $S \in F_D$ and $\delta_D^*(\{q_0\}, w) = S$ by the lemma, so w is accepted by M_D as well.



□

Let us do an example illustrating the above steps.

Example 2.1. Convert the nfa in Figure 1 into an equivalent deterministic machine.

Solution: Since $\delta_N(q_0, 0) = \{q_0, q_1\}$, we introduce the state $\{q_0, q_1\}$ in G_D and add an edge labeled 0 between $\{q_0\}$ and $\{q_0, q_1\}$. In the same way, considering $\delta_N(q_0, 1) = \{q_1\}$ gives us the new state $\{q_1\}$ and an edge labeled 1 between it and $\{q_0\}$. There are now a number of missing edges, so we continue, using the construction of Theorem 2.1. Looking at the state $\{q_0, q_1\}$, we see that there is no outgoing edge labeled 0, so we compute

$$\delta_N^*(q_0, 0) \cup \delta_N^*(q_1, 0) = \{q_0, q_1, q_2\}.$$

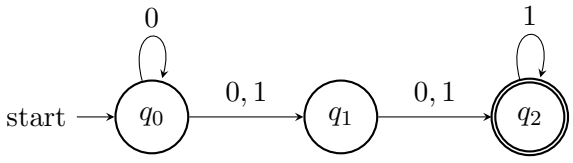


Figure 1

This gives us the new state $\{q_0, q_1, q_2\}$ and the transition

$$\delta_D(\{q_0, q_1\}, 0) = \{q_0, q_1, q_2\}.$$

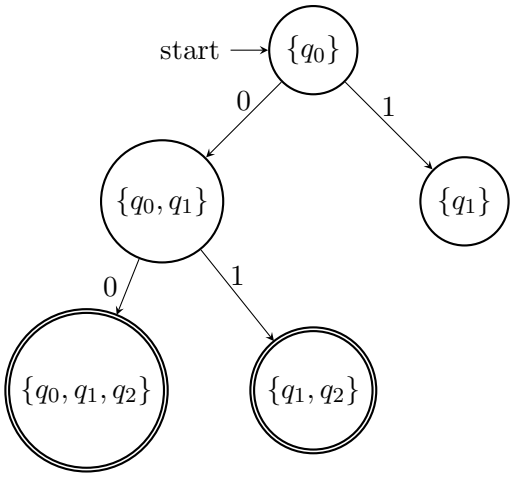


Figure 2

Then, using $a = 1, i = 0, j = 1, k = 2$,

$$\delta_N^*(q_0, 1) \cup \delta_N^*(q_1, 1) \cup \delta_N^*(q_2, 1) = \{q_1, q_2\}$$

makes it necessary to introduce yet another state $\{q_1, q_2\}$. At this point, we have the partially constructed automaton shown in Figure 2. Since there are still some missing edges, we continue until we obtain the complete solution in Figure 3.

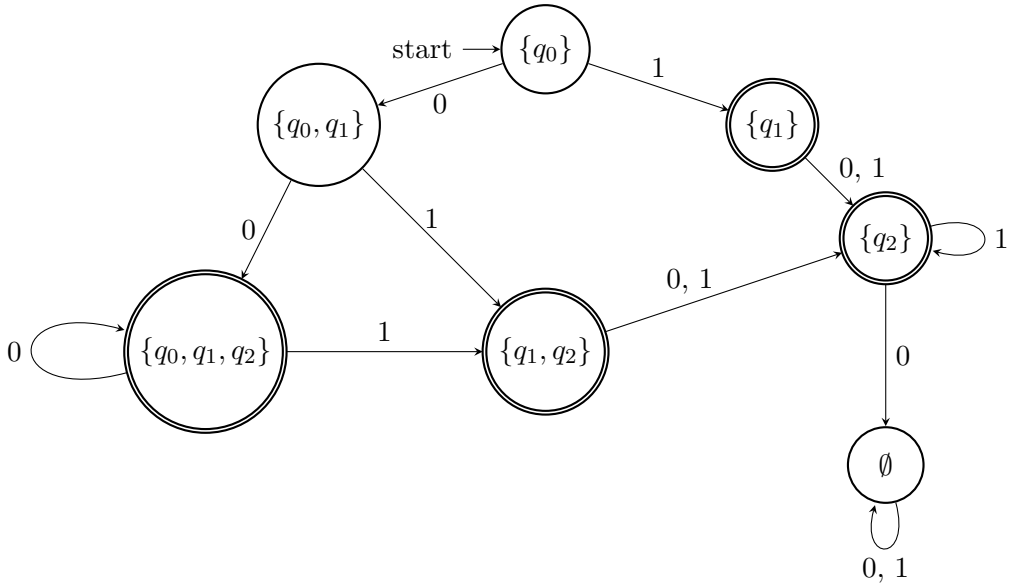


Figure 3

3 Regular Expressions

Definition 3.1. Let Σ be a given alphabet. Then:

1. $\emptyset$, λ, and $a \in \Sigma$ are all regular expressions. These are called **primitive regular expressions**.
2. If r_1 and r_2 are regular expressions, so are $r_1 + r_2$, $r_1 \cdot r_2$, r_1^*, and (r_1).
3. A string is a regular expression if and only if it can be derived from the primitive regular expressions by a finite number of applications of the rules in (2).

Example 3.1. For $\Sigma = \{a, b\}$, the string $(a + b+)$ is *not* a regular expression, because the last '+' has no valid expression on its right-hand side. Similarly, the string (a^*bc) is *not* a regular expression, since $c \notin \Sigma$.

References

[1] Peter Linz, *An Introduction to Formal Languages and Automata*, 6th Edition, Jones & Bartlett Learning, 2016.