Finite Automata and Formal Languages

TMV026/DIT321– LP4 2012

Lecture 3
Ana Bove

March 15th 2012

Overview of today's lecture:

- Some Concepts in Discrete Mathematics
- Central Concepts of Automata Theory

Sets

Definition: A *set* is a collection of well defined and distinct objects.

Some operations on sets:

Union: $S_1 \cup S_2 = \{x \mid x \in S_1 \text{ or } x \in S_2\}$

Intersection: $S_1 \cap S_2 = \{x \mid x \in S_1 \text{ and } x \in S_2\}$

Cartesian Product: $S_1 \times S_2 = \{(x, y) \mid x \in S_1 \text{ and } y \in S_2\}$
Observe this is a collection of ordered pairs!

Complement: $S - A$ is the set of all elements in set S not in set A.
When the set S is known, $S - A$ is sometimes written $\overline{A}$.

Some Particular Sets

Empty set: $\emptyset$ is the set with no elements. We have $\emptyset \subseteq S$ for all sets S.

Singleton sets: Sets with only one element: $\{p_0\}$, $\{p_1\}$

Finite sets: Set with a finite number n of elements:
$\{p_1, \ldots, p_n\} = \{p_1\} \cup \ldots \cup \{p_n\}$

Power sets: $\mathcal{P}ow(S)$ the set of all subsets of the set S.
$\mathcal{P}ow(S) = \{A \mid A \subseteq S\}$.
Observe that $\emptyset \in \mathcal{P}ow(S)$ and $S \in \mathcal{P}ow(S)$.
Also, if $|S| = n$ then $|\mathcal{P}ow(S)| = 2^n$.

(Equivalent) Relations

Definition: A (binary) relation R between two sets A and B is a subset of $A \times B$, that is, $R \subseteq A \times B$.

Notation: $(a, b) \in R$, aRb, $R(a, b)$, (a, b) satisfies R.

Definition: A relation R over a set S, that is $R \subseteq S \times S$, is

Reflexive: $\forall a \in S,\ aRa$

Symmetric: $\forall a, b \in S,\ aRb \Rightarrow bRa$

Transitive: $\forall a, b, c \in S,\ aRb \wedge bRc \Rightarrow aRc$

Definition: A relation R over a set S that is reflexive, symmetric and transitive is called an *equivalence relation* over S.

Example of Relations

Let $S = \{1, 2, 3\}$. Which of these relations are reflexive, symmetric, transitive?

- $R_1 = \{(1, 2)\}$
- $R_2 = \{(1, 2), (2, 3)\}$
- $R_3 = \{(1, 2), (2, 3), (1, 3)\}$
- $R_4 = \{(1, 2), (2, 1)\}$
- $R_5 = \{(1, 2), (2, 1), (1, 1)\}$
- $R_6 = \{(1, 2), (2, 1), (1, 1), (2, 2)\}$
- $R_7 = \{(1, 2), (2, 1), (1, 1), (2, 2), (3, 3)\}$

Partitions

Definition: A set P is a *partition* over the set S if:

- Every element of P is a non-empty subset of S

$$\forall C \in P,\ C \neq \emptyset \wedge C \subseteq S$$

- Elements of P are pairwise disjoint

$$\forall C_1, C_2 \in P,\ C_1 \neq C_2 \Rightarrow C_1 \cap C_2 = \emptyset$$

- The union of the elements of P is equal to S

$$\bigcup_{C \in P} C = S$$

Equivalent Classes

Let R be an equivalent relation over S.

Definition: If $a \in S$, then the *equivalent class* of a in S is the set defined as $[a] = \{b \in S \mid aRb\}$.

Lemma: $\forall a, b \in S,\ [a] = [b]$ *iff* aRb.

Theorem: *The set of all equivalence classes in S with respect to R form a partition over S.*

Note: This partition is called the *quotient* and it is denoted as S/R.

Example: The rational numbers $\mathbb{Q}$ can be formally defined as the equivalence classes of the quotient set $\mathbb{Z} \times \mathbb{Z}^+ / \sim$, where $\sim$ is the equivalence relation defined by $(m_1, n_1) \sim (m_2, n_2)$ iff $m_1 n_2 =_{\mathbb{Z}} m_2 n_1$.

Central Concepts of Automata Theory: Alphabets

Definition: An *alphabet* is a finite, non-empty set of symbols, usually denoted by Σ.
The number of symbols in Σ is denoted as $|\Sigma|$.

Type convention: We will use $a, b, c, \ldots$ to denote symbols.

Note: Alphabets will represent the observable events of the automata.

Example: Some alphabets:

- on/off-switch: $\Sigma = \{\text{Push}\}$
- simple vending machine: $\Sigma = \{5\ kr, \text{choc}\}$
- complex vending machine: $\Sigma = \{5\ kr, 10\ kr, \text{choc}, \text{big choc}\}$
- parity counter: $\Sigma = \{p_0, p_1\}$

Strings or Words

Definition: *Strings/Words* are finite sequence of symbols from some alphabet.

Type convention: We will use $w, x, y, z, \ldots$ to denote words.

Note: A word will represent the *behaviour* of an automaton.

Example: Some behaviours:

- on/off-switch: Push Push Push Push ...
- simple vending machine: 5 *kr* choc 5 *kr* choc 5 *kr* choc ...
- parity counter: p_0p_1 or $p_0p_0p_0p_1p_1p_0$ or ...

Inductive Definition of Σ^*

Definition: Σ^* is the set of all words for a given alphabet Σ.
This can be described inductively in at least 2 different ways:

1. Basis case: the empty word ϵ is in Σ^* (notation: $\epsilon \in \Sigma^*$)
   Inductive step: if $a \in \Sigma$ and $x \in \Sigma^*$ then $ax \in \Sigma^*$
2. Basis case: $\epsilon \in \Sigma^*$
   Inductive step: if $a \in \Sigma$ and $x \in \Sigma^*$ then $xa \in \Sigma^*$

We can (recursively) *define* functions over Σ^* and (inductively) *prove* properties about those functions.

Length

Definition: The *length* function $|\_| : \Sigma^* \to \mathbb{N}$ is defined as:

$$|\epsilon| = 0$$
$$|ax| = 1 + |x|$$

Example: $|p_0p_1p_1p_0p_0| = 5$

Concatenation

Definition: Given the strings x and y, the *concatenation* xy is defined as:

$$\epsilon y = y$$
$$(ax)y = a(xy)$$

Example: Observe that in general $xy \neq yx$.
If $x = p_0p_1p_1$ and $y = p_0p_0$ then $xy = p_0p_1p_1p_0p_0$ and $yx = p_0p_0p_0p_1p_1$.

Lemma: *If Σ has more than one symbol then concatenation is not commutative.*

Power

Of a string: We define x^n as follows:

$$x^0 = \epsilon$$
$$x^{n+1} = xx^n$$

Example: $(p_0 p_1 p_0)^3 = p_0 p_1 p_0 p_0 p_1 p_0 p_0 p_1 p_0$

Of an alphabet: We define Σ^n, the set of words over Σ with length n, as follows:

$$\Sigma^0 = \{\epsilon\}$$
$$\Sigma^{n+1} = \{ax \mid a \in \Sigma,\ x \in \Sigma^n\}$$

Example:
$\{0,1\}^3 = \{000, 001, 010, 011, 100, 101, 110, 111\}$.

Observe: $\Sigma^* = \Sigma^0 \bigcup \Sigma^1 \bigcup \Sigma^2 \ldots$ and $\Sigma^+ = \Sigma^1 \bigcup \Sigma^2 \bigcup \Sigma^3 \ldots$

Reverse Function

Intuitively, $\text{rev}(a_1 \ldots a_n) = a_n \ldots a_1$.

Definition: Formally we can define $\text{rev}(x)$ as:

$$\text{rev}(\epsilon) = \epsilon$$
$$\text{rev}(ax) = \text{rev}(x)a$$

Some Properties

The following properties can be proved by induction:

Lemma: *Concatenation is associative:* $\forall x, y, z.\ x(yz) = (xy)z$.
We shall simply write xyz.

Lemma: $\forall x, y.\ |xy| = |x| + |y|$.

Lemma: $\forall x, y.\ x\epsilon = \epsilon x = x$.

Lemma: $\forall x.\ |x^n| = n|x|$.

Lemma: $\forall \Sigma.\ |\Sigma^n| = |\Sigma|^n$.

Lemma: $\forall x, \mathrm{rev}(\mathrm{rev}(x)) = x$.

Lemma: $\forall x, y.\ \mathrm{rev}(xy) = \mathrm{rev}(y)\mathrm{rev}(x)$.

Some Terminology

Definition: Given x and y words over a certain alphabet Σ:

- x is a *prefix* of y iff there exists z such that $y = xz$
- x is a *suffix* of y iff there exists z such that $y = zx$
- x is a *palindrome* iff $x = \mathrm{rev}(x)$

Languages

Definition: Given an alphabet Σ, a *language* $\mathcal{L}$ is a subset of Σ^*, that is, $\mathcal{L} \subseteq \Sigma^*$.

Note: If $\mathcal{L} \subseteq \Sigma^*$ and $\Sigma \subseteq \Delta$ then $\mathcal{L} \subseteq \Delta^*$.

Note: A language can be either finite or infinite.

Example: Some languages:

- Swedish, English, Spanish, French, ...
- Any programming language
- $\emptyset$, $\{\epsilon\}$ and Σ^* are languages over any Σ
- The set of prime natural numbers $\{1, 3, 5, 7, 11, \ldots\}$

Some Operations on Languages

Definition: Given $\mathcal{L}$, $\mathcal{L}_1$ and $\mathcal{L}_2$ languages, we define the following languages:

Union, Intersection, ... : As for any set

Concatenation: $\mathcal{L}_1\mathcal{L}_2 = \{x_1x_2 \mid x_1 \in \mathcal{L}_1,\ x_2 \in \mathcal{L}_2\}$

Closure: $\mathcal{L}^* = \bigcup_{n\in\mathbb{N}} \mathcal{L}^n$
where $\mathcal{L}^0 = \{\epsilon\}$, $\mathcal{L}^{n+1} = \mathcal{L}^n\mathcal{L}$.

Note: We have then that $\emptyset^* = \{\epsilon\}$ and
$\mathcal{L}^* = \mathcal{L}^0 \cup \mathcal{L}^1 \cup \mathcal{L}^2 \cup \ldots = \{\epsilon\} \cup \{x_1 \ldots x_n \mid n > 0, x_i \in \mathcal{L}\}$

Notation: $\mathcal{L}^+ = \mathcal{L}^1 \cup \mathcal{L}^2 \cup \mathcal{L}^3 \cup \ldots$ and $\mathcal{L}? = \mathcal{L} \cup \{\epsilon\}$.

How to Prove the Equality of Languages?

Given the languages $\mathcal{L}$ and $\mathcal{M}$, how can we prove that $\mathcal{L} = \mathcal{M}$?

A few possibilities:

- Languages are sets so we prove that $\mathcal{L} \subseteq \mathcal{M}$ and $\mathcal{M} \subseteq \mathcal{L}$
- We can reason about the elements in the language:

  Example: $\{a(ba)^n \mid n \geqslant 0\} = \{(ab)^n a \mid n \geqslant 0\}$ can be proved by induction on n.
- Transitivity of equality: $\mathcal{L} = \mathcal{L}_1 = \ldots = \mathcal{L}_m = \mathcal{M}$

Algebraic Laws for Languages

The following equalities hold for any languages $\mathcal{L}$, $\mathcal{M}$ and $\mathcal{N}$:

- Associativity: $\mathcal{L} \cup (\mathcal{M} \cup \mathcal{N}) = (\mathcal{L} \cup \mathcal{M}) \cup \mathcal{N}$, $\mathcal{L} \cap (\mathcal{M} \cap \mathcal{N}) = (\mathcal{L} \cap \mathcal{M}) \cap \mathcal{N}$ and $\mathcal{L}(\mathcal{M}\mathcal{N}) = (\mathcal{L}\mathcal{M})\mathcal{N}$
- Commutative: $\mathcal{L} \cup \mathcal{M} = \mathcal{M} \cup \mathcal{L}$ and $\mathcal{L} \cap \mathcal{M} = \mathcal{M} \cap \mathcal{L}$
- In general, concatenation is not commutative: $\mathcal{L}\mathcal{M} \neq \mathcal{M}\mathcal{L}$
- Distributivity: $\mathcal{L}(\mathcal{M} \cup \mathcal{N}) = \mathcal{L}\mathcal{M} \cup \mathcal{L}\mathcal{N}$ and $(\mathcal{M} \cup \mathcal{N})\mathcal{L} = \mathcal{M}\mathcal{L} \cup \mathcal{N}\mathcal{L}$
- Identity (or neutral): $\mathcal{L} \cup \emptyset = \emptyset \cup \mathcal{L} = \mathcal{L}$ and $\mathcal{L}\{\epsilon\} = \{\epsilon\}\mathcal{L} = \mathcal{L}$
- Annihilator: $\mathcal{L}\emptyset = \emptyset\mathcal{L} = \emptyset$
- Idempotent: $\mathcal{L} \cup \mathcal{L} = \mathcal{L}, \mathcal{L} \cap \mathcal{L} = \mathcal{L}$
- $\emptyset^* = \{\epsilon\}^* = \{\epsilon\}$
- $\mathcal{L}^+ = \mathcal{L}\mathcal{L}^* = \mathcal{L}^*\mathcal{L}$
- $(\mathcal{L}^*)^* = \mathcal{L}^*$

Algebraic Laws for Languages (Cont.)

Note: While

$$\mathcal{L}(\mathcal{M} \cap \mathcal{N}) \subseteq \mathcal{L}\mathcal{M} \cap \mathcal{L}\mathcal{N} \quad \text{and} \quad (\mathcal{M} \cap \mathcal{N})\mathcal{L} \subseteq \mathcal{M}\mathcal{L} \cap \mathcal{N}\mathcal{L}$$

both hold, in general

$$\mathcal{L}\mathcal{M} \cap \mathcal{L}\mathcal{N} \subseteq \mathcal{L}(\mathcal{M} \cap \mathcal{N}) \quad \text{and} \quad \mathcal{M}\mathcal{L} \cap \mathcal{N}\mathcal{L} \subseteq (\mathcal{M} \cap \mathcal{N})\mathcal{L}$$

don't.

Example: Consider the case where

$$\mathcal{L} = \{\epsilon, a\}, \quad \mathcal{M} = \{a\}, \quad \mathcal{N} = \{aa\}$$

Then $\mathcal{L}\mathcal{M} \cap \mathcal{L}\mathcal{N} = \{aa\}$ but $\mathcal{L}(\mathcal{M} \cap \mathcal{N}) = \mathcal{L}\emptyset = \emptyset$.

Functions between Languages

Definition: A function $f : \Sigma^* \to \Delta^*$ between 2 languages should be such that it satisfies

$$f(\epsilon) = \epsilon$$
$$f(xy) = f(x)f(y)$$

Intuitively, $f(a_1 \ldots a_n) = f(a_1) \ldots f(a_n)$.
Notice that $f(a) \in \Delta^*$ if $a \in \Sigma$.

Definition: f is called *coding* iff f is *injective*.

Definition: $f(\mathcal{L}) = \{f(x) \mid x \in \mathcal{L}\}$.

Some Terminology

Definition: A *problem* is the question of deciding if a given string is a member of some particular language.

A “problem” can be expressed as membership in a language.

If $\mathcal{L}$ is a language over Σ then the problem $\mathcal{L}$ is:

$$\text{given } w \in \Sigma^* \text{ decide whether or not } w \text{ is in } \mathcal{L}.$$